User Manual

32

V850E2/Fx4-H

32-bit Microcontroller

V850E2/FK4-H
μPD70F3561

V850E2/FL4-H
μPD70F3564

Renesas Electronics
www.renesas.com

R01UH0002ED0200, Rev. 2.00
April 29, 2013

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: “Standard”, “High Quality”, and “Specific”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as “Specific” without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as “Specific” or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics.

The quality grade of each Renesas Electronics product is “Standard” unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

“Standard”:	Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.
“High Quality”:	Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.
“Specific”:	Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.

8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU products from Renesas. For detailed usage notes on the products covered by this manual, refer to the relevant sections of the manual. If the descriptions under General Precautions in the Handling of MPU/MCU Products and in the body of the manual differ from each other, the description in the body of the manual takes precedence.

1. Handling of unused Pins

   Handle unused pins in accord with the directions given under Handling of Unused Pins in the manual.

   – The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at power-on

   The state of the product is undefined at the moment when power is supplied.

   – The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.

     In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.

     In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited.

   – The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   – When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between products

   Before changing from one product to another, i.e. to one with a different part number, confirm that the change will not lead to problems.

   – The characteristics of MPU/MCU in the same group but having different part numbers may differ because of the differences in internal memory capacity and layout pattern. When changing to products of different part numbers, implement a system-evaluation test for each of the products.

Table of Contents

Notice 2

General Precautions in the Handling of MPU/MCU Products 4

Table of Contents 5

How to use this manual 32

Purpose and target readers 32

Special notations 32

Electrical specifications 32

Additional documents 32

Content of this manual 33

Notation of numbers and symbols 34

Diagrams 35

Trademarks 35

Functional modules descriptions 36

Functional modules abbreviation convention 36

Product specific features 36

Further information 41

Chapter 1 Introduction 42

1.1 V850E2/Fx4-H Product Line Overview 42

1.2 Related Documents 48

1.3 Ordering Information 48

1.4 Product Name Register 49

Chapter 2 Port Functions 50

2.1 V850E2/Fx4-H Port Features 50

2.2 Overview 51

2.2.1 Terms 52

2.2.2 Pin function configuration 53

2.2.3 Pin data input/output 55

2.2.4 Port control logic diagram 57

2.3 Port Group Configuration Registers 58

2.3.1 Writing to protected registers 58

2.3.2 Port control registers overview 58

2.3.3 Port function configuration registers 60

2.3.4 Data input/output registers 68

2.3.5 Configuration of electrical characteristics registers 73

2.4 79

2.5 V850E2/Fx4-H Port Group Configuration 80

2.5.1 Port register protection clusters 80

2.5.2 Common port functions 81

2.5.3 V850E2/FK4-H port functions 86

2.5.4 V850E2/FL4-H port functions . . . 97
2.5.5 Non-port input/output signals . . . 110
2.5.6 Alphabetic pin function list . . . 111
2.5.7 Port and pin functions in stand-by modes . . . 122
2.5.8 Port and pin functions during and after reset . . . 122
2.5.9 Recommended connection of unused pins . . . 123
2.6 Port Filters . . . 125
2.6.1 Port filters assignment . . . 125
2.6.2 Port filters clock supply . . . 129
2.6.3 Port filters reset . . . 131
2.7 Port Filters Functional Description . . . 132
2.7.1 Analog filters . . . 132
2.7.2 Digital filters . . . 135
2.7.3 Filter control registers . . . 139

Chapter 3 CPU System Functions . . . 143
3.1 Overview . . . 143
3.2 Peripheral Protection Unit . . . 145
3.3 Timing Supervision Unit . . . 148
3.4 Memory Protection Unit (MPU) . . . 148
3.5 CPU Access Bus Structures and Latencies . . . 149
3.5.1 CPU Subsystem modules access . . . 149
3.5.2 PBUS/HBUS modules access . . . 150
3.5.3 PBUS Synchronizer . . . 153
3.5.4 Module wait clocks insertion . . . 155
3.6 CPU Subsystem . . . 156
3.6.1 Power and clock domain . . . 156
3.6.2 CPU Subsystem busses overview . . . 157
3.6.3 V850E2K/Fx4-H CPU Subsystem . . . 158
3.6.4 V850E2 system manual . . . 163
3.7 Data flash wait cycle control . . . 164
3.8 HBUS Cross Connection System . . . 166
3.8.1 HBUS connection matrix . . . 168
3.8.2 HBUS arbiters policy . . . 169
3.9 Operation modes . . . 170
3.9.1 Normal operation mode . . . 170
3.9.2 Flash programming mode . . . 170
3.9.3 Boundary Scan mode . . . 171
3.10 Mode pins and JP0 connections . . . 172
3.10.1 Normal operation mode . . . 173
3.10.2 Debug mode . . . 174
3.10.3 Flash programming mode . . . 176
3.10.4 Boundary scan mode . . . 177
3.11 Address Space . . . 178
3.11.1 CPU data address and physical program address space . . . 178
3.11.2 Program and data space . . . 178
3.12 V850E2/Fx4-H CPU Address Map . . . 180
3.12.1 DMA address map . . . 180

3.12.2 V850E2/Fx4-H memory map . . . 181
3.12.3 Memory areas . . . 182
3.13 Back-up RAM (BURAM) . . . 184
3.13.1 Back-up RAM protection . . . 185
3.14 CPU Subsystem - HBUS Bridge . . . 187
3.14.1 Functional description . . . 188
3.14.2 Register overview . . . 190
3.14.3 Master I/F registers details . . . 191
3.14.4 Slave I/F registers details . . . 201
3.15 Write protected Registers . . . 206
3.15.1 Register protection clusters . . . 206
3.15.2 Register protection unlock sequence . . . 207
3.15.3 Register protection and interrupt/emulation break . . . 208
3.15.4 V850E2/Fx4-H write protected registers . . . 209
3.15.5 V850E2/Fx4-H Protection registers overview . . . 211
3.15.6 Control protection clusters registers details . . . 213
3.15.7 Clock monitors protection cluster registers details . . . 215
3.15.8 Port protection clusters registers details . . . 216
3.15.9 Self-programming protection cluster registers details . . . 217

Chapter 4 External Memory Controller (MEMC) . . . 218
4.1 V850E2/Fx4-H MEMC Features . . . 218
4.2 Overview . . . 219
4.2.1 Operation mode, connectable memory types . . . 219
4.2.2 Chip select output function . . . 219
4.2.3 Operation setting function . . . 219
4.2.4 Bus sizing function . . . 219
4.2.5 Data endian setting function . . . 219
4.2.6 Programmable wait setting functions . . . 220
4.2.7 External wait function . . . 220
4.3 Registers . . . 221
4.4 Bus cycle type setting function . . . 233
4.4.1 Multiplexed bus mode . . . 233
4.5 Bus control function . . . 235
4.5.1 Chip select output function . . . 235
4.5.2 Operation enable/operation disable setting function . . . 235
4.5.3 Bus size setting function . . . 236
4.5.4 Data endian setting function . . . 237
4.6 Wait Functions . . . 237
4.6.1 Programmable data wait function . . . 238
4.6.2 External wait function . . . 239
4.6.3 Data setup wait function . . . 240
4.6.4 Data hold wait function . . . 241
4.6.5 Address setup wait function . . . 242
4.6.6 Address hold wait function . . . 243
4.6.7 Idle insertion function . . . 244
4.7 Memory Connection Examples . . . 245
4.7.1 Multiplexed bus mode connection example . . . 245

4.8 Data Flow . . . 246
4.8.1 Data flow during byte access . . . 247

Chapter 5 Interrupt Functions . . . 257
5.1 Exceptions and Interrupts . . . 257
5.1.1 Exception handler switching . . . 259
5.2 V850E2/Fx4-H Exceptions . . . 260
5.2.1 Memory error exceptions MEP . . . 260
5.2.2 System error exceptions SYSERR . . . 261
5.2.3 Code flash error correction . . . 265
5.3 V850E2/Fx4-H Interrupt Requests . . . 269
5.3.1 V850E2/Fx4-H interrupt sources . . . 269
5.3.2 V850E2/Fx4-H FE level non-maskable interrupt sharing . . . 290
5.3.3 V850E2/Fx4-H EI level maskable interrupt sharing . . . 292
5.3.4 V850E2/Fx4-H TAPA EI level maskable interrupt sharing . . . 295
5.3.5 V850E2/Fx4-H DMA interrupt selection . . . 296
5.4 External Interrupts . . . 297
5.4.1 Edge detection configuration . . . 297
5.4.2 External interrupts as trigger and wake-up signals . . . 298
5.5 Interrupt Controller Control Registers . . . 299
5.6 Interrupt Acknowledgment and Restoring . . . 309
5. 6. 1 FE level non-maskable interrupt FENMI . . . 309
5. 6. 2 FE level maskable interrupt FEINT . . . 311
5. 6. 3 EI level maskable interrupt INTn . . . 313
5.7 Interrupt Operation . . . 316
5. 7. 1 Mask function of EI level maskable interrupt INTn . . . 316
5. 7. 2 Interrupt priority level judgment . . . 316
5. 7. 3 Priority mask function . . . 322
5. 7. 4 Pending interrupt report function . . . 322
5. 7. 5 In-service priority clear function . . . 323
5.8 Exception Handler Address Switching Function . . . 323

Chapter 6 DMA/DTS Controller (DMAC) . . . 324
6.1 V850E2/Fx4-H DMA Features . . . 324
6.2 Definition of Terms . . . 329
6.3 General . . . 330
6.3.1 DMA controller (DMAC) function . . . 330
6.3.2 DMA trigger factor register (DTFR) function . . . 330
6.3.3 Data transfer service (DTS) function . . . 330
6.3.4 DTS factor selector (DTSFSL) function . . . 331
6.3.5 DMA access memory map . . . 332
6.3.6 Prioritization of channels . . . 332
6.3.7 Arbitration of transfer requests . . . 333
6.3.8 Stand-by function . . . 334
6.4 DMAC Function . . . 335
6.4.1 Features . . . 335
6.4.2 DMAC setting registers . . . 337
6.4.3 Enabling or disabling writing control registers . . . 347

6.5 DMA Control Registers . . . 348
6.5.1 DTRCx – DMA transfer request control register (x = 0, 1) . . . 348
6.5.2 DTRSn – DMA transfer request select register . . . 349
6.5.3 DSAnL – DMA source address register L . . . 350
6.5.4 DSAnH – DMA source address register H . . . 352
6.5.5 DSCn – DMA source chip select register . . . 353
6.5.6 DNSAnL – DMA next source address register L . . . 354
6.5.7 DNSAnH – DMA next source address register H . . . 355
6.5.8 DNSCn – DMA next source chip select register . . . 356
6.5.9 DDAnL – DMA destination address register L . . . 357
6.5.10 DDAnH – DMA destination address register H . . . 359
6.5.11 DDCn – DMA destination chip select register . . . 360
6.5.12 DNDAnL – DMA next destination address register L . . . 361
6.5.13 DNDAnH – DMA next destination address register H . . . 362
6.5.14 DNDCn – DMA next destination chip select register . . . 363
6.5.15 DTCn – DMA transfer count register . . . 364
6.5.16 DNTCn – DMA next transfer count register . . . 365
6.5.17 DTCCn – DMA transfer count compare register . . . 366
6.5.18 DTCTn – DMA transfer control register . . . 367
6.5.19 DTSn – DMA transfer status register . . . 369
6.6 DMAC Function Details . . . 371
6.6.1 DMAC transfer setting flow . . . 371
6.6.2 DMAC transfer modes . . . 373
6.6.3 DMAC channel priority control . . . 376
6.6.4 Valid DMA transfer request conditions . . . 377
6.6.5 Next address function . . . 378
6.6.6 Aborting/resuming DMA transfer . . . 379
6.6.7 Error response support . . . 380
6.6.8 Stand-by support . . . 380
6.7 DTFR Function . . . 381
6.7.1 Features . . . 381
6.8 DTFR Control Registers . . . 382
6.8.1 DTFRn – DTFRn register . . . 382
6.8.2 DRQCLR – DMA request clear register . . . 383
6.8.3 DRQSTR – DMA request check register . . . 384
6.9 DTS Function . . . 385
6.9.1 Features . . . 385
6.10 DTS Control Registers . . . 388
6.10.1 DTS0TSR – DTS transfer status register [register group A] . . . 389
6.10.2 DTS0TRC – DTS transfer request control register [register group A] . . . 390
6.10.3 DTS0ICR – DTS initialization control register [register group A] . . . 391
6.10.4 DTS0BTR – DTS base table register [register group A] . . . 393
6.10.5 DTS0BVR – DTS base vector register [register group A] . . . 394
6.10.6 DTS0ACR – DTS active channel register [register group A] . . . 395
6.10.7 DTS0TST – DTS TI hold status register [register group A] . . . 396
6.10.8 DTS0HC – DTS TI hold channel number register [register group A] . . . 397
6.10.9 DTS0SAR – DTS source address register [register group B] . . . 399
6.10.10 DTS0DAR – DTS destination address register [register group B] . . . 400

6.10.11 DTS0TCEA – DTS transfer counter or else address register [register group B] 401
6.10.12 DTS0SCS – DTS source address count size register [register group B] . . 403
6.10.13 DTS0DCS – DTS destination address count size register [register group B] . . 404
6.10.14 DTS0CIR – DTS control information register [register group B] 405
6.10.15 DTS0ECSRA – DTS extension address count size/repeat address registers [register group B]. 407
6.11 DTS Function Details 409
6.11.1 Transfer information (TI) 409
6.11.2 DTS transfer setting flow 412
6.11.3 Basic operations of DTS 413
6.11.4 Transfer modes 414
6.11.5 Transfer types 415
6.11.6 Special functions 431
6.11.7 Transfer count 433
6.11.8 Chain function 433
6.11.9 TI hold function 435
6.11.10 Interrupt output function 436
6.11.11 Interrupt and chain of transfers during block transfer 436
6.11.12 TI write back skip function 437
6.11.13 DTS channel priority control 438
6.11.14 Valid DTS transfer request conditions 438
6.11.15 Aborting/resuming DTS transfer 438
6.11.16 Error response support 439
6.11.17 Stand-by support 440
6.12 DTSFSL Function 441
6.12.1 Features 441
6.13 DTSFSL Control Registers 442
6.13.1 DTSENn – DTS transfer enable register 442
6.13.2 DTSHENm – DTS hold enable register (m = 0 to 3) 443
6.14 DTSFSL Function Details 445
6.14.1 Interrupt output function 445

Chapter 7 Flash Memory 446
7.1 Code Flash Memory Overview 446
7.1.1 Code flash memory features 446
7.1.2 Code flash memory map 447
7.1.3 Data flash memory map 447
7.2 Code Flash Memory functional Outline 448
7.2.1 Code flash memory erasure and rewrite 451
7.3 Data Flash Memory 452
7.3.1 Data flash memory features 452
7.3.2 Data flash reading and writing. 452
7.4 Flash Programming with Flash Programmer 453
7.4.1 Programming environment 453
7.4.2 Communication modes 454
7.4.3 Pin connection with flash programmer PG-FP5 455
7.4.4 Flash memory programming control 456

7.5 Code Flash Self-Programming . . . 463
7.5.1 Self-Programming enable . . . 464
7.5.2 Self-Programming library functions . . . 465
7.5.3 Self-Programming internal RAM occupancy . . . 465
7.5.4 Secure Self-Programming (boot cluster swapping) . . . 466
7.5.5 Interrupt handling during flash Self-Programming . . . 470
7.6 Flash Mask Options . . . 471
7.6.1 OPBT0 - Flash mask option register 0 . . . 472

Chapter 8 Data CRC Function A (DCRA) . . . 474
8.1 V850E2/Fx4-H DCRA Features . . . 474
8.2 Functional Overview . . . 476
8.3 Functional Description . . . 477
8.4 Registers . . . 478
8.4.1 DCRA registers overview . . . 478
8.4.2 DCRA registers details . . . 479

Chapter 9 Clock Controller . . . 482
9.1 Clock Controller Overview . . . 483
9.2 General Description of Clock Generation and Control . . . 486
9.2.1 Clock generators . . . 489
9.2.2 Clock selectors . . . 491
9.3 Clock Generators . . . 493
9.3.1 Main Oscillator (MainOsc) clock generator . . . 493
9.3.2 Sub Oscillator (SubOsc) clock generator . . . 496
9.3.3 Low Speed Internal Oscillator (Low Speed IntOsc) clock generator . . . 498
9.3.4 High Speed Internal Oscillator (High Speed IntOsc) clock generator . . . 499
9.3.5 Phase-Locked Loop (PLL) clock generators . . . 502
9.4 Clock Selection . . . 506
9.4.1 Clock domains of Always-On-Area . . . 509
9.4.2 Clock domains of Isolated-Area-0 . . . 514
9.4.3 Clock domains of Isolated-Area-1 . . . 519
9.5 Clock Domain Figures . . . 527
9.6 Frequency Output Function (FOUT) . . . 536
9.6.1 FOUT Clock Divider (FOUTDIV) . . . 537
9.7 Clock Monitor A (CLMA) . . . 539
9.7.1 V850E2/Fx4-H CLMA features . . . 539
9.7.2 CLMA enable and start-up options . . . 542
9.7.3 Functional Overview . . . 544
9.7.4 Functional Description . . . 545
9.7.5 Clock Monitor registers . . . 549
9.8 Clock Controller Registers . . . 555
9.8.1 Writing to protected registers . . . 555
9.8.2 Clock Controller registers overview . . . 555
9.8.3 Clock generators registers . . . 557
9.8.4 Clock selector control register . . . 574

Chapter 10 Stand-by Controller (STBC) 578
10.1 V850E2/Fx4-H Stand-by Controller Features 578
10.2 Stand-by Controller functions 582
10.2.1 Stand-by Controller signal connections 583
10.2.2 Stand-by modes control 584
10.2.3 Stand-by modes overview 586
10.2.4 Clock generators in stand-by 587
10.2.5 Wake-up 588
10.2.6 I/O buffer control 596
10.2.7 Mode transitions 597
10.3 Stand-by mode entry and exit example flows 598
10.3.1 STOP mode 599
10.3.2 RUN mode (Isolated-Area-1 STOP) 602
10.3.3 DEEPSTOP mode 605
10.3.4 RUN mode (Isolated-Area-1 DEEPSTOP) 610
10.3.5 Precaution: Clock generators and early wake-up 614
10.3.6 Application hint: Handling of wake-up events during stand-by mode preparation 616
10.4 Stand-by Controller Registers 619
10.4.1 Writing to protected registers 619
10.4.2 Stand-by Controller registers overview 619
10.4.3 Stand-by Controller control registers details 621
10.4.4 Wake-up factor controller registers details 627
10.4.5 Oscillator wake-up mask registers details 630

Chapter 11 Code Protection and Security 631
11.1 Overview 631
11.2 Flash Programmer and Self-Programming Protection 632
11.3 On-Chip Debug Interface Protection 633
11.3.1 On-Chip Debug enable flag 633
11.3.2 On-Chip Debug ID code 634
11.3.3 On-Chip Debug protection levels summary 634
11.3.4 On-Chip Debug control registers 635

Chapter 12 Reset Controller 638
12.1 Functional Overview 638
12.2 Functional Description 642
12.2.1 Reset flags 642
12.2.2 Power-On-Clear reset (POCRES) 643
12.2.3 Low-Voltage Indicator (LVI) 644
12.2.4 Very-Low-Voltage Indicator (VLVI) 646
12.2.5 External RESET 648
12.2.6 Watchdog Timers reset (WDTAnRES) 650
12.2.7 Software reset (SWRES) 650
12.2.8 Clock Monitors reset (CLMAnRES) 650
12.2.9 Debugger reset (DBRES) 650
12.3 Registers 651
12.3.1 Writing to protected registers 651

12.3.2 Reset Controller registers overview . . . 651
12.3.3 Reset Controller general control registers details . . . 652
12.3.4 Software reset control registers details . . . 654
12.3.5 Low-Voltage Indicator reset control registers . . . 655
12.3.6 Very-Low-Voltage flag control registers . . . 656

Chapter 13 OS Timer (OSTM) . . . 658
13.1 V850E2/Fx4-H OSTM Features . . . 658
13.2 Functional Overview . . . 659
13.3 Functional Description . . . 660
13.3.1 Count clock . . . 661
13.3.2 Interrupt request generation . . . 661
13.3.3 Starting and stopping the timer . . . 662
13.3.4 Interval timer mode . . . 662
13.3.5 Free-run compare mode . . . 666
13.4 OS Timer Registers . . . 668
13.4.1 OS Timer registers overview . . . 668
13.4.2 OS Timer registers details . . . 669

Chapter 14 Window Watchdog Timer A (WDTA) . . . 674
14.1 V850E2/Fx4-H WDTA Features . . . 674
14.2 WDTA Start-up Options . . . 677
14.2.1 V850E2/Fx4-H WDTAn start modes . . . 678
14.3 Functional Overview . . . 679
14.4 Functional Description . . . 680
14.4.1 WDTA after reset release . . . 681
14.4.2 WDTA trigger . . . 684
14.4.3 Error detection . . . 685
14.4.4 75% interrupt output . . . 687
14.4.5 Window function . . . 688
14.5 Application hint: Evaluation of the Watchdog status . . . 689
14.6 WDTA registers . . . 690
14.6.1 WDTA registers overview . . . 690
14.6.2 WDTA registers details . . . 691

Chapter 15 Timer Array Unit A (TAUA) . . . 697
15.1 V850E2/Fx4-H TAUA Features . . . 697
15.2 TAUA Input Selection . . . 701
15.2.1 TAUA0 input selection . . . 701
15.3 Functional Overview . . . 709
15.3.1 Terms . . . 711
15.4 Functional Description . . . 712
15.5 General Operating Procedure . . . 714
15.6 Operation Modes . . . 715
15.7 Concepts of Synchronous Channel Operation . . . 716
15.7.1 Rules . . . 716
15.7.2 Simultaneous start and stop of synchronous channel counters . . . 718

15.8 Simultaneous Rewrite . . . 719
15.8.1 Introduction . . . 719
15.8.2 How to control simultaneous rewrite . . . 721
15.8.3 Other general rules of simultaneous rewrite . . . 722
15.8.4 Types of simultaneous rewrite . . . 723
15.9 Channel Output Modes . . . 731
15.9.1 General procedure for specifying a channel output mode . . . 733
15.9.2 Channel output modes controlled independently by TAUAn signals . . . 734
15.9.3 Channel output modes controlled synchronously by TAUAn signals . . . 736
15.10 Start Timing of Operating Modes . . . 741
15.10.1 Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode . . . 741
15.10.2 Event Mode . . . 742
15.10.3 All other operating modes . . . 742
15.11 TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts . . . 743
15.12 Interrupt Generation upon Overflow . . . 744
15.12.1 Capture Mode . . . 745
15.12.2 Capture and One Count Mode . . . 746
15.12.3 Count Capture Mode . . . 747
15.12.4 Capture and Gate Count Mode . . . 748
15.13 TAUAnTTINm Edge Detection . . . 749
15.14 Assigning DMA Window Addresses . . . 750
15.15 Independent Channel Operation Functions . . . 751
15.16 Independent Channel Interrupt Functions . . . 751
15.16.1 Interval Timer Function . . . 752
15.16.2 TAUAnTTINm Input Interval Timer Function . . . 759
15.16.3 Delay Count Function . . . 765
15.16.4 One-Pulse Output Function . . . 770
15.17 Independent Channel Signal Measurement Functions . . . 775
15.17.1 TAUAnTTINm Input Pulse Interval Measurement Function . . . 776
15.17.2 TAUAnTTINm Input Signal Width Measurement Function . . . 784
15.17.3 Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement) . . . 792
15.17.4 TAUAnTTINm Input Period Count Detection Function . . . 797
15.17.5 Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection) . . . 803
15.17.6 TAUAnTTINm Input Pulse Interval Judgment Function . . . 808
15.17.7 TAUAnTTINm Input Signal Width Judgment Function . . . 813
15.18 Independent Channel Real-Time Functions . . . 818
15.18.1 Real-Time Output Function Type 1 . . . 819
15.18.2 Real-Time Output Function Type 2 . . . 826
15.19 Independent Channel Simultaneous Rewrite Functions . . . 833
15.19.1 Simultaneous Rewrite Trigger Generation Function Type 1 . . . 834
15.19.2 Simultaneous Rewrite Trigger Generation Function Type 2 . . . 840
15.20 Independent Channel One-Phase PWM Function . . . 846
15.20.1 One-Phase PWM Output Function . . . 847
15.21 Other Independent Channel Functions . . . 854
15.21.1 External Event Count Function . . . 855

15.21.2 Clock Divide Function 862
15.21.3 TAUAnTTINm Input Position Detection Function 869
15.22 Synchronous Channel Operation Functions 875
15.23 Synchronous PWM Signal Functions Triggered at Regular Intervals 875
15.23.1 PWM Output Function 876
15.23.2 Trigger Start PWM Output Function 887
15.23.3 Delay Pulse Output Function 898
15.23.4 AD Conversion Trigger Output Function Type 1 914
15.24 Synchronous PWM Signal Functions Triggered by an External Signal 916
15.24.1 One-Shot Pulse Output Function 917
15.24.2 Offset Trigger Output Function 929
15.25 Synchronous Triangle PWM Functions 939
15.25.1 Triangle PWM Output Function 940
15.25.2 Triangle PWM Output Function with Dead Time 951
15.25.3 AD Conversion Trigger Output Function Type 2 974
15.26 Synchronous Real-Time Output Functions 976
15.26.1 Synchronous Real-Time Output Function Type 1 977
15.26.2 Synchronous Real-Time Output Function Type 2 988
15.26.3 Synchronous Real-Time Output Function Type 3 999
15.27 Synchronous Non-Complementary and Complementary Functions 1010
15.27.1 Non-Complementary Modulation Output Function Type 1 1011
15.27.2 Non-Complementary Modulation Output Function Type 2 1024
15.27.3 Complementary Modulation Output Function 1038
15.28 Other Synchronous Channel Functions 1056
15.28.1 Interrupt Culling Function 1056
15.29 Registers 1065
15.29.1 TAUAn registers overview 1065
15.29.2 TAUAn prescaler registers details 1067
15.29.3 TAUAn control registers details 1070
15.29.4 TAUAn output registers details 1080
15.29.5 TAUAn channel output level registers details 1086
15.29.6 TAUAn simultaneous rewrite register details 1087
15.29.7 TAUAn DMA window registers 1090
15.29.8 TAUAn emulation register 1092

Chapter 16 Timer Array Unit B (TAUB) 1093
16.1 V850E2/Fx4-H TAUB Features 1093
16.2 TAUB Input Selection 1100
16.2.1 TAUB1TTIN[7:0] input selections 1100
16.3 Functional Overview 1102
16.3.1 Terms 1104

16.4 Functional Description 1105
16.5 General Operating Procedure 1107
16.6 Operation Modes 1108
16.7 Concepts of Synchronous Channel Operation 1109
16.7.1 Rules 1109
16.7.2 Simultaneous start and stop of synchronous channel counters 1111
16.8 Simultaneous Rewrite 1112
16.8.1 Introduction 1112
16.8.2 How to control simultaneous rewrite 1114
16.8.3 Other general rules of simultaneous rewrite 1115
16.8.4 Types of simultaneous rewrite 1116
16.9 Channel Output Modes 1122
16.9.1 General procedure for specifying a channel output mode. 1124
16.9.2 Channel output modes controlled independently by TAUBn signals 1125
16.9.3 Channel output modes controlled synchronously by TAUBn signals. 1126
16.10 Start Timing of Operating Modes 1129
16.10.1 Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode 1129
16.10.2 Event Mode 1130
16.10.3 All other operating modes 1130
16.11 TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit) 1131
16.12 TAUBnTTINm Edge Detection 1133
16.13 Independent Channel Operation Functions 1134
16.14 Independent Channel Interrupt Functions 1134
16.14.1 Interval Timer Function 1135
16.14.2 TAUBnTTINm Input Interval Timer Function 1143
16.14.3 One-Pulse Output Function. 1149
16.15 Independent Channel Signal Measurement Functions 1154
16.15.1 TAUBnTTINm Input Pulse Interval Measurement Function. 1155
16.15.2 TAUBnTTINm Input Signal Width Measurement Function 1163
16.15.3 Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement) 1171
16.15.4 TAUBnTTINm Input Period Count Detection Function 1176
16.15.5 Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection) 1182
16.15.6 TAUBnTTINm Input Pulse Interval Judgment Function 1187
16.15.7 TAUBnTTINm Input Signal Width Judgment Function. 1192
16.16 Independent Channel Simultaneous Rewrite Functions 1197
16.16.1 Simultaneous Rewrite Trigger Generation Function Type 1 1198
16.17 Other Independent Channel Functions 1204
16.17.1 External Event Count Function 1205
16.17.2 Clock Divide Function 1212
16.17.3 TAUBnTTINm Input Position Detection Function. 1219
16.18 Synchronous Channel Operation Functions 1225
16.19 Synchronous PWM Signal Functions Triggered at Regular Intervals 1225
16.19.1 PWM Output Function 1226
16.19.2 Delay Pulse Output Function. 1237

16.19.3 AD Conversion Trigger Output Function Type 1 1253
16.20 Synchronous PWM Signal Functions Triggered by an External Signal 1255
16.20.1 One-Shot Pulse Output Function 1256
16.21 Synchronous Triangle PWM Functions 1268
16.21.1 Triangle PWM Output Function 1269
16.21.2 Triangle PWM Output Function with Dead Time 1280
16.21.3 AD Conversion Trigger Output Function Type 2 1303
16.22 Registers 1305
16.22.1 TAUBn registers overview 1305
16.22.2 TAUBn prescaler registers details 1306
16.22.3 TAUBn control registers details 1308
16.22.4 TAUBn output registers details 1318
16.22.5 TAUBn channel output level registers details 1321
16.22.6 TAUBn simultaneous rewrite register details 1322
16.22.7 TAUBn emulation register 1325

Chapter 17 Timer Array Unit C (TAUC) 1326
17.1 V850E2/Fx4-H TAUC Features 1326
17.2 Functional Overview 1332
17.2.1 Terms 1333
17.3 Functional Description 1334
17.4 General Operating Procedure 1335
17.5 Operation Modes 1336
17.6 Concepts of Synchronous Channel Operation 1337
17.6.1 Rules 1337
17.6.2 Simultaneous start and stop of synchronous channel counters 1339
17.7 Simultaneous Rewrite 1340
17.7.1 Introduction 1340
17.7.2 How to control simultaneous rewrite 1341
17.7.3 Other general rules of simultaneous rewrite 1342
17.7.4 Types of simultaneous rewrite 1343
17.8 Channel Output Modes 1347
17.8.1 General procedure for specifying a channel output mode 1349
17.8.2 Channel output modes controlled independently by TAUCn signals 1350
17.8.3 Channel output modes controlled synchronously by TAUCn signals 1350
17.9 Start Timing of Operating Modes 1351
17.9.1 Interval Timer Mode 1351
17.9.2 Event Mode 1352
17.10 TAUCnTTOUTm toggle and INTTAUCnIm Generation when Counter start is triggered (MD0-bit) 1353
17.11 Independent Channel Operation Functions 1354
17.12 Independent Channel Interrupt Functions 1354
17.12.1 Interval Timer Function 1355
17.13 Independent Channel Simultaneous Rewrite Functions 1363
17.13.1 Simultaneous Rewrite Trigger Generation Function Type 1 1364

17.14 Synchronous PWM Signal Functions Triggered at Regular Intervals 1370
17.14.1 PWM Output Function . . . 1371
17.15 Registers . . . 1382
17.15.1 TAUCn registers overview . . . 1382
17.15.2 TAUCn prescaler registers details . . . 1383
17.15.3 TAUCn control registers details . . . 1385
17.15.4 TAUCn output registers details . . . 1390
17.15.5 TAUCn channel output level registers details . . . 1391
17.15.6 TAUCn simultaneous rewrite register details . . . 1392
17.15.7 TAUCn emulation register . . . 1395

Chapter 18 Timer Array Unit J (TAUJ) . . . 1396
18.1 V850E2/Fx4-H TAUJ Features . . . 1396
18.2 TAUJ Input Selection . . . 1400
18.2.1 TAUJ0/TAUJ1 input selection . . . 1400
18.3 Functional Overview . . . 1406
18.3.1 Terms . . . 1408
18.4 Functional Description . . . 1409
18.5 General Operating Procedure . . . 1411
18.6 Operation Modes . . . 1412
18.7 Concepts of Synchronous Channel Operation . . . 1413
18.7.1 Rules . . . 1413
18.7.2 Simultaneous start and stop of synchronous channel counters . . . 1415
18.8 Simultaneous Rewrite . . . 1416
18.8.1 Introduction . . . 1416
18.8.2 How to control simultaneous rewrite . . . 1417
18.8.3 Other general rules of simultaneous rewrite . . . 1418
18.8.4 Simultaneous rewrite procedure . . . 1419
18.9 Channel Output Modes . . . 1421
18.9.1 General procedure for specifying a channel output mode . . . 1423
18.9.2 Channel output modes controlled independently by TAUJn signals . . . 1424
18.9.3 Channel output modes controlled synchronously by TAUJn signals . . . 1425
18.10 Start Timing of Operating Modes . . . 1426
18.10.1 Interval Timer Mode, Capture Mode . . . 1426
18.10.2 Other operating modes . . . 1427
18.11 TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts . . . 1428
18.12 Interrupt Generation upon Overflow . . . 1429
18.12.1 Capture Mode . . . 1430
18.12.2 Capture and One Count Mode . . . 1431
18.12.3 Count Capture Mode . . . 1432
18.12.4 Capture and Gate Count Mode . . . 1433
18.13 TAUJnTTINm Edge Detection . . . 1434
18.14 Independent Channel Operation Functions . . . 1435
18.15 Independent Channel Interrupt Functions . . . 1435
18.15.1 Interval Timer Function . . . 1436

18.15.2 TAUJnTTINm Input Interval Timer Function 1443
18.16 Independent Channel Signal Measurement Fuctions 1449
18.16.1 TAUJnTTINm Input Pulse Interval Measurement Function 1450
18.16.2 TAUJnTTINm Input Signal Width Measurement Function 1457
18.16.3 Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement) 1464
18.16.4 TAUJnTTINm Input Period Count Detection Function 1468
18.16.5 Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection) 1474
18.17 Other Independent Channel Functions 1479
18.17.1 TAUJnTTINm Input Position Detection Function 1480
18.18 Synchronous PWM Signal Functions Triggered at Regular Intervals 1486
18.18.1 PWM Output Function 1487
18.19 Registers 1498
18.19.1 TAUJn registers overview 1498
18.19.2 TAUJn prescaler registers details 1499
18.19.3 TAUJn control registers details 1502
18.19.4 TAUJn output registers details 1513
18.19.5 TAUJn simultaneous rewrite register details 1516
18.19.6 TAUJn emulation register 1518

Chapter 19 Real-Time Clock (RTCA) 1519
19.1 V850E2/Fx4-H RTCA Features 1519
19.2 Functional Overview 1522
19.3 Functional Description 1523
19.3.1 Operation modes 1524
19.3.2 Clock counter format 1524
19.3.3 Fixed interval interrupt function 1525
19.3.4 Alarm interrupt function 1526
19.3.5 Clock error correction 1527
19.4 Registers 1531
19.4.1 RTCA registers overview 1531
19.4.2 RTCA control registers details 1533
19.4.3 RTCA sub-counter registers details 1537
19.4.4 RTCA clock counter and buffer registers details 1541
19.4.5 RTCA special counter and buffer registers details 1556
19.4.6 RTCA alarm setting registers details 1560
19.4.7 RTCA emulation register details 1563
19.5 Procedures for Setup, Writing and Reading 1564
19.5.1 Initial setting of the RTCA 1564
19.5.2 Updating clock counters 1566
19.5.3 Reading clock counters 1567
19.5.4 Reading RTCAnSRBU 1570
19.5.5 Writing to RTCAnSUBU 1571
19.5.6 Writing to RTCAnSCMP 1572
19.6 Timing diagrams 1573
19.6.1 Timing of RTCA counter start 1573
19.6.2 Timing of RTCA while counter is enabled 1574

19.6.3 Timing of sub-counter buffer read while counter is enabled 1575

Chapter 20 Motor Control 1576

20.1 Basic structure of motor control 1576

20.2 V850E2/Fx4-H Timer Motor Control Function (TAPA) Features . . 1578

20.3 TAPA0 Hi-Z control input selection 1581

20.4 Functional Overview 1583

20.4.1 Block diagram 1583

20.4.2 Peak and valley interrupts - Peak and valley of timer counter 1584

20.5 Registers 1585

20.5.1 Registers overview 1585

20.5.2 Registers details 1586

20.6 Basic Functions 1591

20.6.1 Asynchronous Hi-Z control function 1591

20.6.2 INT signal output selection function 1598

20.6.3 A/D conversion trigger selection function 1599

20.7 Three-Phase PWM Output with Dead Time 1603

20.7.1 Functional overview 1603

20.7.2 Configuration 1603

20.7.3 Operation example 1606

20.7.4 Setup flow 1615

20.7.5 Example of setting up operation functions 1617

20.7.6 Registers 1624

20.8 High-accuracy Triangle PWM Output with Dead Time 1630

20.8.1 Functional overview 1630

20.8.2 Configuration 1631

20.8.3 Operation example 1634

20.8.4 Setup flow 1647

20.8.5 Example of setting up operation functions 1649

20.8.6 Registers 1657

20.9 Delay Pulse Output with Dead Time 1669

20.9.1 Functional overview 1669

20.9.2 Configuration 1669

20.9.3 Operation example 1672

20.9.4 Setup flow 1678

20.9.5 Example of setting up operation functions 1681

20.9.6 Registers 1688

Chapter 21 Encoder Timer 1694

21.1 Basic structure of Encoder Timer 1694

21.2 V850E2/Fx4-H Encoder Timer (ENCA) Features 1696

21.3 ENCA Functional Overview 1699

21.3.1 Block Diagram 1700

21.3.2 Preliminary knowledge for understanding basic specifications 1701

21.4 ENCA Control Registers 1702

21.5 ENCA Functional Description 1713

21.5.1 Timer counter operation 1713

21.5.2 Up/down control of timer counter . . . 1715
21.5.3 Timer counter clear control by encoder input . . . 1719
21.5.4 Functions of ENCAnCCR0 . . . 1720
21.5.5 Functions of ENCAnCCR1 . . . 1721
21.6 ENCA Setting Sequences . . . 1724
21.6.1 Encoder timer setting procedure . . . 1724
21.7 A/D Trigger Encoder Capture . . . 1727
21.7.1 Functional overview . . . 1727
21.7.2 Configuration . . . 1727
21.7.3 Operation example . . . 1728
21.7.4 Setup flow . . . 1729
21.7.5 Example of setting up operation functions . . . 1730
21.7.6 Registers . . . 1732
21.8 Synchronized Timer Operation . . . 1734
21.8.1 Functional Overview . . . 1734
21.8.2 Configuration . . . 1734
21.8.3 Operation example . . . 1735
21.8.4 Setup flow . . . 1736
21.8.5 Setting up operation functions . . . 1736
21.8.6 Registers . . . 1737
21.9 Trigger Pulse Width Measurement . . . 1739
21.9.1 Functional overview . . . 1739
21.9.2 Configuration . . . 1739
21.9.3 Operation example . . . 1741
21.9.4 Setup flow . . . 1744
21.9.5 Example of setting up operation functions . . . 1747
21.9.6 Registers . . . 1750

Chapter 22 PWM Diagnostic . . . 1756
22.1 PWM Diagnostic functional overview . . . 1756
22.1.1 Basic concept and definitions . . . 1756
22.1.2 PWM generation . . . 1757
22.1.3 Channel-to-channel delay . . . 1758
22.1.4 A/D Converter trigger delay . . . 1759
22.1.5 Synchronous PWM groups multiplexer and A/D Converter control . . . 1760
22.1.6 A/D Converter conversion result memory transfer . . . 1762
22.1.7 Diagnostic value evaluation . . . 1762
22.2 PWM Delay (DLYA) . . . 1764
22.2.1 DLYA features . . . 1764
22.2.2 PWM Delay (DLYA) bypass control . . . 1768
22.2.3 Functional Overview . . . 1770
22.2.4 Registers . . . 1771
22.2.5 Procedure for Setup . . . 1776
22.2.6 Timing diagrams . . . 1776
22.3 PWM Diagnostic timing and trigger generation (PMCA) . . . 1777
22.3.1 Features . . . 1777
22.3.2 Connections . . . 1780
22.3.3 Functional Overview . . . 1787
22.3.4 Registers . . . 1789

22.3.5 Procedures for Setup, Writing and Reading 1793

Chapter 23 Asynchronous Serial Interface E (URTE) 1794
23.1 V850E2/Fx4-H URTEn Features 1794
23.2 Functional Overview 1799
23.3 Configuration 1800
23.4 URTE Registers 1801
23.5 Interrupt Request Signals 1817
23.5.1 Transmission interrupt request INTUAEnTIT 1817
23.5.2 Reception interrupt request INTUAEnTIR 1818
23.5.3 Status interrupt request INTUAEnTIS 1819
23.6 Operation 1819
23.6.1 Data formats 1819
23.6.2 BF transmission/reception format 1821
23.6.3 BF transmission 1823
23.6.4 BF reception 1824
23.6.5 Transmission data consistency check 1826
23.6.6 URTEn transmission 1827
23.6.7 Continuous transmission procedure 1828
23.6.8 URTEn reception 1830
23.6.9 Reception errors 1832
23.6.10 Parity types and operations 1833
23.6.11 Digital receive data noise filter 1834
23.7 Baud Rate Generator 1835

Chapter 24 LIN Master Controller (LMA) 1836
24.1 V850E2/Fx4-H LMAn Features 1836
24.2 LIN Master Scheduler Counters (CNTA) 1842
24.2.1 CNTAm registers 1842
24.3 Functional Overview 1844
24.4 Functional Description 1846
24.4.1 UART through mode 1846
24.4.2 UART buffer mode 1847
24.4.3 LIN master modes 1853
24.4.4 Automatic checksum function 1865
24.4.5 Scheduler 1866
24.5 LMAn Registers 1871
24.5.1 LMAn registers overview 1871
24.5.2 LMAn registers details 1873

Chapter 25 CAN Controller (FCN) 1888
25.1 V850E2/Fx4-H FCN Features 1888
25.2 FCN0 and FCN1 connection 1893
25.3 CAN baudrate and time quanta 1895
25.4 Features 1896
25.4.1 Overview of functions 1896
25.4.2 Configuration 1898

25.5 Internal Registers of CAN Controller . . . 1899
25.5.1 CAN Controller configuration . . . 1899
25.5.2 CAN Controller Registers Overview . . . 1901
25.5.3 Register bit configuration . . . 1903
25.6 Bit Set/Clear Function . . . 1908
25.7 Control Registers . . . 1910
25.7.1 CAN Controller global registers . . . 1910
25.7.2 CAN channel registers . . . 1919
25.7.3 Message buffer registers . . . 1941
25.8 CAN Controller Initialization . . . 1950
25.8.1 Initialization of CAN Controller . . . 1950
25.8.2 Initialization of message buffer . . . 1950
25.8.3 Redefinition of message buffer . . . 1950
25.8.4 Transition from initialization mode to operation mode . . . 1952
25.9 Message Reception . . . 1953
25.9.1 Message reception . . . 1953
25.9.2 Receive data read . . . 1954
25.9.3 Receive history list function . . . 1955
25.9.4 Mask function . . . 1957
25.9.5 Multi buffer receive block function . . . 1959
25.9.6 Remote frame reception . . . 1960
25.10 Message Transmission . . . 1962
25.10.1 Message transmission . . . 1962
25.10.2 Transmit history list function . . . 1964
25.10.3 Automatic block transmission (ABT) . . . 1966
25.10.4 Transmission abort process . . . 1968
25.10.5 Remote frame transmission . . . 1969
25.11 Power Saving Modes . . . 1970
25.11.1 CAN Controller sleep mode . . . 1970
25.11.2 CAN Controller stop mode . . . 1973
25.11.3 Example of using power saving modes . . . 1974
25.12 Interrupt Function . . . 1975
25.13 Diagnosis Functions and Special Operational Modes . . . 1976
25.13.1 Receive-only mode . . . 1976
25.13.2 Single-shot mode . . . 1977
25.13.3 Self-test mode . . . 1978
25.13.4 Receive/transmit operation in each operation mode . . . 1979
25.14 Time Stamp Function . . . 1980
25.14.1 Time stamp function . . . 1980
25.15 Baudrate Settings . . . 1981
25.15.1 Baudrate setting conditions . . . 1981
25.15.2 Clock prescaler and baudrate generator settings . . . 1985
25.16 Operation of the CAN Controller . . . 1987
25.16.1 Initialization . . . 1987
25.16.2 Message transmission . . . 1993
25.16.3 Message reception . . . 2007
25.16.4 Power save modes . . . 2012

Chapter 26 Diagnostic CAN Controller (DCN) 2019
26.1 V850E2/Fx4-H DCN Features 2019
26.2 Introduction 2022
26.3 Overview of Functions 2024
26.4 Architecture 2025
26.4.1 CPU interface 2026
26.4.2 Global module control 2026
26.4.3 CAN interrupt generator 2026
26.4.4 Message control (MSG Ctrl) 2026
26.4.5 Arbitration logic 2027
26.4.6 RXONLY_CH CAN machine 2027
26.4.7 DIAG_CH CAN machine 2028
26.5 Module Initialisation and Control 2029
26.5.1 Global Module initialisation and control 2030
26.5.2 Message buffer initialisation and configuration 2039
26.5.3 Message buffer to CAN I/F channel assignment 2040
26.5.4 DCN module initialisation and control 2045
26.5.5 CAN bit time programming 2045
26.5.6 Transitions for operational modes of DIAG_CH 2046
26.5.7 Transition for operational modes of RXONLY_CH 2048
26.6 Module Interrupts 2050
26.7 Message Reception of RXONLY Channel 2054
26.7.1 Principal reception process 2054
26.7.2 Reception History 2054
26.7.3 Reception of remote frames 2055
26.8 Message Transmission 2055
26.9 Operational Modes of RXONLY_CH 2056
26.9.1 Receive–only mode 2056
26.9.2 Mirror mode 2061
26.9.3 Mirror mode with TIF 2063
26.10 Transitions for Buffer Assignment 2064
26.11 Register Description 2067
26.11.1 Register bit configuration 2067
26.11.2 DCN global registers 2075
26.11.3 DCN module registers 2081
26.11.4 DCN message buffers registers 2098

Chapter 27 Clocked Serial Interface G (CSIG) 2104
27.1 V850E2/Fx4-H CSIG Features 2104
27.1.1 Data consistency check 2109
27.2 Functional Overview 2110
27.3 Functional Description 2112
27.3.1 Master/slave mode 2112
27.3.2 Master/slave connections 2113
27.3.3 Transmission clock selection 2115
27.3.4 Data transfer modes 2116
27.3.5 Data length selection 2117

27.3.6 Serial data direction select function . . . 2119
27.3.7 Communication in slave mode . . . 2120
27.3.8 CSIG interrupts . . . 2121
27.3.9 Handshake function . . . 2123
27.3.10 Loop-back mode . . . 2126
27.3.11 Error detection . . . 2127
27.4 CSIG Control Registers . . . 2130
27.5 Operating Procedure Example . . . 2142

Chapter 28 Clocked Serial Interface H (CSIH) . . . 2144
28.1 V850E2/Fx4-H CSIH Features . . . 2144
28.1.1 Data consistency check . . . 2148
28.2 Functional Overview . . . 2149
28.3 Functional Description . . . 2151
28.3.1 Operating modes (master/slave) . . . 2152
28.3.2 Master/slave connections . . . 2153
28.3.3 Chip selection (CS) features . . . 2155
28.3.4 The job concept . . . 2157
28.3.5 Chip select timing details . . . 2158
28.3.6 Transmission clock selection . . . 2159
28.3.7 CSIH buffer memory . . . 2160
28.3.8 Data transfer modes . . . 2162
28.3.9 Data length selection . . . 2163
28.3.10 Serial data direction selection . . . 2166
28.3.11 Communication in slave mode . . . 2167
28.3.12 CSIH interrupt requests . . . 2168
28.3.13 Handshake function . . . 2176
28.3.14 Error detection . . . 2179
28.3.15 Loop-back mode . . . 2188
28.4 CSIH Control Registers . . . 2189
28.4.1 CSIH registers details . . . 2190
28.5 Operating Procedures . . . 2213
28.5.1 Procedures in direct access mode . . . 2213
28.5.2 Procedures in transmit-only buffer mode . . . 2218
28.5.3 Procedures in dual buffer mode . . . 2222
28.5.4 Procedures in FIFO mode . . . 2228

Chapter 29 I2C Interface (IICB) . . . 2232
29.1 V850E2/Fx4-H IICB Features . . . 2232
29.2 I2C Interface Port Settings . . . 2234
29.3 Functional Overview . . . 2235
29.4 I^2C Bus Mode Functions . . . 2237
29.4.1 Pin configuration . . . 2237
29.5 I^2C Bus Definition . . . 2238
29.5.1 Start Condition . . . 2239
29.5.2 Addresses . . . 2239
29.5.3 Extension code . . . 2240
29.5.4 Transfer direction specification . . . 2240

29.5.5 Acknowledge (ACK) . . . 2241
29.5.6 Data . . . 2242
29.5.7 Stop condition . . . 2242
29.5.8 Wait state . . . 2243
29.5.9 Arbitration . . . 2245
29.6 Registers . . . 2246
29.7 Operation . . . 2270
29.7.1 Single transfer mode . . . 2270
29.7.2 Continuous transfer mode . . . 2275
29.7.3 Arbitration . . . 2280
29.7.4 Entering and exiting wait state . . . 2281
29.7.5 Extension code . . . 2286
29.8 Interrupt Request Signals . . . 2287
29.8.1 Single transfer mode . . . 2288
29.8.2 Continuous transfer mode . . . 2291
29.9 Interrupt Outputs and Statuses . . . 2297
29.9.1 Single transfer mode (master device operation) . . . 2298
29.9.2 Single transfer mode (slave device operation: during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1)) . . . 2301
29.9.3 Single transfer mode (slave device operation: during extension code reception (IICBnSTR0.IICBnSSEX bit = 1)) . . . 2305
29.9.4 Single transfer mode (non-participation in communications) . . . 2309
29.9.5 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): operation as slave after arbitration loss) 2310
29.9.6 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss) . . . 2312
29.9.7 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss (during extension code transfer)) . . . 2318
29.9.8 Continuous transfer mode (master device operation (reception)) . . . 2319
29.9.9 Continuous transfer mode (master device operation (transmission)) . . . 2322
29.9.10 Continuous transfer mode (slave device operation (reception): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1)) . . . 2325
29.9.11 Continuous transfer mode (slave device operation (reception): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1)) . . . 2329
29.9.12 Continuous transfer mode (slave device operation (transmission): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1)) . . . 2333
29.9.13 Continuous transfer mode (slave device operation (transmission): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1)) . . . 2337
29.9.14 Continuous transfer mode (non-participation in communications) . . . 2341
29.9.15 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): operation as slave after arbitration loss) . . . 2342
29.9.16 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss) . . . 2344
29.9.17 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss (during extension code transfer)) . . . 2349
29.10 Setting Sequence . . . 2351
29.10.1 Single master environment . . . 2351

29.10.2 Multi-master environment 2355

Chapter 30 FlexRay™ (FLX) 2363

30.1 V850E2/Fx4-H FLXn Features 2363

30.2 E-Ray Overview 2366

30.2.1 Conventions 2366
30.2.2 Definition 2366
30.2.3 References 2366
30.2.4 Terms and abbreviations 2366
30.2.5 Functional overview 2367
30.2.6 Block diagram 2369
30.2.7 Host CPU interface timing 2371
30.2.8 Reset timing 2371

30.3 Programmer's Model 2372

30.3.1 Register map 2372
30.3.2 E-Ray registers 2377
30.3.3 Special registers 2382
30.3.4 Interrupt registers 2389
30.3.5 CC control registers 2410
30.3.6 CC status registers 2434
30.3.7 Message buffer control registers 2451
30.3.8 Message buffer status registers 2457
30.3.9 Identification Registers 2470
30.3.10 Input buffer 2472
30.3.11 Output buffer 2481

30.4 Functional Description 2492

30.4.1 Communication cycle 2492
30.4.2 Communication modes 2494
30.4.3 Clock synchronization 2495
30.4.4 Error handling 2497
30.4.5 Communication controller states 2499
30.4.6 Network management 2514
30.4.7 Filtering and masking 2514
30.4.8 Transmit process 2517
30.4.9 Receive process 2520
30.4.10 FIFO function 2522
30.4.11 Message handling 2524
30.4.12 Message RAM 2534
30.4.13 Module interrupt 2543

30.5 Appendix 2545

30.5.1 Assignment of FlexRay Configuration Parameters 2545

30.6 Cautions 2547

30.6.1 Loop back mode operates only at 10 MBit/s. 2547
30.6.2 Noise following a dynamic frame that delays idle detection may fail to stop slot counting for the remainder of the dynamic segment 2547
30.6.3 Register FLXnRCV displays wrong value. 2548
30.6.4 After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored. 2548

30.6.5 Sync frame overflow flag FLXnEIR.FLXnSFO may be set if slot counter is greater than 1024. . . . 2549
30.6.6 Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames. . . . 2549
30.6.7 Initial rate correction value of an integrating node is zero if pMicroInitialOffsetA,B = 00H. . . . 2550
30.6.8 Incorrect rate and/or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame. . . . 2551
30.6.9 Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received. . . . 2551
30.6.10 Rate correction set to zero in case of SyncCalcResult=MISSING_TERM. 2552
30.6.11 A sequence of received WUS may generate redundant FLXnSIR.FLXnWUPA/B events. . . . 2552
30.6.12 Erroneous cycle offset during startup after abort of startup or normal operation by a READY or FREEZE command. . . . 2553
30.6.13 First WUS following received valid WUP may be ignored. . . . 2553
30.6.14 READY command accepted in READY state. . . . 2554
30.6.15 Slot Status vPOC!SlotMode is reset immediately when entering HALT state (CCSV.SLM[1:0] = "00"). . . . 2554
30.6.16 Received messages not stored in Message RAM when in Loop Back Mode. . 2554

Chapter 31 Ethernet Controller (ETHA) . . . 2555
31.1 V850E2/Fx4-H ETHAn Features . . . 2555
31.2 General . . . 2558
31.2.1 Functions. . . . 2558
31.3 Configuration . . . 2559
31.3.1 System configuration. . . . 2559
31.3.2 Interrupt requests and sources . . . 2561
31.4 Registers for Controlling the Ethernet Controller . . . 2562
31.4.1 Ethernet Controller register overview . . . 2562
31.4.2 MAC control registers . . . 2566
31.4.3 Statistics counters . . . 2598
31.4.4 FIFO controller control registers . . . 2637
31.4.5 DMAC control registers in Ethernet Controller . . . 2671
31.4.6 Control registers of DMAC for transmit checksum . . . 2682
31.5 MAC/FIFO Function . . . 2692
31.5.1 Frame format. . . . 2692
31.5.2 Frame transmission. . . . 2697
31.5.3 Frame reception . . . 2702
31.5.4 MAC control function. . . . 2704
31.5.5 Serial management interface . . . 2708
31.5.6 Address filtering . . . 2713
31.5.7 Statistics counters . . . 2718
31.6 Dedicated DMAC . . . 2719
31.6.1 DMAC overview. . . . 2719
31.6.2 DMA . . . 2720
31.6.3 Descriptor mechanism . . . 2722

31.7 Receive Checksum . . . 2738
31.7.1 Processing by software . . . 2738
31.8 Transmit Checksum . . . 2740
31.8.1 Configuration of transmit checksum descriptor . . . 2741
31.9 Cautions . . . 2742
31.9.1 Cautions on FIFO . . . 2742

Chapter 32 Random Number Generator A (RNGA) . . . 2743
32.1 V850E2/Fx4-H RNGA Features . . . 2743
32.2 Functional Overview . . . 2745
32.3 Functional Description . . . 2745
32.3.1 RNGA status . . . 2745
32.3.2 RNGA start and reset . . . 2745
32.4 Registers . . . 2746
32.4.1 Registers overview . . . 2746
32.4.2 Registers details . . . 2746

Chapter 33 Key Return Function (KR) . . . 2747
33.1 V850E2/Fx4-H KR Features . . . 2747
33.2 Functional Overview . . . 2749
33.3 Functional Description . . . 2750
33.3.1 Interrupt request KRnTIKR . . . 2750
33.4 Registers . . . 2751
33.4.1 Key Return Function registers overview . . . 2751
33.4.2 Key Return Function registers details . . . 2751

Chapter 34 A/D Converter A (ADCA) . . . 2752
34.1 V850E2/Fx4-H ADCA Features . . . 2752
34.2 H/W Trigger Expansion . . . 2756
34.2.1 ADCAn H/W trigger selection . . . 2756
34.2.2 ADCAn H/W trigger edge selection . . . 2757
34.2.3 ADCA0 H/W trigger tables . . . 2758
34.2.4 ADCA1 H/W trigger selections . . . 2762
34.3 Functional Overview . . . 2764
34.4 Cautions . . . 2766
34.5 Functional Description . . . 2766
34.5.1 Basic Operation . . . 2768
34.5.2 Clock usage . . . 2769
34.5.3 Channels and channel groups . . . 2769
34.5.4 A/D conversion modes . . . 2771
34.5.5 Starting A/D conversion (start trigger modes) . . . 2774
34.5.6 Stopping A/D conversion . . . 2776
34.5.7 Stand-by mode . . . 2778
34.5.8 Pausing and resuming A/D conversion (ADCHALT mode) . . . 2778
34.5.9 Resolution, sampling and conversion times . . . 2779
34.5.10 Interrupt generation . . . 2780
34.5.11 Storage of A/D conversion result . . . 2781

34.5.12 Result check functions . . . 2784
34.5.13 Channel S&H function . . . 2786
34.5.14 Self-diagnosis functions . . . 2793
34.5.15 Discharge function . . . 2800
34.5.16 Buffer amplifier function . . . 2801
34.5.17 Stabilization control . . . 2801
34.6 Registers . . . 2802
34.6.1 ADCA registers overview . . . 2802
34.6.2 Control registers details . . . 2804
34.6.3 Conversion status registers . . . 2815
34.6.4 S/W trigger registers details . . . 2819
34.6.5 ADCA conversion result registers details . . . 2821
34.6.6 A/D conversion result upper/lower limit comparison registers details . . . 2828
34.6.7 Diagnose functions registers . . . 2832
34.6.8 Emulation register . . . 2835

Chapter 35 Voltage Comparator (VCPC) . . . 2836
35.1 V850E2/Fx4-H VCPC Features . . . 2836
35.2 Overview . . . 2839
35.2.1 Description . . . 2839
35.2.2 Stand-by mode . . . 2841
35.3 Voltage Comparator Registers . . . 2842

Chapter 36 On-chip Debug Unit (OCD) . . . 2845
36.1 V850E2/Fx4-H On-chip Debug Features . . . 2845
36.1.1 Modules behaviour during emulation break . . . 2845
36.1.2 Signal masking . . . 2847
36.2 Functional Overview . . . 2847
36.3 Emulation Break Control . . . 2850
36.4 Connection with On-Chip Debug Emulator . . . 2851
36.5 Cautions on using On-Chip Debugging . . . 2851

Chapter 37 Boundary Scan . . . 2852
37.1 Outline . . . 2852
37.2 JTAG interface . . . 2852
37.3 Entering Boundary Scan mode . . . 2852
37.4 Boundary scan features . . . 2853
37.5 Boundary Scan applicable pins . . . 2853
37.6 DID - Boundary scan ID register . . . 2854

Chapter 38 Power Supply . . . 2855
38.1 Power supply pins naming . . . 2855
38.2 Power supply schemes . . . 2856
38.2.1 Power supply of the digital circuits . . . 2856
38.2.2 V850E2/FK4-H power supply scheme . . . 2857
38.2.3 V850E2/FL4-H power supply scheme . . . 2858

38.3 Power-up and down procedures 2859
38.3.1 Power Sequencer 2861
38.3.2 Initial power-up and final power-down 2863
38.3.3 DEEPSTOP entry and wake-up 2864
38.3.4 Power-Fail 2869
38.3.5 Other power supplies 2871

Revision History 2873
Revision History Rev. 0.02 2876
Revision History Rev. 1.00 2878
Revision History Rev. 1.01 2882
Revision History Rev. 1.02 2884
Revision History Rev. 2.00 2886

Index 2887

How to use this manual

Purpose and target readers

This manual is designed to provide the user with an understanding of the hardware functions of the microcontroller. It is intended for users designing application systems incorporating the microcontroller. A basic knowledge of electric circuits, logical circuits, and microcontrollers is necessary in order to use this manual.

Special notations

Following special notations are used throughout this document:

Note Additional remark or tip

Caution Item deserving extra attention

Electrical specifications

This manual does not present any electrical specifications.
Refer to the Data Sheet for detailed definitions of all electrical properties.
For information about the Data Sheet document, refer to the section *"Related Documents"* in the chapter *"Introduction"*.

Additional documents

Following types of documents are available for the V850E2/Fx4-H microcontrollers. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document
Data sheet	Hardware overview and electrical characteristics	Refer to the section *"Related Documents"* in the chapter *"Introduction"*
User manual: Hardware	Hardware specifications (pin assignments, memory maps, functional modules specifications and operation description) Note: Refer to the application notes for details on using functional modules.	
User manual: 32-bit Microprocessor Core Architecture	Description of CPU, its instruction set and processor protection functions	
Application note	Information on using peripheral functions and application examples, sample programs and information on writing programs in assembly language and C	Available from Renesas Electronics Web site
Renesas technical update	Product specifications, updates on documents, etc.	

Content of this manual

In the following brief hints are given where to find certain information about the V850E2/Fx4-H microcontrollers.

Product overview

Refer to the chapter “Introduction” for an overview of the features of all target microcontrollers and their block diagrams.
Order codes for all devices and a list of related documents is given here as well.

CPU core functions

The functions of the CPU core (e.g. instruction set, processor protection functions, etc.) are not subject to this manual. Refer to the separate CPU core manual, shown in the section “Related Documents” in the chapter “Introduction”.

CPU Subsystem functions

The functions of the CPU Subsystem (including address map, operation modes, etc.) are described in the chapter “CPU System Function”.
The section “Write protected Registers” in this chapter describes how to deal with registers, that feature special write protection facilities.
If the microcontroller has separate bus systems beside the CPU Subsystem to connect certain functional modules, refer to the chapter “Bus Architecture”.

Port functions

The chapter “Port Functions” describes all input/output port related functions, such as port sharing, I/O buffer control, port filters.
The features and electrical properties of the I/O buffers are not subject to this manual, but are described in the Data Sheet.

Interrupt functions

Refer to the chapter “Interrupt Controller”.
Note that the function of each interrupt source is not described here, but in the related chapter of the module, that generates the interrupt.

DMA/DTS functions

Refer to the chapter “DMA/DTS Controller” or “DMA Controller”, if the target microcontroller does not feature DTS functions.
Note that the function of each DMA/DTS trigger source is not described here, but in the related chapter of the module, that generates the trigger signal.

Flash memory

For microcontrollers with on-chip flash memory refer to the chapter “Flash Memory” for information about the flash memories structure and features, programming facilities, etc.

Stand-by functions

How to set the microcontroller in stand-by modes and wake it up again is described in the chapter “Stand-by Controller (STBC)”.

Code protection and security

Facilities to protect program code in on-chip flash memory (if available) from illegal read-out via external flash programming equipment or debuggers is described in the chapter “Code Protection and Security”.

Clock supply The chapter “Clock Controller” describes the generation and operation of all clocks, provide to the entire microcontroller.

Resets The sources that can generate reset signals to all or dedicated internal modules and how to control them is described in the chapter “Reset Controller”.

Functional modules The description of most functional modules, like timers, serial interfaces, etc. is provided in separate chapters. These chapters have a certain structure of information presentation. Refer to the section *“Functional modules descriptions”*.

Debugging The main features on the On-Chip Debug Unit of the microcontroller is described in the chapter “On-chip Debug Unit (OCD)”.
Note that the description of the external debugger tool is not subject to this manual.

Power supply The chapter “Power Supply” provides information which modules of the microcontrollers are supplied by which external power supply pins.
Note that the specification of the external power supply is not subject to this manual. Refer to the Data Sheet for detailed definitions of the power supply.

Boundary scan If the target microcontroller supports boundary scan testing, refer to the chapter “Boundary Scan” for information about available Boundary Scan features.

Notation of numbers and symbols

Symbols Symbols and notation are used as follows:

- Weight in data notation: Left is high order column, right is low order column
- Active low notation: $\overline{xxx}$ (pin or signal name is over-scored) or /xxx (slash before signal name)
- Memory map address: High order at high stage and low order at low stage

Numeric notation

- Binary: xxxx or xxx_B
- Decimal: xxxx
- Hexadecimal: $xxxx_H$ or 0x xxxx

Numeric prefixes represent different factors, depending on the measure:

Prefix	Powers of 2	Powers of 10
k (kilo)	–	$10^3 = 1000$
K (Kilo)	$2^{10} = 1024$	–
M (Mega)	$2^{20} = 1024^2 = 1,048,576$	$10^6 = 1000^2 = 1,000,000$
G (Giga)	$2^{30} = 1024^3 = 1,073,741,824$	$10^9 = 1000^3 = 1,000,000,000$
m (milli)	–	$10^{-3} = 0.001$
µ (micro)	–	$10^{-6} = 0.001^2 = 0.000,001$
p (piko)	–	$10^{-9} = 0.001^3 = 0.000,000,001$
	For example used for • address and memory spaces in bytes: KB, MB, GB	For example used for • frequencies: kHz, MHz, GHz • times: ms, µs • resistance: kΩ, MΩ • capacitance: µF, pF

Register contents X, x = don't care

Diagrams

Block diagrams do not necessarily show the exact wiring in hardware but the functional structure.

Timing diagrams are for functional explanation purposes only, without any relevance to the real hardware implementation.

Trademarks

All trademarks are the property of their respective owners.

Functional modules descriptions

Most of the chapters provide a technical description of a certain functional module.

These chapters are split into two parts:

- The first section describes all properties of the functional module specific to the target product of the user manual, such as instances, register base addresses, input/output signal names, etc.
- The subsequent sections describe the features of the functional module as a generic description. The generic description is common to all user manuals of products, that feature this module.

Functional modules abbreviation convention

Each functional module has a unique abbreviation, for instance

TAUA for the Timer Array Unit A

This shortcut is used in names for various purposes:

- The module registers and their bits names are preceded by this shortcut, for instance

  TAUAnTS for the TAUAn channel start trigger register

  The index “n” denotes the instance number of the module, refer to the next section and the key words *“Instances”* and *“Instances index n”*.
- The base address of the module registers include the by this shortcut, for instance

  <TAUAn_base> for the base address of the TAUAn registers
- The input/output signals of the module are preceded by this shortcut, for instance

  TAUAnTTIN0 for the TAUAn channel 0 input signal
- The names of the module interrupts includes the module shortcut, for instance

  INTTAUAnI0 for the TAUAn channel 0 interrupt

Product specific features

The product specific section is always structured by a set of identical key words.

For the naming of signals product specific section serves also as some kind of interface between the generic module description and all other parts of the document.
This means that the names of signals, used in the generic module description, may be translated to other names, that are used in the other document chapters.
This name translation is given in form of tables, as the following as an example:

Module signals	Function	Connected to
Module shortcut:		
Name used in generic module description	Brief functional description	Name used in remaining document

The following lists the key words for product specific definitions. As examples, definitions of different modules are used.

Cautions

1. The following product specific definitions are only used as examples and do not define any properties of the target product of this document.
2. Consequently the functional modules, used for examples purposes, may not be available with the target product of this document.

Instances The devices of the target product may contain different numbers of the functional module, so called instances. The “Instances” paragraph specifies the number of instances for all devices of the target product.

Table 0-1 Example: Instances of TAUAn

Timer Array Unit A	Device_1	Device_2	Device_3
Instance	2	4	2
Name	TAUA0 to TAUA1	TAUA0 to TAUA3	TAUA0 to TAUA1

Instances index n

Throughout the following generic module description, an instance of a module is identified by the index "n", for instance

TAUAnTS for the TAUAn channel start trigger register

“n” counts from 0 to the number of instances minus one.

Other indices

In case other indices, except “n” for instances, are used throughout the generic module description, they are specified here.

Channel index m

Timer Array Unit A has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7 , 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All module register addresses in the generic description are given as address offsets to a base address, that is individual to a certain module instance n. For each module instance n the individual base address is given here.

Table 0-2 Example: Register base addresses <TAUAn_base>

TAUAn instance	<TAUAn_base> address
TAUA0	FF80 8000_H
TAUA1	FF80 9000_H
TAUA2	FF80 $A000_H$
TAUA3	FF80 $B000_H$
TAUA4	FF80 $C000_H$

Clock supply The clock signals of each instance n of the module and their connection to other functional modules of the device are given here.
A figure shows the modules clock supply options.

Table 0-3 Example: TAUAn clock supply

TAUAn instance	TAUAn clock	Connected to
TAUA0	PCLK	Clock Generator CKSCLK_006
TAUA1	PCLK	Clock Generator CKSCLK_104
TAUA2	PCLK	Clock Generator CKSCLK_111
TAUA3	PCLK	Clock Generator CKSCLK_106
TAUA4	PCLK	Clock Generator CKSCLK_105

Interrupts and DMA/DTS The interrupt signals of each instance n of the module and their connections to other functional modules of the device are given here.

Table 0-4 Example: TAUAn interrupt and DMA/DTS requests

TAUAn signals	Function	Connected to
TAUA0:		
INTTAUA0I0 to INTTAUA0I7	Channel 0 to 7 interrupt	Interrupt Controller INTTAUA0I0 to INTTAUA0I7
INTTAUA0I8	Channel 8 interrupt	Interrupt Controller INTTAUA0I8 DMA Controller trigger 15 DTS Controller trigger 9
INTTAUA0I9 to 15	Channels 9 to 14 interrupt	not connected
TAUA1:		
...	...	...

Internal signals Signals of some modules are connected to other device's modules. Such connections are defined here.

Table 0-5 Example: VIn internal signals

VIn signal	Function	Connected to
VI0EN	VIO0 enable	VI0 I/F control
ES_VSYNC	VSYNC signal output	MVO0 EVSYNC
ES_HSYNC	HSYNC signal output	MVO0 EHSYNC

H/W reset The signals, that reset each instance n of the module, are listed here.

Table 0-6 Example: TAUAn reset signals

TAUAn	Reset signal
TAUA0	• Reset Controller SYSRES • Reset upon Isolated-Area-0 wake-up from DEEPSTOP mode
TAUA1 to TAUA4	• Reset Controller SYSRES • Reset upon Isolated-Area-1 wake-up from DEEPSTOP mode

I/O signals The input/output signals of each instance n of the module and their connections to ports and other functional modules of the device are given here.

Table 0-7 Example: TAUAn I/O signals

TAUA signal	Function	Connected to
TAUA0:		
TAUA0TTIN0	Channel 0 input	Port TAUA0I0[a] or FCN0 TSOUT or port URTE0RX[b]
TAUA0TTIN1	Channel 1 input	Port TAUA0I1[a] or FCN1 TSOUT or port URTE1RX[b]
TAUA0TTIN2 to TAUA0TTIN15	Channel 2 to 15 input	Port TAUA0I2[a] to TAUA0I115[a]
TAUA0TTOUT0 to TAUA0TTOUT15	Channel 0 to 15 output	Port TAUA0O0 to TAUA0O15
TAUA1:		
...	...	...

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

b) Refer to *15.2 "TAUA Input Selection" on page 662* for details.

Special definitions If the functional module needs any particular definitions for its operation, which are product dependent, these are defined here.

Further information

For further information see *http://www.renesas.com*.

Chapter 1 Introduction

1.1 V850E2/Fx4-H Product Line Overview

(1) V850E2/FK4-H product line overview

Table 1-1 V850E2/FK4-H product series overview (1/2)

Part number:			μPD70F3561
Internal memory	Instruction flash		2 MB
	Data flash		64 KB
	CPU RAM		144 KB
	HBUS-RAM		96 KB
	Back-up RAM		16 KB
External memory interface (MEMC)			provided
CPU	CPU System		V850E2M
	FPU		provided
	CPU frequency		160 MHz max. (+ 5% with SSCG)
	System Protection Functions (SPF)	MPU	provided
		SRP	provided
		TSU	provided
		PPU	provided
	Instruction cache		8 KB/ 2 way associative (4 KB/ way)
DMA/DTS	DMA		16 channels
	DTS		provided
Operating clock	Main Oscillator (MainOsc)		4 MHz to 20 MHz
	Low Speed Internal Oscillator (LS IntOsc)		240 KHz typ.
	High Speed Internal Oscillator (HS IntOsc)		8 MHz typ.
	Sub Oscillator (SubOsc)		32768 Hz typ.
	PLL0 (SSCG0)		160 MHz max. (+ 5% with SSCG)
	PLL1		120 MHz max.
	PLL2 (SSCG2)		120 MHz max. (+ 5% with SSCG)
I/O ports			135
A/D converter (ADCA)			24 +16 channels, 12 bit, 6 S & H

Table 1-1 V850E2/FK4-H product series overview (2/2)

Part number:		μPD70F3561
Timers	Timer Array Unit A (TAUA), 16 bit	1 units x 16 channels
	Timer Array Unit B (TAUB), 16 bit	2 units x 16 channels
	Timer Array Unit C (TAUC), 16 bit	4 units x 8 channels
	Timer Array Unit J (TAUJ), 32 bit	2 units x 4 channels
	PWM Diagnosis (PMCA)	1 unit (56 channels)
	PWM Delay (DLYA)	1 unit
	Realtime Clock (RTCA) and calibration	1 unit
	Window Watchdog (WDTA)	2 channels
	Operating System Timer (OSTM)	1 channel
	Motor Control (TAPA)	1 channel
	Encoder Timer (ENCA)	1 channel
Serial interfaces	CAN I/F (FCN)	3 channels (64 messages buffer) 1 channel (128 messages buffer)
	Diagnostic CAN I/F (DCN)	1 channel (128 messages buffer)
	UART I/F (URTE) with LIN Master Controller (LM)	12 channels
	Synchronous I/F (CSIG)	2 channels
	Synchronous I/F (CSIH)	3 channels
	I^2C I/F (IICB)	1 channel
Other interfaces	Flexray Controller (FLX	1 unit (2 channels)
	Ethernet Controller (ETHA)	1 unit (via MMI interface)
External interrupts	Maskable	16
	Non-maskable (NMI)	1
Other functions	Power-On-Clear	provided
	Voltage Comparators	2 channels
	Clock Monitors (CLMA)	provided for MainOsc, HS IntOsc, PLL0 supervision
	Random Number Generator (RNGA)	1 channel
	Data CRC (DCRA)	1 channel
	Key Return (KR)	8 channels
	Wake-up signal output	provided
	Auxiliary frequency output (FOUT)	provided
	On-Chip debug (OCD)	provided
	Boundary Scan	provided
Voltage supply	System supply	3.0 V to 5.5 V[a]
	Port supply	3.0 V to 5.5 V[a]
Operating Temperature		-40° C … +110° C[a]
Package		176-pin HLQFP

a) Refer to the Electrical Target Specification.

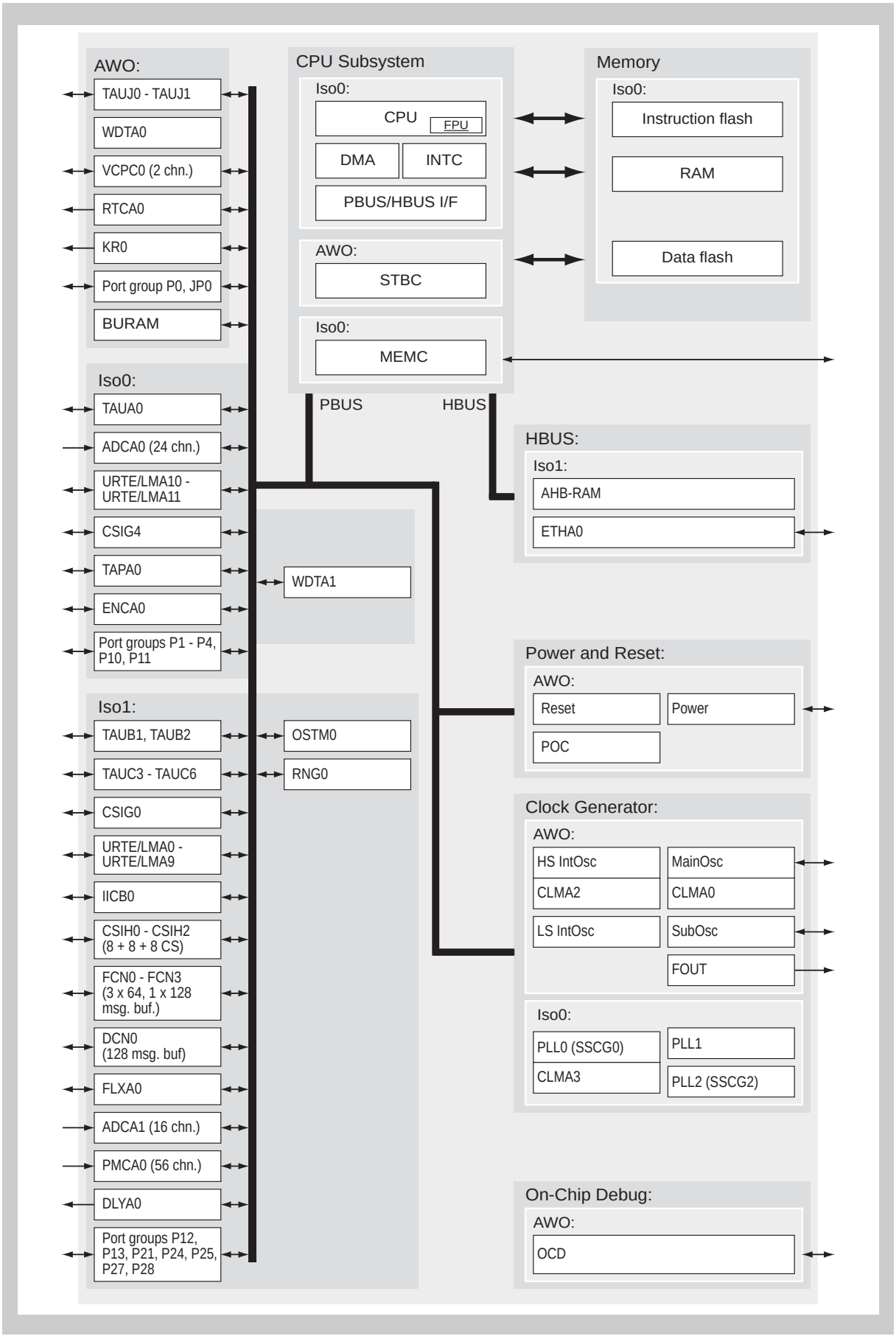


Figure 1-1 V850E2/FK4 block diagram

(2) V850E2/FL4-H product line overview

Table 1-2 V850E2/FL4-H product series overview (1/2)

Part number:			μPD70F3564
Internal memory	Instruction flash		2 MB
	Data flash		64 KB
	CPU RAM		144 KB
	HBUS-RAM		96 KB
	Back-up RAM		16 KB
External memory interface (MEMC)			provided
CPU	CPU System		V850E2M
	FPU		provided
	CPU frequency		160 MHz max. (+ 5% with SSCG)
	System Protection Functions (SPF)	MPU	provided
		SRP	provided
		TSU	provided
		PPU	provided
	Instruction cache		8 KB/ 2 way associative (4 KB/ way)
DMA/DTS	DMA		16 channels
	DTS		provided
Operating clock	Main Oscillator (MainOsc)		4 MHz to 20 MHz
	Low Speed Internal Oscillator (LS IntOsc)		240 KHz typ.
	High Speed Internal Oscillator (HS IntOsc)		8 MHz typ.
	Sub Oscillator (SubOsc)		32768 Hz typ.
	PLL0 (SSCG0)		160 MHz max. (+ 5% with SSCG)
	PLL1		120 MHz max.
	PLL2 (SSCG2)		120 MHz max. (+ 5% with SSCG)
I/O ports			161
A/D converter (ADCA)			2 x 24 channels, 12 bit, 6 S & H
Timers	Timer Array Unit A (TAUA), 16 bit		1 units x 16 channels
	Timer Array Unit B (TAUB), 16 bit		2 units x 16 channels
	Timer Array Unit C (TAUC), 16 bit		5 units x 8 channels
	Timer Array Unit J (TAUJ), 32 bit		2 units x 4 channels
	PWM Diagnosis (PMCA)		1 unit (64 channels)
	PWM Delay (DLYA)		1 unit
	Realtime Clock (RTCA) and calibration		1 unit
	Window Watchdog (WDTA)		2 channels
	Operating System Timer (OSTM)		1 channel
	Motor Control (TAPA)		1 channel
	Encoder Timer (ENCA)		1 channel

Table 1-2 V850E2/FL4-H product series overview (2/2)

Part number:		μPD70F3564
Serial interfaces	CAN I/F (FCN)	3 channels (64 messages buffer) 2 channels (128 messages buffer)
	Diagnostic CAN I/F (DCN)	1 channel (128 messages buffer)
	UART I/F (URTE) with LIN Master Controller (LM)	12 channels
	Synchronous I/F (CSIG)	3 channels
	Synchronous I/F (CSIH)	3 channels
	I^2C I/F (IICB)	1 channel
Other interfaces	Flexray Controller (FLX)	1 unit (2 channels)
	Ethernet Controller (ETHA)	1 unit (via MMI interface)
External interrupts	Maskable	16
	Non-maskable (NMI)	1
Other functions	Power-On-Clear	provided
	Voltage Comparators (VCPC)	2 channels
	Clock Monitors (CLMA)	provided for MainOsc, HS IntOsc, PLL0 supervision
	Random Number Generator (RNGA)	1 channel
	Data CRC (DCRA)	1 channel
	Key Return (KR)	8 channels
	Wake-up signal output	provided
	Auxiliary frequency output (FOUT)	provided
	On-Chip debug (OCD)	provided
	Boundary Scan	provided
Voltage supply	System supply	3.0 V to 5.5 V[a]
	Port supply	3.0 V to 5.5 V[a]
Operating Temperature		-40° C … +110° C[a]
Package		272-pin PBGA

a) Refer to the Electrical Target Specification.

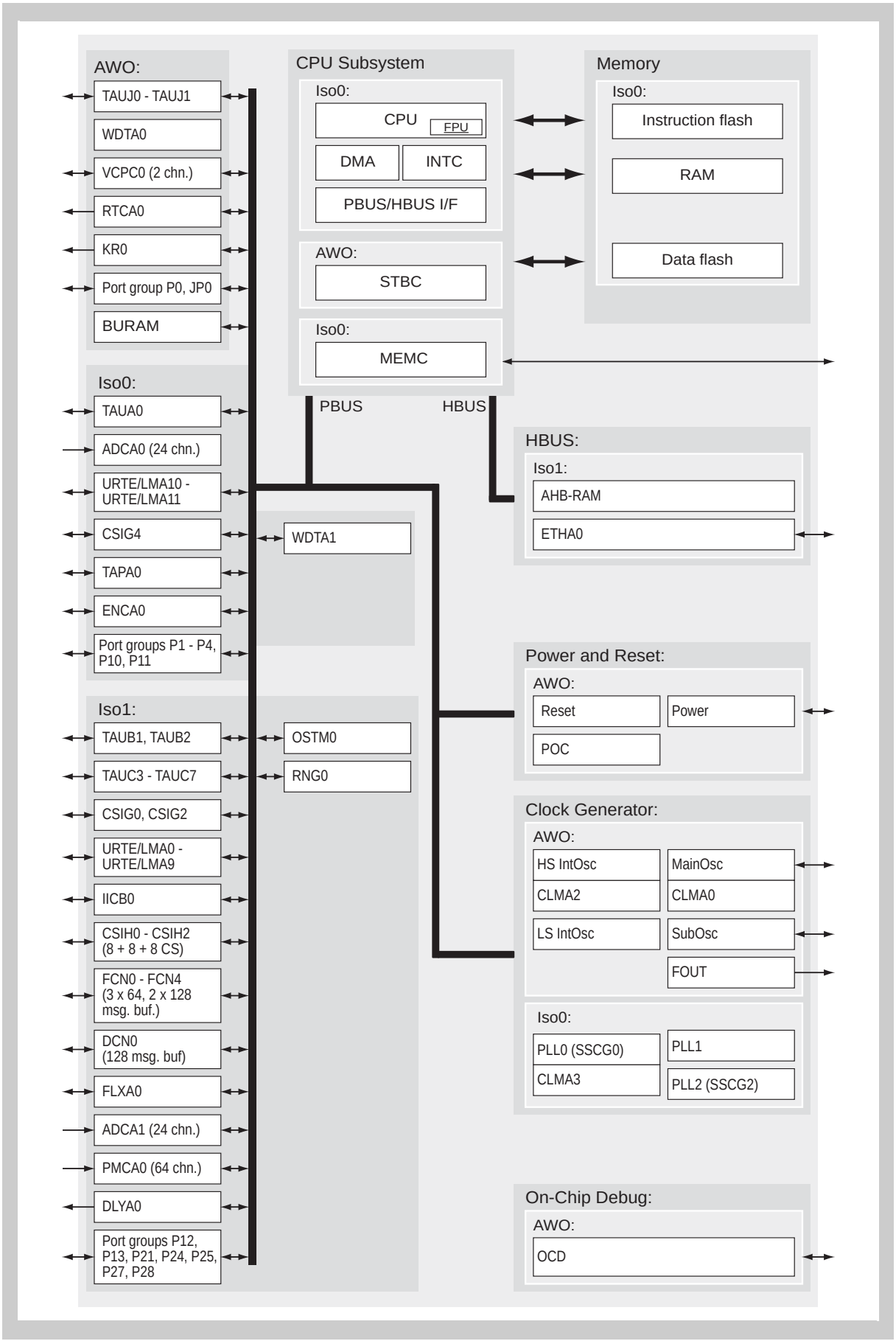


Figure 1-2 V850E2/FL4 block diagram

1.2 Related Documents

Table 1-3 Related documents

Document number[a]	Title
R01US0001EJxxxx	User's Manual: V850E2M 32-bit Microcontroller Core Architecture
<tbd>	Electrical Target Specification V850E2/FK4-H
EASE-DS-0032-x.x	Electrical Target Specification V850E2/FL4-H
R20UT0008EJxxxx	User's Manual: PG-FP5 Flash Memory Programmer
U17638EJxVxUM00	User's Manual: QB-V850MINI, QB-V850MINIL On-Chip Debug Emulator
U20281EExVxUM00	User's Manual: Flash Self-programming Library FSL - T05
U20279EExVxUM00	User's Manual: Data Flash Access Library FDL - T05

a) "x" denotes the current document revision numbers.

1.3 Ordering Information

Table 1-4 V850E2/Fx4-H ordering information

Series name	Device name	Renesas order code	Remarks
FK4-H-2M	µPD70F3561	UPD70F3561GM(Ax)-GBK-AX	
FL4-H-2M	µPD70F3564	UPD70F3564F1(Ax)-JN1-A	

1.4 Product Name Register

Information about the product code, version, etc. can be read via the product name register PRDNAME.

(1) PRDNAME - Product name register

This register holds the product information.

Access This register can be read in 32-bit units.

Address FF47 0028$_H$

Initial Value 00xx xxxx xxxx xxx 0000 xxxx xxxx xxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	PRDNUM[13:0]													
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	PRDVMAJ[3:0]				PRDVMIN [7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 1-5 PRDNAME register contents

Bit position	Bit name	Function
29 to 16	PRD NUM[13:0]	Product number 0DE9$_H$: µPD70F3561 0DEC$_H$: µPD70F3564
11 to 8	PRD VMAJ[3:0]	Major product version number
7 to 0	PRD VMIN[7:0]	Minor product version number

Note The product version number is composed as follows:

PRDVMAJ.PRDVMIN

Chapter 2 Port Functions

This chapter contains a generic description of the Port control functions.

The first section describes all properties specific to the V850E2/Fx4-H, such as port groups, register base addresses, etc.

The second section describes the features of the port control functions that apply to all ports.

The third section summarizes the individual functions of all pins of V850E2/Fx4-H microcontrollers.

Finally the function of analog and digital filters, which are implemented at some pins, are described.

2.1 V850E2/Fx4-H Port Features

Port groups

The V850E2/Fx4-H microcontrollers have following number of port groups:

Table 2-1 Port groups of V850E2/Fx4-H

Port groups	V850E2/FK4-H	V850E2/FL4-H
Number	12	15
Names	P0 to P4, P10 to P12, P21, P25, P27, JP0	P0 to P4, P10 to P13, P21, P24, P25, P27, JP0

Port groups index n

Throughout this chapter, the individual port groups are identified by the index “n”, for example, PMCn for the port mode control register of Pn.

Register addresses

All port and JTAG port control register addresses are given as address offsets from the individual base addresses <PORTn_base> and <JPORT0_base>. The base addresses <PORTn_base> and <JPORT0_base> are specified in the following table:

Table 2-2 Port base addresses <PORTn_base> and <JPORT0_base>

<PORTn_base> address	<JPORT0_base> address
FF40 0000_H	FF44 0000_H

2.2 Overview

The microcontroller has various pins for input/output functions, known as ports. The ports are organized in port groups.

The microcontroller also has several control registers to allocate other than general purpose input/output functions to the pins.

For a description of the terms pin, port, or port group, see the following section *"Terms"*.

Features summary

- Configuration possible for individual pins.
- The following features can be selected for most of the pins:
  - One out of four input buffer characteristics
  - Output current limit
  - Open drain emulation
  - Pull-up or pull-down resistor connection
- The following registers are offered for most of the ports:
  - Direct register for reading the pin values
  - Port register
  - Port set/reset register
  - Register for output inversion

2.2.1 Terms

In this chapter, the following terms are used:

- **Pin**

  Denotes the physical pin. Every pin is denoted by a unique pin number. The pin numbers depend on the package and are given in the Electrical Target Specification.

  Most of the pins can be used in several modes. Thus the pin name depends on the selected mode.

- **Port group**

  Denotes a group of ports. The ports of a port group have a common set of port mode control registers.

- **Port mode / Port**

  A pin in port mode works as a general purpose input/output pin. It is then called “port”.

  The corresponding name is Pn_m. For example, P0_7 denotes port 7 of port group 0. It is referenced as “port P0_7”.

- **Alternative mode**

  In alternative mode, a pin can be used for various non-general purpose input/output functions, for example as the input/output pin of on-chip peripherals.

  The corresponding pin name depends on the selected function. For example, pin INTP0 denotes the pin for one of the external interrupt inputs.

  Note that two different names can refer to the same physical pin, for example P0_0 and INTP0. The different names indicate the function in which the pin is being operated.

JTAG ports The JTAG port group JP0 is used for connecting the debugger for on-chip debugging purposes. Therefore it present a special port group, as the ports of JP0 are not available for application purposes during a debug session. During normal operation, i.e. without debugger, the JP0 ports can be used in the same way as all others.

The JTAG port group JP0 control registers and their control bits have the same names as the other port groups, registers and bits, but are identified by a “J” prefix.

Note Throughout this chapter the description of all ports and their registers apply also to the JTAG ports, unless otherwise noted.

2.2.2 Pin function configuration

The pins can operate in three different general modes:

- Port mode (PMCn.PMCn_m = 0)
  In port mode the pin operates as a general purpose I/O port. PMn.PMn_m selects input or output.
- S/W I/O control alternative mode (PMCn.PMCn_m = 1, PIPCn.PIPCn_m = 0)
  In S/W I/O control alternative mode the pin is operated by an alternative function. The selection between input or output is done by S/W via the PMn.PMn_m control bits.
- Direct I/O control alternative mode (PMCn.PMCn_m = 1, PIPCn.PIPCn_m = 1)
  In direct I/O control alternative mode the pin is operated by an alternative function. In contrast to the S/W I/O control alternative mode the input/output control is also handled by the alternative function, thus the S/W doesn't have to care about.

An overview of the register settings is given in the tables below.

Table 2-3 Pin function configuration (overview)

Mode	Control bits			I/O
	PMCn_m	PMn_m	PIPCn_m	
Port	0	0	X	O
		1[a]		I
S/W I/O control alternative	1	0	0	O
		1	0	I
Direct I/O control alternative		X	1	controlled by alternative function

a) The input buffer must be enabled (PIBCn.PIBCn_m = 1)

If a pin is operated in an alternative mode (PMCn.PMCn_m = 1), one out of up to four different alternative functions can be selected by the PFCn and PFCEn registers.

Selection of one of the alternative input and output functions:

- S/W I/O control alternative functions (PIPCn.PIPCn_m = 0):
  - outputs (PMn_m = 0): ALT_OUT1 to ALT_OUT4
  - inputs (PMn_m = 1): ALT_IN1 to ALT_IN4
- Direct I/O control alternative functions (PIPCn.PIPCn_m = 1):
  - input/out of ALT_OUT1 to ALT_OUT4 and ALT_IN1 to ALT_IN4 is directly controlled by the alternative function

Table 2-4 Alternative mode selection overview (PMCn.PMCn_m = 1)

Function	Registers				I/O
	PIPC[a]	PM[a]	PFCE	PFC	
Alternative output mode 1 (ALT_OUT1)	0	0	0	0	O
Alternative input mode 1 (ALT_IN1)		1			I
Alternative output mode 2 (ALT_OUT2)		0	0	1	O
Alternative input mode 2 (ALT_IN2)		1			I
Alternative output mode 3 (ALT_OUT3)		0	1	0	O
Alternative input mode 3 (ALT_IN3)		1			I
Alternative output mode 4 (ALT_OUT4)		0	1	1	O
Alternative input mode 4 (ALT_IN4)		1			I

a) If PIPCn.PIPCn_m = 1, the I/O direction is directly controlled by the alternative function and PM is ignored.

Caution In case a certain alternative input function is available via multiple ports, only one port must be configured to use this alternative input function. All other ports must be configured to use other signals.

PMn/PMCn register write

The port mode register PMn and port mode control register PMCn can be manipulated in two different ways:

- Direct PMn/PMCn write
  New value can be written directly to the PMn/PMCn register.
- Indirect PMn/PMCn bit set/reset
  An indirect way to set or reset a PMn/PMCn bit is possible by using following registers:
  - Port mode set reset register PMSRn
    If the bit PMSRn.PMSRn(m+16) = 1, the value of bit PMSRn.PMSRn_m determines the value of PMn.PMn_m.
    Thus PMn_m can be set/reset without a direct write to PMn.
  - Port mode control set reset register PMCSRn
    If the bit PMCSRn.PMCSRn(m+16) = 1, the value of bit PMCSRn.PMCSRn_m determines the value of PMCn.PMn_m.
    Thus PMn_m can be set/reset without a direct write to PMCn.

  The indirect PMn/PMCn set/reset operation provides access to single bits of the PMn/PMCn register, while leaving all other register bits untouched.

Both ways to manipulate a PMn/PMCn bit can be used concurrently.

Note It is recommended to use the indirect PMn/PMCn bit set/reset method for changing a single bit or concurrently several bits of the PMn/PMCn register, since all other bits are not modified and can be independently treated by other S/W modules, for instance in interrupt service routines.

2.2.3 Pin data input/output

In the following the registers are described, used for data input and output.

Depending on the pin mode, the source of the data to be output and the data read via the PPRn register differs.

Output data

In *port mode* (PMCn.PMCn_m = 0) the data of Pn.Pn_m is output to pin Pn_m.

In *alternative mode* (PMCn.PMCn_m = 1) the pin Pn_m output is determined by the alternative function.

Input data

A read operation of the PPRn register returns either the value of the Pn_m pin, the associated bit of the port register Pn.Pn_m or the data output by an alternative function.

The source of the data read via PPRn depends on the pin mode and the setting of several control bits.

The table below summarizes the different PPRn read modes.

Note PBDCn_m is not included in the table, as it can be set to 1 for reading the Pn_m pin level in all modes.

Table 2-5 PPRn_m read values

PMC n_m	PM n_m	PIBC n_m	PIPC n_m	PODC n_m	Mode	PPRn_m read value
0	1	0	X	X	Port input, input buffer disabled	Pn.Pn_m register
		1		X	Port input, input buffer enabled	Pn_m pin
	0	X		0	Port push-pull output	Pn.Pn_m register[a]
				1	Port open-drain output	
1	1	X	0	X	S/W I/O control alternative input	Pn_m pin
	0			0	S/W I/O control alternative push-pull output	Alternative function output[a]
				1	S/W I/O control alternative open-drain output	
	X		1	0	Direct I/O control alternative input/ push-pull output	If alternative functions sets port in • input: PPRn_m = Pn_m pin • output: PPRn_m = alternative function output[a]
				1	Direct I/O control alternative input/ open-drain output	

a) If PBDCn_m = 1, Pn_m pin level is read via PPRn_m.

The control registers in the table above have following effects:

- PBDCn.PBDCn_m (see table footnote)
  This bit forces to read the Pn_m pin level via PPRn_m, thus enabling a bi-directional mode, where the level of pin Pn_m can also be read back if the port is operated in an output mode.
- PMCn.PMCn_m
  This bit selects port mode (PMCn_m = 0) or alternative mode (PMCn_m = 1).

- PMn.PMn_m
  This bit selects input (PMn_m = 1) or output (PMn_m = 0) in port mode (PMCn_m = 0) and S/W I/O control alternative function mode (PMCn_m = 1, PIPCn_m = 0).
- PIBCn.PIBCn_m
  This bit disables (PIBCn_m = 0) or enables (PIBCn_m = 1) the input buffer in input port mode (PMCn_m = 0 and PMn_m = 1). If the input buffer is disabled, PPRn_m reads the Pn.Pn_m bit, otherwise the Pn_m pin level is returned.
- PIPCn.PIPCn_m
  This bit selects between the S/W and direct I/O control alternative mode.
- PODCn.PODCn_m
  This bit selects between push-pull (PODCn_m = 0) and open-drain (PODCn_m = 1) output.

Pn register write

The data to be output via port Pn_m in port mode (PMCn.PMCn_m = 0) is held in the port register Pn. The Pn data can be manipulated in two different ways:

- Direct Pn write
  New data can be written directly to the Pn register.
- Indirect Pn bit set/reset/not
  An indirect way to set (Pn_m = 1), reset (Pn_m = 0), or invert ($\overline{Pn\_m} \rightarrow Pn\_m$) a Pn bit is possible using two registers:
  - Port set reset register PSRn
    If the bit PSRn.PSRn(m+16) = 1, the value of bit PSRn.PSRn_m determines the value of Pn.Pn_m.
    Thus Pn_m can be set/reset without a direct write to Pn.
  - Port NOT register PNOTn
    Setting PNOTn.PNOTn_m = 1 inverts the bit Pn.Pn_m without a direct write to Pn_m.

  The indirect Pn set/reset/not operation provides access to single bits of the Pn register, while leaving all other Pn bits untouched.

Both ways to manipulate a Pn bit can be used concurrently.

Note It is recommended to use the indirect Pn bit set/reset/not method for changing a single bit or concurrently several bits of the Pn register, since all other bits are not modified and can independently be treated by other S/W modules, for instance in interrupt service routines.

Caution If a port Pn_m

- provides an alternative output ALT_OUTx and input function ALT_INx
- and is used in alternative output mode ALT_OUTx (PMCn.PMCn_m = 1, PMn.PMn_m = 0)
- and the bi-directional mode is enabled (PBDCn.PBDCn_m = 1) for reading the Pn_m level via PPRn.PPRn_m,

the Pn_m output, i.e. of ALT_OUTx, is internally fed back to the alternative input function ALT_INx.

2.2.4 Port control logic diagram

The following diagram shows the logical circuitry of the port control functions.

Note The diagram is only a logical reference and does not show the real circuitry.

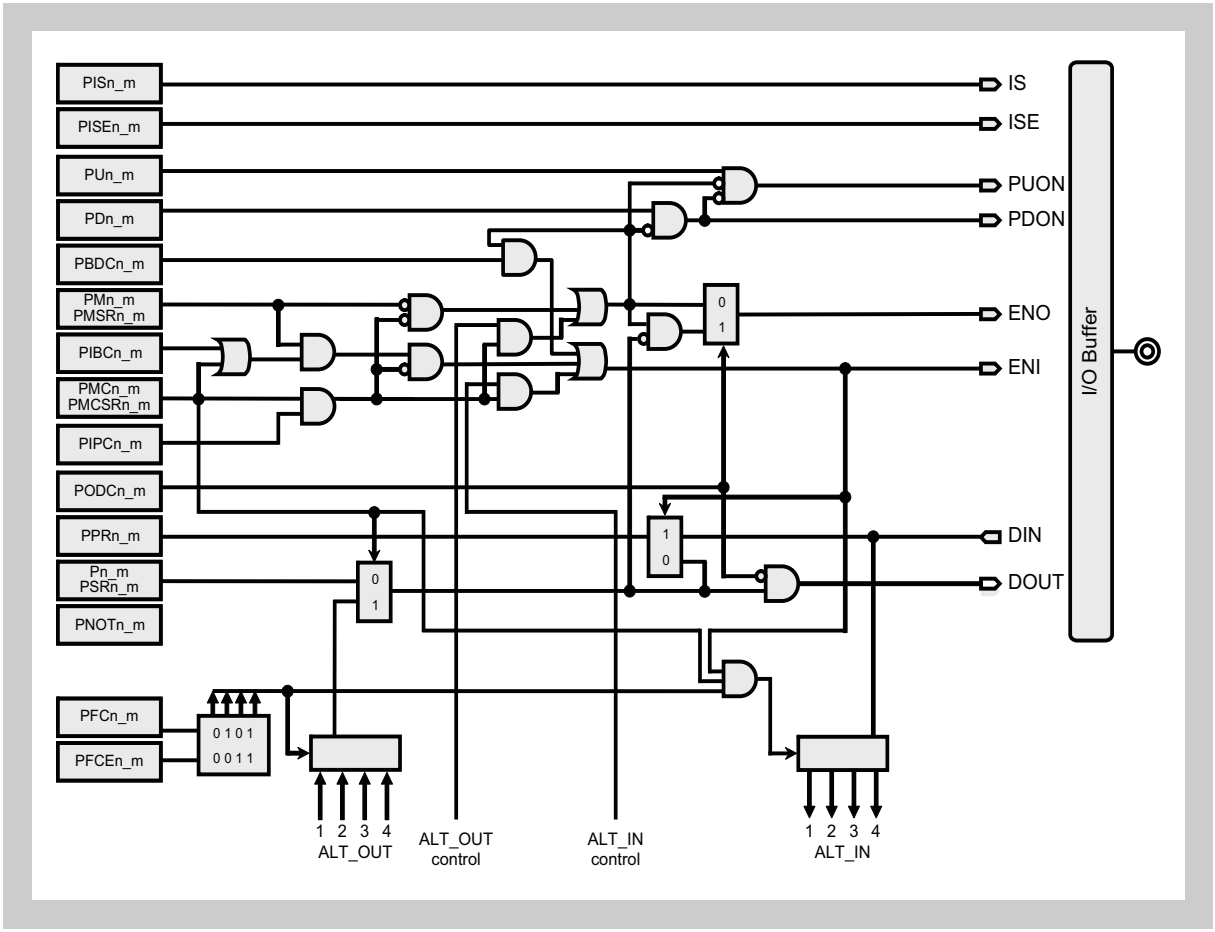


Figure 2-1 Port control logic diagram

The signals to the I/O buffer in the diagram above have the following general function:

Buffer control signal	General function
DS	port drive strength control
IS, ISE	input buffer selection
PUON/PDON	pull-up/-down register control
ENO/ENI	output/input buffer enable
DIN/DOUT	port data in/out

2.3 Port Group Configuration Registers

This section starts with an overview of all configuration registers and then presents all registers in detail. The configuration registers are grouped as follows:

- *"Pin function configuration registers"*
- *"Pin data input/output"*
- *"Configuration of electrical characteristics registers"*

2.3.1 Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Following port registers feature this special write protection:

- Port drive strength control registers PDSCn, JPDSC0
- Port open drain control registers PODCn, JPODC0

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

2.3.2 Port control registers overview

The following registers are used for the configuration of the individual pins of the port groups:

Note Some of the registers, listed in the table below, are not available for all port groups n. Refer to the section *"V850E2/Fx4-HPort Group Configuration"* below for information which registers are available for the individual port groups.

Table 2-6 Registers for port group configuration (1/2)

Register name	Shortcut	Address
Port function configuration:		
Port mode control register	PMCn	<PORTn_base> + 0400_H + n x 4
	JPMC0	<JPORT0_base> + 0040_H
Port mode control set reset register	PMCSRn	<PORTn_base> + 0900_H + n x 4
	JPMCSR0	<JPORT0_base> + 0090_H
Port IP control register	PIPCn	<PORTn_base> + 4200_H + n x 4
Port mode register	PMn	<PORTn_base> + 0300_H + n x 4
	JPM0	<JPORT0_base> + 0030_H
Port mode set reset register	PMSRn	<PORTn_base> + 0800_H + n x 4
	JPMSR0	<JPORT0_base> + 0080_H
Port input buffer control register	PIBCn	<PORTn_base> + 4000_H + n x 4
	JPIBC0	<JPORT0_base> + 0400_H
Port function control register	PFCn	<PORTn_base> + 0500_H + n x 4
	JPFC0	<JPORT0_base> + 0050_H

Table 2-6 Registers for port group configuration (2/2)

Register name	Shortcut	Address
Port function control expansion register	PFCEn	<PORTn_base> + 0600_H + n x 4
	JPFCE0	<JPORT0_base> + 0060_H
Data input/output:		
Port bi-direction control register	PBDCn	<PORTn_base> + 4100_H + n x 4
	JPBDC0	<JPORT0_base> + 0410_H
Port pin read register	PPRn	<PORTn_base> + 0200_H + n x 4
	JPPR0	<JPORT0_base> + 0020_H
Port register	Pn	<PORTn_base> + 0000_H + n x 4
	JP0	<JPORT0_base> + 0000_H
Port NOT register	PNOT0	<PORTn_base> + 0700_H + n x 4
	JPNOT0	<JPORT0_base> + 0070_H
Port set reset register	PSRn	<PORTn_base> + 0100_H + n x 4
	JPSR0	<JPORT0_base> + 0010_H
Configuration of electrical characteristics:		
Pull-up option register	PUn	<PORTn_base> + 4300_H + n x 4
	JPU0	<JPORT0_base> + 0430_H
Pull-down option register	PDn	<PORTn_base> + 4400_H + n x 4
	JPD0	<JPORT0_base> + 0440_H
Port drive strength control register	PDSCn	<PORTn_base> + 4600_H + n x 4
	JPDSC0	<JPORT0_base> + 0460_H
Port open drain control register	PODCn	<PORTn_base> + 4500_H + n x 4
	JPODC0	<JPORT0_base> + 0450_H
Port input buffer selection register	PISn	<PORTn_base> + 4700_H + n x 4
	JPIS0	<JPORT0_base> + 0470_H
Port input buffer selection expansion register	PISEn	<PORTn_base> + 4800_H + n x 4
	JPISE0	<JPORT0_base> + 0480_H

<PORTn_base> The base address <PORTn_base> of the port contorl registers is defined in the first section of this chapter under the key word “Register addresses”.

JTAG port registers The following register descriptions do not explicitely reference the JTAG port registers. However all description apply also to the respective JTAG port registers, but the base address of the JTAG port registers is different:

<JPORT0_base> The base addresses <JPORT0_base> of the JTAG port control registers is defined in the first section of this chapter under the key word “Register addresses”.

Initial register values The initial values after reset release depend on the port, and are not described in the following register descriptions, but are given in the section *“V850E2/Fx4-H Port Groups Configuration”*.

2.3.3 Port function configuration registers

(1) PMCn/JPMC0 - Port mode control register

This register specifies whether the individual pins of port group n are in port mode or in alternative mode.

Access PMCn: This register can be read/written in 16-bit units.
JPMC0: This register can be read/written in 8-bit units.

Address PMCn: <PORTn_base> + 0400_H + n x 4
JPMC0: <JPORT0_base> + 0040_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMCn_15	PMCn_14	PMCn_13	PMCn_12	PMCn_11	PMCn_10	PMCn_9	PMCn_8	PMCn_7	PMCn_6	PMCn_5	PMCn_4	PMCn_3	PMCn_2	PMCn_1	PMCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPMC0 are named JPMC0_[7:0].

Table 2-7 PMCn/JPMC0 register contents

Bit position	Bit name	Function
15 to 0	PMCn_[15:0]	Specifies the operation mode of the corresponding pin: 0: Port mode 1: Alternative mode

Cautions

1. Setting PMCn.PMCn_m = 1 to use a port in alternative mode does not hand over I/O control to the alternative function. If the alternative function requires direct I/O control, PIPCn.PIPCn_m must also be set to 1.
2. Setting PMCn.PMCn_m = 1 to use a port in alternative mode may also require to configure a port filter, if this port is used as a signal input. The input signal may be passed through a noise filter, that may need to be configured. Refer to the section *"Port Filters"* in this chapter.

(2) PMCSRn/JPMCSR0 - Port mode control set reset register

This register provides an alternative method to write data to the PMCn register.

The register's upper 16 bit PMCSRn_[31:16] specify which PMCn.PMCn_m bit will be modified by the corresponding bit of the lower 16 bit PMCSRn_[15:0].

Access These registers can be read/written in 32-bit units.
Bits 31 to 16 are always read as 0000_H.
Reading bits 15 to 0 returns the value of register PMCn/JPMC0.

Address PMCSRn: <PORTn_base> + 0900_H + n x 4
JPMCSR0: <JPORT0_base> + 0090_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMCSRn_31	PMCSRn_30	PMCSRn_29	PMCSRn_28	PMCSRn_27	PMCSRn_26	PMCSRn_25	PMCSRn_24	PMCSRn_23	PMCSRn_22	PMCSRn_21	PMCSRn_20	PMCSRn_19	PMCSRn_18	PMCSRn_17	PMCSRn_16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMCSRn_15	PMCSRn_14	PMCSRn_13	PMCSRn_12	PMCSRn_11	PMCSRn_10	PMCSRn_9	PMCSRn_8	PMCSRn_7	PMCSRn_6	PMCSRn_5	PMCSRn_4	PMCSRn_3	PMCSRn_2	PMCSRn_1	PMCSRn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPMCSR0 are named JPMCSR0_[31:0].

Table 2-8 PMCSRn/JPMCSR0 register contents

Bit position	Bit name	Function
31 to 16	PMCSR n_[31:16]	PMCSRn_m specifies whether the value of the corresponding lower bit PMCSRn_m value is written to PMCn_m: 0: PMCn_m is independent of PMCSRn_m 1: PMCn_m is PMCSRn_m Example: If PMCSRn.PMCSRn_31 = 1, the value of bit PMCSRn.PMCSRn_15 is written to bit PMCn.PMCn_15 and output.
15 to 0	PMCSR n_[15:0]	Specifies the PMCn_m value if the corresponding upper bit PMCSRn_(m+16) is 1: 0: PMCn_m = 0 1: PMCn_m = 1

(3) PIPCn - Port IP control register

This register specifies whether the I/O direction of pin Pn_m is controlled by the port mode register PMn.PMn_m or by an alternative function.

If pin Pn_m is operated in alternative mode (PMCn.PMCn_m = 1) and the alternative function requires to directly control the I/O direction of Pn_m, PIPCn.PIPCn_m must be set to 1 as well. This hands over I/O control to the alternative function and overrules the PMn.PMn_m setting.

Access PIPCn: This register can be read/written in 16-bit units.

Address PIPCn: <PORTn_base> + 4200_H + n x 4

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIPCn_15	PIPCn_14	PIPCn_13	PIPCn_12	PIPCn_11	PIPCn_10	PIPCn_9	PIPCn_8	PIPCn_7	PIPCn_6	PIPCn_5	PIPCn_4	PIPCn_3	PIPCn_2	PIPCn_1	PIPCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-9 PIPCn register contents

Bit position	Bit name	Function
15 to 0	PIPCn_[15:0]	Specifies the I/O control mode: 0: I/O mode is selected by PMn.PMn_m (S/W I/O control) 1: I/O mode is selected by peripheral function (direct I/O control)

(4) PMn/JPM0 - Port mode register

The PMn register specifies whether the individual pins of the port group n are in input mode or in output mode.

Access PMn: This register can be read/written in 16-bit units.
JPM0: This register can be read/written in 8-bit units.

Address PMn: <PORTn_base> + 0300_H + n x 4
JPM0: <JPORT0_base> + 0030_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PM n_15	PM n_14	PM n_13	PM n_12	PM n_11	PM n_10	PM n_9	PM n_8	PM n_7	PM n_6	PM n_5	PM n_4	PM n_3	PM n_2	PM n_1	PM n_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPM0 are named JPM0_[7:0].

Table 2-10 PMn/JPM0 register contents

Bit position	Bit name	Function
15 to 0	PM n_[15:0]	Specifies input/output mode of the corresponding pin: 0: Output mode (output enabled) 1: Input mode (output disabled)

Notes
1. To use a port in input port mode (PMCn.PMCn_m = 0 and PMn.PMn_m = 1), the input buffer must be enabled (PIBCn.PIBCn_m = 1).
2. By default, PMn_m specifies the I/O direction in port mode (PMCn.PMCn_m = 0) and alternative mode (PMCn.PMCn_m=1), since PIPCn.PIPCn_m = 0 after reset.

(5) PMSRn/JPMSR0 - Port mode set reset register

This register provides an alternative method to write data to the PMn register.

The register's upper 16 bit PMSRn_[31:16] specify which PMn.PMn_m bit will be modified by the corresponding bit of the lower 16 bit PMSRn_[15:0].

Access These registers can be read/written in 32-bit units.
Bits 31 to 16 are always read as 0000_H.
Reading bits 15 to 0 returns the value of register PMn/JPM0.

Address PMSRn: <PORTn_base> + 0800_H + n x 4
JPMSR0: <JPORT0_base> + 0080_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PMSRn_31	PMSRn_30	PMSRn_29	PMSRn_28	PMSRn_27	PMSRn_26	PMSRn_25	PMSRn_24	PMSRn_23	PMSRn_22	PMSRn_21	PMSRn_20	PMSRn_19	PMSRn_18	PMSRn_17	PMSRn_16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMSRn_15	PMSRn_14	PMSRn_13	PMSRn_12	PMSRn_11	PMSRn_10	PMSRn_9	PMSRn_8	PMSRn_7	PMSRn_6	PMSRn_5	PMSRn_4	PMSRn_3	PMSRn_2	PMSRn_1	PMSRn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPMSR0 are named JPMSR0_[31:0].

Table 2-11 PMSRn/JPMSR0 register contents

Bit position	Bit name	Function
31 to 16	PMSR n_[31:16]	PMSRn_m specifies whether the value of the corresponding lower bit PMSRn_m value is written to PMn_m: 0: PMn_m is independent of PMSRn_m 1: PMn_m is PMSRn_m Example: If PMSRn.PMSRn_31 = 1, the value of bit PMSRn.PMSRn_15 is written to bit PMn.PMn_15 and output.
15 to 0	PMSR n_[15:0]	Specifies the PMn_m value if the corresponding upper bit PMSRn_(m+16) is 1: 0: PMn_m = 0 1: PMn_m = 1

(6) PIBCn/JPIBC0 - Port input buffer control register

In input port mode (PMCn.PMCn_m = 0 and PMn.PMn_m = 1) this register enables/disables the port pin's input buffer.

Access PIBCn: This register can be read/written in 16-bit units.
JPIBC0: This register can be read/written in 8-bit units.

Address PIBCn: <PORTn_base> + 4000_H + n x 4
JPIBC0: <JPORT0_base> + 0400_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIBCn_15	PIBCn_14	PIBCn_13	PIBCn_12	PIBCn_11	PIBCn_10	PIBCn_9	PIBCn_8	PIBCn_7	PIBCn_6	PIBCn_5	PIBCn_4	PIBCn_3	PIBCn_2	PIBCn_1	PIBCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPIBC0 are named JPIBC0_[7:0].

Table 2-12 PIBCn/JPIBC0 register contents

Bit position	Bit name	Function
15 to 0	PIBC n_[15:0]	Enables/disables the input buffer: 0: Input buffer disabled 1: Input buffer enabled

Note When the input buffer is disabled, it does not consume current even when the pin level is Hi-Z state. Thus the pin does not need to be fixed to a high or low level externally.

(7) PFCn/JPFC0 - Port function control register

This register, together with register PFCEn, specifies an alternative function of the pins.

Some alternative functions require direct I/O control of pin Pn_m. For such alternative functions PIPCn.PIPCn_m must be set to 1 as well.
For other alternative functions, input/output must be specified by PMn.PMn_m.

Access PFCn: This register can be read/written in 16-bit units.
JPFC0: This register can be read/written in 8-bit units.

Address PFCn: <PORTn_base> + 0500_H + n x 4
JPFC0: <JPORT0_base> + 0050_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFCn_15	PFCn_14	PFCn_13	PFCn_12	PFCn_11	PFCn_10	PFCn_9	PFCn_8	PFCn_7	PFCn_6	PFCn_5	PFCn_4	PFCn_3	PFCn_2	PFCn_1	PFCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPFC0 are named JPFC0_[7:0].

Table 2-13 PFCn/JPFC0 register contents

Bit position	Bit name	Function
15 to 0	PFCn_[15:0]	Specifies the alternative function of a pin. See *Table 2-4 "Alternative mode selection overview (PMCn.PMCn_m = 1)" on page 54* for details.

(8) PFCEn/JPFCE0 - Port function control expansion register

This register, together with register PFCn, specifies an alternative function of the pins.

Some alternative functions require direct I/O control of pin Pn_m. For such alternative functions PIPCn.PIPCn_m must be set to 1 as well.
For other alternative functions, input/output must be specified by PMn.PMn_m.

Access PFCEn: This register can be read/written in 16-bit units.
JPFCE0: This register can be read/written in 8-bit units.

Address PFCEn: <PORTn_base> + 0600_H + n x 4
JPFCE0: <JPORT0_base> + 0060_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PFCEn_15	PFCEn_14	PFCEn_13	PFCEn_12	PFCEn_11	PFCEn_10	PFCEn_9	PFCEn_8	PFCEn_7	PFCEn_6	PFCEn_5	PFCEn_4	PFCEn_3	PFCEn_2	PFCEn_1	PFCEn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPFCE0 are named JPFCE0_[7:0].

Table 2-14 PFCEn/JPFCE0 register contents

Bit position	Bit name	Function
15 to 0	PFCEn_[15:0]	Specifies the alternative function of a pin. See *Table 2-4 "Alternative mode selection overview (PMCn.PMCn_m = 1)" on page 54* for details.

2.3.4 Data input/output registers

(1) PBDCn/JPBDC0 - Port bi-direction control register

This register enables the input buffer of Pn_m, if its output buffer is enabled as well.
Thus the concerned port Pn_m is operated in bi-directional mode and the Pn_m pin level is read via PPRn.PPRn_m.

Note If Pn_m is not configured as output, the input buffer can not be activated via the PBDCn/JPBDCn register.

Access PBDCn: This register can be read/written in 16-bit units.
JPBDC0: This register can be read/written in 8-bit units.

Address PBDCn: <PORTn_base> + 4100_H + n x 4
JPBDC0: <JPORT0_base> + 0410_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PBDCn_15	PBDCn_14	PBDCn_13	PBDCn_12	PBDCn_11	PBDCn_10	PBDCn_9	PBDCn_8	PBDCn_7	PBDCn_6	PBDCn_5	PBDCn_4	PBDCn_3	PBDCn_2	PBDCn_1	PBDCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPBDC0 are named JPBDC0_[7:0].

Table 2-15 PBDCn/JPBDC0 register contents

Bit position	Bit name	Function
15 to 0	PBDCn_[15:0]	Enables/disables bi-directional mode of the corresponding pin: 0: Bi-directional mode disabled 1: Bi-directional mode enabled

(2) PPRn/JPPR0 - Port pin read register

This register reflects the actual level of pin Pn_m, the value of the Pn.Pn_m bit or the level of an alternative output function. The value which is read depends on various control settings as described in *Table 2-5 "PPRn_m read values" on page 55* .

Access PPRn: This register can be read/written in 16-bit units.
JPPR0: This register can be read/written in 8-bit units.

Address PPRn: <PORTn_base> + 0200_H + n x 4
JPPR0: <JPORT0_base> + 0020_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PPRn_15	PPRn_14	PPRn_13	PPRn_12	PPRn_11	PPRn_10	PPRn_9	PPRn_8	PPRn_7	PPRn_6	PPRn_5	PPRn_4	PPRn_3	PPRn_2	PPRn_1	PPRn_0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Note The control bits of the JTAG port register JPPR0 are named JPPR0_[7:0].

Table 2-16 PPRn/JPPR0 register contents

Bit position	Bit name	Function
15 to 0	PPRn_[15:0]	Pin Pn_m, Pn.Pn_m value or alternative function output.

(3) Pn/JP0 - Port register

This register holds the data Pn.Pn_m to be output via the related port Pn_m in output port mode (PMCn.PMCn_m = 0 and PMn.PMn_m = 0).

Access Pn: This register can be read/written in 16-bit units.
JP0: This register can be read/written in 8-bit units.

Address Pn: <PORTn_base> + 0000_H + n x 4
JP0: <JPORT0_base> + 0000_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pn_15	Pn_14	Pn_13	Pn_12	Pn_11	Pn_10	Pn_9	Pn_8	Pn_7	Pn_6	Pn_5	Pn_4	Pn_3	Pn_2	Pn_1	Pn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JP0 are named JP0_[7:0].

Table 2-17 Pn/JP0 register contents

Bit position	Bit name	Function
15 to 0	Pn_[15:0]	Sets the output level of pin m (m = 0 to 15): 0: Outputs low level 1: Outputs high level

Note The bits of this register can be manipulated by different means, refer to *2.2.3 "Pin data input/output" on page 55* under the keyword "Pn register write".

(4) PNOTn/JPNOT0 - Port NOT register

This register allows to invert a bit Pn_m of the port register Pn without directly writing to Pn.

Access PNOTn: This register can be read/written in 16-bit units.
JPNOT0: This register can be read/written in 8-bit units.
These registers are always read as 0000_H.

Address PNOTn: <PORTn_base> + 0700_H + n x 4
JPNOT0: <JPORT0_base> + 0070_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PNOT n_15	PNOT n_14	PNOT n_13	PNOT n_12	PNOT n_11	PNOT n_10	PNOT n_9	PNOT n_8	PNOT n_7	PNOT n_6	PNOT n_5	PNOT n_4	PNOT n_3	PNOT n_2	PNOT n_1	PNOT n_0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Note The control bits of the JTAG port register JPNOT0 are named JPNOT0_[7:0].

Table 2-18 PNOTn/JPNOT0 register contents

Bit position	Bit name	Function
15 to 0	PNOT n_[15:0]	Specifies if Pn.Pn_m is inverted: 0: Pn.Pn_m is not inverted (Pn_m → Pn_m) 1: Pn.Pn_m is inverted ($\overline{Pn\_m}$ → Pn_m)

(5) PSRn/JPSR0 - Port set reset register

This register provides an alternative method to write data to the Pn register.

The register's upper 16 bit PSRn_[31:16] specify which Pn.Mn_m bit will be modified by the corresponding bit of the lower 16 bit PSRn_[15:0].

Access These registers can be read/written in 32-bit units.
Bits 31 to 16 are always read as 0000_H.
Reading bits 15 to 0 returns the value of register Pn/JP0.

Address PSRn: <PORTn_base> + 0100_H + n x 4
JPSR0: <JPORT0_base> + 0010_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PSRn_31	PSRn_30	PSRn_29	PSRn_28	PSRn_27	PSRn_26	PSRn_25	PSRn_24	PSRn_23	PSRn_22	PSRn_21	PSRn_20	PSRn_19	PSRn_18	PSRn_17	PSRn_16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSRn_15	PSRn_14	PSRn_13	PSRn_12	PSRn_11	PSRn_10	PSRn_9	PSRn_8	PSRn_7	PSRn_6	PSRn_5	PSRn_4	PSRn_3	PSRn_2	PSRn_1	PSRn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPSR0 are named JPSR0_[31:0].

Table 2-19 PSRn/JPSR0 register contents

Bit position	Bit name	Function
31 to 16	PSRn_[31:16]	PSRn_m specifies whether the value of the corresponding lower bit PSRn_m value is written to Pn_m: 0: Pn_m is independent of PSRn_m 1: Pn_m is PSRn_m Example: If PSRn.PSRn31 = 1, the value of bit PSRn.PSRn_15 is written to bit Pn.Pn_15 and output.
15 to 0	PSRn_[15:0]	Specifies the Pn_m value if the corresponding upper bit PSRn_(m+16) is 1: 0: Pn_m = 0 1: Pn_m = 1

2.3.5 Configuration of electrical characteristics registers

(1) PUn/JPU0 - Pull-up option register

This register specifies whether a pull-up resistor is connected to an input pin.

Access PUn: This register can be read/written in 16-bit units.
JPU0: This register can be read/written in 8-bit units.

Address PUn: <PORTn_base> + 4300_H + n x 4
JPU0: <JPORT0_base> + 0430_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PU n_15	PU n_14	PU n_13	PU n_12	PU n_11	PU n_10	PU n_9	PU n_8	PU n_7	PU n_6	PU n_5	PU n_4	PU n_3	PU n_2	PU n_1	PU n_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPU0 are named JPU0_[7:0].

Table 2-20 PUn/JPU0 register contents

Bit position	Bit name	Function
15 to 0	PU n_[15:0]	Specifies whether a pull-up resistor is connected to the corresponding pin: 0: No pull-up resistor connected 1: Pull-up resistor connected

Notes
1. If a pin is configured that both a pull-up resistor (PUn.PUn_m = 1) and a pull-down resistor (PDn.PDn_m = 1) are connected, the pull-down resistor is automatically selected and the pull-up resistor is not connected.
2. The pull-up resistor has no effect when the pin is operated in output mode.

(2) PDn/JPD0 - Pull-down option register

This register specifies whether a pull-down resistor is connected to an input pin.

Access PDn: This register can be read/written in 16-bit units.
JPD0: This register can be read/written in 8-bit units.

Address PDn: <PORTn_base> + 4400_H + n x 4
JPD0: <JPORT0_base> + 0440_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PD n_15	PD n_14	PD n_13	PD n_12	PD n_11	PD n_10	PD n_9	PD n_8	PD n_7	PD n_6	PD n_5	PD n_4	PD n_3	PD n_2	PD n_1	PD n_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPD0 are named JPD0_[7:0].

Table 2-21 PDn/JPD0 register contents

Bit position	Bit name	Function
15 to 0	PD n_[15:0]	Specifies whether a pull-down resistor is connected to the corresponding pin: 0: No pull-down resistor connected 1: Pull-down resistor connected

Notes
1. If a pin is configured that both a pull-up resistor (PUn.PUn_m = 1) and a pull-down resistor (PDn.PDn_m = 1) are connected, the pull-down resistor is automatically selected and the pull-up resistor is not connected.
2. The pull-down resistor has no effect when the pin is operated in output mode.

(3) PDSCn/JPDSC0 - Port drive strength control register

This register enables the driver strength control function.

Protection Writing to this register is protected by a special sequence of instructions. Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access These registers can be read/written in 32-bit units.
The bits 31 to 16 must always be written with "0" and "0" is returned when read.

Address PDSCn: <PORTn_base> + 4600_H + n x 4
JPDSC0: <JPORT0_base> + 0460_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDSCn_15	PDSCn_14	PDSCn_13	PDSCn_12	PDSCn_11	PDSCn_10	PDSCn_9	PDSCn_8	PDSCn_7	PDSCn_6	PDSCn_5	PDSCn_4	PDSCn_3	PDSCn_2	PDSCn_1	PDSCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPDSC0 are named JPDSC0_[31:0].

Table 2-22 PDSCn/JPDSC0 register contents

Bit position	Bit name	Function
15 to 0	PDSC n_[15:0]	Enables/disables output current limiting function: 0: Current limitation enabled 1: Current limitation disabled

(4) PODCn/JPODC0 - Port open drain control register

This register selects push-pull or open-drain as output buffer function.

Protection Writing to this register is protected by a special sequence of instructions. Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access These registers can be read/written in 32-bit units.
The bits 31 to 16 must always be written with "0" and "0" is returned when read.

Address PODCn: <PORTn_base> + 4500_H + n x 4
JPODC0: <JPORT0_base> + 0450_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PODCn_15	PODCn_14	PODCn_13	PODCn_12	PODCn_11	PODCn_10	PODCn_9	PODCn_8	PODCn_7	PODCn_6	PODCn_5	PODCn_4	PODCn_3	PODCn_2	PODCn_1	PODCn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPODC0 are named JPODC0_[31:0].

Table 2-23 PODCn/JPODC0 register contents

Bit position	Bit name	Function
15 to 0	PODC n_[15:0]	Specifies the output buffer function: 0: Push-pull 1: Open-drain

(5) PISn/JPIS0 - Port input buffer selection register

This register specifies the input buffer characteristics.

A port can have up to four different input buffer characteristics.
The type of input characteristic is selected by the

- port input buffer selection register PISn
- port input buffer selection expansion register PISEn

Table 2-24 Port input buffer characteristic selection

PISEn_m	PISn_m	Input buffer characteristic
0	0	CMOS (type 1)
	1	Schmitt2 (type 2)
1	0	Schmitt1 (type 3)
	1	Schmitt4 (type 4)

Refer to the Electrical Target Specification for electrical characteristics of the different types and which types are available for each port.

Access PISn: This register can be read/written in 16-bit units.
JPIS0: This register can be read/written in 8-bit units.

Address PISn: <PORTn_base> + 4700_H + n x 4
JPIS0: <JPORT0_base> + 0470_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PIS n_15	PIS n_14	PIS n_13	PIS n_12	PIS n_11	PIS n_10	PIS n_9	PIS n_8	PIS n_7	PIS n_6	PIS n_5	PIS n_4	PIS n_3	PIS n_2	PIS n_1	PIS n_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPIS0 are named JPIS0_[7:0].

Table 2-25 PISn/JPIS0 register contents

Bit position	Bit name	Function
15 to 0	PIS n_[15:0]	Specifies the input buffer characteristic of port m (m = 0 to 15) together with the bits PISEn[15:0].

(6) PISEn/JPISE0 - Port input buffer selection expansion register

This register specifies the input buffer characteristics together with the port input selection register PISn.

If a port has up to five input buffer characteristics, the port input selection advanced register PISAn is also valid.

Access PISEn: This register can be read/written in 16-bit units.
JPISE0: This register can be read/written in 8-bit units.

Address PISEn: <PORTn_base> + 4800_H + n x 4
JPISE0: <JPORT0_base> + 0480_H

Initial Value Refer to the section *"V850E2/Fx4-H Port Groups Configuration"*.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PISEn_15	PISEn_14	PISEn_13	PISEn_12	PISEn_11	PISEn_10	PISEn_9	PISEn_8	PISEn_7	PISEn_6	PISEn_5	PISEn_4	PISEn_3	PISEn_2	PISEn_1	PISEn_0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note The control bits of the JTAG port register JPISE0 are named JPISE0_[7:0].

Table 2-26 PISEn/JPISE0 register contents

Bit position	Bit name	Function
15 to 0	PISEn_[15:0]	Specifies the input buffer characteristic of port m (m = 0 to 15) together with the bits PISn[15:0]. Refer to the PISn register description for how to select the input buffer characteristic.

2.4

2.5 V850E2/Fx4-H Port Group Configuration

This section provides

- an overview of the port register protection clusters, refer to the section *"Port registers protection clusters"*
- general information for all ports , refer to the section *"Common port functions"*
- details of all port groups and their associated control registers for each device, refer to the sections
  - *"V850E2K/FK4-H port functions"*
  - *"V850E2K/FL4-H port functions"*
- a list of input/output signals with port functionality, refer to the section *"Non-port input/putput signals"*
- an alphabetic pin functions list and the ports, the functions can be assigned to, refer to the section *"Alphabetic pin function list"*
- a description of the port status during and after reset and in stand-by modes, refer to the section *"Port functions during/after reset and in stand-by modes"*
- recommendations concerning unused pins, refer to the section *"Recommended connection of unused pins"*.

2.5.1 Port register protection clusters

Several registers of certain port groups n are bundled in port protection clusters:

Table 2-27 Port protection clusters

Port protection cluster	Port groups
1	JP0
2	P0
3	P1 to P4, P10, P11
4	P12, P13, P21, P24 to P29

For further information concerning port register protection refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

2.5.2 Common port functions

This section provides information about special port functions, common to all devices.

(1) Initialization of port control registers

The port control registers are initialized by the following reset signals:

Table 2-28 Port control registers reset signals

Port group	Power domain	Reset signal
JP0, P0, P5	Always-On-Area	• Reset Controller: SYSRES
P1 to P4, P10, P11	Isolated-Area-0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
P12, P13, P21, P24 to P29	Isolated-Area-1	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

(2) P0_0: $\overline{\text{RESETOUT}}$

After reset release P0_0 outputs a $\overline{\text{RESETOUT}}$ signal, which is low level after reset release. P0_0 is configured as follows after reset release:

- PM0.PM0_0 = 0: port output
- PODC0.PODC0_0 = 1: open-drain output

Since P0.P0_0 = 0 after reset, low level is output.

Any change of the P0_0 configuration terminates the $\overline{\text{RESETOUT}}$ output.

Note Since the $\overline{\text{RESETOUT}}$ signal is activated by all reset events, thus also when an internal reset is applied, it can be used for reset of external devices.

Caution Once asserted the $\overline{\text{RESETOUT}}$ remains on low level. It must be de-asserted by changing the port configuration of P0_0 after reset release.

(3) JP_0 to JP_5: Debug interface

If the debug reset $\overline{\text{DCUTRST}}$ is at high level at reset release, the port of the JP0 port group are used for the debugger interface:

- JP0_0: DCUTDI input
- JP0_1: DCUTDO output
- JP0_2: DCUTCK input
- JP0_3: DCUTMS
- JP0_4: $\overline{\text{DCUTRST}}$
- JP0_5: $\overline{\text{DCUTRDY}}$

Consequently all port and alternative modes on these pins can not be used while the debugger is connected.

Refer to the chapter *"On-chip Debug Unit (OCD)"* and the section *"Operation Modes"* of chapter *"CPU System Functions"* for details.

Note In order to connect the debugger via the JP0 pins the flash mask option OPBT0.OPBT0[31] has to be set to 1.

(4) JP0_0, JP0_1, JP0_2: Flash programmer

These ports are used for connecting a flash programmer.

Refer to the chapter *"Flash Memory"* and the section *"Operation Modes"* of chapter *"CPU System Functions"* for details.

(5) Mode pins

Following ports are used as mode setting signals in combination with the FLMD0 pin:

- P0_1: FLMD1
- P0_2: MODE0
- P0_3: MODE1

Refer to the section *"Operation Modes"* of chapter *"CPU System Functions"* for details.

(6) Permanent inputs

Permanent input means, that the input to a port is unconditionally connected to another module. Thus settings of the port control registers do not impact this connection.

Following ports are permanently connected to dedicated function modules:

Table 2-29 Permanent input functions

Port	Permanent input to
P10_6	ADCA0I6
P10_7	ADCA0I7
P10_8	ADCA0I8
P10_9	ADCA0I9
P10_10	ADCA0I10
P10_11	ADCA0I11
P10_12	ADCA0I12
P10_13	ADCA0I13
P10_14	ADCA0I14
P10_15	ADCA0I15
P11_0	ADCA0I16
P11_1	ADCA0I17
P11_2	ADCA0I18
P11_3	ADCA0I19

Port	Permanent input to
P11_4	ADCA0I20
P11_5	ADCA0I21
P11_6	ADCA0I22
P11_7	ADCA0I23
P12_0	ADCA1I0
P12_1	ADCA1I1
P12_2	ADCA1I2
P12_3	ADCA1I3
P12_4	ADCA1I4
P12_5	ADCA1I5
P12_6	ADCA1I6
P12_7	ADCA1I7
P12_8	ADCA1I8
P12_9	ADCA1I9

Port	Permanent input to
P12_10	ADCA1I10
P12_11	ADCA1I11
P12_12	ADCA1I12
P12_13	ADCA1I13
P12_14	ADCA1I14
P12_15	ADCA1I15
P13_0	ADCA1I16
P13_1	ADCA1I17
P13_2	ADCA1I18
P13_3	ADCA1I19
P13_4	ADCA1I20
P13_5	ADCA1I21
P13_6	ADCA1I22
P13_7	ADCA1I23

Notes

1. If the ports of the port groups P10 to P13 shall be used in input port mode, set
   - ADCA0CTL1.ADCA0GPS = 1 to use P10 or P11 in input port mode
   - ADCA1CTL1.ADCA1GPS = 1 to use P12 or P13 in input port mode
2. The input pins of the ADCA0 simultaneous sampling channels ADCA0I0 to ADCA0I5 do not provide port functionality.

(7) Direct I/O control (PIPC)

Some modules take over the input and output control of the used ports automatically.
These ports have to be set in alternative mode by setting PMCn.PMCn_m, PFCn.PFCn_m and PFCEn.PFCEn_m accordingly and I/O control has to be handed over to the module by setting

PIPCn.PIPCn_m = 1.

The setting of PMn.PMn_m has no more effect for these ports.

The following table lists all alternative modes, where PIPCn.PIPCn_m has to be set to 1.
Note that not all functions in the table below are available for all devices.

Table 2-30 Alternative modes with PIPCn.PIPCn_m = 1 (1/2)

Port	Function	Alternative mode
Clocked Serial Interfaces G (CSIG):		
P0_14	CSIG0SO	ALT_OUT4
P0_15	CSIG0SC	ALT_IN4/ALT_OUT4
P4_4	CSIG0SO	ALT_OUT2
P4_5	CSIG0SC	ALT_IN2/ALT_OUT2
P3_1	CSIG2SO	ALT_OUT4
P3_2	CSIG2SC	ALT_IN4/ALT_OUT4
P4_10	CSIG2SO	ALT_OUT2
P4_11	CSIG2SC	ALT_IN2/ALT_OUT2
P25_0	CSIG2SO	ALT_OUT3
P25_2	CSIG2SC	ALT_IN3/ALT_OUT3
P0_1	CSIG4SO	ALT_OUT2
P0_3	CSIG4SC	ALT_IN2/ALT_OUT2
P3_6	CSIG4SO	ALT_OUT4
P3_5	CSIG4SC	ALT_IN4/ALT_OUT4
P4_7	CSIG4SO	ALT_OUT2
P4_8	CSIG4SC	ALT_IN2/ALT_OUT2
P25_4	CSIG4SO	ALT_OUT3
P25_5	CSIG4SC	ALT_IN3/ALT_OUT3
Clocked Serial Interfaces H (CSIH):		
P4_1	CSIH0SO	ALT_OUT3
P4_2	CSIH0SC	ALT_IN3/ALT_OUT3
P1_8	CSIH1SO	ALT_OUT3
P1_9	CSIH1SC	ALT_IN3/ALT_OUT3
P1_3	CSIH2SO	ALT_OUT4
P1_2	CSIH2SC	ALT_IN4/ALT_OUT4
P21_3	CSIH2SO	ALT_OUT2
P21_2	CSIH2SC	ALT_IN2/ALT_OUT2
External Memory Controller (MEMC):		
P21_2	$\overline{\text{MEMC0BEN1}}$	ALT_OUT1
P21_3	$\overline{\text{MEMC0BEN0}}$	ALT_OUT1
P21_4	$\overline{\text{MEMC0WR}}$	ALT_OUT1
P21_5	$\overline{\text{MEMC0RD}}$	ALT_OUT1
P21_9	$\overline{\text{MEMC0CS2}}$	ALT_OUT1
P21_10	$\overline{\text{MEMC0CS3}}$	ALT_OUT1
P21_11	$\overline{\text{MEMC0CS4}}$	ALT_OUT1
P21_15	$\overline{\text{MEMC0ASTB}}$	ALT_OUT1
P25_0	MEMC0AD0	ALT_IN1/ALT_OUT1
P25_1	MEMC0AD1	ALT_IN1/ALT_OUT1
P25_2	MEMC0AD2	ALT_IN1/ALT_OUT1
P25_3	MEMC0AD3	ALT_IN1/ALT_OUT1

Table 2-30 Alternative modes with PIPCn.PIPCn_m = 1 (2/2)

Port	Function	Alternative mode
P25_4	MEMC0AD4	ALT_IN1/ALT_OUT1
P25_5	MEMC0AD5	ALT_IN1/ALT_OUT1
P25_6	MEMC0AD6	ALT_IN1/ALT_OUT1
P25_7	MEMC0AD7	ALT_IN1/ALT_OUT1
P25_8	MEMC0AD8	ALT_IN1/ALT_OUT1
P25_9	MEMC0AD9	ALT_IN1/ALT_OUT1
P25_10	MEMC0AD10	ALT_IN1/ALT_OUT1
P25_11	MEMC0AD11	ALT_IN1/ALT_OUT1
P25_12	MEMC0AD12	ALT_IN1/ALT_OUT1
P25_13	MEMC0AD13	ALT_IN1/ALT_OUT1
P25_14	MEMC0AD14	ALT_IN1/ALT_OUT1
P25_15	MEMC0AD15	ALT_IN1/ALT_OUT1
P27_0	MEMC0A16	ALT_OUT1
P27_1	MEMC0A17	ALT_OUT1
P27_2	MEMC0A18	ALT_OUT1
P27_5	$\overline{\text{MEMC0ASTB}}$	ALT_OUT2
Ethernet Controller (ETHA):		
P25_15	ETH0MDI	ALT_IN2
P25_15	ETH0MDO	ALT_OUT2

2.5.3 V850E2/FK4-H port functions

This section summarizes all port functions of the V850E2/FK4-H devices and its port control registers.

(1) General I/O functions

The table below shows all alternative functions, that can be applied to the V850E2/FK4-H ports.
It also gives the settings of the control bits PMCn_m, PFCn_m, PFCEn_m and PMn_m to the respective port into the different modes.

Table 2-31 V850E2/FK4-H general I/O functions (1/5)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group 0 (Always-On-Area, E0VDD/E0VSS power supply):								
P0_0	TAUJ1I0	TAUJ1O0	$\overline{\text{CSIG4SSI}}$		ADCA0TRG0		INTP0	
P0_1	TAUJ1I1	TAUJ1O1		CSIG4SO[a]	URTE2RX		INTP1	TAUA0O1
P0_2	TAUJ1I2	TAUJ1O2	CSIG4SI	RTCA0OUT	ADCA0TRG2	URTE2TX	INTP2	TAUA0O2
P0_3	TAUJ1I3	TAUJ1O3	CSIG4SC[a]		ADCA0TRG1		INTP3/ TAPA0ESO	
P0_4		FCN0TX					INTP11	
P0_5	FCN0RX						INTP12	
P0_6	FCN1RX	URTE11TX	KR0I1	CSIH2CSS1	NMI			
P0_7	URTE11RX	FCN1TX	KR0I2	CSIH2CSS2	INTP4			
P0_8	FCN2RX	URTE10TX	KR0I3	CSIH2CSS3	INTP5	TAUA0O5		IICB0SDA[b]
P0_9	URTE10RX	FCN2TX	KR0I4	CSIH2CSS4	INTP6	TAUA0O6		IICB0SCL[b]
P0_10	FCN3RX	URTE11TX			INTP9			
P0_11	URTE11RX	FCN3TX			INTP10			
P0_12	TAUJ0I0	TAUJ0O0	KR0I0		INTP8		$\overline{\text{CSIG0SSI}}$	CSCXFOUT
P0_13	TAUJ0I1	TAUJ0O1	KR0I5	CSIH2CSS5	INTP7	FCN5TX	CSIG0SI	
P0_14	TAUJ0I2	TAUJ0O2	KR0I6	CSIH2CSS6	FCN5RX	TAUB1O13		CSIG0SO[a]
P0_15	TAUJ0I3	TAUJ0O3	KR0I7	CSIH2CSS7		TAUB1O14	CSIG0SC[a]	
Port group 1 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P1_1	TAUA0I1	TAUA0O1		TAUC4O1	ENCA0AIN		FCN1RX	FCN0TX
P1_2	TAUA0I2	TAUA0O2		TAUC4O2	ENCA0BIN	TAPA0UP	CSIH2SI	FCN1TX
P1_3	TAUA0I3	TAUA0O3		TAUC4O5	ENCA0ZIN	TAPA0UN		CSIH2SO[a]
P1_4	TAUA0I4	TAUA0O4		TAUC4O6	ENCA0TIN0	TAPA0VP	CSIH2SC[a]	
P1_5	TAUA0I5	TAUA0O5		TAUC4O9	ENCA0TIN1	TAPA0VN	CSIH2RY	
P1_6	TAUA0I6	TAUA0O6		TAUC4O10	$\overline{\text{CSIH3SSI}}$	TAPA0WP	$\overline{\text{CSIH2SSI}}$	CSIH2CSS0
P1_7	TAUA0I7	TAUA0O7		TAUC4O13	CSIH1SI	TAPA0WN	FCN0RX	CSIH2CSS1
P1_8	TAUA0I8	TAUA0O8		TAUC4O14		CSIH1SO[a]	FCN2RX	URTE4TX

Table 2-31 V850E2/FK4-H general I/O functions (2/5)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P1_9	TAUA0I9	TAUA0O9	INTP3	FLX0TXENA	CSIH1SC[a]		URTE4RX	FCN2TX
P1_10	TAUA0I10	TAUA0O10	FLX0RXDA	URTE3TX	CSIH1RY		INTP4	
P1_11	TAUA0I11	TAUA0O11	URTE3RX	FLX0TXDA		CSIH1CSS0	INTP5	
P1_12	TAUA0I12	TAUA0O12	FLX0RXDB	URTE4TX		CSIH1CSS1	INTP6	
P1_13	TAUA0I13	TAUA0O13	URTE4RX	FLX0TXDB		CSIH1CSS2	INTP7	
P1_14	TAUA0I14	TAUA0O14	FLX0STPWT	URTE5TX		CSIH1CSS3	INTP8	
P1_15	TAUA0I15	TAUA0O15	URTE5RX	FLX0TXENB		CSIH1CSS4	INTP9	
Port group 2 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P2_0				URTE6TX		CSIH1CSS5	INTP10	
P2_1			URTE6RX	URTE7TX	INTP0	CSIH1CSS6	TAUJ1I3	TAUJ1O3
P2_2			URTE7RX		INTP1	CSIH1CSS7	TAUJ1I2	TAUJ1O2
Port group 3 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P3_0	TAUJ1I1	TAUJ1O1	TAUA0I0	TAUA0O0				
P3_1	TAUB2I1	TAUB2O1	TAUA0I1	TAUA0O1				
P3_2	TAUB2I2	TAUB2O2	TAUA0I2	TAUA0O2	KR0I7			
P3_3	TAUB2I3	TAUB2O3	TAUA0I3	TAUA0O3	KR0I6			
P3_4	TAUB2I5	TAUB2O5	TAUA0I4	TAUA0O4	KR0I5		CSIG0RY	
P3_5	TAUB2I6	TAUB2O6	TAUA0I5	TAUA0O5	KR0I4		CSIG0SC[a]	
P3_6	TAUB2I7	TAUB2O7	TAUA0I6	TAUA0O6				CSIG0SO[a]
P3_7	TAUB2I9	TAUB2O9	TAUA0I7	TAUA0O7	URTE7RX		CSIG0SI	URTE3TX
P3_8	TAUB2I10	TAUB2O10	TAUA0I8	TAUA0O8	INTP11	URTE6TX	URTE3RX	
P3_9	TAUB2I11	TAUB2O11	TAUA0I9	TAUA0O9	URTE5RX	FCN5TX	INTP12	URTE3TX
Port group 4 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P4_0	TAUB1I1	TAUB1O1	TAUA0I13	TAUA0O13	CSIH0SI	URTE7TX		
P4_1	TAUB1I2	TAUB1O2	TAUA0I14	TAUA0O14		CSIH0SO[a]	URTE2RX	FCN3TX
P4_2	TAUB1I3	TAUB1O3	TAUA0I15	TAUA0O15	CSIH0SC[a]		FCN3RX	URTE2TX
P4_3	TAUB1I5	TAUB1O5	CSIG0SI	URTE10TX	CSIH0RY		INTP10	
P4_4	INTP2	TAUB1O6	URTE10RX	CSIG0SO[a]	$\overline{\text{CSIH0SSI}}$	CSIH0CSS0	ENCA0TIN0	
P4_5	TAUB1I7	TAUB1O7	CSIG0SC[a]		KR0I3	CSIH0CSS1	ENCA0TIN1	
P4_6	TAUB1I9	TAUB1O9	CSIG4SI	URTE11TX	KR0I2	CSIH0CSS2	ENCA0AIN	
P4_7	INTP4	TAUB1O10	URTE11RX	CSIG4SO[a]	KR0I1	CSIH0CSS3	ENCA0BIN	
P4_8	TAUB1I11	TAUB1O11	CSIG4SC[a]		KR0I0	CSIH0CSS4	ENCA0ZIN	
P4_9	TAUB1I13	TAUB1O13		CSIG0RY		CSIH0CSS5		URTE8TX
P4_10	TAUB1I14	TAUB1O14	CSIG4RY		INTP15	CSIH0CSS6	URTE8RX	
P4_11	TAUB1I15	TAUB1O15				CSIH0CSS7		

Table 2-31 V850E2/FK4-H general I/O functions (3/5)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group 10 (Isolated-Area-0, A0VDD/A0VSS power supply):[c]								
P10_6								
P10_7								
P10_8								
P10_9	ADCA0TRG0							
P10_10	ADCA0TRG1							
P10_11	ADCA0TRG2							
P10_12								
P10_13								
P10_14								
P10_15								
Port group 11 (Isolated-Area-0, A0VDD/A0VSS power supply):[c]								
P11_0								
P11_1								
P11_2								
P11_3								
P11_4								
P11_5								
P11_6								
P11_7								
Port group 12 (Isolated-Area-1, A1VDD/A1VSS power supply):[d]								
P12_0								
P12_1								
P12_2								
P12_3								
P12_4								
P12_5								
P12_6								
P12_7								
P12_8								
P12_9	ADCA1TRG0							
P12_10	ADCA1TRG1							
P12_11	ADCA1TRG2							
P12_12								
P12_13								
P12_14								

Table 2-31 V850E2/FK4-H general I/O functions (4/5)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P12_15								
Port group 21 (Isolated-Area-1, B0VDD/VSS power supply):								
P21_2	INTP10	$\overline{\text{MEMC0BEN1}}$	CSIH2SI			IICB0SDA[b]	TAUB2I13	TAUB2O13
P21_3	INTP11	$\overline{\text{MEMC0BEN0}}$		CSIH2SO[a]		IICB0SCL[b]	TAUB2I14	TAUB2O14
P21_4	INTP12	$\overline{\text{MEMC0WR}}$	CSIH2SC[a]					TAUC3O1
P21_5	INTP13	$\overline{\text{MEMC0RD}}$	CSIH2RY			URTE9TX		TAUC3O2
P21_6	INTP14	MEMC0CLK	$\overline{\text{CSIH2SSI}}$	CSIH2CSS0	URTE9RX			TAUC3O5
P21_7	$\overline{\text{MEMC0WAIT}}$			CSIH2CSS1		URTE0TX		TAUC3O6
P21_8	INTP15			CSIH2CSS2	URTE0RX	PMCA0 MSEL0		TAUC3O9
P21_9		$\overline{\text{MEMC0CS2}}$[a]		CSIH2CSS3		PMCA0 MSEL1		TAUC3O10
P21_10		$\overline{\text{MEMC0CS3}}$[a]		CSIH2CSS4	URTE1RX	PMCA0 MSEL2		TAUC3O13
P21_11		$\overline{\text{MEMC0CS4}}$[a]		CSIH2CSS5		URTE1TX		TAUC3O14
Port group 25 (Isolated-Area-1, B0VDD/VSS power supply):								
P25_0	MEMC0AD0[a]		INTP6	ETH0TXD3	URTE6RX			TAUC6O1
P25_1	MEMC0AD1[a]			ETH0TXD2	CSIG2SI	URTE6TX		TAUC6O2
P25_2	MEMC0AD2[a]			ETH0TXD1				TAUC6O5
P25_3	MEMC0AD3[a]			ETH0TXD0	CSIG4SI	URTE7TX		TAUC6O6
P25_4	MEMC0AD4[a]		INTP7	ETH0TXEN	URTE7RX	CSIG4SO[a]		TAUC6O9
P25_5	MEMC0AD5[a]		ETH0 REFCLK	ETH0TXER	CSIG4SC[a]			TAUC6O10
P25_6	MEMC0AD6[a]		ETH0CRSDV					TAUC6O13
P25_7	MEMC0AD7[a]		ETH0RXER					TAUC6O14
P25_8	MEMC0AD8[a]		ETH0RXD0					TAUC5O1
P25_9	MEMC0AD9[a]		ETH0RXD1					TAUC5O2
P25_10	MEMC0AD10[a]		ETH0RXD2	CSIH2CSS6				TAUC5O5
P25_11	MEMC0AD11[a]		ETH0RXD3	CSIH2CSS7				TAUC5O6
P25_12	MEMC0AD12[a]		ETH0RXDV			IICB0SDA[b]		TAUC5O9
P25_13	MEMC0AD13[a]		ETH0RXCLK			IICB0SCL[b]		TAUC5O10
P25_14	MEMC0AD14[a]		INTP5	ETH0MDC	URTE5RX	FCN3TX		TAUC5O13
P25_15	MEMC0AD15[a]		ETH0MDI[a]	ETH0MDO[a]	FCN3RX	URTE5TX		TAUC5O14

Table 2-31 V850E2/FK4-H general I/O functions (5/5)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group 27 (Isolated-Area-1, B0VDD/VSS power supply):								
P27_0	INTP0	MEMC0A16[a]			ADCA1TRG2	PMCA0 MSEL0		
P27_1	INTP1	MEMC0A17[a]	ETH0COL		ADCA1TRG1	PMCA0 MSEL1		
P27_2	INTP2	MEMC0A18[a]	ETH0TXCLK		ADCA1TRG0	PMCA0 MSEL2		
P27_4	INTP4							
P27_5	INTP5			$\overline{\text{MEMC0ASTB}}$[a]				
Port group JP0 (Always-On-Area, E0VDD/E0VSS power supply):								
JP0_0	INTP0	VCPC1OUT	TAUJ0I0	TAUJ0O0				
JP0_1	INTP1	VCPC0OUT	TAUJ0I1	TAUJ0O1				
JP0_2	INTP2	CSCXFOUT	TAUJ0I2	TAUJ0O2				
JP0_3	INTP3		TAUJ0I3	TAUJ0O3				
JP0_4								
JP0_5	NMI	RTCA0OUT						

a) When using this alternative mode, set PIPCn.PIPCn_m = 1. The module controls the I/O setting and PMn_m has no effect.

b) Refer to the section *"I²C Interface Port Settings"* in the "I²C Interface (IICB)" chapter for details about the correct configuration of the I²C Interface ports.

c) The ports of port groups 10 and 11 are also used as permanent inputs to the A/D Converter ADCA0. If the P10 or P11 ports shall be used as port inputs, set the ADCA0CTL1.ADCA0GPS = 1.

d) The ports of port groups 12 and 13 are also used as permanent inputs to the A/D Converter ADCA1. If the P12 or P13 ports shall be used as port inputs, set the ADCA1CTL1.ADCA1GPS = 1.

(2) V850E2/FK4-H port control registers

The following table summarizes all V850E2/FK4-H port control registers, their addresses and intitial values.

Tables legend

A: Register address

I: Initial value

B: Available bits
- 1: available, x: not available
- right: bit 0, left: bit 15

Table 2-32 V850E2/FK4-H port (groups 0 to 3) control registers (1/2)

Register		Port group n =			
		0	1	2	3
Pn	A:	FF40 0000$_H$	FF40 0004$_H$	FF40 0008$_H$	FF40 000C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PSRn	A:	FF40 0100$_H$	FF40 0104$_H$	FF40 0108$_H$	FF40 010C$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PNOTn	A:	FF40 0700$_H$	FF40 0704$_H$	FF40 0708$_H$	FF40 070C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PPRn	A:	FF40 0200$_H$	FF40 0204$_H$	FF40 0208$_H$	FF40 020C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PMn	A:	FF40 0300$_H$	FF40 0304$_H$	FF40 0308$_H$	FF40 030C$_H$
	I:	FFFE$_H$	FFFF$_H$	0007$_H$	0FFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PMCn	A:	FF40 0400$_H$	FF40 0404$_H$	FF40 0408$_H$	FF40 040C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PFCn	A:	FF40 0500$_H$	FF40 0504$_H$	FF40 0508$_H$	FF40 050C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxx1 1111 1111 1111
PFCEn	A:	FF40 0600$_H$	FF40 0604$_H$	FF40 0608$_H$	FF40 060C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 11xx
PMSRn	A:	FF40 0800$_H$	FF40 0804$_H$	FF40 0808$_H$	FF40 080C$_H$
	I:	0000 FFFE$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PMCSRn	A:	FF40 0900$_H$	FF40 0904$_H$	FF40 0908$_H$	FF40 090C$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111

Table 2-32 V850E2/FK4-H port (groups 0 to 3) control registers (2/2)

Register		Port group n =			
		0	1	2	3
PIBCn	A:	FF40 4000$_H$	FF40 4004$_H$	FF40 4008$_H$	FF40 400C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PBDCn	A:	FF40 4100$_H$	FF40 4104$_H$	FF40 4108$_H$	FF40 410C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PIPCn	A:	FF40 4200$_H$	FF40 4204$_H$	FF40 4208$_H$	FF40 420C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PUn	A:	FF40 4300$_H$	FF40 4304$_H$	FF40 4308$_H$	FF40 430C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PDn	A:	FF40 4400$_H$	FF40 4404$_H$	FF40 4408$_H$	FF40 440C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PODCn	A:	FF40 4500$_H$	FF40 4504$_H$	FF40 4508$_H$	FF40 450C$_H$
	I:	0000 0001$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PDSCn	A:	FF40 4600$_H$	FF40 4604$_H$	FF40 4608$_H$	FF40 460C$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PISn	A:	FF40 4700$_H$	FF40 4704$_H$	FF40 4708$_H$	FF40 470C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PISEn	A:	FF40 4800$_H$	FF40 4804$_H$	FF40 4808$_H$	FF40 480C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxxx xx11 1111 1111
PPCMDn	A:	FF40 4C00$_H$	FF40 4C04$_H$	FF40 4C08$_H$	FF40 4C0C$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B00$_H$	FF40 4B04$_H$	FF40 4B08$_H$	FF40 4B0C$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-33 V850E2/FK4-H port (groups 4, 10 to 12) control registers (1/2)

Register		Port group n =			
		4	10	11	12
Pn	A:	FF40 0010$_H$	FF40 0028$_H$	FF40 002C$_H$	FF40 0030$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PSRn	A:	FF40 0110$_H$	FF40 0128$_H$	FF40 012C$_H$	FF40 0130$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PNOTn	A:	FF40 0710$_H$	FF40 0728$_H$	FF40 072C$_H$	FF40 0730$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PPRn	A:	FF40 0210$_H$	FF40 0228$_H$	FF40 022C$_H$	FF40 0230$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMn	A:	FF40 0310$_H$	FF40 0328$_H$	FF40 032C$_H$	FF40 0330$_H$
	I:	0FFF$_H$	FF40$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCn	A:	FF40 0410$_H$	FF40 0428$_H$	–	FF40 0304$_H$
	I:	0000$_H$	0000$_H$		0000$_H$
	B:	xxxx 1111 1111 1111	xxxx 111x xxxx xxxx		xxxx 111x xxxx xxxx
PFCn	A:	FF40 0510$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PFCEn	A:	FF40 0610$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PMSRn	A:	FF40 0810$_H$	FF40 0828$_H$	FF40 082C$_H$	FF40 0830$_H$
	I:	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCSRn	A:	FF40 0910$_H$	FF40 0928$_H$		FF40 0930$_H$
	I:	0000 0000$_H$	0000 0000$_H$		0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx		xxxx 111x xxxx xxxx
PIBCn	A:	FF40 4010$_H$	FF40 4028$_H$	FF40 402C$_H$	FF40 4030$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PBDCn	A:	FF40 4110$_H$	FF40 4128$_H$	FF40 412C$_H$	FF40 4130$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PIPCn	A:	FF40 4210$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			

Table 2-33 V850E2/FK4-H port (groups 4, 10 to 12) control registers (2/2)

Register		Port group n =			
		4	10	11	12
PUn	A:	FF40 4310$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PDn	A:	FF40 4410$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PODCn	A:	FF40 4510$_H$	FF40 4528$_H$	FF40 452C$_H$	FF40 4530$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PDSCn	A:	FF40 4610$_H$	–	–	–
	I:	0000 0000$_H$			
	B:	xxxx 1111 1111 1111			
PISn	A:	FF40 4710$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PISEn	A:	FF40 4810$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PPCMDn	A:	FF40 4C10$_H$	FF40 4C28$_H$	FF40 4C2C$_H$	FF40 4C30$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B10$_H$	FF40 4B28$_H$	FF40 4B2C$_H$	FF40 4B30$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-34 V850E2/FK4-H port (groups 21, 25, 27, JP) control registers (1/3)

Register		Port group n =			
		21	25	27	JP
Pn	A:	FF40 0054$_H$	FF40 0064$_H$	FF40 006C$_H$	FF44 0000$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PSRn	A:	FF40 0154$_H$	FF40 0164$_H$	FF40 016C$_H$	FF44 0010$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PNOTn	A:	FF40 0754$_H$	FF40 0764$_H$	FF40 076C$_H$	FF44 0070$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PPRn	A:	FF40 0254$_H$	FF40 0264$_H$	FF40 026C$_H$	FF44 0020$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111

Table 2-34 V850E2/FK4-H port (groups 21, 25, 27, JP) control registers (2/3)

Register		Port group n =			
		21	25	27	JP
PMn	A:	FF40 0354$_H$	FF40 0364$_H$	FF40 036C$_H$	FF44 0030$_H$
	I:	FFFF$_H$	FFFF$_H$	00FF$_H$	FF$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PMCn	A:	FF40 0454$_H$	FF40 0464$_H$	FF40 046C$_H$	FF44 0040$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PFCn	A:	FF40 0554$_H$	FF40 0564$_H$	FF40 056C$_H$	FF44 0050$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx1x x111	xxxx xxxx xx11 1111
PFCEn	A:	FF40 0654$_H$	FF40 0664$_H$	FF40 066C$_H$	–
	I:	0000$_H$	0000$_H$	0000$_H$	
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xxxx x111	
PMSRn	A:	FF40 0854$_H$	FF40 0864$_H$	FF40 086C$_H$	FF44 0080$_H$
	I:	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 00FF$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PMCSRn	A:	FF40 0954$_H$	FF40 0964$_H$	FF40 096C$_H$	FF44 0090$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PIBCn	A:	FF40 4054$_H$	FF40 4064$_H$	FF40 406C$_H$	FF44 0400$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PBDCn	A:	FF40 4154$_H$	FF40 4164$_H$	FF40 416C$_H$	FF44 0410$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PIPCn	A:	FF40 4254$_H$	FF40 4264$_H$	FF40 426C$_H$	FF44 4200$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PUn	A:	FF40 4354$_H$	FF40 4364$_H$	FF40 436C$_H$	FF44 0430$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PDn	A:	FF40 4454$_H$	FF40 4464$_H$	FF40 446C$_H$	FF44 0440$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	00$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PODCn	A:	FF40 4554$_H$	FF40 4564$_H$	FF40 456C$_H$	FF44 0450$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PDSCn	A:	FF40 4654$_H$	FF40 4664$_H$	FF40 466C$_H$	FF44 0460$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111

Table 2-34 V850E2/FK4-H port (groups 21, 25, 27, JP) control registers (3/3)

Register		Port group n =			
		21	25	27	JP
PISn	A:	FF40 4754_H	FF40 4764_H	FF40 $476C_H$	FF44 0470_H
	I:	0000_H	0000_H	0000_H	00_H
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PISEn	A:	FF40 4854_H	FF40 4864_H	FF40 $486C_H$	FF44 0480_H
	I:	0000_H	0000_H	0000_H	00_H
	B:	xxxx 1111 1111 11xx	1111 1111 1111 1111	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111
PPCMDn	A:	FF40 $4C54_H$	FF40 $4C64_H$	FF40 $4C6C_H$	FF44 $04C0_H$
	I:	00_H	00_H	00_H	00_H
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 $4B54_H$	FF40 $4B64_H$	FF40 $4B6C_H$	FF44 $04B0_H$
	I:	00_H	00_H	00_H	00_H
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

2.5.4 V850E2/FL4-H port functions

This section summarizes all port functions of the V850E2/FL4-H devices and its port control registers.

(1) General I/O functions

The table below shows all alternative functions, that can be applied to the V850E2/FL4-H ports.
It also gives the settings of the control bits PMCn_m, PFCn_m, PFCEn_m and PMn_m to the respective port into the different modes.

Table 2-35 V850E2/FL4-H general I/O functions (1/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group 0 (Always-On-Area, E0VDD/E0VSS power supply):								
P0_0	TAUJ1I0	TAUJ1O0	$\overline{\text{CSIG4SSI}}$		ADCA0TRG0		INTP0	
P0_1	TAUJ1I1	TAUJ1O1		CSIG4SO[a]	URTE2RX		INTP1	TAUA0O1
P0_2	TAUJ1I2	TAUJ1O2	CSIG4SI	RTCA0OUT	ADCA0TRG2	URTE2TX	INTP2	TAUA0O2
P0_3	TAUJ1I3	TAUJ1O3	CSIG4SC[a]		ADCA0TRG1		INTP3/ TAPA0ESO	
P0_4		FCN0TX					INTP11	
P0_5	FCN0RX						INTP12	
P0_6	FCN1RX	URTE11TX	KR0I1	CSIH2CSS1	NMI			
P0_7	URTE11RX	FCN1TX	KR0I2	CSIH2CSS2	INTP4			
P0_8	FCN2RX	URTE10TX	KR0I3	CSIH2CSS3	INTP5	TAUA0O5		IICB0SDA[b]
P0_9	URTE10RX	FCN2TX	KR0I4	CSIH2CSS4	INTP6	TAUA0O6		IICB0SCL[b]
P0_10	FCN3RX	URTE11TX			INTP9			
P0_11	URTE11RX	FCN3TX			INTP10			
P0_12	TAUJ0I0	TAUJ0O0	KR0I0		INTP8	FCN4TX	$\overline{\text{CSIG0SSI}}$	CSCXFOUT
P0_13	TAUJ0I1	TAUJ0O1	KR0I5	CSIH2CSS5	INTP7	FCN5TX	CSIG0SI	
P0_14	TAUJ0I2	TAUJ0O2	KR0I6	CSIH2CSS6	FCN5RX	TAUB1O13		CSIG0SO[a]
P0_15	TAUJ0I3	TAUJ0O3	KR0I7	CSIH2CSS7	FCN4RX	TAUB1O14	CSIG0SC[a]	
Port group 1 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P1_1	TAUA0I1	TAUA0O1		TAUC4O1	ENCA0AIN		FCN1RX	FCN0TX
P1_2	TAUA0I2	TAUA0O2		TAUC4O2	ENCA0BIN	TAPA0UP	CSIH2SI	FCN1TX
P1_3	TAUA0I3	TAUA0O3		TAUC4O5	ENCA0ZIN	TAPA0UN		CSIH2SO[a]
P1_4	TAUA0I4	TAUA0O4		TAUC4O6	ENCA0TIN0	TAPA0VP	CSIH2SC[a]	
P1_5	TAUA0I5	TAUA0O5		TAUC4O9	ENCA0TIN1	TAPA0VN	CSIH2RY	
P1_6	TAUA0I6	TAUA0O6		TAUC4O10	$\overline{\text{CSIH1SSI}}$	TAPA0WP	$\overline{\text{CSIH2SSI}}$	CSIH2CSS0
P1_7	TAUA0I7	TAUA0O7		TAUC4O13	CSIH1SI	TAPA0WN	FCN0RX	CSIH2CSS1
P1_8	TAUA0I8	TAUA0O8		TAUC4O14		CSIH1SO[a]	FCN2RX	URTE4TX

Table 2-35 V850E2/FL4-H general I/O functions (2/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P1_9	TAUA0I9	TAUA0O9	INTP3	FLX0TXENA	CSIH1SC[a]		URTE4RX	FCN2TX
P1_10	TAUA0I10	TAUA0O10	FLX0RXDA	URTE3TX	CSIH1RY		INTP4	
P1_11	TAUA0I11	TAUA0O11	URTE3RX	FLX0TXDA		CSIH1CSS0	INTP5	
P1_12	TAUA0I12	TAUA0O12	FLX0RXDB	URTE4TX		CSIH1CSS1	INTP6	
P1_13	TAUA0I13	TAUA0O13	URTE4RX	FLX0TXDB		CSIH1CSS2	INTP7	
P1_14	TAUA0I14	TAUA0O14	FLX0STPWT	URTE5TX		CSIH1CSS3	INTP8	
P1_15	TAUA0I15	TAUA0O15	URTE5RX	FLX0TXENB		CSIH1CSS4	INTP9	
Port group 2 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P2_0			FCN4RX	URTE6TX		CSIH1CSS5	INTP10	
P2_1			URTE6RX	URTE7TX	INTP0	CSIH1CSS6	TAUJ1I3	TAUJ1O3
P2_2			URTE7RX	FCN4TX	INTP1	CSIH1CSS7	TAUJ1I2	TAUJ1O2
P2_3								
Port group 3 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P3_0	TAUJ1I1	TAUJ1O1	TAUA0I0	TAUA0O0				
P3_1	TAUB2I1	TAUB2O1	TAUA0I1	TAUA0O1				
P3_2	TAUB2I2	TAUB2O2	TAUA0I2	TAUA0O2	KR0I7			
P3_3	TAUB2I3	TAUB2O3	TAUA0I3	TAUA0O3	KR0I6		CSIG2RY	
P3_4	TAUB2I5	TAUB2O5	TAUA0I4	TAUA0O4	KR0I5		CSIG0RY	
P3_5	TAUB2I6	TAUB2O6	TAUA0I5	TAUA0O5	KR0I4		CSIG0SC[a]	
P3_6	TAUB2I7	TAUB2O7	TAUA0I6	TAUA0O6				CSIG0SO[a]
P3_7	TAUB2I9	TAUB2O9	TAUA0I7	TAUA0O7	URTE7RX		CSIG0SI	URTE3TX
P3_8	TAUB2I10	TAUB2O10	TAUA0I8	TAUA0O8	INTP11	URTE6TX	URTE3RX	
P3_9	TAUB2I11	TAUB2O11	TAUA0I9	TAUA0O9	URTE5RX	FCN5TX	INTP12	URTE3TX
P3_10	TAUB2I13	TAUB2O13	TAUA0I10	TAUA0O10	FCN5RX	URTE5TX		
P3_11	TAUB2I14	TAUB2O14	TAUA0I11	TAUA0O11	URTE6RX		INTP13	
P3_12	TAUB2I15	TAUB2O15	TAUA0I12	TAUA0O12	URTE7RX	URTE6TX	INTP14	
Port group 4 (Isolated-Area-0, E1VDD/E1VSS power supply):								
P4_0	TAUB1I1	TAUB1O1	TAUA0I13	TAUA0O13	CSIH0SI	URTE7TX		
P4_1	TAUB1I2	TAUB1O2	TAUA0I14	TAUA0O14		CSIH0SO[a]	URTE2RX	FCN3TX
P4_2	TAUB1I3	TAUB1O3	TAUA0I15	TAUA0O15	CSIH0SC[a]		FCN3RX	URTE2TX
P4_3	TAUB1I5	TAUB1O5	CSIG0SI	URTE10TX	CSIH0RY		INTP10	
P4_4	INTP2	TAUB1O6	URTE10RX	CSIG0SO[a]	$\overline{\text{CSIH0SSI}}$	CSIH0CSS0	ENCA0TIN0	
P4_5	TAUB1I7	TAUB1O7	CSIG0SC[a]		KR0I3	CSIH0CSS1	ENCA0TIN1	
P4_6	TAUB1I9	TAUB1O9	CSIG4SI	URTE11TX	KR0I2	CSIH0CSS2	ENCA0AIN	
P4_7	INTP4	TAUB1O10	URTE11RX	CSIG4SO[a]	KR0I1	CSIH0CSS3	ENCA0BIN	
P4_8	TAUB1I11	TAUB1O11	CSIG4SC[a]		KR0I0	CSIH0CSS4	ENCA0ZIN	

Table 2-35 V850E2/FL4-H general I/O functions (3/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P4_9	TAUB1I13	TAUB1O13	CSIG2SI	CSIG0RY		CSIH0CSS5		URTE8TX
P4_10	TAUB1I14	TAUB1O14	CSIG4RY	CSIG2SO	INTP15	CSIH0CSS6	URTE8RX	
P4_11	TAUB1I15	TAUB1O15	CSIG2SC[a]			CSIH0CSS7		
Port group 10 (Isolated-Area-0, A0VDD/A0VSS power supply):[c]								
P10_6								
P10_7								
P10_8								
P10_9	ADCA0TRG0							
P10_10	ADCA0TRG1							
P10_11	ADCA0TRG2							
P10_12								
P10_13								
P10_14								
P10_15								
Port group 11 (Isolated-Area-0, A0VDD/A0VSS power supply):[c]								
P11_0								
P11_1								
P11_2								
P11_3								
P11_4								
P11_5								
P11_6								
P11_7								
Port group 12 (Isolated-Area-1, A1VDD/A1VSS power supply):[d]								
P12_0								
P12_1								
P12_2								
P12_3								
P12_4								
P12_5								
P12_6								
P12_7								
P12_8								
P12_9	ADCA1TRG0							
P12_10	ADCA1TRG1							
P12_11	ADCA1TRG2							

Table 2-35 V850E2/FL4-H general I/O functions (4/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P12_12								
P12_13								
P12_14								
P12_15								
Port group 13 (Isolated-Area-1, A1VDD/A1VSS power supply):[d]								
P13_0								
P13_1								
P13_2								
P13_3								
P13_4								
P13_5								
P13_6								
P13_7								
Port group 21 (Isolated-Area-1, B0VDD/VSS power supply):								
P21_0			ETH0COL					
P21_1			ETH0TXCLK					
P21_2	INTP10	$\overline{\text{MEMC0BEN1}}$	CSIH2SI			IICB0SDA[b]	TAUB2I13	TAUB2O13
P21_3	INTP11	$\overline{\text{MEMC0BEN0}}$		CSIH2SO[a]		IICB0SCL[b]	TAUB2I14	TAUB2O14
P21_4	INTP12	$\overline{\text{MEMC0WR}}$	CSIH2SC[a]					TAUC3O1
P21_5	INTP13	$\overline{\text{MEMC0RD}}$	CSIH2RY			URTE9TX		TAUC3O2
P21_6	INTP14	MEMC0CLK	$\overline{\text{CSIH2SSI}}$	CSIH2CSS0	URTE9RX			TAUC3O5
P21_7	$\overline{\text{MEMC0WAIT}}$			CSIH2CSS1		URTE0TX		TAUC3O6
P21_8	INTP15			CSIH2CSS2	URTE0RX	PMCA0 MSEL0		TAUC3O9
P21_9		$\overline{\text{MEMC0CS2}}$[a]		CSIH2CSS3		PMCA0 MSEL1		TAUC3O10
P21_10		$\overline{\text{MEMC0CS3}}$[a]		CSIH2CSS4	URTE1RX	PMCA0 MSEL2		TAUC3O13
P21_11		$\overline{\text{MEMC0CS4}}$[a]		CSIH2CSS5		URTE1TX		TAUC3O14
Port group 24 (Isolated-Area-1, B0VDD/VSS power supply):								
P24_0						URTE8TX		
P24_1			INTP8		URTE8RX			
P24_2					FCN5RX	URTE9TX		
P24_3			INTP9		URTE9RX	FCN5TX		
P24_4					FCN4RX	URTE0TX		
P24_5			INTP10		URTE0RX	FCN4TX		
P24_6						URTE1TX		

Table 2-35 V850E2/FL4-H general I/O functions (5/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
P24_7			INTP11		URTE1RX			
P24_8								TAUC7O1
P24_9								TAUC7O2
P24_10								TAUC7O5
P24_11								TAUC7O6
P24_12								TAUC7O9
P24_13								TAUC7O10
P24_14			ETH0COL					TAUC7O13
P24_15			ETH0TXCLK					TAUC7O14
Port group 25 (Isolated-Area-1, B0VDD/VSS power supply):								
P25_0	MEMC0AD0[a]		INTP6	ETH0TXD3	URTE6RX	CSIG2SO[a]		TAUC6O1
P25_1	MEMC0AD1[a]			ETH0TXD2	CSIG2SI	URTE6TX		TAUC6O2
P25_2	MEMC0AD2[a]			ETH0TXD1	CSIG2SC[a]			TAUC6O5
P25_3	MEMC0AD3[a]			ETH0TXD0	CSIG4SI	URTE7TX		TAUC6O6
P25_4	MEMC0AD4[a]		INTP7	ETH0TXEN	URTE7RX	CSIG4SO[a]		TAUC6O9
P25_5	MEMC0AD5[a]		ETH0 REFCLK	ETH0TXER	CSIG4SC[a]			TAUC6O10
P25_6	MEMC0AD6[a]		ETH0CRSDV					TAUC6O13
P25_7	MEMC0AD7[a]		ETH0RXER					TAUC6O14
P25_8	MEMC0AD8[a]		ETH0RXD0					TAUC5O1
P25_9	MEMC0AD9[a]		ETH0RXD1					TAUC5O2
P25_10	MEMC0AD10[a]		ETH0RXD2	CSIH2CSS6				TAUC5O5
P25_11	MEMC0AD11[a]		ETH0RXD3	CSIH2CSS7				TAUC5O6
P25_12	MEMC0AD12[a]		ETH0RXDV			IICB0SDA[b]		TAUC5O9
P25_13	MEMC0AD13[a]		ETH0RXCLK			IICB0SCL[b]		TAUC5O10
P25_14	MEMC0AD14[a]		INTP5	ETH0MDC	URTE5RX	FCN3TX		TAUC5O13
P25_15	MEMC0AD15[a]		ETH0MDI[a]	ETH0MDO[a]	FCN3RX	URTE5TX		TAUC5O14
Port group 27 (Isolated-Area-1, B0VDD/VSS power supply):								
P27_0	INTP0	MEMC0A16[a]			ADCA1TRG2	PMCA0 MSEL0		
P27_1	INTP1	MEMC0A17[a]	ETH0COL		ADCA1TRG1	PMCA0 MSEL1		
P27_2	INTP2	MEMC0A18[a]	ETH0TXCLK		ADCA1TRG0	PMCA0 MSEL2		
P27_4	INTP4							
P27_5	INTP5			$\overline{\text{MEMC0ASTB}}$[a]				

Table 2-35 V850E2/FL4-H general I/O functions (6/6)

Port mode	Alternative mode							
PMCn_m = 0	PMCn_m = 1							
	PFCEn_m = 0, PFCn_m = 0		PFCEn_m = 0, PFCn_m = 1		PFCEn_m = 1, PFCn_m = 0		PFCEn_m = 1, PFCn_m = 1	
	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0	PMn_m = 1	PMn_m = 0
	ALT_IN1	ALT_OUT1	ALT_IN2	ALT_OUT2	ALT_IN3	ALT_OUT3	ALT_IN4	ALT_OUT4
Port group JP0 (Always-On-Area, E0VDD/E0VSS power supply):								
JP0_0	INTP0	VCPC1OUT	TAUJ0I0	TAUJ0O0				
JP0_1	INTP1	VCPC0OUT	TAUJ0I1	TAUJ0O1				
JP0_2	INTP2	CSCXFOUT	TAUJ0I2	TAUJ0O2				
JP0_3	INTP3		TAUJ0I3	TAUJ0O3				
JP0_4								
JP0_5	NMI	RTCA0OUT						

a) When using this alternative mode, set PIPCn.PIPCn_m = 1. The module controls the I/O setting and PMn_m has no effect.

b) Refer to the section *"I²C Interface Port Settings"* in the "I²C Interface (IICB)" chapter for details about the correct configuration of the I²C Interface ports.

c) The ports of port groups 10 and 11 are also used as permanent inputs to the A/D Converter ADCA0. If the P10 or P11 ports shall be used as port inputs, set the ADCA0CTL1.ADCA0GPS = 1.

d) The ports of port groups 12 and 13 are also used as permanent inputs to the A/D Converter ADCA1. If the P12 or P13 ports shall be used as port inputs, set the ADCA1CTL1.ADCA1GPS = 1.

(2) V850E2/FL4-H port control registers

The following table summarizes all V850E2/FL4-H port control registers, their addresses and intitial values.

Tables legend

A: Register address

I: Initial value

B: Available bits
- 1: available, x: not available
- right: bit 0, left: bit 15

Table 2-36 V850E2/FL4-H port (groups 0 to 3) control registers (1/2)

Register		Port group n =			
		0	1	2	3
Pn	A:	FF40 0000$_H$	FF40 0004$_H$	FF40 0008$_H$	FF40 000C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PSRn	A:	FF40 0100$_H$	FF40 0104$_H$	FF40 0108$_H$	FF40 010C$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PNOTn	A:	FF40 0700$_H$	FF40 0704$_H$	FF40 0708$_H$	FF40 070C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PPRn	A:	FF40 0200$_H$	FF40 0204$_H$	FF40 0208$_H$	FF40 020C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PMn	A:	FF40 0300$_H$	FF40 0304$_H$	FF40 0308$_H$	FF40 030C$_H$
	I:	FFFE$_H$	FFFF$_H$	0007$_H$	0FFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PMCn	A:	FF40 0400$_H$	FF40 0404$_H$	FF40 0408$_H$	FF40 040C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxx1 1111 1111 1111
PFCn	A:	FF40 0500$_H$	FF40 0504$_H$	FF40 0508$_H$	FF40 050C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxx1 1111 1111 1111
PFCEn	A:	FF40 0600$_H$	FF40 0604$_H$	FF40 0608$_H$	FF40 060C$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxx1 1111 1111 1111
PMSRn	A:	FF40 0800$_H$	FF40 0804$_H$	FF40 0808$_H$	FF40 080C$_H$
	I:	0000 FFFE$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PMCSRn	A:	FF40 0900$_H$	FF40 0904$_H$	FF40 0908$_H$	FF40 090C$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx x111	xxx1 1111 1111 1111

Table 2-36 V850E2/FL4-H port (groups 0 to 3) control registers (2/2)

Register		Port group n =			
		0	1	2	3
PIBCn	A:	FF40 4000_H	FF40 4004_H	FF40 4008_H	FF40 $400C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PBDCn	A:	FF40 4100_H	FF40 4104_H	FF40 4108_H	FF40 $410C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PIPCn	A:	FF40 4200_H	FF40 4204_H	FF40 4208_H	FF40 $420C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PUn	A:	FF40 4300_H	FF40 4304_H	FF40 4308_H	FF40 $430C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx xv111	xxx1 1111 1111 1111
PDn	A:	FF40 4400_H	FF40 4404_H	FF40 4408_H	FF40 $440C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PODCn	A:	FF40 4500_H	FF40 4504_H	FF40 4508_H	FF40 $450C_H$
	I:	0000 0001_H	0000 0000_H	0000 0000_H	0000 0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PDSCn	A:	FF40 4600_H	FF40 4604_H	FF40 4608_H	FF40 $460C_H$
	I:	0000 0000_H	0000 0000_H	0000 0000_H	0000 0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx xv111	xxx1 1111 1111 1111
PISn	A:	FF40 4700_H	FF40 4704_H	FF40 4708_H	FF40 $470C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PISEn	A:	FF40 4800_H	FF40 4804_H	FF40 4808_H	FF40 $480C_H$
	I:	0000_H	0000_H	0000_H	0000_H
	B:	1111 1111 1111 1111	1111 1111 1111 111x	xxxx xxxx xxxx 1111	xxx1 1111 1111 1111
PPCMDn	A:	FF40 $4C00_H$	FF40 $4C04_H$	FF40 $4C08_H$	FF40 $4C0C_H$
	I:	00_H	00_H	00_H	00_H
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 $4B00_H$	FF40 $4B04_H$	FF40 $4B08_H$	FF40 $4B0C_H$
	I:	00_H	00_H	00_H	00_H
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-37 V850E2/FL4-H port (groups 4, 10 to 12) control registers (1/2)

Register		Port group n =			
		4	10	11	12
Pn	A:	FF40 0010$_H$	FF40 0028$_H$	FF40 002C$_H$	FF40 0030$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 1111 1111	xxxx xxxx 1111 1111	1111 1111 1111 1111
PSRn	A:	FF40 0110$_H$	FF40 0128$_H$	FF40 012C$_H$	FF40 0130$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PNOTn	A:	FF40 0710$_H$	FF40 0728$_H$	FF40 072C$_H$	FF40 0730$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PPRn	A:	FF40 0210$_H$	FF40 0228$_H$	FF40 022C$_H$	FF40 0230$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMn	A:	FF40 0310$_H$	FF40 0328$_H$	FF40 032C$_H$	FF40 0330$_H$
	I:	0FFF$_H$	FF40$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCn	A:	FF40 0410$_H$	FF40 0428$_H$	–	FF40 0304$_H$
	I:	0000$_H$	0000$_H$		0000$_H$
	B:	xxxx 1111 1111 1111	xxxx 111x xxxx xxxx		xxxx 111x xxxx xxxx
PFCn	A:	FF40 0510$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PFCEn	A:	FF40 0610$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PMSRn	A:	FF40 0810$_H$	FF40 0828$_H$	FF40 082C$_H$	FF40 0830$_H$
	I:	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PMCSRn	A:	FF40 0910$_H$	FF40 0928$_H$		FF40 0930$_H$
	I:	0000 0000$_H$	0000 0000$_H$		0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx		xxxx 111x xxxx xxxx
PIBCn	A:	FF40 4010$_H$	FF40 4028$_H$	FF40 402C$_H$	FF40 4030$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PBDCn	A:	FF40 4110$_H$	FF40 4128$_H$	FF40 412C$_H$	FF40 4130$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PIPCn	A:	FF40 4210$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			

Table 2-37 V850E2/FL4-H port (groups 4, 10 to 12) control registers (2/2)

Register		Port group n =			
		4	10	11	12
PUn	A:	FF40 4310$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PDn	A:	FF40 4410$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PODCn	A:	FF40 4510$_H$	FF40 4528$_H$	FF40 452C$_H$	FF40 4530$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx 1111 1111 1111	1111 1111 11xx xxxx	xxxx xxxx 1111 1111	1111 1111 1111 1111
PDSCn	A:	FF40 4610$_H$	–	–	–
	I:	0000 0000$_H$			
	B:	xxxx 1111 1111 1111			
PISn	A:	FF40 4710$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PISEn	A:	FF40 4810$_H$	–	–	–
	I:	0000$_H$			
	B:	xxxx 1111 1111 1111			
PPCMDn	A:	FF40 4C10$_H$	FF40 4C28$_H$	FF40 4C2C$_H$	FF40 4C30$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B10$_H$	FF40 4B28$_H$	FF40 4B2C$_H$	FF40 4B30$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-38 V850E2/FL4-H port (groups 13, 21, 24, 25) control registers (1/3)

Register		Port group n =			
		13	21	24	25
Pn	A:	FF40 0034$_H$	FF40 0054$_H$	FF40 0060$_H$	FF40 0064$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PSRn	A:	FF40 0134$_H$	FF40 0154$_H$	FF40 0160$_H$	FF40 0164$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PNOTn	A:	FF40 0734$_H$	FF40 0754$_H$	FF40 0760$_H$	FF40 0764$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PPRn	A:	FF40 0234$_H$	FF40 0254$_H$	FF40 0260$_H$	FF40 0264$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111

Table 2-38 V850E2/FL4-H port (groups 13, 21, 24, 25) control registers (2/3)

Register		Port group n =			
		13	21	24	25
PMn	A:	FF40 0334$_H$	FF40 0354$_H$	FF40 0360$_H$	FF40 0364$_H$
	I:	00FF$_H$	FFFF$_H$	FFFF$_H$	FFFF$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMCn	A:	_	FF40 0454$_H$	FF40 0460$_H$	FF40 0464$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PFCn	A:	_	FF40 0554$_H$	FF40 0560$_H$	FF40 0564$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PFCEn	A:	_	FF40 0654$_H$	FF40 0660$_H$	FF40 0664$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 11xx	1111 1111 1111 1111	1111 1111 1111 1111
PMSRn	A:	FF40 0834$_H$	FF40 0854$_H$	FF40 0860$_H$	FF40 0864$_H$
	I:	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$	0000 FFFF$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PMCSRn	A:		FF40 0954$_H$	FF40 0960$_H$	FF40 0964$_H$
	I:		0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PIBCn	A:	FF40 4034$_H$	FF40 4054$_H$	FF40 4060$_H$	FF40 4064$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PBDCn	A:	FF40 4134$_H$	FF40 4154$_H$	FF40 4160$_H$	FF40 4164$_H$
	I:	0000$_H$	0000$_H$	0000$_H$	0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PIPCn	A:	_	FF40 4254$_H$	FF40 4260$_H$	FF40 4264$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PUn	A:	_	FF40 4354$_H$	FF40 4360$_H$	FF40 4364$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PDn	A:	_	FF40 4454$_H$	FF40 44600$_H$	FF40 4464$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PODCn	A:	FF40 4534$_H$	FF40 4554$_H$	FF40 4560$_H$	FF40 4564$_H$
	I:	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:	xxxx xxxx 1111 1111	xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PDSCn	A:	_	FF40 4654$_H$	FF40 4660$_H$	FF40 4664$_H$
	I:		0000 0000$_H$	0000 0000$_H$	0000 0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111

Table 2-38 V850E2/FL4-H port (groups 13, 21, 24, 25) control registers (3/3)

Register		Port group n =			
		13	21	24	25
PISn	A:	_	FF40 4754$_H$	FF40 4760$_H$	FF40 4764$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PISEn	A:	_	FF40 4854$_H$	FF40 4860$_H$	FF40 4864$_H$
	I:		0000$_H$	0000$_H$	0000$_H$
	B:		xxxx 1111 1111 1111	1111 1111 1111 1111	1111 1111 1111 1111
PPCMDn	A:	FF40 4C34$_H$	FF40 4C54$_H$	FF40 4C60$_H$	FF40 4C64$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	1111 1111	1111 1111	1111 1111	1111 1111
PPROTSn	A:	FF40 4B34$_H$	FF40 4B54$_H$	FF40 4B60$_H$	FF40 4B64$_H$
	I:	00$_H$	00$_H$	00$_H$	00$_H$
	B:	xxxx xxx1	xxxx xxx1	xxxx xxx1	xxxx xxx1

Table 2-39 V850E2/FL4-H port (groups 27, JP) control registers (1/2)

Register		Port group n =			
		27	JP		
Pn	A:	FF40 006C$_H$	FF44 0000$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PSRn	A:	FF40 016C$_H$	FF44 0010$_H$		
	I:	0000 0000$_H$	0000 0000$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PNOTn	A:	FF40 076C$_H$	FF44 0070$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PPRn	A:	FF40 026C$_H$	FF44 0020$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PMn	A:	FF40 036C$_H$	FF44 0030$_H$		
	I:	FFFF$_H$	FF$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PMCn	A:	FF40 046C$_H$	FF44 0040$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PFCn	A:	FF40 056C$_H$	FF44 0050$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx1x x111	xxxx xxxx xx11 1111		
PFCEn	A:	FF40 066C$_H$	_		
	I:	0000$_H$			
	B:	xxxx xxxx xxxx x111			

Table 2-39 V850E2/FL4-H port (groups 27, JP) control registers (2/2)

Register		Port group n =			
		27	JP		
PMSRn	A:	FF40 086C$_H$	FF44 0080$_H$		
	I:	0000 FFFF$_H$	0000 00FF$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PMCSRn	A:	FF40 096C$_H$	FF44 0090$_H$		
	I:	0000 0000$_H$	0000 0000$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PIBCn	A:	FF40 406C$_H$	FF44 0400$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PBDCn	A:	FF40 416C$_H$	FF44 0410$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PIPCn	A:	FF40 426C$_H$	FF44 4200$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PUn	A:	FF40 436C$_H$	FF44 0430$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PDn	A:	FF40 446C$_H$	FF44 0440$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PODCn	A:	FF40 456C$_H$	FF44 0450$_H$		
	I:	0000 0000$_H$	0000 0000$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PDSCn	A:	FF40 466C$_H$	FF44 0460$_H$		
	I:	0000 0000$_H$	0000 0000$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PISn	A:	FF40 476C$_H$	FF44 0470$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PISEn	A:	FF40 486C$_H$	FF44 0480$_H$		
	I:	0000$_H$	00$_H$		
	B:	xxxx xxxx xx11 x111	xxxx xxxx xx11 1111		
PPCMDn	A:	FF40 4C6C$_H$	FF44 04C0$_H$		
	I:	00$_H$	00$_H$		
	B:	1111 1111	1111 1111		
PPROTSn	A:	FF40 4B6C$_H$	FF44 04B0$_H$		
	I:	00$_H$	00$_H$		
	B:	xxxx xxx1	xxxx xxx1		

2.5.5 Non-port input/output signals

Following signals are input/output via pins without port functionality. Thus no port control registers are available for these pins:

Table 2-40 General pin functions

Mode signals	• FLMD0 • $\overline{\text{RESET}}$
Oscillator/clock connections	• X1, X2 • XT1, XT2
Voltage Comparators	• VCPC0IN, VCPC1IN
ADCA0 simultaneous sampling channels	• ADCA0I0 to ADCA0I5
Power Sequencer	• WAKE • PWGD
Power supply	• all power supply and voltage reference signals • PTCTL1

2.5.6 Alphabetic pin function list

The following table lists all pin signals in alphabetic order nad the ports, they can be assigned to.

Note The table shows all V850E2/Fx4-H signals and does not note the availability of a signal on a certain device.

Table 2-41 Alphabetic pin function list (1/11)

Pin name	I/O	Pin function	Port
ADCA0I0	I	A/D Converter 0 input channel 0	P10_0
ADCA0I1	I	A/D Converter 0 input channel 1	P10_1
ADCA0I2	I	A/D Converter 0 input channel 2	P10_2
ADCA0I3	I	A/D Converter 0 input channel 3	P10_3
ADCA0I4	I	A/D Converter 0 input channel 4	P10_4
ADCA0I5	I	A/D Converter 0 input channel 5	P10_5
ADCA0I6	I	A/D Converter 0 input channel 6	P10_6
ADCA0I7	I	A/D Converter 0 input channel 7	P10_7
ADCA0I8	I	A/D Converter 0 input channel 8	P10_8
ADCA0I9	I	A/D Converter 0 input channel 9	P10_9
ADCA0I10	I	A/D Converter 0 input channel 10	P10_10
ADCA0I11	I	A/D Converter 0 input channel 11	P10_11
ADCA0I12	I	A/D Converter 0 input channel 12	P10_12
ADCA0I13	I	A/D Converter 0 input channel 13	P10_13
ADCA0I14	I	A/D Converter 0 input channel 14	P10_14
ADCA0I15	I	A/D Converter 0 input channel 15	P10_15
ADCA0I16	I	A/D Converter 0 input channel 16	P11_0
ADCA0I17	I	A/D Converter 0 input channel 17	P11_1
ADCA0I18	I	A/D Converter 0 input channel 18	P11_2
ADCA0I19	I	A/D Converter 0 input channel 19	P11_3
ADCA0I20	I	A/D Converter 0 input channel 20	P11_4
ADCA0I21	I	A/D Converter 0 input channel 21	P11_5
ADCA0I22	I	A/D Converter 0 input channel 22	P11_6
ADCA0I23	I	A/D Converter 0 input channel 23	P11_7
ADCA0TRG0	I	A/D Converter 0 trigger 0	P0_0, P10_9
ADCA0TRG1	I	A/D Converter 0 trigger 1	P0_3, P10_10
ADCA0TRG2	I	A/D Converter 0 trigger 2	P0_2, P10_11
ADCA1I0	I	A/D Converter 1 input channel 0	P12_0
ADCA1I1	I	A/D Converter 1 input channel 1	P12_1
ADCA1I2	I	A/D Converter 1 input channel 2	P12_2
ADCA1I3	I	A/D Converter 1 input channel 3	P12_3
ADCA1I4	I	A/D Converter 1 input channel 4	P12_4
ADCA1I5	I	A/D Converter 1 input channel 5	P12_5
ADCA1I6	I	A/D Converter 1 input channel 6	P12_6
ADCA1I7	I	A/D Converter 1 input channel 7	P12_7

Table 2-41 Alphabetic pin function list (2/11)

Pin name	I/O	Pin function	Port
ADCA1I8	I	A/D Converter 1 input channel 8	P12_8
ADCA1I9	I	A/D Converter 1 input channel 9	P12_9
ADCA1I10	I	A/D Converter 1 input channel 10	P12_10
ADCA1I11	I	A/D Converter 1 input channel 11	P12_11
ADCA1I12	I	A/D Converter 1 input channel 12	P12_12
ADCA1I13	I	A/D Converter 1 input channel 13	P12_13
ADCA1I14	I	A/D Converter 1 input channel 14	P12_14
ADCA1I15	I	A/D Converter 1 input channel 15	P12_15
ADCA1I16	I	A/D Converter 1 input channel 16	P13_0
ADCA1I17	I	A/D Converter 1 input channel 17	P13_1
ADCA1I18	I	A/D Converter 1 input channel 18	P13_2
ADCA1I19	I	A/D Converter 1 input channel 19	P13_3
ADCA1I20	I	A/D Converter 1 input channel 20	P13_4
ADCA1I21	I	A/D Converter 1 input channel 21	P13_5
ADCA1I22	I	A/D Converter 1 input channel 22	P13_6
ADCA1I23	I	A/D Converter 1 input channel 23	P13_7
ADCA1TRG0	I	A/D Converter 1 trigger 0	P12_9, P27_0
ADCA1TRG1	I	A/D Converter 1 trigger 1	P12_10, P27_1
ADCA1TRG2	I	A/D Converter 1 trigger 2	P12_11, P27_2
AnVDD	–	A/D Converter 0 voltage supply	–
AnVREFM	–	A/D Converter 0 negative reference voltage	–
AnVREFP	–	A/D Converter 0 positive reference voltage	–
AnVSS	–	A/D Converter 0 ground	–
BnVDD	–	Port buffer voltage supply	–
BnVSS	–	Port buffer ground	–
CSCXFOUT	O	Clock Controller FOUT	JP0_2
CSIG0RY	I/O	Clocked Serial I/F G 0 handshake signal	P3_4
	O		P4_9
CSIG0SC	I/O	Clocked Serial I/F G 0 data clock	P0_15, P_3_5, P4_5
CSIG0SI	I	Clocked Serial I/F G 0 serial data input	P0_13, P3_7, P4_3
CSIG0SO	O	Clocked Serial I/F G 0 serial data output	P0_14, P3_6, P4_4
$\overline{\text{CSIG0SSI}}$	I	Clocked Serial I/F G 0 slave select input signal	P0_12
CSIG2RY	I/O	Clocked Serial I/F G 2 handshake signal	P3_3
CSIG2SC	I/O	Clocked Serial I/F G 2 data clock	P3_2, P4_11, P25_2
CSIG2SI	I	Clocked Serial I/F G 2 serial data input	P3_0, P4_9, P25_1
CSIG2SO	O	Clocked Serial I/F G 2 serial data output	P3_1, P4_10, P25_0
CSIG4RY	I	Clocked Serial I/F G 4 handshake signal	P4_10
CSIG4SC	I/O	Clocked Serial I/F G 4 data clock	P0_3, P4_8, P3_5, P25_5
CSIG4SI	I	Clocked Serial I/F G 4 serial data input	P0_2, P4_6, P3_7, P25_3
CSIG4SO	O	Clocked Serial I/F G 4 serial data output	P0_1, P4_7, P3_6, P25_4
$\overline{\text{CSIG4SSI}}$	I	Clocked Serial I/F G 0 slave select input signal	P0_0

Table 2-41 Alphabetic pin function list (3/11)

Pin name	I/O	Pin function	Port
CSIH0RY	I/O	Clocked Serial I/F H 0 handshake signal	P4_3
CSIH0SC	I/O	Clocked Serial I/F H 0 data clock	P4_2
CSIH0SI	I	Clocked Serial I/F H 0 serial data input	P4_0
CSIH0SO	O	Clocked Serial I/F H 0 serial data output	P4_1
CSIH0SS0	O	Clocked Serial I/F H 0 chip select output 0	P4_4
CSIH0SS1	O	Clocked Serial I/F H 0 chip select output 1	P4_5
CSIH0SS2	O	Clocked Serial I/F H 0 chip select output 2	P4_6
CSIH0SS3	O	Clocked Serial I/F H 0 chip select output 3	P4_7
CSIH0SS4	O	Clocked Serial I/F H 0 chip select output 4	P4_8
CSIH0SS5	O	Clocked Serial I/F H 0 chip select output 5	P4_9
CSIH0SS6	O	Clocked Serial I/F H 0 chip select output 6	P4_10
CSIH0SS7	O	Clocked Serial I/F H 0 chip select output 7	P4_11
$\overline{\text{CSIH0SSI}}$	I	Clocked Serial I/F H 0 slave select input signal	P4_4
CSIH1RY	I/O	Clocked Serial I/F H 1 handshake signal	P1_10
CSIH1SC	I/O	Clocked Serial I/F H 1 data clock	P1_9
CSIH1SI	I	Clocked Serial I/F H 1 serial data input	P1_7
CSIH1SO	O	Clocked Serial I/F H 1 serial data output	P1_8
CSIH1SS0	O	Clocked Serial I/F H 1 chip select output 0	P1_11
CSIH1SS1	O	Clocked Serial I/F H 1 chip select output 1	P1_12
CSIH1SS2	O	Clocked Serial I/F H 1 chip select output 2	P1_13
CSIH1SS3	O	Clocked Serial I/F H 1 chip select output 3	P1_14
CSIH1SS4	O	Clocked Serial I/F H 1 chip select output 4	P1_15
CSIH1SS5	O	Clocked Serial I/F H 1 chip select output 5	P2_0
CSIH1SS6	O	Clocked Serial I/F H 1 chip select output 6	P2_1
CSIH1SS7	O	Clocked Serial I/F H 1 chip select output 7	P2_2
$\overline{\text{CSIH1SSI}}$	I	Clocked Serial I/F H 1 slave select input signal	P1_6
CSIH2RY	I/O	Clocked Serial I/F H 2 handshake signal	P1_5, P21_5
CSIH2SC	I/O	Clocked Serial I/F H 2 data clock	P1_4, P21_4
CSIH2SI	I	Clocked Serial I/F H 2 serial data input	P1_2, P21_2
CSIH2SO	O	Clocked Serial I/F H 2 serial data output	P1_3, P21_3
CSIH2SS0	O	Clocked Serial I/F H 2 chip select output 0	P1_6, P21_6
CSIH2SS1	O	Clocked Serial I/F H 2 chip select output 1	P1_7, P0_6, P21_7
CSIH2SS2	O	Clocked Serial I/F H 2 chip select output 2	P0_7, P21_8
CSIH2SS3	O	Clocked Serial I/F H 2 chip select output 3	P0_8, P21_9
CSIH2SS4	O	Clocked Serial I/F H 2 chip select output 4	P0_9, P21_10
CSIH2SS5	O	Clocked Serial I/F H 2 chip select output 5	P0_13, P21_11
CSIH2SS6	O	Clocked Serial I/F H 2 chip select output 6	P0_14, P25_10
CSIH2SS7	O	Clocked Serial I/F H 2 chip select output 7	P0_15, P25_11
$\overline{\text{CSIH2SSI}}$	I	Clocked Serial I/F H 2 slave select input signal	P1_6, P21_6
$\overline{\text{DCUTRDY}}$	O	Debug I/F ready signal	JP0_5
DCUTCK	I	Debug I/F clock	JP0_2

Table 2-41 Alphabetic pin function list (4/11)

Pin name	I/O	Pin function	Port
DCUTDI	I	Debug I/F data input	JP0_0
DCUTDO	O	Debug I/F data output	JP0_1
DCUTMS	I	Debug I/F mode select	JP0_3
$\overline{\text{DCUTRST}}$	I	Debug I/F reset	JP0_4
ENCA0AIN	I	ENCA0 encoder input (phase A)	P1_1, P4_6
ENCA0BIN	I	ENCA0 encoder input (phase B)	P1_2, P4_7
ENCA0TIN0	I	ENCA0 capture trigger input 0	P1_4, P4_4
ENCA0TIN1	I	ENCA0 capture trigger input 1	P1_5, P4_5
ENCA0ZIN	I	ENCA0 encoder input (phase Z)	P1_3, P4_8
EnVDD	–	Port buffer voltage supply	–
EnVSS	–	Port buffer ground	–
ETH0COL	I	Collision detection	P21_0, P24_14
ETH0MDC			P25_14
ETH0MDI			P25_15
ETH0MDO			P25_15
ETH0REFCLK			P25_5
ETH0CRSDV	I	Carrier detection	P25_6
ETH0RXCLK	I	Reception clock	P25_13
ETH0RXD0	I	Reception data 0	P25_8
ETH0RXD1	I	Reception data 1	P25_9
ETH0RXD2	I	Reception data 2	P25_10
ETH0RXD3	I	Reception data 3	P25_11
ETH0RXDV	I	Reception data valid	P25_12
ETH0RXER	I	Reception error	P25_7
ETH0TXCLK	O	Transmission clock	P21_1, P24_15
ETH0TXD0	O	Transmission data 0	P25_3
ETH0TXD1	O	Transmission data 1	P25_2
ETH0TXD2	O	Transmission data 2	P25_1
ETH0TXD3	O	Transmission data 3	P25_0
ETH0TXEN	O	Transmission data enable	P25_4
ETH0TXER	O	Transmission error	P25_5
FCN0RX	I	CAN I/F 0 receive input	P0_5, P1_7
FCN0TX	O	CAN I/F 0 transmit output	P0_4, P1_1
FCN1RX	I	CAN I/F 1 receive input	P0_6, P1_1
FCN1TX	O	CAN I/F 1 transmit output	P0_7, P1_2
FCN2RX	I	CAN I/F 2 receive input	P0_8, P1_8
FCN2TX	O	CAN I/F 2 transmit output	P0_9, P1_9
FCN3RX	I	CAN I/F 3 receive input	P4_2, P0_10, P25_15
FCN3TX	O	CAN I/F 3 transmit output	P4_1, P0_11, P25_14
FCN4RX	I	CAN I/F 4 receive input	P0_15, P2_0, P24_4, P29_0

Table 2-41 Alphabetic pin function list (5/11)

Pin name	I/O	Pin function	Port
FCN4TX	O	CAN I/F 4 transmit output	P0_12, P2_2, P24_5, P29_1
FCN5RX	I	CAN I/F 5 receive input	P0_14, P3_10, P24_2
FCN5TX	O	CAN I/F 5 transmit output	P0_13, P3_9, P24_3
FLCS0SCI	I	Flash programmer synchroneous I/F clock input	JP0_2
FLCS0SI	I	Flash programmer synchroneous I/F data input	JP0_0
FLCS0SO	O	Flash programmer synchroneous I/F clock output	JP0_1
FLMD0	I	Primary operating mode select pin	–
FLMD1	I	Secondary operating mode select pin	P0_1
FLUR0RTX	I/O	Flash programmer asynchroneous I/F data input/output	JP0_0
FLX0RXDA	I	Flexray I/F 0 receiver input A	P1_10
FLX0RXDB	I	Flexray I/F 0 receiver input B	P1_12
FLX0STPWT	I	Flexray I/F 0 stop watch trigger	P1_14
FLX0TXDA	O	Flexray I/F 0 transmitter output A	P1_11
FLX0TXDB	O	Flexray I/F 0 transmitter output B	P1_13
FLX0TXENA	O	Flexray I/F 0 transmitter enable A	P1_9
FLX0TXENB	O	Flexray I/F 0 transmitter enable B	P1_15
FVDDn	–	Flash voltage supply	–
IICB0SCL	I/O	I^2C Interface 0 clock signal	P0_9, P21_3, P25_13
IICB0SDA	I/O	I^2C Interface 0 data/address signal	P0_8, P21_2, P25_12
INTP0	I	External interrupt input 0	P0_0, P2_1, JP0_0, P27_0
INTP1	I	External interrupt input 1	P0_1, P2_2, JP0_1, P27_1
INTP2	I	External interrupt input 2	P0_2, P4_4, JP0_2, P27_2
INTP3	I	External interrupt input 3	P0_3, P1_9, JP0_3, P27_3
INTP4	I	External interrupt input 4	P0_7 P1_10, P4_7, P27_4
INTP5	I	External interrupt input 5	P0_8, P1_11, P25_14, P27_5
INTP6	I	External interrupt input 6	P0_9, P1_12, P25_0, P28_0
INTP7	I	External interrupt input 7	P0_13, P1_13, P25_4, P28_1
INTP8	I	External interrupt input 8	P0_12, P1_14, P24_1, P28_2
INTP9	I	External interrupt input 9	P0_10, P1_15, P24_3, P28_3
INTP10	I	External interrupt input 10	P0_11, P2_0, P4_3, P21_2, P24_5
INTP11	I	External interrupt input 11	P3_8, P21_3, P24_7
INTP12	I	External interrupt input 12	P3_9, P21_4
INTP13	I	External interrupt input 13	P21_5
INTP14	I	External interrupt input 14	P21_6
INTP15	I	External interrupt input 15	P4_10, P21_8
KRI0	I	Key Return input 0	P0_12, P4_8
KRI1	I	Key Return input 1	P0_6, P4_7

Table 2-41 Alphabetic pin function list (6/11)

Pin name	I/O	Pin function	Port
KRI2	I	Key Return input 2	P0_7, P4_6
KRI3	I	Key Return input 3	P0_8, P4_5
KRI4	I	Key Return input 4	P0_9, P3_5
KRI5	I	Key Return input 5	P0_13, P3_4
KRI6	I	Key Return input 6	P0_14, P3_3
KRI7	I	Key Return input 7	P0_15, P3_2
MEMC0A16	0	External Memory I/F address 16	P27_0
MEMC0A17	0	External Memory I/F address 17	P27_1
MEMC0A18	0	External Memory I/F address 18	P27_2
MEMC0AD0	I/O	External Memory I/F data/address 0	P25_0
MEMC0AD1	I/O	External Memory I/F data/address 1	P25_1
MEMC0AD2	I/O	External Memory I/F data/address 2	P25_2
MEMC0AD3	I/O	External Memory I/F data/address 3	P25_3
MEMC0AD4	I/O	External Memory I/F data/address 4	P25_4
MEMC0AD5	I/O	External Memory I/F data/address 5	P25_5
MEMC0AD6	I/O	External Memory I/F data/address 6	P25_6
MEMC0AD7	I/O	External Memory I/F data/address 7	P25_7
MEMC0AD8	I/O	External Memory I/F data/address 8	P25_8
MEMC0AD9	I/O	External Memory I/F data/address 9	P25_9
MEMC0AD10	I/O	External Memory I/F data/address 10	P25_10
MEMC0AD11	I/O	External Memory I/F data/address 11	P25_11
MEMC0AD12	I/O	External Memory I/F data/address 12	P25_12
MEMC0AD13	I/O	External Memory I/F data/address 13	P25_13
MEMC0AD14	I/O	External Memory I/F data/address 14	P25_14
MEMC0AD15	I/O	External Memory I/F data/address 15	P25_15
$\overline{\text{MEMC0ASTB}}$	O	External Memory I/F address strobe	P21_15, P27_5
MEMC0CLK	O	External Memory I/F clock	P21_6
$\overline{\text{MEMC0CS2}}$	O	External Memory I/F chip select 2	P21_9
$\overline{\text{MEMC0CS3}}$	O	External Memory I/F chip select 3	P21_10
$\overline{\text{MEMC0CS4}}$	O	External Memory I/F chip select 4	P21_11
$\overline{\text{MEMC0BEN0}}$	O	External Memory I/F byte enable 0	P21_2
$\overline{\text{MEMC0BEN1}}$	O	External Memory I/F byte enable 1	P21_3
$\overline{\text{MEMC0RD}}$	O	External Memory I/F read strobe	P21_5
$\overline{\text{MEMC0WAIT}}$	I	External Memory I/F wait input	P21_7
$\overline{\text{MEMC0WR}}$	O	External Memory I/F write strobe	P21_4
NMI	I	External non-maskable interrupt	P0_6, JP0_5
OSCVDD	–	Oscillator voltage supply	–
OSCVSS	–	Oscillator ground	–
PMCA0MSEL0	O	External multiplexer address select signal 0	P21_8, P27_0
PMCA0MSEL1	O	External multiplexer address select signal 1	P21_9, P27_1
PMCA0MSEL2	O	External multiplexer address select signal 2	P21_10, P27_2

Table 2-41 Alphabetic pin function list (7/11)

Pin name	I/O	Pin function	Port
PTCTL1	O	External power transistor control	–
PWGD	I	External power supply stable indicator	–
REGnC	–	Voltage regulators capacitor connections	–
REGnVDD	–	Votage regulators input	–
REGnVSS	–	Votage regulators ground	–
$\overline{\text{RESET}}$	I	External reset input	–
$\overline{\text{RESETOUT}}$	O	Reset output	P0_0
RTCA0OUT	O	Real-Time Clock 1 second interval output	P0_2, JP0_5
TAPA0UP	O	Motor control output U phase	P1_2
TAPA0UN	O		P1_3
TAPA0VP	O	Motor control output V phase	P1_4
TAPA0VN	O		P1_5
TAPA0WP	O	Motor control output W phase	P1_6
TAPA0WN	O		P1_7
TAUA0I0	I	Timer Array Unit A 0 channel 0 input	P3_0
TAUA0I1	I	Timer Array Unit A 0 channel 1input	P1_1, P3_1
TAUA0I2	I	Timer Array Unit A 0 channel 2 input	P1_2, P3_2
TAUA0I3	I	Timer Array Unit A 0 channel 3 input	P1_3, P3_3
TAUA0I4	I	Timer Array Unit A 0 channel 4 input	P1_4, P3_4
TAUA0I5	I	Timer Array Unit A 0 channel 5 input	P1_5, P3_5
TAUA0I6	I	Timer Array Unit A 0 channel 6 input	P1_6, P3_6
TAUA0I7	I	Timer Array Unit A 0 channel 7 input	P1_7, P3_7
TAUA0I8	I	Timer Array Unit A 0 channel 8 input	P1_8, P3_8
TAUA0I9	I	Timer Array Unit A 0 channel 9 input	P1_9, P3_9
TAUA0I10	I	Timer Array Unit A 0 channel 10 input	P1_10, P3_10
TAUA0I11	I	Timer Array Unit A 0 channel 11 input	P1_11, P3_11
TAUA0I12	I	Timer Array Unit A 0 channel 12 input	P1_12
TAUA0I13	I	Timer Array Unit A 0 channel 13 input	P1_13, P4_0
TAUA0I14	I	Timer Array Unit A 0 channel 14 input	P1_14, P4_1
TAUA0I15	I	Timer Array Unit A 0 channel 15 input	P1_15, P4_2
TAUA0O0	O	Timer Array Unit A 0 channel 0 ouput	P3_0
TAUA0O1	O	Timer Array Unit A 0 channel 1 output	P0_1, P1_1, P3_1
TAUA0O2	O	Timer Array Unit A 0 channel 2 output	P0_2, P1_2, P3_2
TAUA0O3	O	Timer Array Unit A 0 channel 3 output	P1_3, P3_3
TAUA0O4	O	Timer Array Unit A 0 channel 4 output	P1_4, P3_4
TAUA0O5	O	Timer Array Unit A 0 channel 5 output	P0_8, P1_5, P3_5
TAUA0O6	O	Timer Array Unit A 0 channel 6 output	P0_9, P1_6, P3_6
TAUA0O7	O	Timer Array Unit A 0 channel 7 output	P1_7, P3_7
TAUA0O8	O	Timer Array Unit A 0 channel 8 output	P1_8, P3_8
TAUA0O9	O	Timer Array Unit A 0 channel 9 output	P1_9, P3_9
TAUA0O10	O	Timer Array Unit A 0 channel 10 output	P1_10, P3_10

Table 2-41 Alphabetic pin function list (8/11)

Pin name	I/O	Pin function	Port
TAUA0O11	O	Timer Array Unit A 0 channel 11 output	P1_11, P3_11
TAUA0O12	O	Timer Array Unit A 0 channel 12 output	P1_12
TAUA0O13	O	Timer Array Unit A 0 channel 13 output	P1_13, P4_0
TAUA0O14	O	Timer Array Unit A 0 channel 14 output	P1_14, P4_1
TAUA0O15	O	Timer Array Unit A 0 channel 15 output	P1_15, P4_2
TAUB1I1	I	Timer Array Unit B 1 channel 1input	P4_0
TAUB1I2	I	Timer Array Unit B 1 channel 2 input	P4_1
TAUB1I3	I	Timer Array Unit B 1 channel 3 input	P4_2
TAUB1I5	I	Timer Array Unit B 1 channel 5 input	P4_3
TAUB1I7	I	Timer Array Unit B 1 channel 7 input	P4_5
TAUB1I9	I	Timer Array Unit B 1 channel 9 input	P4_6
TAUB1I11	I	Timer Array Unit B 1 channel 11 input	P4_8
TAUB1I13	I	Timer Array Unit B 1 channel 13 input	P4_9
TAUB1I14	I	Timer Array Unit B 1 channel 14 input	P4_10
TAUB1I15	I	Timer Array Unit B 1 channel 15 input	P4_11
TAUB1O1	O	Timer Array Unit B 1 channel 1 output	P4_0
TAUB1O2	O	Timer Array Unit B 1 channel 2 output	P4_1
TAUB1O3	O	Timer Array Unit B 1 channel 3 output	P4_2
TAUB1O5	O	Timer Array Unit B 1 channel 5 output	P4_3
TAUB1O6	O	Timer Array Unit B 1 channel 6 output	P4_4
TAUB1O7	O	Timer Array Unit B 1 channel 7 output	P4_5
TAUB1O9	O	Timer Array Unit B 1 channel 9 output	P4_6
TAUB1O10	O	Timer Array Unit B 1 channel 10 output	P4_7
TAUB1O11	O	Timer Array Unit B 1 channel 11 output	P4_8
TAUB1O13	O	Timer Array Unit B 1 channel 13 output	P4_9, P0_14
TAUB1O14	O	Timer Array Unit B 1 channel 14 output	P4_10, P0_15
TAUB1O15	O	Timer Array Unit B 1 channel 15 output	P4_11
TAUB2I1	O	Timer Array Unit B 2 channel 1 input	P3_1
TAUB2I2	O	Timer Array Unit B 2 channel 2 input	P3_2
TAUB2I3	O	Timer Array Unit B 2 channel 3 input	P3_3
TAUB2I5	O	Timer Array Unit B 2 channel 5 input	P3_4
TAUB2I6	O	Timer Array Unit B 2 channel 6 input	P3_5
TAUB2I7	O	Timer Array Unit B 2 channel 7 input	P3_6
TAUB2I9	O	Timer Array Unit B 2 channel 9 input	P3_7
TAUB2I10	O	Timer Array Unit B 2 channel 10 input	P3_8
TAUB2I11	O	Timer Array Unit B 2 channel 11 input	P3_9
TAUB2I13	O	Timer Array Unit B 2 channel 13 input	P3_10, P21_2
TAUB2I14	O	Timer Array Unit B 2 channel 14 input	P3_11, P21_3
TAUB2I15	O	Timer Array Unit B 2 channel 15 input	P3_12
TAUB2O1	O	Timer Array Unit B 2 channel 1 output	P3_1
TAUB2O2	O	Timer Array Unit B 2 channel 2 output	P3_2

Table 2-41 Alphabetic pin function list (9/11)

Pin name	I/O	Pin function	Port
TAUB2O3	O	Timer Array Unit B 2 channel 3 output	P3_3
TAUB2O5	O	Timer Array Unit B 2 channel 5 output	P3_4
TAUB2O6	O	Timer Array Unit B 2 channel 6 output	P3_5
TAUB2O7	O	Timer Array Unit B 2 channel 7 output	P3_6
TAUB2O9	O	Timer Array Unit B 2 channel 9 output	P3_7
TAUB2O10	O	Timer Array Unit B 2 channel 10 output	P3_8
TAUB2O11	O	Timer Array Unit B 2 channel 11 output	P3_9
TAUB2O13	O	Timer Array Unit B 2 channel 13 output	P3_10, P21_2
TAUB2O14	O	Timer Array Unit B 2 channel 14 output	P3_11, P21_3
TAUB2O15	O	Timer Array Unit B 2 channel 15 output	P3_12
TAUC3O1	O	Timer Array Unit C 3 channel 1 output	P21_4
TAUC3O2	O	Timer Array Unit C 3 channel 2 output	P21_5
TAUC3O5	O	Timer Array Unit C 3 channel 5 output	P21_6
TAUC3O6	O	Timer Array Unit C 3 channel 6 output	P21_7
TAUC3O9	O	Timer Array Unit C 3 channel 9 output	P21_8
TAUC3O10	O	Timer Array Unit C 3 channel 10 output	P21_9
TAUC3O13	O	Timer Array Unit C 3 channel 13 output	P21_10
TAUC3O14	O	Timer Array Unit C 3 channel 14 output	P21_11
TAUC4O1	O	Timer Array Unit C 4 channel 1 output	P1_1
TAUC4O2	O	Timer Array Unit C 4 channel 2 output	P1_2
TAUC4O5	O	Timer Array Unit C 4 channel 5 output	P1_3
TAUC4O6	O	Timer Array Unit C 4 channel 6 output	P1_4
TAUC4O9	O	Timer Array Unit C 4 channel 9 output	P1_5
TAUC4O10	O	Timer Array Unit C 4 channel 10 output	P1_6
TAUC4O13	O	Timer Array Unit C 4 channel 13 output	P1_7
TAUC4O14	O	Timer Array Unit C 4 channel 14 output	P1_8
TAUC5O1	O	Timer Array Unit C 5 channel 1 output	P25_8
TAUC5O2	O	Timer Array Unit C 5 channel 2 output	P25_9
TAUC5O5	O	Timer Array Unit C 5 channel 5 output	P25_10
TAUC5O6	O	Timer Array Unit C 5 channel 6 output	P25_11
TAUC5O9	O	Timer Array Unit C 5 channel 9 output	P25_12
TAUC5O10	O	Timer Array Unit C 5 channel 10 output	P25_13
TAUC5O13	O	Timer Array Unit C 5 channel 13 output	P25_14
TAUC5O14	O	Timer Array Unit C 5 channel 14 output	P25_15
TAUC6O1	O	Timer Array Unit C 6 channel 1 output	P25_0
TAUC6O2	O	Timer Array Unit C 6 channel 2 output	P25_1
TAUC6O5	O	Timer Array Unit C 6 channel 5 output	P25_2
TAUC6O6	O	Timer Array Unit C 6 channel 6 output	P25_3
TAUC6O9	O	Timer Array Unit C 6 channel 9 output	P25_4
TAUC6O10	O	Timer Array Unit C 6 channel 10 output	P25_5
TAUC6O13	O	Timer Array Unit C 6channel 13 output	P25_6

Table 2-41 Alphabetic pin function list (10/11)

Pin name	I/O	Pin function	Port
TAUC6O14	O	Timer Array Unit C 6 channel 14 output	P25_7
TAUC7O1	O	Timer Array Unit C 7 channel 1 output	P24_8
TAUC7O2	O	Timer Array Unit C 7 channel 2 output	P24_9
TAUC7O5	O	Timer Array Unit C 7 channel 5 output	P24_10
TAUC7O6	O	Timer Array Unit C 7 channel 6 output	P24_11
TAUC7O9	O	Timer Array Unit C 7 channel 9 output	P24_12
TAUC7O10	O	Timer Array Unit C 7 channel 10 output	P24_13
TAUC7O13	O	Timer Array Unit C 7 channel 13 output	P24_14
TAUC7O14	O	Timer Array Unit C 7 channel 14 output	P24_15
TAUJ0I0	I	Timer Array Unit J 0 channel 0 input	P0_12, JP0_0
TAUJ0I1	I	Timer Array Unit J 0 channel 1 input	P0_13, JP0_1
TAUJ0I2	I	Timer Array Unit J 0 channel 2 input	P0_14, JP0_2
TAUJ0I3	I	Timer Array Unit J 0 channel 3 input	P0_15, JP0_3
TAUJ0O0	O	Timer Array Unit J 0 channel 0 output	P0_12, JP0_0
TAUJ0O1	O	Timer Array Unit J 0 channel 1 output	P0_13, JP0_1
TAUJ0O2	O	Timer Array Unit J 0 channel 2 output	P0_14, JP0_2
TAUJ0O3	O	Timer Array Unit J 0 channel 3 output	P0_15, JP0_3
TAUJ1I0	I	Timer Array Unit J 1 channel 0 input	P0_0
TAUJ1I1	I	Timer Array Unit J 1 channel 1 input	P0_1, P3_0
TAUJ1I2	I	Timer Array Unit J 1 channel 2 input	P0_2, P2_2
TAUJ1I3	I	Timer Array Unit J 1 channel 3 input	P0_3, P2_1
TAUJ1O0	O	Timer Array Unit J 1 channel 0 output	P0_0
TAUJ1O1	O	Timer Array Unit J 1 channel 1 output	P0_1, P3_0
TAUJ1O2	O	Timer Array Unit J 1 channel 2 output	P0_2, P2_2
TAUJ1O3	O	Timer Array Unit J 1 channel 3 output	P0_3, P2_1
URTE0RX	I	Asynchroneous Serial I/F 0 receive data input	P21_6, P21_8, P24_5
URTE0TX	O	Asynchroneous Serial I/F 0 receive data output	P21_5, P21_7, P24_4
URTE1RX	I	Asynchroneous Serial I/F 1 receive data input	P24_5, P21_8, P24_7
URTE1TX	O	Asynchroneous Serial I/F 1 receive data output	P24_4, P21_7, P24_6
URTE2RX	I	Asynchroneous Serial I/F 2 receive data input	P0_1, P4_1
URTE2TX	O	Asynchroneous Serial I/F 2 receive data output	P0_2, P4_2
URTE3RX	I	Asynchroneous Serial I/F 3 receive data input	P1_11, P3_8
URTE3TX	O	Asynchroneous Serial I/F 3 receive data output	P1_10, P3_7, P3_9
URTE4RX	I	Asynchroneous Serial I/F 4 receive data input	P1_9, P1_13
URTE4TX	O	Asynchroneous Serial I/F 4 receive data output	P1_8, P1_12
URTE5RX	I	Asynchroneous Serial I/F 5 receive data input	P1_15, P3_9, P25_14
URTE5TX	O	Asynchroneous Serial I/F 5 receive data output	P1_14, P3_10, P25_15
URTE6RX	I	Asynchroneous Serial I/F 6 receive data input	P2_1, P3_11, P25_0
URTE6TX	O	Asynchroneous Serial I/F 6 receive data output	P2_0, P3_8, P25_1
URTE7RX	I	Asynchroneous Serial I/F 7 receive data input	P2_2, P3_7, P25_4
URTE7TX	O	Asynchroneous Serial I/F 7 receive data output	P2_1, P4_0, P25_3

Table 2-41 Alphabetic pin function list (11/11)

Pin name	I/O	Pin function	Port
URTE8RX	I	Asynchroneous Serial I/F 8 receive data input	P4_10, P24_1
URTE8TX	O	Asynchroneous Serial I/F 8 receive data output	P4_9, P24_0
URTE9RX	I	Asynchroneous Serial I/F 9 receive data input	P21_6, P24_3
URTE9TX	O	Asynchroneous Serial I/F 9 receive data output	P21_5, P24_2
URTE10RX	I	Asynchroneous Serial I/F 10 receive data input	P0_9, P4_4
URTE10TX	O	Asynchroneous Serial I/F 10 receive data output	P0_8, P4_3
URTE11RX	I	Asynchroneous Serial I/F 11 receive data input	P0_7, P0_11, P4_7
URTE11TX	O	Asynchroneous Serial I/F 11 receive data output	P0_6, P0_10, P4_6
VCPC0IN	I	Voltage Comparator 0 compare input voltage	–
VCPC0OUT	O	Voltage Comparator 0 compare output signal	JP0_0
VCPC1IN	I	Voltage Comparator 1 compare input voltage	–
VCPC1OUT	O	Voltage Comparator 1 compare output signal	JP0_1
WAKE	O	DEEPSTOP mode indicator	–
X1, X2	–	Main oscillator resonator connections	–
XT1, XT2	–	Sub oscillator resonator connections	–

2.5.7 Port and pin functions in stand-by modes

Details about the port functions in stand-by modes are given in the chapter *"Stand-by Controller"* and in the section *"Power-up and down procedures"* of chapter *"Power Supply".*

2.5.8 Port and pin functions during and after reset

Table 2-42 Ports and pins function during and after reset

Port	During reset	After reset	Comment
JP0_0 to JP0_3, JP0_5	high impedance	mode dependent	These ports are also used for the On-Chip Debug I/F. Thus their function after reset depends on the operation mode.[a]
JP0_4	input with internal pull-down resistor	mode dependent	
P0_0	open drain output (low level)	open drain output (low level)	This port functions as $\overline{\text{RESETOUT}}$ during and after reset. Refer to " *2.5.2 "Common port functions" on page 81* for details.
P0_1	input	input	This port is also used as FLMD1 mode selection input.[a]
P0_2 to P0_15	high impedance	high impedance	
Port groups P1 to P4, P10 to P13, P21, P24, P25 to P29	high impedance	high impedance	
FLMD0	input	input	
$\overline{\text{RESET}}$	input	input	
WAKE	output	output	
PWGD	input	input	
ADCA0I0 to ADCA0I5	input	input	

a) Refer to the section *"Mode pins and JP0 connections"* in the *"CPU System Functions"* chapter.

2.5.9 Recommended connection of unused pins

In the following recommendations are given how to treat pins, which are not used on the application board.

As a reference the final table lists all pins which must be connected in any case.

(1) Unused pins with port functionality

Table 2-43 Recommended connection of unused pins with port functionality

Port	After reset	Connect to	Comment
P0_0/$\overline{\text{RESETOUT}}$	open drain output	leave open	
P0_1/FLMD1[a]	The function of these ports depends on the operation mode. Refer to the section *"Mode pins and JP0 connections"* in the *"CPU System Functions"* chapter.		
P0_2/MODE0			
P0_3/MODE1			
JP0_4/ $\overline{\text{DCUTRST}}$[a]			
JP0_0 to JP0_3, JP0_5			
P0_4 to P0_15	high impedance[b]	leave open	If the high impedance reset configuration is changed to • output: leave open • input: connect to E0VDD or E0VSS via resistor
Port groups P1 to P4	high impedance[b]	leave open	If the high impedance reset configuration is changed to • output: leave open • input: connect to E1VDD or E1VSS via resistor
Port groups P10, P11	high impedance[b]	leave open	If the high impedance reset configuration is changed to • output: leave open • input: connect to A0VDD or A0VSS via resistor
Port groups P12, P13	high impedance[b]	leave open	If the high impedance reset configuration is changed to • output: leave open • input: connect to A1VDD or A1VSS via resistor
Port groups P21, P24 to P29	high impedance[b]	leave open	If the high impedance reset configuration is changed to • output: leave open • input: connect to B0VDD or B0VSS via resistor

a) Refer also to *"Mandatory connection of pins"* below.
b) Rfer to the note below..

Note It is recommended to leave pins, which are in high impedance state after reset release, in this state, if they are not used by the application.

(2) Unused pins without port functionality

Table 2-44 Recommended connection of unused pins without port functionality

Pin	Connect to	Comment
X1	connect to OSCVSS	If MainOsc not used
X2	leave open	
XT1	connect to OSCVSS	If SubOsc not used
XT2	leave open	
$\overline{\text{RESET}}$	connect to E0VDD	If external $\overline{\text{RESET}}$ is not used, the device is only reset by on-chip Power-On-Clear POCRES
VCPC0IN, VCPC1IN	connect to E0VDD or E0VSS via resistor	If Voltage Comparators not used
ADCA0I0 to ADCA0I5	connect to A0VDD or A0VSS via resistor	
WAKE	leave open	If external voltage supply for isolated areas is not switched off in DEEPSTOP
PWGD	connect to E0VDD	
PTCTL1	leave open	If the V850E2/Fx4-H is used without external power transistor, but external voltage regulator to provide stabilized CVDD
N.C.	leave open or connect to ground	Ground means E1VSS or B0VSS orCVSS

(3) Mandatory connection of pins

Table 2-45 Mandatory connection pins

Pin	Comment
FLMD0 FLMD1/P0_1 $\overline{\text{DCUTRST}}$/JP0_4	FLMD0 determines the operation mode of the device. FLMD1 and $\overline{\text{DCUTRST}}$ must not be left unconnected. Refer to the section *"Mode pins and JP0 connections"* in the *"CPU System Functions"* chapter.
REG0VDD/REG0VSS	All power supply pins must always be connected, although the external power supplies for the isolated areas (REGnVDD, CVDD) may be switched off in DEEPSTOP stand-by mode.
CVDD, CVSS	
AnVDD, AnVREFP	
AnVSS, AnVREFM	
B0VDD, B0VSS	
EnVDD, EnVSS	
OSCVDD, OSCVSS	
FVDD, VSS	

2.6 Port Filters

The input signals at some pins are passed through a filter to remove noise and glitches. The microcontroller supports different types of analog and digital filters.

The first section provides an overview, which port input signals are equipped with which kind of filter type, their control registers and bits, the register addresses, and the power domain they are located on.
The last paragraph of this section informs about the clock signals for the port filters

A detailed description of the analog and digital filter types and their control registers follows in section *"Port Filters Functional Description"* below.

2.6.1 Port filters assignment

The following tables list the input signals, which are equipped with an analog filter or a digital filter.

(1) Input signals with analog filters type A

The analog filters type A are controlled by means of the following register:

- Filter control registers FCLAnCTLm (m = 0 to 7)
  For each port with an analog filter a dedicated register FCLAnCTLm is provided, whereas each group "n" can handle up to 8 input signals "m".

Table 2-46 Input signals with analog filters type A (1/2)

Signal	FCLA instance		
	Register		Address
INTP0	FCLA0	CTL0	FF41 4000$_H$
INTP1		CTL1	FF41 4004$_H$
INTP2		CTL2	FF41 4008$_H$
INTP3		CTL3	FF41 400C$_H$
INTP4		CTL4	FF41 4010$_H$
INTP5		CTL5	FF41 4014$_H$
INTP6		CTL6	FF41 4018$_H$
INTP7		CTL7	FF41 401C$_H$
INTP8	FCLA1	CTL0	FF41 4020$_H$
INTP9		CTL1	FF41 4024$_H$
INTP10		CTL2	FF41 4028$_H$
INTP11		CTL3	FF41 402C$_H$
INTP12		CTL4	FF41 4030$_H$
INTP13		CTL5	FF41 4034$_H$
INTP14		CTL6	FF41 4038$_H$
INTP15		CTL7	FF41 403C$_H$
NMI	FCLA2	CTL0	FF41 4040$_H$
TAPA0ESO		CTL1	FF41 4044$_H$

Table 2-46 Input signals with analog filters type A (2/2)

Signal	FCLA instance		
	Register		Address
KR0	FCLA3	CTL0	FF41 4060$_H$
KR1		CTL1	FF41 4064$_H$
KR2		CTL2	FF41 4068$_H$
KR3		CTL3	FF41 406C$_H$
KR4		CTL4	FF41 4070$_H$
KR5		CTL5	FF41 4074$_H$
KR6		CTL6	FF41 4078$_H$
KR7		CTL7	FF41 407C$_H$

(2) Input signals with analog filters type B

The analog filters type B are controlled by means of the following register:

- Filter control registers FCLAnCTLm (m = 0 to 7)
  For each port with an analog filter a dedicated register FCLAnCTLm is provided, whereas each group “n” can handle up to 8 input signals “m”.

Table 2-47 Input signals with analog filters type B

Signal	FCLA instance		
	Register		Address
TAUJ0I0	FCLA4	CTL0	FF41 4080$_H$
TAUJ0I1		CTL1	FF41 4084$_H$
TAUJ0I2		CTL2	FF41 4088$_H$
TAUJ0I3		CTL3	FF41 408C$_H$
TAUJ1I0		CTL4	FF41 4090$_H$
TAUJ1I1		CTL5	FF41 4094$_H$
TAUJ1I2		CTL6	FF41 4098$_H$
TAUJ1I3		CTL7	FF41 409C$_H$

(3) Input signals with analog filters type C

The analog filters type C have no control registers.

Table 2-48 Input signals with analog filters type C

Signal	FCLA instance		
	Register		Address
Always-On-Area:			
FLMD0	–	–	–
FLMD1		–	–
PWGD		–	–
$\overline{\text{RESET}}$		–	–

(4) Input signals with digital filters type D

The digital filters type D are controlled by means of the following register:

- Filter control registers FCLAnCTLm (m = 0 to 7)
  For each port with a digital filter a dedicated register FCLAnCTLm is provided, whereas each group “n” can handle up to 8 input signals “m”.
- Digital noise elimination control registers DNFAnCTL
  Each DNFAnCTL control register can handle a group “n” with up to 16 input signals with digital filters.
- Digital noise elimination enable registers DNFAnEN
  The bits DNFAnNFEN[15:0] of this register enable digital filters of the group “n” of up to 16 input signals.

Caution If digital filtering shall not be applied to the input signal, the filter bypass must be activated by FCLAnCTLm.FCLAnBYPSm = 1.

Table 2-49 Input signals with digital filters type D (1/3)

Signal	DNFA instance				FCLA instance		
	Register	Address	Filter enable bit		Register		Address
TAUA0I0	DNFA0CTL DNFA0EN DNFA0ENH DNFA0ENL	FF41 1000$_H$ FF41 1004$_H$ FF41 1008$_H$ FF41 100C$_H$	DNFA0EN.DNFA0	NFEN0	FCLA5	CTL0	FF41 5000$_H$
TAUA0I1				NFEN1		CTL1	FF41 5004$_H$
TAUA0I2				NFEN2		CTL2	FF41 5008$_H$
TAUA0I3				NFEN3		CTL3	FF41 500C$_H$
TAUA0I4				NFEN4		CTL4	FF41 5010$_H$
TAUA0I5				NFEN5		CTL5	FF41 5014$_H$
TAUA0I6				NFEN6		CTL6	FF41 5018$_H$
TAUA0I7				NFEN7		CTL7	FF41 501C$_H$
TAUA0I8				NFEN8	FCLA6	CTL0	FF41 5020$_H$
TAUA0I9				NFEN9		CTL1	FF41 5024$_H$
TAUA0I10				NFEN10		CTL2	FF41 5028$_H$
TAUA0I11				NFEN11		CTL3	FF41 502C$_H$
TAUA0I12				NFEN12		CTL4	FF41 5030$_H$
TAUA0I13				NFEN13		CTL5	FF41 5034$_H$
TAUA0I14				NFEN14		CTL6	FF41 5038$_H$
TAUA0I15				NFEN15		CTL7	FF41 503C$_H$

Table 2-49 Input signals with digital filters type D (2/3)

Signal	DNFA instance				FCLA instance		
	Register	Address	Filter enable bit		Register		Address
URTE10RX	DNFA1CTL DNFA1EN DNFA1ENH DNFA1ENL	FF41 1020$_H$ FF41 1024$_H$ FF41 1028$_H$ FF41 102C$_H$	DNFA1EN.DNFA1	NFEN0	FCLA7	CTL0	FF41 5040$_H$
URTE11RX				NFEN1		CTL1	FF41 5044$_H$
CSIG4SC				NFEN2		CTL2	FF41 5048$_H$
CSIG4SI				NFEN3		CTL3	FF41 504C$_H$
CSIG4RY				NFEN4		CTL4	FF41 5050$_H$
$\overline{\text{CSIG4SSI}}$				NFEN5		CTL5	FF41 5054$_H$
ADCA0TRG0				NFEN6	FCLA8	CTL0	FF41 5060$_H$
ADCA0TRG1				NFEN7		CTL1	FF41 5064$_H$
ADCA0TRG2				NFEN8		CTL2	FF41 5068$_H$
ENCA0AIN				NFEN9		CTL3	FF41 506C$_H$
ENCA0BIN				NFEN10		CTL4	FF41 5070$_H$
ENCA0ZIN				NFEN11		CTL5	FF41 5074$_H$
ENCA0TIN0				NFEN12		CTL6	FF41 5078$_H$
ENCA0TIN1				NFEN13		CTL7	FF41 507C$_H$
TAUB1I1	DNFA2CTL DNFA2EN DNFA2ENH DNFA2ENL	FF41 2000$_H$ FF41 2004$_H$ FF41 2008$_H$ FF41 200C$_H$	DNFA5EN.DNFA5	NFEN0	FCLA9	CTL0	FF41 6000$_H$
TAUB1I2				NFEN1		CTL1	FF41 6004$_H$
TAUB1I3				NFEN2		CTL2	FF41 6008$_H$
TAUB1I5				NFEN3		CTL3	FF41 600C$_H$
TAUB1I7				NFEN5		CTL5	FF41 6014$_H$
TAUB1I9				NFEN8	FCLA10	CTL0	FF41 6020$_H$
TAUB1I11				NFEN10		CTL2	FF41 6028$_H$
TAUB1I13				NFEN11		CTL3	FF41 602C$_H$
TAUB1I14				NFEN12		CTL4	FF41 6030$_H$
TAUB1I15				NFEN13		CTL5	FF41 6034$_H$
TAUB2I1	DNFA5CTL DNFA5EN DNFA5ENH DNFA5ENL	FF41 2060$_H$ FF41 2064$_H$ FF41 2068$_H$ FF41 206C$_H$	DNFA5EN.DNFA5	NFEN0	FCLA15	CTL0	FF41 60C0$_H$
TAUB2I2				NFEN1		CTL1	FF41 60C4$_H$
TAUB2I3				NFEN2		CTL2	FF41 60C8$_H$
TAUB2I5				NFEN3		CTL3	FF41 60CC$_H$
TAUB2I6				NFEN4		CTL4	
TAUB2I7				NFEN5		CTL5	FF41 60D4$_H$
TAUB2I9				NFEN8	FCLA16	CTL0	FF41 60E0$_H$
TAUB2I10				NFEN9		CTL1	
TAUB2I11				NFEN10		CTL2	FF41 60E8$_H$
TAUB2I13				NFEN11		CTL3	FF41 60EC$_H$
TAUB2I14				NFEN12		CTL4	FF41 60F0$_H$
TAUB2I15				NFEN13		CTL5	FF41 60F4$_H$

Table 2-49 Input signals with digital filters type D (3/3)

Signal	DNFA instance				FCLA instance		
	Register	Address	Filter enable bit		Register		Address
ADCA1TRG0	DNFA8CTL DNFA8EN DNFA8ENH DNFA8ENL	FF41 20C0$_H$ FF41 20C4$_H$ FF41 20C8$_H$ FF41 20CC$_H$	DNFA8EN.DNFA8	NFEN4	FCLA21	CTL4	FF41 6190$_H$
ADCA1TRG1				NFEN5		CTL5	FF41 6194$_H$
ADCA1TRG2				NFEN6		CTL6	FF41 6198$_H$
CSIH0SC	DNFA9CTL DNFA9EN DNFA9ENH DNFA9ENL	FF41 2oE0$_H$ FF41 2oE4$_H$ FF41 2oE8$_H$ FF41 2oEC$_H$	DNFA9EN.DNFA9	NFEN0	FCLA22	CTL0	FF41 61A0$_H$
CSIH0RY				NFEN1		CTL1	FF41 61A4$_H$
CSIH0SI				NFEN2		CTL2	FF41 61A8$_H$
$\overline{\text{CSIH0SSI}}$				NFEN3		CTL3	FF41 61AC$_H$
CSIH1SC				NFEN4		CTL4	FF41 61B0$_H$
CSIH1RY				NFEN5		CTL5	FF41 61B4$_H$
CSIH1SI				NFEN6		CTL6	FF41 61B8$_H$
$\overline{\text{CSIH1SSI}}$				NFEN7		CTL7	FF41 61BC$_H$
CSIH2SC				NFEN8	FCLA23	CTL0	FF41 61C0$_H$
CSIH2RY				NFEN9		CTL1	FF41 61C4$_H$
CSIH2SI				NFEN10		CTL2	FF41 61C8$_H$
$\overline{\text{CSIH2SSI}}$				NFEN11		CTL3	FF41 61CC$_H$
CSIG0SC	DNFA10CTL DNFA10EN DNFA10ENH DNFA10ENL	FF41 2100$_H$ FF41 2104$_H$ FF41 2108$_H$ FF41 210C$_H$	DNFA10EN.DNFA10	NFEN0	FCLA24	CTL0	FF41 61E0$_H$
CSIG0RY				NFEN1		CTL1	FF41 61E4$_H$
CSIG0SI				NFEN2		CTL2	FF41 61E8$_H$
$\overline{\text{CSIG0SSI}}$				NFEN3		CTL3	FF41 61EC$_H$
CSIG2SC				NFEN8	FCLA25	CTL0	FF41 6200$_H$
CSIG2RY				NFEN9		CTL1	FF41 6204$_H$
CSIG2SI				NFEN10		CTL2	FF41 6208$_H$
URTE0RX	DNFA11CTL DNFA11EN DNFA11ENH DNFA11ENL	FF41 2120$_H$ FF41 2124$_H$ FF41 2128$_H$ FF41 212C$_H$	DNFA11EN.DNFA11	NFEN4	FCLA26	CTL4	FF41 6230$_H$
URTE1RX				NFEN5		CTL5	FF41 6234$_H$
URTE2RX				NFEN8	FCLA27	CTL0	FF41 6240$_H$
URTE3RX				NFEN9		CTL1	FF41 6244$_H$
URTE4RX				NFEN10		CTL2	FF41 6248$_H$
URTE5RX				NFEN11		CTL3	FF41 624C$_H$
URTE6RX				NFEN12		CTL4	FF41 6250$_H$
URTE7RX				NFEN13		CTL5	FF41 6254$_H$
URTE8RX				NFEN14		CTL6	FF41 6258$_H$
URTE9RX				NFEN15		CTL7	FF41 625C$_H$

2.6.2 Port filters clock supply

Following table sumamrizes the clock supplies for the port filter types on the different power domains:

Table 2-50 Port filter clock supply

Power domain	Filters clocks	Connected to
Always-On-Area	PCLK	Clock Controller CKSCLK_A02
Isolated-Area-0	PCLK	Clock Controller CKSCLK_005
	DNFATCKI	Clock Controller CKSCLK_016
Isolated-Area-1	PCLK	Clock Controller CKSCLK_101
	DNFATCKI	Clock Controller CKSCLK_128

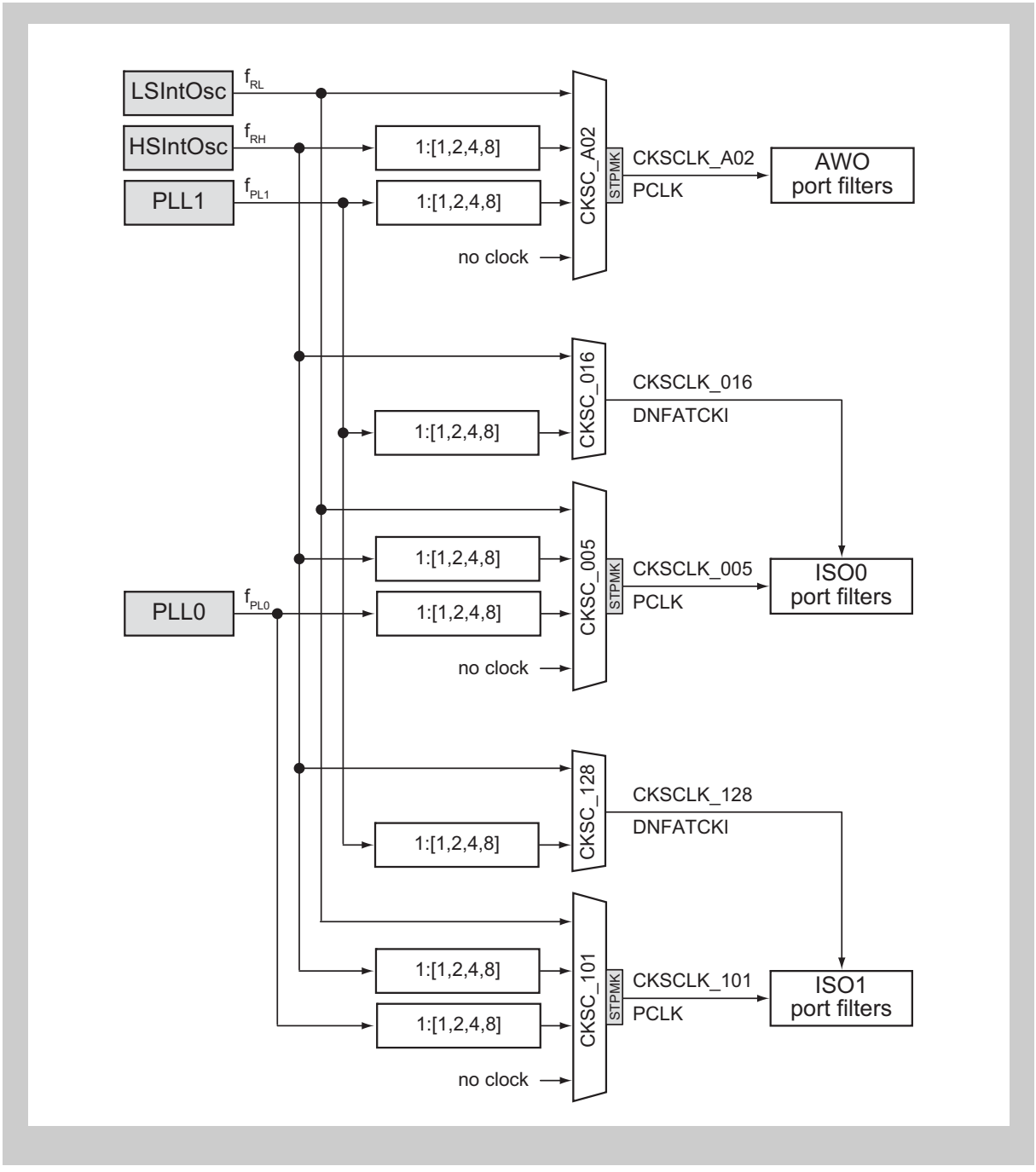


Figure 2-2 Port filters clock supply

2.6.3 Port filters reset

The port filters and its registers are initialized by the following reset signal:

Table 2-51 Port filters reset signal

Port filters power domain	Reset signal
Always-On-Area	• Reset Controller: SYSRES
Isolated-Area-0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
Isolated-Area-1	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

2.7 Port Filters Functional Description

Depending on the purpose of the external input signal to be filtered, the external signals are passed through different filter types:

- Analog filters
  Analog filter have a fixed filter characteristic.
  - Type A: analog filtered signals with edge or level detection
    The output signals are used to signal an external event, whereas the timing of the external signal is not of concern, but its level or level changes.
    An external interrupt is a typical case for such event signals.
  - Type B: analog filtered signals with filter bypass option
    The timing of the filtered output signals is retained with this filter type. Bypassing of the filter is also possible.
    A typical example for such signals is a timer input signal to measure its frequency.
  - Type C: analog filter only
    The input signal are always passed through an analog filter, which can not be bypassed.
    Such filters are typically used for external $\overline{\text{RESET}}$ inputs and mode signals.
- Digital filters
  The characteristic of a digital filter can be adjusted to the application's needs.
  - Type D: configurable digital filtered signals with filter bypass option
    The timing of the filtered output signals is retained with this filter type. Bypassing of the filter is also possible.
    A typical example for such signals is a timer input signal to measure its frequency.

2.7.1 Analog filters

Analog filter characteristic

The characteristics of the analog filter as well as of the level and edge detectors are specified in the Electrical Target Specification document.

Analog filters control registers

For each input signal, that is equipped with an analog filter, a dedicated control register FCLAnCTLm is provided.

The registers are ordered in groups of 8 registers with the same index n. The register index m ranges from 0 to 7:

FCLA group n: FCLAnCTL0 to FCLAnCTL7

The assignment of the input signals to the control registers and their addresses is given in the tables in the previous section *"Port filters assignments"*.

Analog filter in stand-by mode

All analog filters are located on the Always-On-Area.
The behaviour of an analog filter and its wake-up capability depend on the filter type. Refer to the description of the analog filter types below.

(1) Analog filter type A

The block diagram of the analog filter type A is shown in the diagram below.

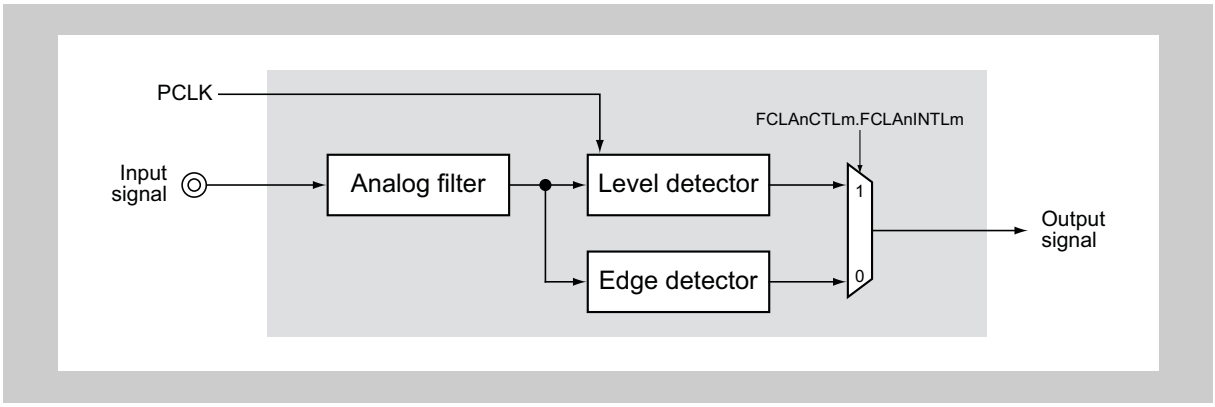


Figure 2-3 Analog filter type A block diagram

After passing the external signal through an analog filter to eliminate noise and spikes, the event detection evaluates the level or any level change, i.e. an edge, of the signal and generates an output accordingly.

The detection mode is selected by the control bit FCLAnCTLm.FCLAnINTLm:

- FCLAnINTLm = 0: edge detection mode
  The detection of a rising or falling edge can be activated separately by FCLAnCTLm.FCLAnINTRm and FCLAnCTLm.FCLAnINTFm respectively.
- FCLAnINTLm = 1: level detection mode
  The detection of a high level or low level can be selected by FCLAnCTLm.FCLAnINTRm.

The table below summarizes the detection conditions of the analog filter.

Table 2-52 Analog filter event detection conditions

FCLAnINTLm	FCLAnINTFm	FCLAnINTRm	Edge detection	Level detection
0	0	0	no detection	not active
		1	rising edge	
	1	0	falling edge	
		1	both edges	
1	X	0	not active	low level
		1		high level

Default configuration

The default configuration of the analog filter type A is as follows:

- analog filter with edge detection

Analog filter type A in stand-by mode

In case the clock PCLK is stopped in stand-by mode, the analog filter type A can only operate with the edge detection. Thus set FCLAnINTLm = 0 and select the required edge detection if the input signal shall be used as a stand-by mode wake-up signal.

(2) Analog filter type B

The block diagram of the analog filter type B is shown in the diagram below.

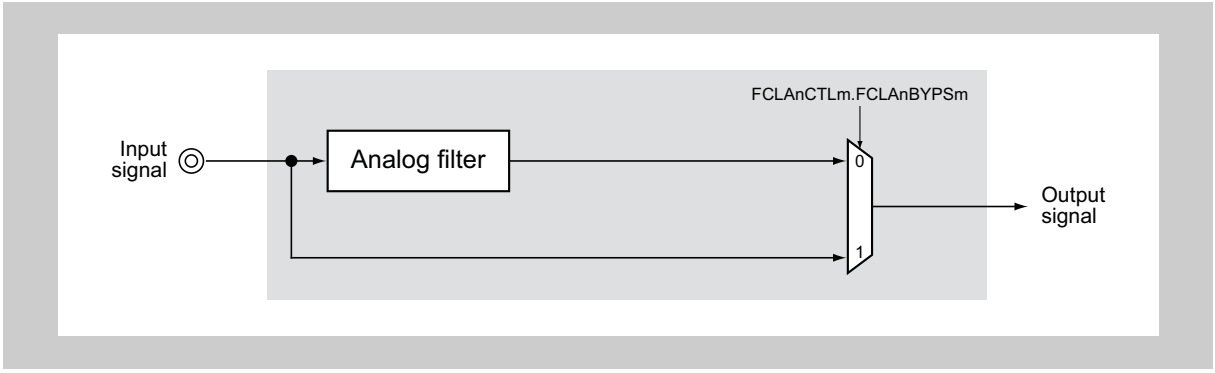


Figure 2-4 Analog filter type B block diagram

The analog filter can optionally be bypassed:

- DCLAnCTLm.FCLAnBYPSm = 0: the filtered signal is output
- DCLAnCTLm.FCLAnBYPSm = 1: the unfiltered input signal is output

Default configuration

The default configuration of the analog filter type B is as follows:

- analog filter active

Analog filter type B in stand-by mode

The output signal of an analog filter type B can always be used as a stand-by mode wake-up signal.

(3) Analog filter type C

The block diagram of the analog filter type C is shown in the diagram below.

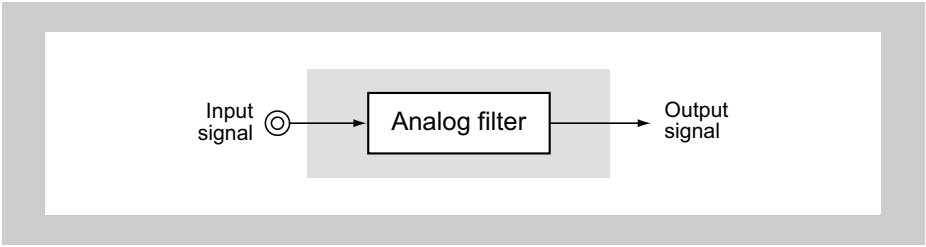


Figure 2-5 Analog filter type C block diagram

The output signal is always the analog filtered input signal.

Analog filter type C in stand-by mode

The output signal of an analog filter type C can always be used as a stand-by mode wake-up signal.

2.7.2 Digital filters

Digital filter characteristic

The digital filters allow to adjust the filter characteristics to the needs of the application.

The input signal is sampled with the sampling frequency f_s.

If a specified number of successive samples yield the same (high or low) level, the signal level is judged as valid and the filter output signal is set accordingly.

If an external signal level change is detected within the specified number of samples (same level samples s), the signal level is judged as noise - or a spike - and the filter output signal does not change.

The length of an external signal pulse to be judged as noise depends on the sampling frequency and the specified number of same level samples s.

Both parameters can be specified:

- DNFAnCTL.DNFAnPRS[2:0] allows to select the sampling frequency to

$$f_s = \frac{f_{DNFATCKI}}{2^{DNFAnPRS[2:0]}}$$

  where $f_{DNFATCKI}$ is the frequency of the DNFATCKI clock.

- DNFAnCTL.DNFAnNFSTS[1:0] determines the number s of same level samples:

$$s = DNFAnNFSTS[1:0] + 2$$

External signal pulses, shorter than

$$t_{wDNF(min)} = (s-1) \bullet \frac{1}{f_s}$$

are always suppressed. That means also, pulses with a width $\geq t_{wDNF(min)}$ may pass the filter.

External signal pulses, longer than

$$t_{wDNF(max)} = s \bullet \frac{1}{f_s}$$

are always judged as valid and are passed on to the filter output. That means also, pulses with a width $\leq t_{wDNF(max)}$ may be suppressed.

Consequently, external signal pulses, with a t_{wDNF} width in the range

$$(s-1) \bullet \frac{1}{f_s} \leq t_{wDNF} \leq s \bullet \frac{1}{f_s}$$

may be suppressed or judged as valid.

The filter effects a delay t_{dDNF} between the filter input and output pulse in the range

$$(s-1) \bullet \frac{1}{f_s} + \frac{2}{f_{DNFATCKI}} \leq t_{dDNF} \leq s \bullet \frac{1}{f_s} + \frac{3}{f_{DNFATCKI}}$$

The filter operation is illustrated in the figure below with DNFAnNFSTS[1:0] = 01_B, i.e. s = 3 same level samples.

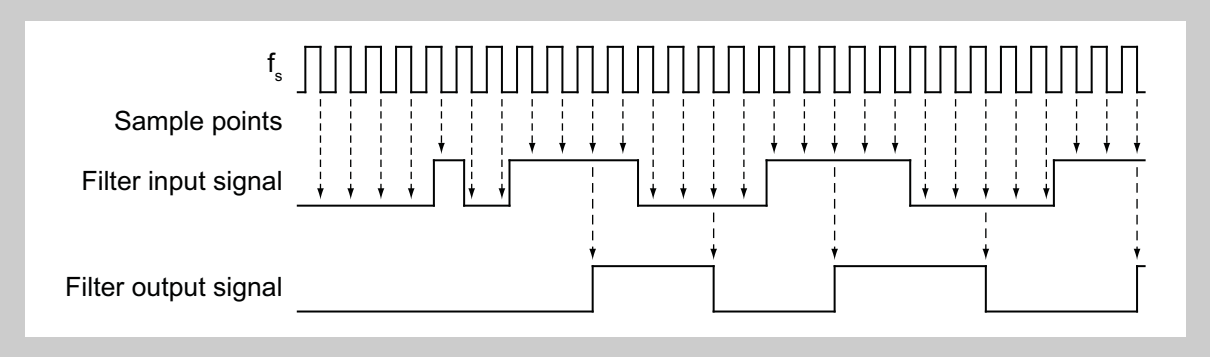


Figure 2-6 Digital filter function

Digital filter groups

The input signals with digital filters are ordered in groups of up to 16 signals. The digital filter characteristics, specified by DNFAnCTL.DNFAnPRS[2:0] and DNFAnNFSTS[1:0] apply to the filters of the entire group.
However the digital filter for each signal can be enabled respectively disabled separately by DNFAnEN.DNAFnNFENm.

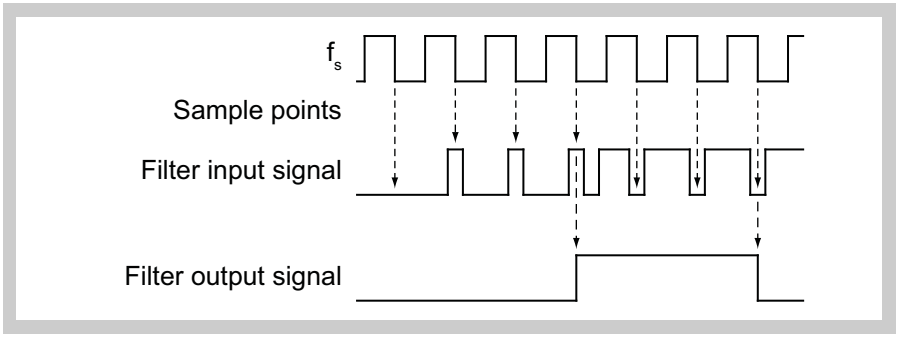
Cautions

1. After enabling the digital filter by DNFAnEN.DNAFnNFENm = 1, the digital filter is in normal operation after the time period

   $$DNFAnNFSTS[1:0] \times 1/f_s + 2 \times 1/f_{DNFATCKI}$$

   and may generate unintended output signals during that time period.
   Wait the above time span before enabling the function, the signal is supplied to.

2. If the levels of the external signal changes multiple times within a sample period $1/f_s$ (undersampling), the digital filter may not detect these level changes and thus may not suppress such signals.
   An example is show in the figure below (for s = 3).



Digital filter in stand-by mode

- DEEPSTOP mode:
  All digital filter are located on an isolated power domain, that can be switched off during DEEPSTOP. Thus they are not in operation, when the respective power domain is in DEEPSTOP mode.
- STOP mode:
  Digital noise elimination requires the clock supply DNFATCKI to operate. Since the DNFATCKI clock is never stopped, digitally filtered signals can always serve as STOP mode wake-up event.

Digital filters control registers

For each group of up to 16 digital filters a common digital filter setup register DNFAnCTL and digital filter enable register DNFAnEN is provided with the same index n.

While the filter setup by DNFAnCTL effects the entire group, the control bits DNFAnENm in the filter enable register DNFAnEN allows to enable respectively disable each filter separately. The register index m is in the range from 0 to 15:

DNFAnCTL is the control register of group n for the digital filters m = 0 to 15, enabled/disabled by the DNFAnEN.DNAFnEN0 to DNFAnEN.DNAFnEN15 control bits.

The edge detection setup is done via the filter dedicated control register FCLAnCTLm.

The FCLAnCTLm registers are ordered in groups of 8 registers with the same index n. The register index m is in the range from 0 to 7:

FCLA group n: FCLAnCTL0 to FCLAnCTL7

The assignment of the input signals to the control registers and their addresses is given in the table *“Input signals with digital filters”* in the previous section *“Analog and digital filter assignments”*.

Caution Do not change any control register settings, while the concerned digital filter is enabled by DNFAnEN.DNAFnENm = 1. Otherwise an unintended filter output may be generated.

(1) Digital filter type D

The block diagram of the digital filter type D is shown in the diagram below.

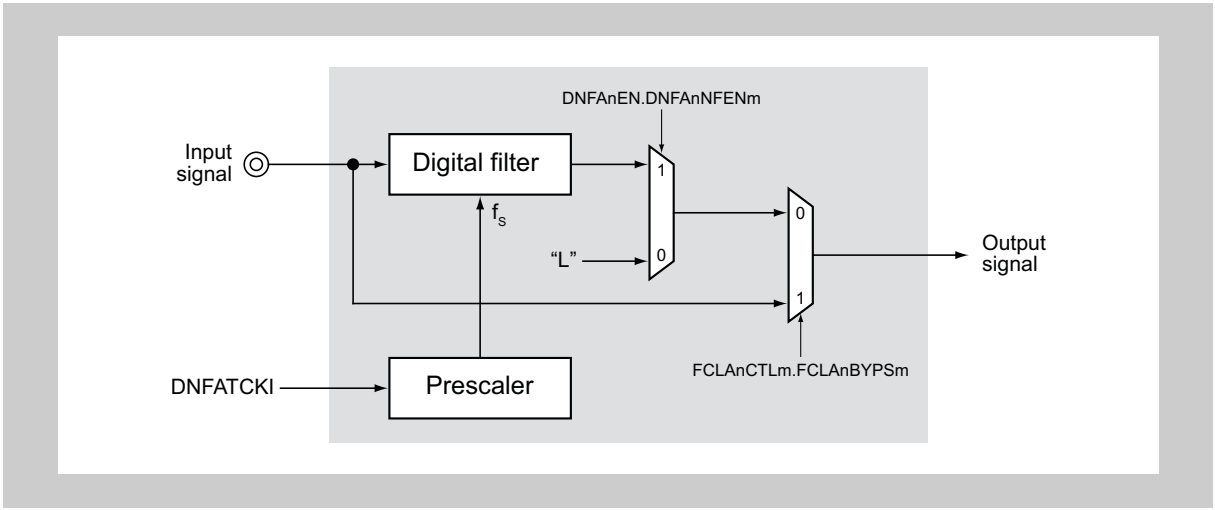


Figure 2-7 Digital filter type D block diagram

The output signal depends on register settings, as shown in the table below:

Table 2-53 Output options of digital filter type D

FCLAnCTLm. FCLAnBYPSm	DNFAnEN. DNFAnNFENm	Output signal
0	0	fixed low level (input signal blocked)
	1	filtered input signal
1	X	not filtered input signal (filter bypass)

Default configuration

The default configuration of the digital filter type D is as follows:

- input signal blocked

Cautions

1. Per default, the input signal is blocked. Thus the digital filter type D must be configured to let the - filtered or non-filtered - input signal pass.
2. If digital filtering shall not be applied to the input signal, the filter bypass must be activated by FCLAnCTLm.FCLAnBYPSm = 1.

2.7.3 Filter control registers

The analog and digital filters are controlled and operated by the following registers:

Table 2-54 Filter registers overview

Register Name	Shortcut	Address
Filter control register m	FCLAnCTLm	The addresses are given in the tables in the previous section *"Port filters assignments"*.
Digital noise elimination control register	DNFAnCTL	
Digital noise elimination enable register	DNFAnEN	

(1) FCLAnCTLm – Filter control register

This register controls the analog and digital filter operation.

Access This register can be read/written in 8-bit units.

Address The assignment of the input signals to the FCLAnCTLm registers and their addresses is given in the tables in the previous section *"Port filters assignments"*.

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
FCLAn BYPSm	0	0	0	0	FCLAn INTLm	FCLAn INTFm	FCLAn INTRm
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-55 FCLAnCTLm register contents

Bit position	Bit name	Function
7	FCLAn BYPSm	Filter bypass control: 0: filter bypass disabled 1: filter bypass enabled **Note:** This bit is only effective for analog filters type B and digital filters type D.
2	FCLAn INTLm	Level/edge detection mode selection 0: edge detection enabled 1: level detection enabled **Note:** This bit is only effective for analog filters type A.
1	FCLAn INTFm	• In level detection mode (FCLAnINTLm = 1): FCLAnINTFm has no effect • In edge detection mode (FCLAnINTLm = 0): falling edge detection control 0: falling edge detection disabled 1: falling edge detection enabled **Note:** This bit is only effective for analog filters type A.
0	FCLAn INTRm	• In level detection mode (FCLAnINTLm = 1): detection level selection 0: low level detection 1: high level detection • In edge detection mode (FCLAnINTLm = 0): rising edge detection control 0: rising edge detection disabled 1: rising edge detection enabled **Note:** This bit is only effective for analog filters type A.

(2) DNFAnCTL – Digital noise elimination control register

This register specifies the filter characteristics of the digital noise elimination filter.

Note This register is only effective for digital filters type D.

Access This register can be read/written in 8-bit units.

Address The assignment of the input signals to the DNFAnCTL registers and their addresses is given in the tables in the previous section *"Port filters assignments"*.

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	DNFAnNFSTS[1:0]		0	0	DNFAnPRS[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-56 DNFAnCTL register contents

Bit position	Bit name	Function
6 to 5	DNFAn NFSTS[1:0]	DNFAnNFSTS[1:0] specifies the number of same level samples, i.e. the number of samples with the same level to judge an external signal pulse as valid.
2 to 0	DNFAn PRS[2:0]	Digital filter sampling clock selection

DNFAnNFSTS[1:0]	Number of same level samples
00_B	2 samples
01_B	3 samples
10_B	4 samples
11_B	5 samples

DNFAnPRS[2:0]	Sampling clock frequency
000_B	DNFATCKI / 1
001_B	DNFATCKI / 2
010_B	DNFATCKI / 4
011_B	DNFATCKI / 8
100_B	DNFATCKI / 16
101_B	DNFATCKI / 32
110_B	DNFATCKI / 64
111_B	DNFATCKI / 128

(3) DNFAnEN – Digital noise elimination enable register

This register enables/disables the digital noise elimination for a certain input signal.

Note This register is only effective for digital filters type D.

Access This register can be read/written in 16-bit units.
The high byte DNFAnNFEN[15:8] and low byte DNFAnNFEN[7:0] can also be separately accessed in 8-bit units via the registers DNFAnENH.DNFAnNFEN[15:8] and DNFAnENL.DNFAnNFEN[7:0].

Address The assignment of the input signals to DNFAnEN registers and their addresses is given in the tables in the previous section *"Port filters assignments"*.

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
DNFAn NFEN15	DNFAn NFEN14	DNFAn NFEN13	DNFAn NFEN12	DNFAn NFEN11	DNFAn NFEN10	DNFAn NFEN9	DNFAn NFEN8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DNFAn NFEN7	DNFAn NFEN6	DNFAn NFEN5	DNFAn NFEN4	DNFAn NFEN3	DNFAn NFEN2	DNFAn NFEN1	DNFAn NFEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 2-57 DNFAnEN register contents

Bit position	Bit name	Function
15 to 0	DNFAn NFEN[15:0]	Digital noise elimination control 0: digital noise elimination disabled 1: digital noise elimination enabled

Chapter 3 CPU System Functions

This chapter describes the registers of the CPU, the operation modes, the address space and the memory areas.

3.1 Overview

The CPU is founded on Harvard architecture and it supports a RISC instruction set. Basic instructions can be executed in one clock period. Optimized seven-stage pipelining is supported. This improves instruction execution speed.

In order to make the microcontroller ideal for use in digital control applications, a 32-bit hardware multiplier enables this CPU to support multiply instructions, saturated multiply instructions, bit operation instructions, etc.

CPU

- CPU
  - Core: V850E2M CPU
  - Architecture: V850E2-V3 Architecture Class
- Instruction execution times:

Device	Minimum instruction execution time[a]	Maximum CPU clock[a]
V850E2/Fx4-H	6.25 ns	160 MHz

a) The maximum CPU clock exceeds the given values, if the CPU clock is dithering due to usage of a Spread Spectrum PLL (SSCG), and thus the minimum instruction execution time is shortened.

- 32 x 32 bits general registers
- 7 stage pipeline
- 2 way superscalar
- Internal 32-bit architecture
- 4 GB linear address space for program and data

- Processor protection functions
  - Memory Protection Unit (MPU)
    Protection against illegal execution from or data manipulation of CPU memory areas (up to 5 areas in instruction space, up to 6 areas in data space)
  - System Register Protection (SRP)
    Protection against damage to the system registers by a non-trusted programs
  - Peripheral Protection Unit (PPU)
    Protection against illegal access to a peripheral modules
  - Timing Supervision Unit (TSU)
    Protection against inappropriate CPU time possession by a non-trusted program, and resources and time of disabling interrupts can be managed

Instruction set

- V850E2 instruction set compatible to former V850 instruction sets plus additional powerful instructions for reduced code size and increasing execution speed
- Signed multiplication operations in 1 clock
  - 16 bits x 16 bits → 32 bits
  - 32 bits x 32 bits → 32 bits or 64 bits
  - 32 bits x 32 bits → 64 bits
- Saturated operation instructions with overflow/underflow detection
- 32-bit shift instructions in1 clock
- Bit manipulation instructions (bit set, clear, not, test)
- Load/store instructions with long/short format
- Signed load instructions
- MAC operation
  32 bit x 32 bit + 64 bit → 64 bit
- Floating-point operations conformance to ANSI/IEEE standard 754-1985 (IEEE binary floating-point operation standard)

3.2 Peripheral Protection Unit

PPU base address

The addresses of the Peripheral Protection Unit registers description in the “V850E2M 32-bit Microprocessor Core Architecture User’s Manual” are give as offset addresses. The base address is give below:

<PPU_base> = FFFF 5100_H

PPU areas and registers

The control registers for each protected area comprises 4 registers:

- PPVn – Validating general peripheral device protection
- PPTn – Specification protection type of general peripheral device
- PPPn – Specification of OS peripheral device
- PPSn – Specification of special peripheral device

These registers are numbered with n = 0 to 4.

The bits in these 32-bit wide registers are named

- PPVn.PPVnm – Validating general peripheral device protection
- PPTn.PPTnm – Specification protection type of general peripheral device
- PPPn.PPPnm – Specification of OS peripheral device
- PPSn.PPSnm – Specification of special peripheral device

with m = 0 to 31.

The table below defines the protected address ranges, their control registers and bits and the modules in the respective address area.

Table 3-1 PPU protected areas and modules (1/3)

Protection range size	Protection control		Module name	Address range
	Registers PPVn, PPTn, PPPn, PPSn n =	Bits PPVnm, PPTnm, PPPnm, PPSnm m =		
1120 B	0	0	INTC	FFFF 6000_H - FFFF $645F_H$
8 B		1	Code flash cache control	FFFF 6480_H - FFFF 6487_H
256 B		22	MEMC	FFFF 7200_H - FFFF $72FF_H$
256 B		23	DMAC (DMAC	FFFF 7300_H - FFFF $73FF_H$
256 B		24		FFFF 7400_H - FFFF $74FF_H$
512 B		25		FFFF 7500_H - FFFF $76FF_H$
512 B		26		FFFF 7700_H - FFFF $78FF_H$
512 B		27		FFFF 7900_H - FFFF $7AFF_H$
512 B		28		FFFF $7B00_H$ - FFFF $7CFF_H$
512 B		29		FFFF $7D00_H$ - FFFF $7EFF_H$

Table 3-1 PPU protected areas and modules (2/3)

Protection range size	Protection control		Module name	Address range
	Registers PPVn, PPTn, PPPn, PPSn n =	Bits PPVnm, PPTnm, PPPnm, PPSnm m =		
64 KB	1	0	Port Pnm control	FF40 0000$_H$ - FF40 FFFF$_H$
		1	Port filters control	FF41 0000$_H$ - FF41 FFFF$_H$
		2	Clock Controller Stand-by Controller Reset Controller	FF42 0000$_H$ - FF42 FFFF$_H$
		4	Port JPnm control	FF44 0000$_H$ - FF44 FFFF$_H$
		5	FE level non-maskable interrupt sharing	FF45 0000$_H$ - FF45 FFFF$_H$
		8	FCN0	FF48 0000$_H$ - FF48 FFFF$_H$
		9		FF49 0000$_H$ - FF49 FFFF$_H$
		10	FCN1	FF4A 0000$_H$ - FF4A FFFF$_H$
		11		FF4B 0000$_H$ - FF4B FFFF$_H$
		12	FCN2	FF4C 0000$_H$ - FF4C FFFF$_H$
		13		FF4D 0000$_H$ - FF4D FFFF$_H$
		14	FCN3	FF4E 0000$_H$ - FF4E FFFF$_H$
		15		FF4F 0000$_H$ - FF4F FFFF$_H$
		16	FCN4	FF50 0000$_H$ - FF50 FFFF$_H$
		17		FF51 0000$_H$ - FF51 FFFF$_H$
		18	DCN0	FF52 0000$_H$ - FF52 FFFF$_H$
		19		FF53 0000$_H$ - FF53 FFFF$_H$
		24	FLX0	FF58 0000$_H$ - FF58 FFFF$_H$
		28	URTE0/LMA0	FF5C 0000$_H$ - FF5C FFFF$_H$
		29	URTE1/LMA1	FF5D 0000$_H$ - FF5D FFFF$_H$
		30	URTE2/LMA2	FF5E 0000$_H$ - FF5E FFFF$_H$
		31	URTE3/LMA3	FF5F 0000$_H$ - FF5F FFFF$_H$
64 KB	2	0	URTE4/LMA4	FF60 0000$_H$ - FF60 FFFF$_H$
		1	URTE5/LMA5	FF61 0000$_H$ - FF61 FFFF$_H$
		2	URTE6/LMA6	FF62 0000$_H$ - FF62 FFFF$_H$
		3	URTE7/LMA7	FF63 0000$_H$ - FF63 FFFF$_H$
		4	URTE8/LMA8	FF64 0000$_H$ - FF64 FFFF$_H$
		5	URTE9/LMA9	FF65 0000$_H$ - FF65 FFFF$_H$
		6	URTE10/LMA10	FF66 0000$_H$ - FF66 FFFF$_H$
		7	URTE11/LMA11	FF67 0000$_H$ - FF66 FFFF$_H$
		12	CSIH0	FF6C 0000$_H$ - FF6C FFFF$_H$
		13	CSIH1	FF6D 0000$_H$ - FF6D FFFF$_H$
		14	CSIH2	FF6E 0000$_H$ - FF6E FFFF$_H$
		16	CSIG0	FF70 0000$_H$ - FF70 FFFF$_H$
		18	CSIG2	FF72 0000$_H$ - FF72 FFFF$_H$

Table 3-1 PPU protected areas and modules (3/3)

Protection range size	Protection control		Module name	Address range
	Registers PPVn, PPTn, PPPn, PPSn n =	Bits PPVnm, PPTnm, PPPnm, PPSnm m =		
		20	CSIG4	FF74 0000$_H$ - FF74 FFFF$_H$
		22	Back-up RAM	FF76 0000$_H$ - FF76 FFFF$_H$
		23	FE level maskable interrupt selection, TAUAn input selections, TAUJn input selections, FCNn signal selections,	FF77 0000$_H$ - FF77 FFFF$_H$
4 KB	3	0	OSTM0	FF80 0000$_H$ - FF80 0FFF$_H$
		2	CLMA0	FF80 2000$_H$ - FF80 2FFF$_H$
		3	CLMA1	FF80 3000$_H$ - FF80 3FFF$_H$
		4	CLMA2	FF80 4000$_H$ - FF80 4FFF$_H$
		5	CLMA3	FF80 5000$_H$ - FF80 5FFF$_H$
		6	WDTA0	FF80 6000$_H$ - FF80 6FFF$_H$
		7	WDTA1	FF80 7000$_H$ - FF80 7FFF$_H$
		8	TAUA0	FF80 8000$_H$ - FF80 8FFF$_H$
		9	TAUB1	FF80 9000$_H$ - FF80 9FFF$_H$
		10	TAUB2	FF80 A000$_H$ - FF80 AFFF$_H$
		11	TAUC3	FF80 B000$_H$ - FF80 BFFF$_H$
		12	TAUC4	FF80 C000$_H$ - FF80 CFFF$_H$
		13	TAUC5	FF80 D000$_H$ - FF80 DFFF$_H$
		14	TAUC6	FF80 E000$_H$ - FF80 EFFF$_H$
		15	TAUC7	FF80 F000$_H$ - FF80 FFFF$_H$
		16	PMCA0, DLYA0	FF81 0000$_H$ - FF81 0FFF$_H$
		17	TAUJ0	FF81 1000$_H$ - FF81 1FFF$_H$
		18	TAUJ1	FF81 2000$_H$ - FF81 2FFF$_H$
		20	RTCA0	FF81 4000$_H$ - FF81 4FFF$_H$
		21	TAPA0	FF81 5000$_H$ - FF81 5FFF$_H$
		24	VCPC0	FF81 8000$_H$ - FF81 8FFF$_H$
		25	ENCA0	FF81 9000$_H$ - FF81 9FFF$_H$
		27	FOUT	FF81 B000$_H$ - FF81 BFFF$_H$
		29	ADCA0	FF81 D000$_H$ - FF81 DFFF$_H$
		30	ADCA1	FF81 E000$_H$ - FF81 EFFF$_H$
		31	DCRA0	FF81 F000$_H$ - FF81 FFFF$_H$
4 KB	4	0	IICAB0	FF82 0000$_H$ - FF82 0FFF$_H$
		10	RNGA0	FF82 A000$_H$ - FF82 AFFF$_H$
		11	KR0	FF82 B000$_H$ - FF82BFFF$_H$

3.3 Timing Supervision Unit

TSU base address

The addresses of the Timing Supervision Unit registers description in the “V850E2M 32-bit Microprocessor Core Architecture User’s Manual” are give as offset addresses. The base address is give below:

<TSU_base> = FFFF 5000_H

3.4 Memory Protection Unit (MPU)

Caution

DPA5 is always used in "base/mask specification mode".

If the address of an accessed data is located below the configured upper boundary address of DPA5 and due to the length of the data, the access is crossing the upper boundary, an access violation will not be generated, although upper bytes of the unaligned access are located above the configured upper boundary of DPA5.

Therefore it is recommended to avoid any unaligned accesses, which could lead to crossing of the upper boundary of the DPA5 region.

Refer to the “V850E2M 32-bit Microprocessor Core Architecture User’s Manual” Architecture User’s Manual (R01US0001EJxxxx) for the DPA5U and DPA5L registers description.

3.5 CPU Access Bus Structures and Latencies

The CPU accesses the configuration, control and status registers of all functional modules of the microcontroller in different ways, depending on where they are located:

- modules of the CPU Subsystem:
  refer the section to *"CPU Subsystem modules access"* below
- modules externally to the CPU Subsystem (PBUS/HBUS modules):
  refer the section to *"PBUS/HBUS modules access"* below.

Note For a detailed description of the CPU Subsystem refer to the section *"CPU Subsystem"* below in this chapter.

3.5.1 CPU Subsystem modules access

For accessing the registers of modules on the CPU Subsystem the dedicated busses LSPB and GSPB are provided. Both are controlled only by the CPU.

Table 3-2 V850E2 CPU Subsystem control busses

Registers of module	CPU master bus	
	LSPB	GSPB
Interrupt Controller (INTC)	R/W	–
Timing Supervision Unit (TSU)	R/W	–
Peripheral Protection Unit (PPU)	R/W	–
DMA/DTS Controller (DMAC)	–	R/W
HBUS slave I/F	–	R/W
HBUS master I/F	–	R/W
PBUS I/F	–	R/W
Data flash I/F	–	R/W

H/W lock function The LSPB and GSPB busses support the hardware lock function.
Thus bit manipulation instructions (CLR1, NOT1, SET1, TST1) can be applied to all registers accessed via the LSPB and GSPB bus, provided the register allows 8-bit access.

Byte/halfword access The LSPB and GSPB busses support byte and halfword accesses.
Thus each byte or halfword of all registers accessed via the LSPB and GSPB bus can be accessed separately.

3.5.2 PBUS/HBUS modules access

The CPU communicates with PBUS and HBUS modules - these are not located on the CPU Subsystem - via the CPU Subsystem's PBUS and HBUS master interfaces.

H/W lock function

The PBUS and HBUS master I/F do not support the hardware lock function.

PBUS byte/halfword access

All PBUS modules are accessed on 32-bit word aligned addresses. Further each read or write is carried out with 32-bit width on the bus.

However the CPU can also perform byte (load/store byte) and halfword (load/store halfword) access instructions with the following results:

- Byte read accesses
  - The CPU instructions LD.B and SLD.B (load of signed byte data) loads the word aligned byte to bits [7:0] of a CPU general purpose register. The general purpose register bits [31:8] will be sign extended.
  - The CPU instructions LD.BU and SLD.BU (load of unsigned byte data) loads the word aligned byte to bits [7:0] of a CPU general purpose register. The general purpose register bits [31:8] will be zero-extended.
- Halfword read accesses
  - The CPU instructions LD.H and SLD.H (load of signed halfword data) loads the word aligned halfword to bits [15:0] of a CPU general purpose register. The general purpose register bits [31:16] will be sign extended.
  - The CPU instructions LD.HU and SLD.H loads the word aligned halfword to bits [15:0] of a CPU general purpose register. The general purpose register bits [31:16] will be zero-extended.
- Write access
  A byte (CPU instructions ST.B, SST.B) or halfword (CPU instructions ST.H, SST.H) write access corrupts the not written bytes of the word address.

Caution Do not perform byte or halfword write accesses, unless the write target is only 8 bit (for byte access) or 16 bit respectively (for halfword access) wide, so that the upper 3 bytes or 2 bytes respectively can not become corrupted.

HBUS access

All accesses via the HBUS have to be performed in 32-bit width.
Thus halfword and byte accesses are not permitted.

Bus clocks

The CPU Subsystem is supplied with the CKSCLK_000 clock (CPUCLK). If the CPU (respectively the DMA Controller) accesses modules, external to the CPU Subsystem, these modules are members of different clock domains. Thus the bus clocks may have to be synchronized, when the CPU accesses the concerned modules.

This section explains the bus structures and its clocks and defines access latencies, when different bus clocks have to be synchronized.

Bus clocks notation

The bus clocks PCLK and HCLK have an index, that indicates the clock, the bus is operating with. For instance, $PCLK_{000}$ denotes a PBUS with the PCLK clock synchronous to CKSCLK_000, or in general:

- $PCLK_{mn}$ is a PBUS clock synchronous to CKSCLK_mn
- $HCLK_{mn}$ is a HBUS clock synchronous to CKSCLK_mn

Caution

Since access to modules on clock domains with bus synchronizers may induce latencies of several cycles of the CPUCLK, the flow of the CPU program may become inconsistent.
That means a CPU instruction, that accesses CPU Subsystem modules (in particular INTC, DMAC), follows an instruction, that accesses a module with bus synchronizer latencies, the access to the CPU Subsystem module may become effective before the first one (module access via bus synchronizer).

Refer to the further descriptions in this section for details.

(1) V850E2/Fx4-H CPU access bus structures

The following diagram shows the bus structure for CPU accesses to other modules of the V850E2/Fx4-H devices.

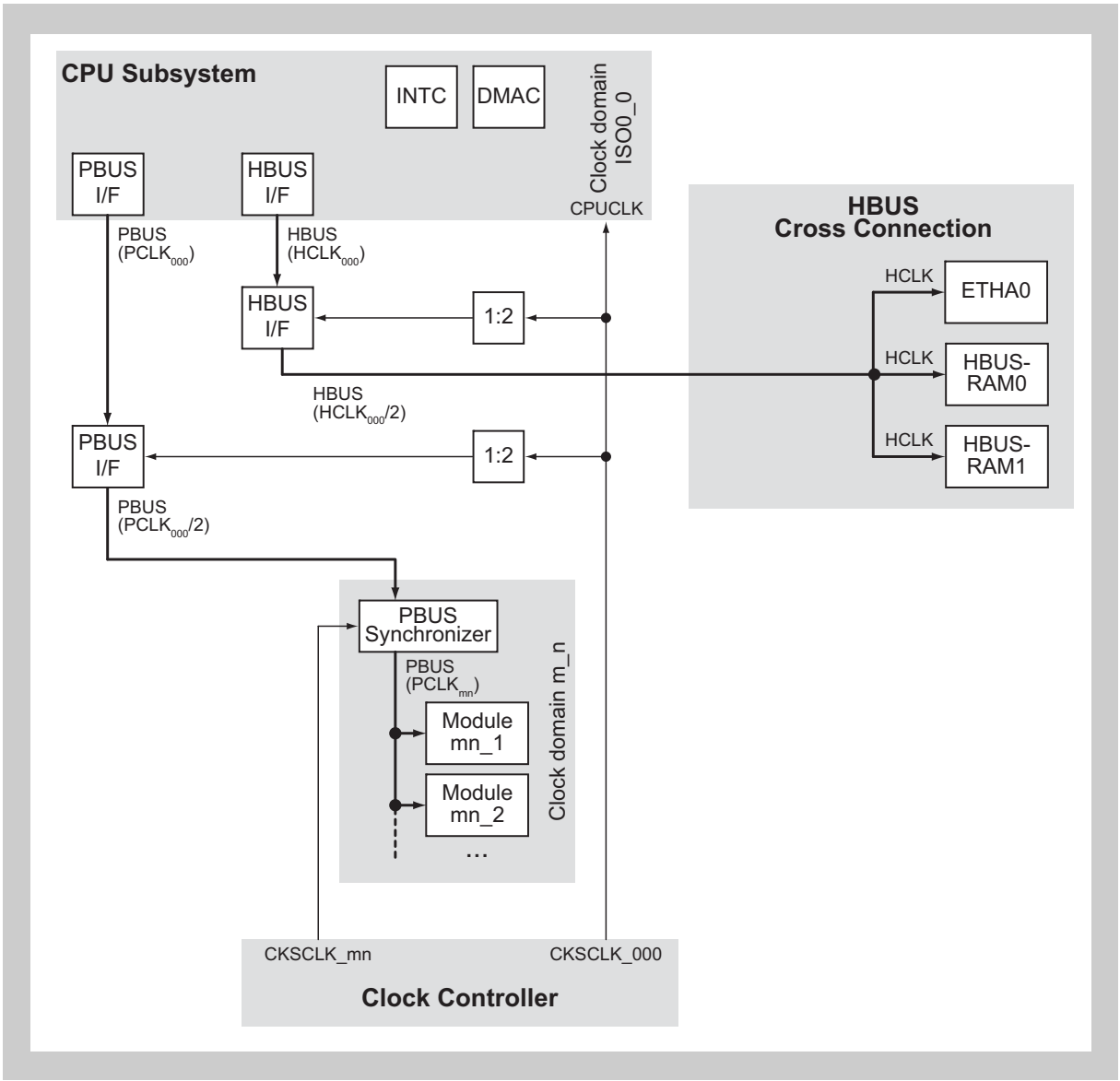


Figure 3-1 V850E2/Fx4-H CPU access bus structures

The CPU accesses

- all PBUS modules with the $PCLK_{000}/2$ clock
- all HBUS modules with the $HCLK_{000}/2$ clock

CPU Subsystem modules

The CPU accesses modules of the CPU Subsystem (DMA Controller, Interrupt Controller) via CPU Subsystem internal busses. The bus clock of these busses is CPUCLK (CKSCLK_000). Thus no bus synchronization is required.

Clock domains m_n

All PBUS interfaces are supplied with clocks from different clock domains CKSCLK_mn.

Thus CPU accesses with $PCLK_{000}/2$ = CKSCLK_000/2 to the clock domain's m_n PBUS interfaces are synchronized to the target clock domain CKSCLK_mn.
Refer to the section *"PBUS Synchronizer"* below for information about the PBUS synchronizers latencies.

HBUS modules on HBUS Cross Connection

All HBUS interfaces of modules of the HBUS Cross Connection system are directly supplied with CKSCLK_000/2, that is synchronous to the CPUCLK domain clock CKSCLK_000. Thus no synchronizer is required.

3.5.3 PBUS Synchronizer

The PBUS synchronizers synchronize the bus clock $PCLK_{000}/2$ from the CPU to the target clock domain's PBUS clock $PCLK_{mn}$.

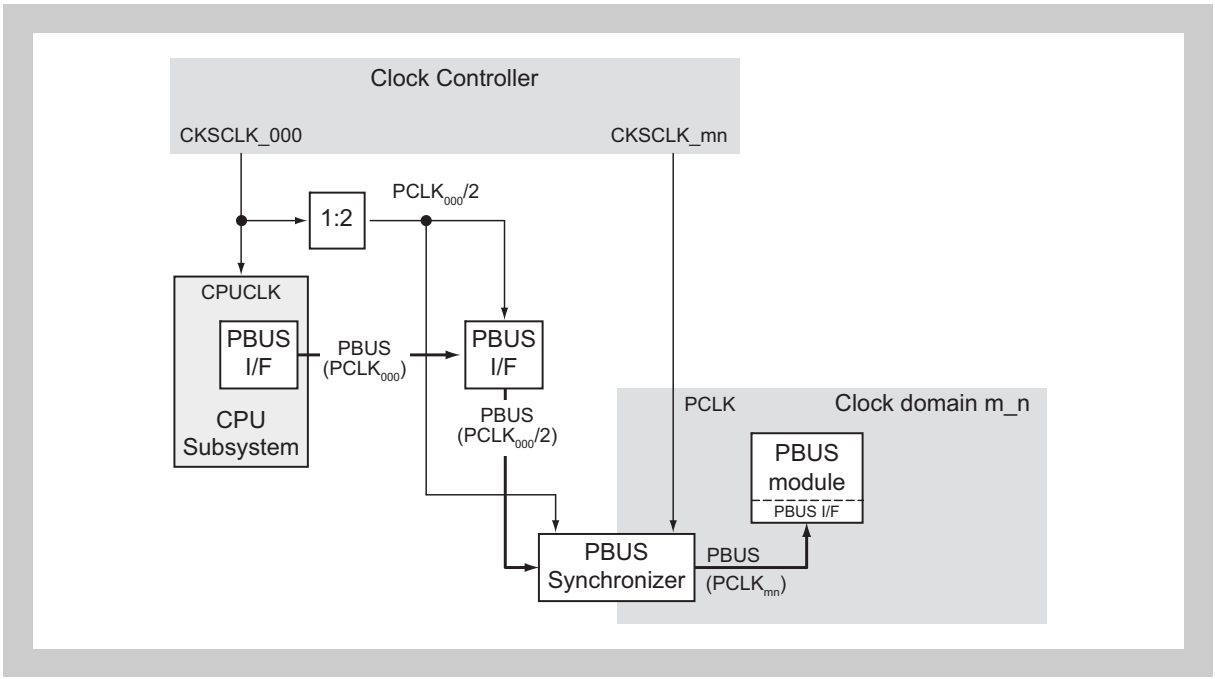


Figure 3-2 PBUS synchronizer

The synchronizers generate latencies during CPU accesses to the PBUS modules on the clock domain m_n.

The maximum latency of a clock domain m_n PBUS access is calculated by the following formula:

$$\text{Latency cycles} = 4 \times PCLK_{000}/2 + 4 \times PCLK_{mn}$$

Notes

1. The synchronization latency always applies, even if both clocks are configured to use same clock source or frequency.
2. The bus synchronizers are shown in the clock domain figures in section *"Clock Domain Figures"* in the chapter *"Clock Controller"*.

(1) Overall PBUS latencies

The overall access latency of a CPU access to a PBUS module is calculated as follows:

- Read access:

$$T_{accRD} = 16 \cdot PCLK_{000} + 6 \cdot PCLK_{mn} + W_{RD} \cdot PCLK_{mn}$$

- Write access:

$$T_{accWR} = 6 \cdot PCLK_{000} + 6 \cdot PCLK_{mn} + W_{WR} \cdot PCLK_{mn}$$

W_{RD} and W_{WR} specify the number of wait clocks, induced by the PBUS module upon read respectively write access. Refer to the section *"Module wait clocks insertion"* below for the W_{RD} and W_{WR} values of the different modules.

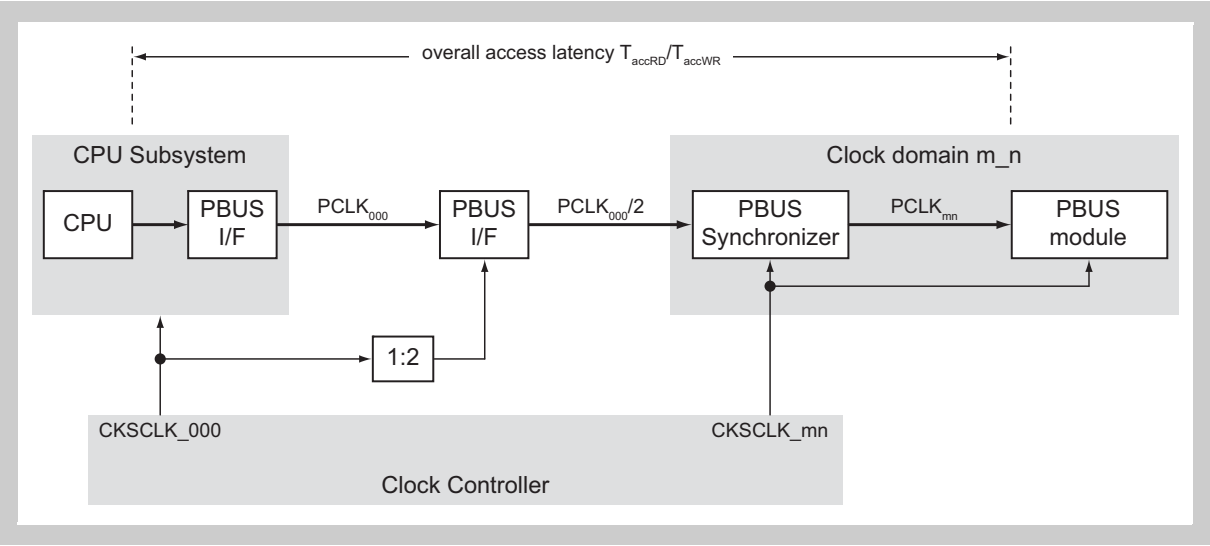


Figure 3-3 Overall PBUS access latency

CPU write access

If the PBUS is not occupied, i.e. the PBUS is idle, the write access is immediately forwarded and the CPU continues with the next instruction.

If the PBUS is occupied because of a former CPU write or a DMA access, the CPU is stopped until the PBUS is idle.

CPU read access

Upon a CPU read access to any PBUS module, CPU operation stops until the requested data is available.

3.5.4 Module wait clocks insertion

The following modules insert wait states when accessed by the CPU.

Note All other module registers, which are not listed in the following table, do not inserted any wait clocks (W_{RD} = W_{WR} = 0).

Table 3-3 Module registers access wait clocks

Module	Registers	Read wait clocks W_{RD}	Write wait clocks W_{WR}
FlexRay (FLX)	All registers	1 PCLK	0 PCLK
	FlexRay RAM write through input buffer: FLXnWRDSm FLXnWRH[3:1]	n.a. (write only)	0 PCLK
	FlexRay RAM read through output buffer: FLXnRDDSm FLXnRDH[3:1]	2 PCLK	n.a. (read only)
CAN Controller (FCN)	FCNnMmDAT[7:0]B FCNnMmDTLGB FCNnMmSTRB FCNnDNBMRX[3:0] FCNnMmCTL FCNnCMLISTR FCNnCMLOSTR	2 PCLK	2 PCLK
	FCNnCMRGRX FCNnCMTGTX FCNnMmDAT[6,4,2,0]H FCNnMmMID0H FCNnMmMID1H	2 PCLK	1 PCLK
	FCNnMmDAT[4,0]W FCNnMmMID0W	3 PCLK	2 PCLK
	FCNnCMCLCTL.FCNnCMCLSERC	n.a. (write only)	1 to 10 PCLK
	FCNnCMCLCTL except FCNnCMCLSERC bit	0 PCLK	3 PCLK
Diagnostic CAN Controller (DCN)	DCNnMmDAT[7:0]B DCNnMmDTLGB DCNnMmSTRB DCNnDNBMRX[3:0] DCNnMmCTL DCNnCMLISTR DCNnCMLOSTR	2 PCLK	2 PCLK
	DCNnCMRGRX DCNnCMTGTX DCNnMmDAT[6,4,2,0]H DCNnMmMID0H DCNnMmMID1H	2 PCLK	1 PCLK
	DCNnMmDAT[4,0]W DCNnMmMID0W	3 PCLK	2 PCLK
	DCNnCMCLCTL.DCNnCMCLSERC	n.a. (write only)	1 to 10 PCLK
	DCNnCMCLCTL except DCNnCMCLSERC bit	0 PCLK	3 PCLK

3.6 CPU Subsystem

This section gives an overview about the CPU Subsystems.

The CPU Subsystems comprise

- the CPU and CPU dedicated components, e.g. the processor protection functions
- busses to instruction and data memory
- various interfaces to the other microcontroller modules, e.g. the PBUS and data flash interface
- the Interrupt Controller (INTC)
- the DMA/DTS Controller (DMAC)
- On-chip Debug unit
- several bus systems with bus arbiters that allow access of the bus masters to all other modules

3.6.1 Power and clock domain

The CPU Subsystem lies on the Isolated-Area-0 and its clock CPUCLK is supplied from the domain clock CKSCLK_000.

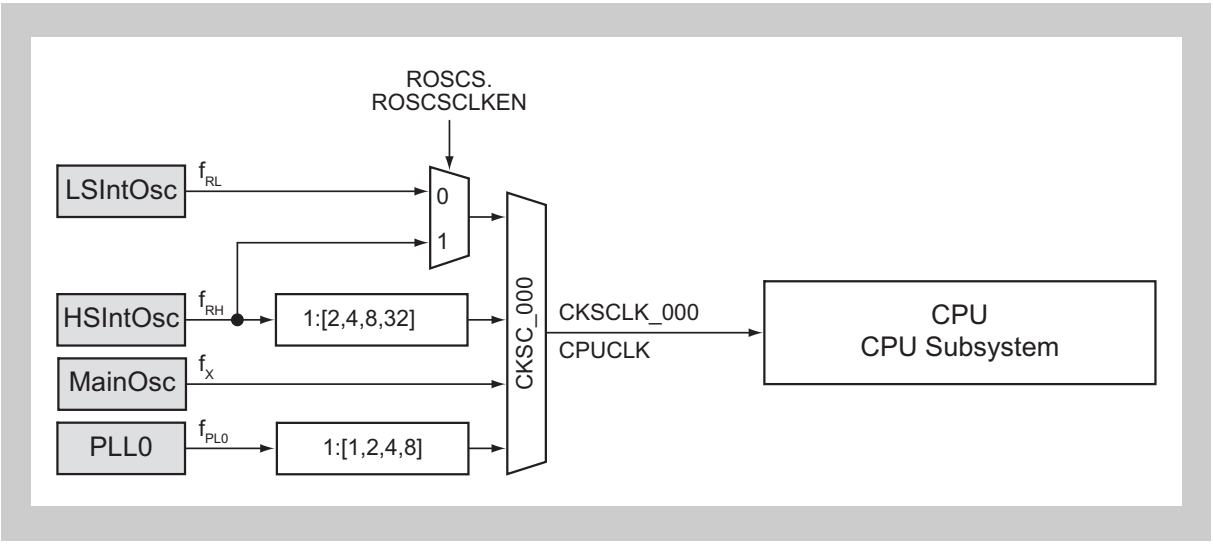


Figure 3-4 CPU Subsystem clock supply

CPU Subsystem H/W reset

The CPU Subsystem and its registers are initialized by the following reset signal:

Table 3-4 CPU Subsystem reset signal

CPU Subsystem	Reset signal
CPU Subsystem	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

3.6.2 CPU Subsystem busses overview

The table below gives an overview about the CPU Subsystem busses and their general purpose.

Refer to the following section for detailed information about the CPU Subsystem.

Table 3-5 Bus systems

CPU Subsystem	Bus	Purpose	Bus master
internal	Local System Peripheral Bus LSPB	Access to registers of • Time Supervision Unit TSU • Peripheral Protection Unit PPU • Interrupt Controller INTC	CPU
	CPU system bus	CPU access to • external HBUS slaves (via HBUS master I/F) • PBUS/GSPB bus • Data flash	CPU
	DMA data bus	DMAC access to HBUS • Data RAM • external HBUS slaves (via HBUS master I/F) • PBUS/GSPB bus • Data flash	DMAC
	HBUS data bus	External HBUS master access (via HBUS slave I/F) to • Data RAM • Code flash • external HBUS slaves (via HBUS master I/F) • PBUS/GSPB bus • Data flash	external HBUS master (via HBUS slave I/F)
	Global System Peripheral Bus GSPB	Access to registers of • DMA/DTS Controller DMAC • HBUS bridge • PBUS I/F • data flash I/F	CPU, DMAC, external HBUS master (via HBUS slave I/F)[a]
external	PBUS	Access to all PBUS modules via the PBUS master I/F	CPU, DMAC, external HBUS master (via HBUS slave I/F)[a]
	HBUS	Access to all HBUS slave modules via the HBUS master I/F	CPU, DMAC, external HBUS master (via HBUS slave I/F)[a]

a) Refer to the following sections more information about the buses arbitration.

3.6.3 V850E2K/Fx4-H CPU Subsystem

The diagram below shows a CPU Subsystem block diagram of the V850E2K/Fx4-H product series.

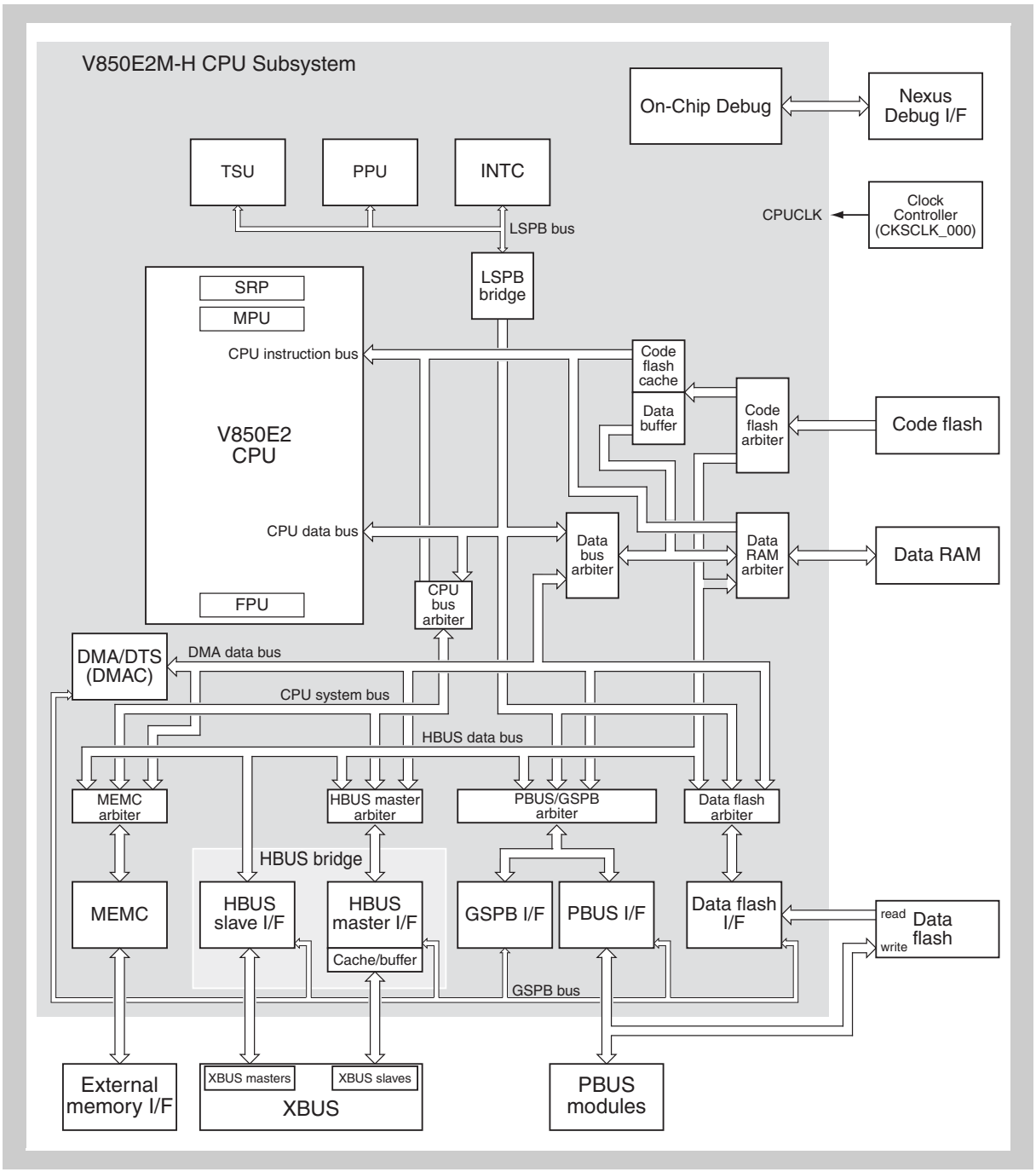


Figure 3-5 V850E2/Fx4-H CPU Subsystem

All busses for transferring data between the various modules are controlled by three masters:

- CPU
- DMA/DTS Controller (DMAC)
- HBUS master via HBUS slave I/F

Table 3-6 V850E2/Fx4-H CPU Subsystem data/instruction busses

Master	Bus	Code flash	Data RAM	Data flash	PBUS/ GSPB I/F	HBUS master I/F	MEMC
CPU	Instruction (128 bit)	R	R	–	–	–	R[a]
	Data (32 bit)	R	R/W	R/W	R/W	R/W	R/W
DMA/DTS Controller	DMA data bus (32 bit)	R	R/W	R	R/W	R/W	R/W
HBUS slave I/F	HBUS data bus (32 bit)	R	R/W	R	R/W	R/W	R/W

a) CPU instruction fetches from these sources may take a considerable amount of CPU clock cycles, resulting in a loss of CPU performance. Thus instruction fetches from these sources are not recommended.

(1) Code flash access

Table 3-7 Code flash access

<table>
<tr><td rowspan="4">Code flash</td><td rowspan="4">Code flash arbiter/ Data buffer</td><td rowspan="4">Code flash cache</td><td colspan="2">← CPU instruction bus</td></tr>
<tr><td rowspan="3">Data bus arbiter</td><td>← CPU data bus</td></tr>
<tr><td>← DMA data bus</td></tr>
<tr><td colspan="2">← HBUS data bus</td></tr>
</table>

(2) Data RAM access

Table 3-8 Data RAM access

Data RAM	Data RAM arbiter	← CPU instruction bus	
		Data bus arbiter	← CPU data bus
			← DMA data bus
		← HBUS data bus	

Caution Before fetching any instruction code from the data RAM, make sure to initialize the 16-byte boundary area of the data RAM that contains the instruction code to fetch.

A 16-byte boundary area is an area from the address XXXX $XXX0_H$ to XXXX $XXXF_H$.

For initializing the data RAM, any data values can be written, but the data RAM area must be initialized before an instruction is fetched from it.

If an instruction is fetched from an uninitialized data RAM area, a memory protection exception (MEP) might occur.

Note In general, it is recommended to initialize all the data RAM before reading it.

(3) Data flash I/F access

Table 3-9 Data flash I/F access

Data flash I/F	Data flash arbiter	← CPU data bus
		← HBUS data bus
		← DMA data bus

Data flash wait cycles Concerning data flash access refer also to the section *"Data flash wait cycle control"* below in this chapter.

(4) PBUS I/F, GSPB I/F access

Table 3-10 PBUS I/F, GSPB I/F access

PBUS I/F GSPB I/F	PBUS/ GSPB arbiter	← CPU data bus
		← HBUS data bus
		← DMA data bus

(5) HBUS master I/F access

Table 3-11 HBUS master I/F access

<table>
<tr><td rowspan="4">HBUS master I/F</td><td rowspan="4">HBUS master arbiter</td><td rowspan="2">CPU bus arbiter</td><td>← CPU instruction bus</td></tr>
<tr><td>← CPU data bus</td></tr>
<tr><td colspan="2">← HBUS data bus</td></tr>
<tr><td colspan="2">← DMA data bus</td></tr>
</table>

(6) MEMC access

Table 3-12 MEMC access

<table>
<tr><td rowspan="4">MEMC</td><td rowspan="4">MEMC master arbiter</td><td rowspan="2">CPU bus arbiter</td><td>← CPU instruction bus</td></tr>
<tr><td>← CPU data bus</td></tr>
<tr><td colspan="2">← HBUS data bus</td></tr>
<tr><td colspan="2">← DMA data bus</td></tr>
</table>

(7) CPU Subsystem busses arbitration policies

The following tables specify the arbitration policies of the CPU Subsystem busses arbiters.

Table 3-13 1-stage arbiters arbitration policies

Arbiter	Policy	Masters
Code flash cache/Data buffer	Fixed priority	High: DMA/CPU data bus (via data bus arbiter)
		Low: CPU instruction bus
Code flash	Round robin	HBUS
		Code flash cache/Data buffer
Data RAM	Round robin	CPU instruction bus
		CPU data bus/DMA (via data bus arbiter)
		HBUS
Data bus	Fixed priority	High: DMA
		Low: CPU data bus
CPU bus	Fixed priority	High: CPU data bus
		Low: CPU instruction bus

Table 3-14 2-stage arbiters arbitration policies

Arbiter	Policy	Masters
HBUS master	Stage 1:	
	Round robin	HBUS
		CPU system bus (from CPU bus arbiter)
	Stage 2:	
	Fixed priority:	High: DMA
		Low: HBUS/CPU system bus (from stage 1)
MEMC	Stage 1:	
	Round robin	HBUS
		CPU system bus (from CPU bus arbiter)
	Stage 2:	
	Fixed priority:	High: DMA
		Low: HBUS/CPU system bus (from stage 1)
PBUS/GSPB	Stage 1:	
	Round robin	HBUS
		CPU data bus
	Stage 2:	
	Fixed priority:	High: DMA
		Low: HBUS/CPU data bus (from stage 1)
Data flash	Stage 1:	
	Round robin	HBUS
		CPU data bus
	Stage 2:	
	Fixed priority:	High: DMA
		Low: HBUS/CPU data bus (from stage 1)

(8) CPU Subsystem HBUS bridge

For details concerning the CPU Subsystem HBUS bridge refer to *3.14 “CPU Subsystem - HBUS Bridge” on page 187* .

3.6.4 V850E2 system manual

Detailed descriptions of the various CPU Subsystem functions can be found in the following documents:

Table 3-15 CPU Subsystem functions information sources

Function	V850E2M Architecture (R01US0001EJxxxx)	This manual (R01UH0002ED0200)
V850E2 CPU (including instruction set)	√	–
Floating-Point Unit (FPU)	√	–
Processor protection functions (MPU, SPR, PPU, TPU)	√	–
DMA/DTS Controller (DMAC)	–	√
Interrupt Controller (INTC)	–	√
HBUS bridge	–	√
Code/Data flash	–	√
Data RAM	–	√

3.7 Data flash wait cycle control

CPU read accesses to the data flash require insertion of some wait cycles to access the flash memory.

The minimum number of wait cycles, in units of CPU Subsystem clock cycles CPUCLK, depends on the CPUCLK frequency and has to be selected via the wait cycle control register DCLKWAIT, as shown in the table below.

Caution The data flash wait cycle control register DCLKWAIT has to be set before the first CPU access to the data flash.

Table 3-16 Data flash wait cycle setting

CPU clock frequency[a]	DCLKWAIT register	CPU clock wait cycles
≤ 10 MHz	≥ 04_H	≥ 4
≤ 20 MHz	≥ 05_H	≥ 5
≤ 30 MHz	≥ 06_H	≥ 6
≤ 40 MHz	≥ 07_H	≥ 7
≤ 50 MHz	≥ 08_H	≥ 8
≤ 60 MHz	≥ 09_H	≥ 9
≤ 70 MHz	≥ $0A_H$	≥ 10
≤ 80 MHz	≥ $0B_H$	≥ 11
≤ 90 MHz	≥ $0C_H$	≥ 12
≤ 100 MHz	≥ $0D_H$	≥ 13
≤ 110 MHz	≥ $0E_H$	≥ 14
≤ 120 MHz	≥ $0F_H$	≥ 15
≤ 130 MHz	≥ 10_H	≥ 16
≤ 140 MHz	≥ 11_H	≥ 17
≤ 150 MHz	≥ 12_H	≥ 18
≤ 160 MHz	≥ 13_H	≥ 19

a) If the CPU clock CPUCLK is supplied by the PLL0 in SSCG mode, the given CPU clock frequency refers to the center frequency.

(1) DCLKWAIT – data flash wait cycle control register

This register is used to select the number of wait cycles for accessing the data flash.

Access This register can be read/written in 8-bit units.

Address FF43 6000$_H$

Initial Value 17$_H$

7	6	5	4	3	2	1	0
0	0	0	WAIT[4:0]				
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-17 DCLKWAIT register contents

Bit position	Bit name	Function
4 to 0	WAIT[4:0]	Sets the number of wait cycles, as shown in the table below: 04$_H$: 4 wait cycles 05$_H$: 5 wait cycles 06$_H$: 6 wait cycles 07$_H$: 7 wait cycles 08$_H$: 8 wait cycles 09$_H$: 8 wait cycles 0A$_H$: 9 wait cycles 0B$_H$: 10 wait cycles 0C$_H$: 11 wait cycles 0D$_H$: 12 wait cycles 0E$_H$: 13 wait cycles 0F$_H$: 14 wait cycles 10$_H$: 15 wait cycles 11$_H$: 16 wait cycles 12$_H$: 17 wait cycles 13$_H$: 18 wait cycles all others: prohibited

3.8 HBUS Cross Connection System

The HBUS Cross Connection system interconnects the following functional modules:

- CPU Subssytem
- Ethernet Controller ETHA0
- HBUS-RAM0
- HBUS-RAM1

(1) HBUS clock supply

The HBUS Cross Connection and all connected modules are supplied by the CPU Subssytem clock domain CKSC_000 divided by 2. The maximum center frequency is 80 MHz.

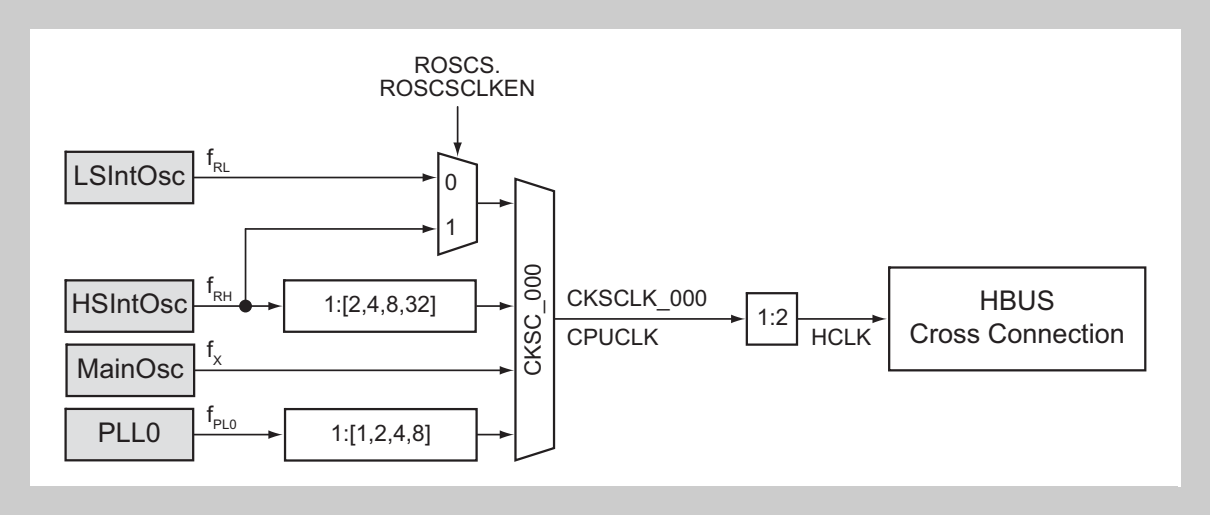


Figure 3-6 HBUS clock supply

(2) HBUS functional summary

The HBUS Cross Connection system enables several bus masters to access various slave modules.

The following diagram shows the HBUS participants and their data port properties.

Notes
1. The diagram below does not show any control ports, which give access to the modules registers.
2. The arrows in the diagram below show the direction of the data flow.

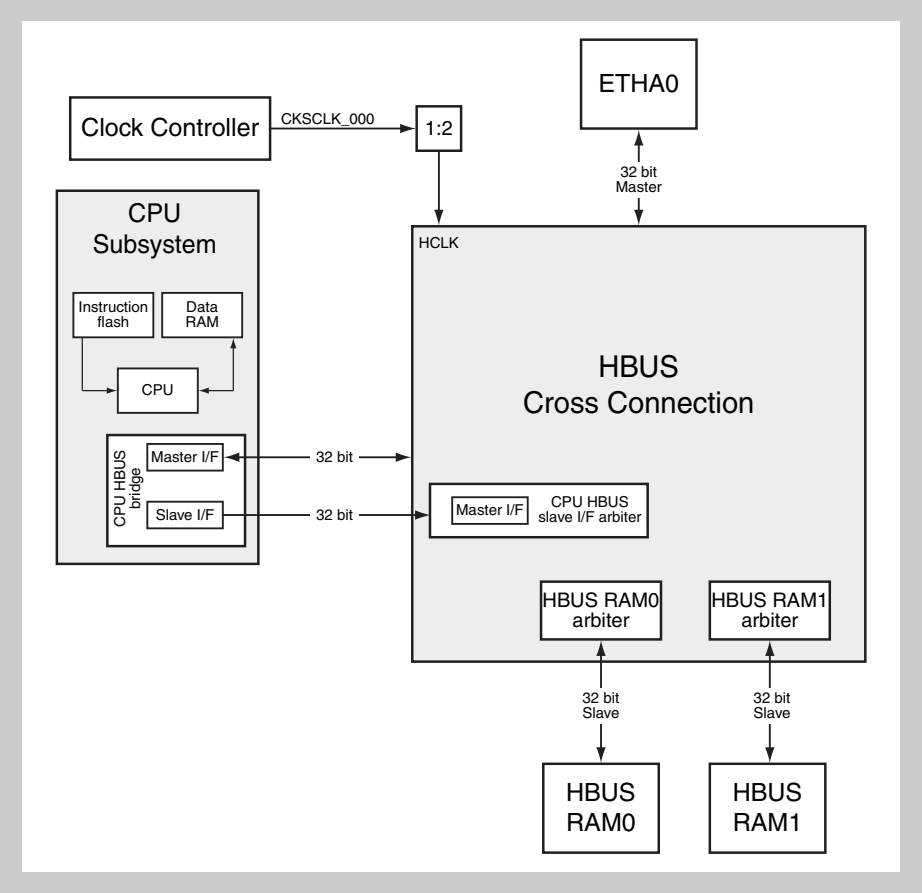


Figure 3-7 HBUS Cross Connection

(3) HBUS data ports

The HBUS data ports are used to interchange data between the different HBUS participants and are shown in the tables below.

Table 3-18 HBUS master data ports

Master ports	Width
Ethernet Controller data	32 bit
CPU Subsystem HBUS master I/F data	32 bit

Table 3-19 HBUS slave data ports

Slave ports	Width
HBUS-RAM0 data	32 bit
HBUS-RAM1 data	32 bit
CPU Subsystem HBUS slave I/F data	32 bit

3.8.1 HBUS connection matrix

The following diagram shows the control busses of the masters to the slaves.

Note The arrows in the diagram below show the direction of access control, thus do not show the data flow directions.

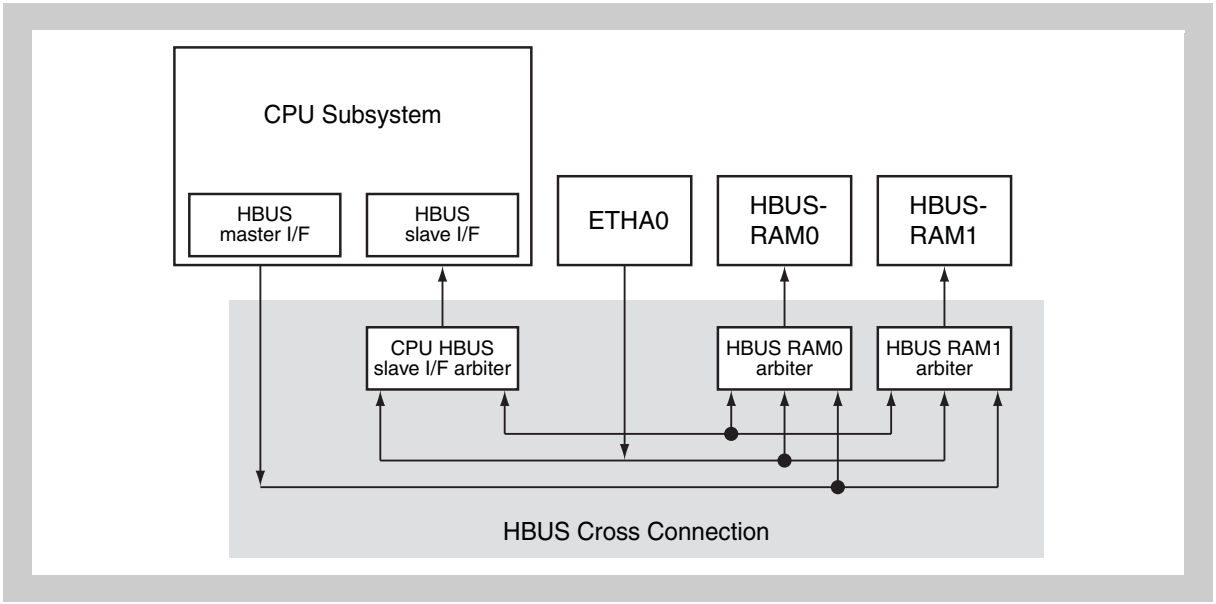


Figure 3-8 HBUS control busses

The HBUS Cross Connection system allows connectivity between the different bus masters and slaves as indicated in the following tables.

Table 3-20 HBUS data path cross connection matrix

Master data ports	Slave data ports		
	HBUS-RAM0	HBUS-RAM1	CPU Subsystem HBUS slave I/F
Ethernet Controller data	R/W	R/W	R/W
CPU Subsystem HBUS master I/F data	R/W	R/W	–

3.8.2 HBUS arbiters policy

The HBUS arbiters

- HBUS-RAM0 arbiter
- HBUS-RAM1 arbiter
- CPU Subsystem HBUS slave I/F arbiter

implement round robin arbitration policy.

3.9 Operation modes

This section describes the operation modes of the V850E2/Fx4-H and how the modes are selected.

The following operation modes are available:

- Normal operation mode
- Serial flash programming mode
- Boundary Scan mode

After release of the Power-On-Clear reset or external $\overline{\text{RESET}}$ the microcontroller starts to fetch instructions from an internal boot ROM which contains the internal firmware. The firmware checks the FLMD0 pin, and optionally also the FLMD1, MODE0, MODE1 and P0_4 pins, to set the operation mode after reset release according to the table below.

Table 3-21 Selection of operation modes

Pins					Operation mode
FLMD0	FLMD1 (P0_1)	MODE0 (P0_2)	MODE1 (P0_3)	P0_4	
VSS	VSS	X	X	X	Normal operation mode
	VDD	X	X	X	Setting prohibited
VDD	VSS	X	X	X	Serial flash programming mode
	VDD	VSS	VSS	VSS	Boundary Scan mode
		VDD	X	X	Setting prohibited

For starting the microcontroller in a certain operation mode properly the mode pins and the ports of the JP0 port group have to be set up correctly, as described in the section *"Mode pins and JP0 connections"* below.

3.9.1 Normal operation mode

In normal operation mode, the internal flash memory is not re-programmed.

After reset release, the firmware branches to the start of the active boot swap cluster. User program execution is started.

Debug mode Debug mode is basically a normal operation mode, but the debugger changes the device into debug mode via $\overline{\text{DCUTRST}}$ upon reset release.

3.9.2 Flash programming mode

In serial flash programming mode, the internal flash memory is erased and reprogrammed.

After reset release, the firmware initiates loading of the user's program code from the external flash programmer and programs the flash memory.

After detaching the external flash programmer, the microcontroller can be started up with the new user's program in normal operation mode.

For more information see chapter *"Flash Memory"*.

3.9.3 Boundary Scan mode

In Boundary Scan mode the microcontroller enters the interconnection test method according to IEEE Std. 1149.1.

3.10 Mode pins and JP0 connections

This section describes the wiring of the

- port group JP0 (JP0_0 to JP0_5)
- mode signals FLMD0, FLMD1 (P0_1), MODE0 (P0_2), MODE1 (P0_3)
- P0_4 port

in different microcontroller modes.

The function of the above mentioned pins depends on the mode:

Table 3-22 JP0 and mode pin functions

Pin/port	Operation mode				
	Normal operation	Debugger	Flash programming		Boundary scan
			Asynchronous	Synchronous	
JP0_0	I/O port	DCUTDI	FLUR0RTX	FLCS0SI	DCUTDI
JP0_1	I/O port	DCUTDO	I/O port	FLCS0SO	DCUTDO
JP0_2	I/O port	DCUTCK	I/O port	FLCS0SCI	DCUTCK
JP0_3	I/O port	DCUTMS	I/O port	I/O port	DCUTMS
JP0_4	I/O port	$\overline{\text{DCUTRST}}$	I/O port	I/O port	$\overline{\text{DCUTRST}}$
JP0_5	I/O port	$\overline{\text{DCUTRDY}}$	I/O port	I/O port	I/O port
FLMD0	FLMD0	FLMD0	FLMD0	FLMD0	FLMD0
P0_1	FLMD1/I/O port[a]	FLMD1/I/O port[a]	FLMD1/I/O port[a]	FLMD1/I/O port[a]	FLMD1/I/O port[a]
P0_2	I/O port	I/O port	I/O port	I/O port	MODE0
P0_3	I/O port	I/O port	I/O port	I/O port	MODE1
P0_4	I/O port	I/O port	I/O port	I/O port	VSS[b]

a) FLMD1 is used for mode setting, when the reset is released. Afterwards P0_1 can be used for general port and alternative functions.

b) In boundary scan mode low level must be input to P0_4.

Notes

1. FLMD0 has no port functionality and is exclusively used for mode setting.
2. Usage as I/O port means, the port can be used in port input/output and alternative modes.

Pull-up/pull-down resistors

The size of the resistors, shown in the following descriptions, depends on several parameters like current flow through the resistors in active level, secure level detection with maximum specified leakage current and maximum drive level of the driving port. Refer to the Electrical Target Specification for details.

3.10.1 Normal operation mode

In normal operation mode the JP0 port group and the ports used for MODE0, MODE1 and P0_4 can be used for general port and alternative functions.

Normal operation mode is selected by

FLMD0	FLMD1 (P0_1)[a]	MODE0 (P0_2)	MODE1 (P0_3)	P0_4
low level	low level	don't care	don't care	don't care

a) FLMD1 can be used for general port and alternative functions after reset release.

FLMD0 has to be low level and therefore needs to be connected to GND via the resistor R1.

FLMD1 and $\overline{\text{DCUTRST}}$ have to be low level at reset release and therefore need to be connected to GND via the resistors R2. Afterwards P0_1 and JP0_4 can be used for general port and alternative functions.

The target value of R1 is 82 kΩ. For the exact dimensioning of the FLMD0 pull-down resistor R1 refer to the Electrical Target Specification.

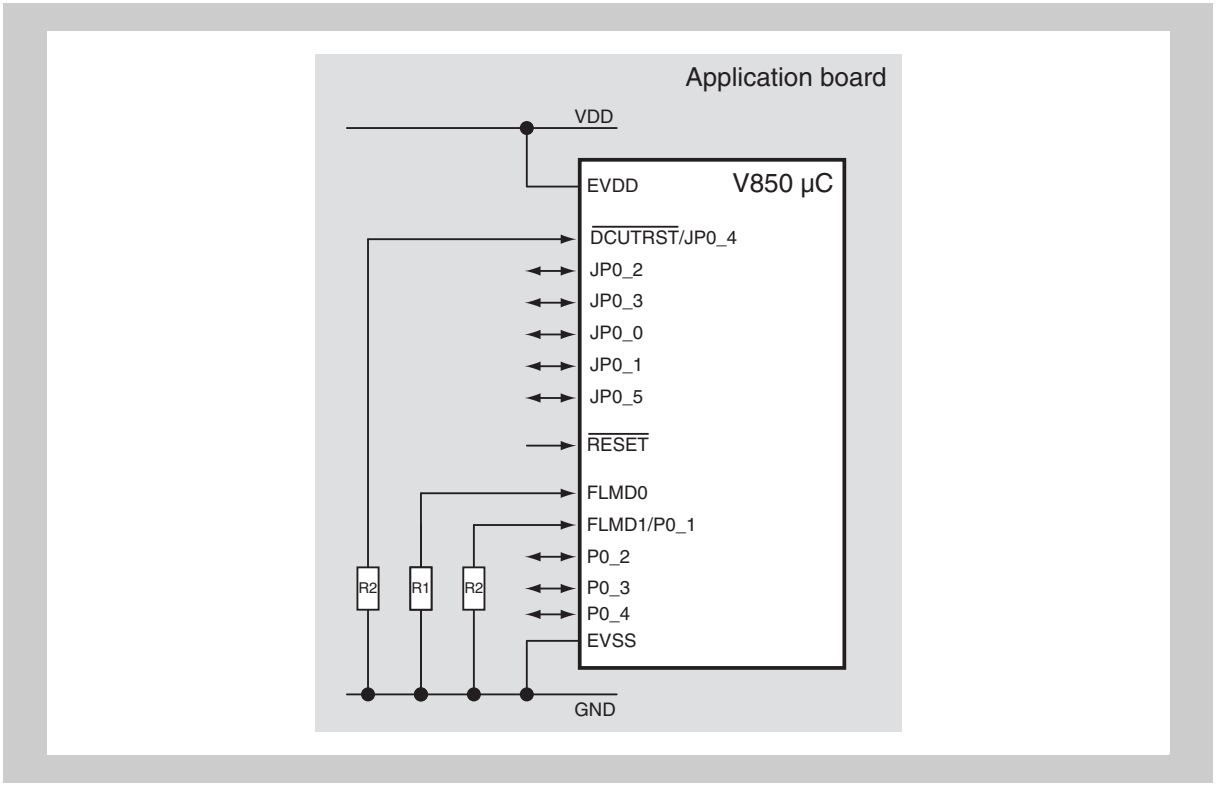


Figure 3-9 Pin connections in normal operation mode

3.10.2 Debug mode

In debug mode the debugger is connected to the pins of the port group JP0. The ports used for MODE0, MODE1 and P0_4 can be used for general port and alternative functions.

Debug mode is basically a normal operation mode, but the debugger changes the device into debug mode via $\overline{\text{DCUTRST}}$ upon reset release.

Debug mode is selected by

FLMD0	FLMD1 (P0_1)[a]	MODE0 (P0_2)	MODE1 (P0_3)	P0_4
low level	low level	don't care	don't care	don't care

a) FLMD1 can be used for general port and alternative functions after reset release.

TDO, $\overline{\text{TRDY}}$ pull-up resistors

Since the microcontroller's DCUTDO and $\overline{\text{DCUTRDY}}$ outputs are in high-impedance state during $\overline{\text{RESET}}$, the R3 resistors maintain a high level at the On-chip Debugger's TDO and $\overline{\text{TRDY}}$ inputs during reset time.
The pull-up resistors R3 at TDO and $\overline{\text{TRDY}}$ may be recommended to be mounted (for instance on the debugger adapter board/cable or on the application board) in order to avoid temporarily undefined levels of TDO and $\overline{\text{TRDY}}$. Refer to the debugger's description for details.
The R3 resistors are not required in normal operation.

Notes

1. In debug mode on-chip pull-up resistors are automatically connected to DCUTCK, DCUTMS and DCUTDI.
2. In debug mode all ports of the debugger interface are automatically configured and are not affected by any port register settings.

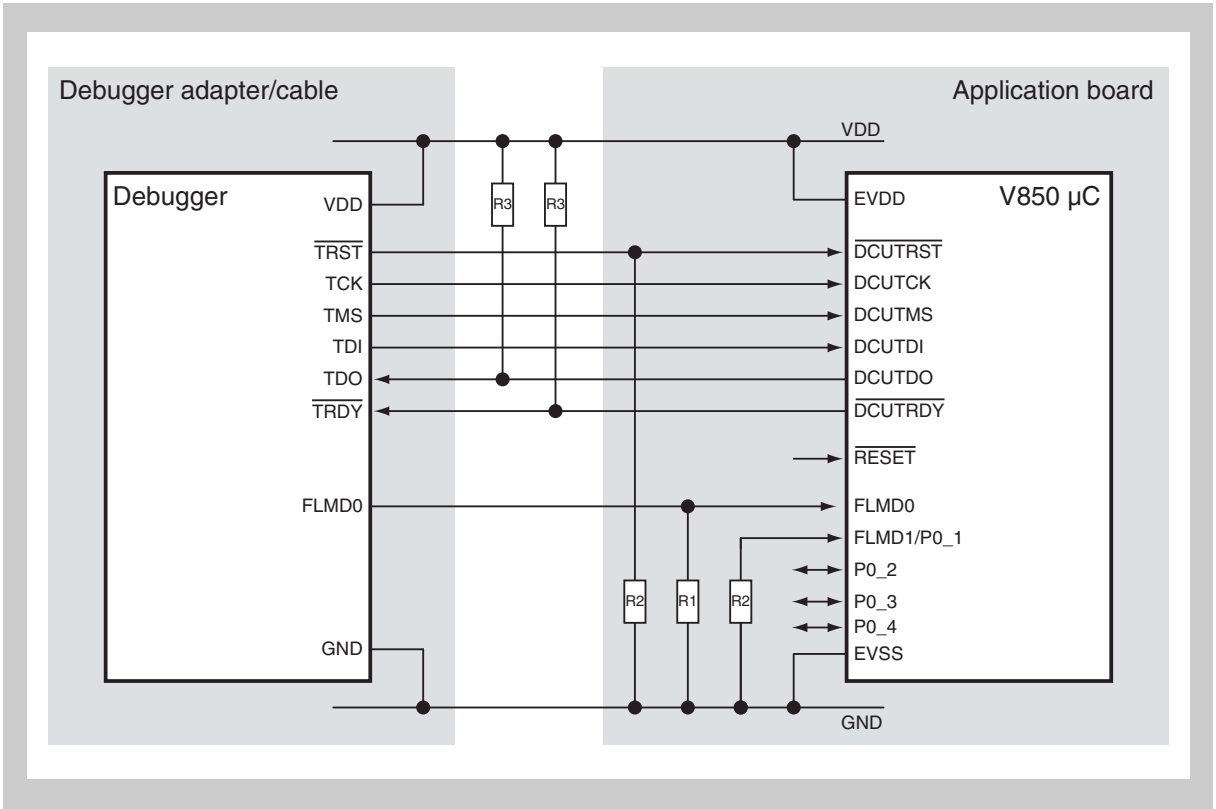


Figure 3-10 Pin connections in debug mode

Further information For further information about the debugger refer to the chapter *“On-Chip Debug Unit (OCD)”* and the debugger documentation

“User’s Manual QB-V850MINI, QB-V850MINIL”,

document number U17638EJxVxUM00 (xVx denotes the current version number).

3.10.3 Flash programming mode

In flash programming mode some ports of port group JP0 are connected to the flash programmer.

Flash programming mode is selected by

FLMD0	FLMD1 (P0_1)	MODE0 (P0_2)	MODE1 (P0_3)	P0_4
high level	low level	don't care	don't care	don't care

The PG-FP5 flash programmer communicates with the V850 microcontroller via

- a single wire asynchronous interface FLUR0RTX
  Ports JP0_1 to JP0_5 and the ports used for MODE0 and MODE1 can be used for general port and alternative functions.
- a 3-wire synchronous interface FLCS0SCI, FLCS0SI, FLCS0SO
  Ports JP0_3 to JP0_5 and the ports used for MODE0 and MODE1 can be used for general port and alternative functions.

Flash programming mode is determined by FLMD0 at high and FLMD1 at low level. Both can be connected to the PG-FP5 flash programmer, that sets the pins to the correct levels. However FLMD1 connection to the flash programmer is not mandatory since FLMD1 is already set to low via the pull-down resistor R2.

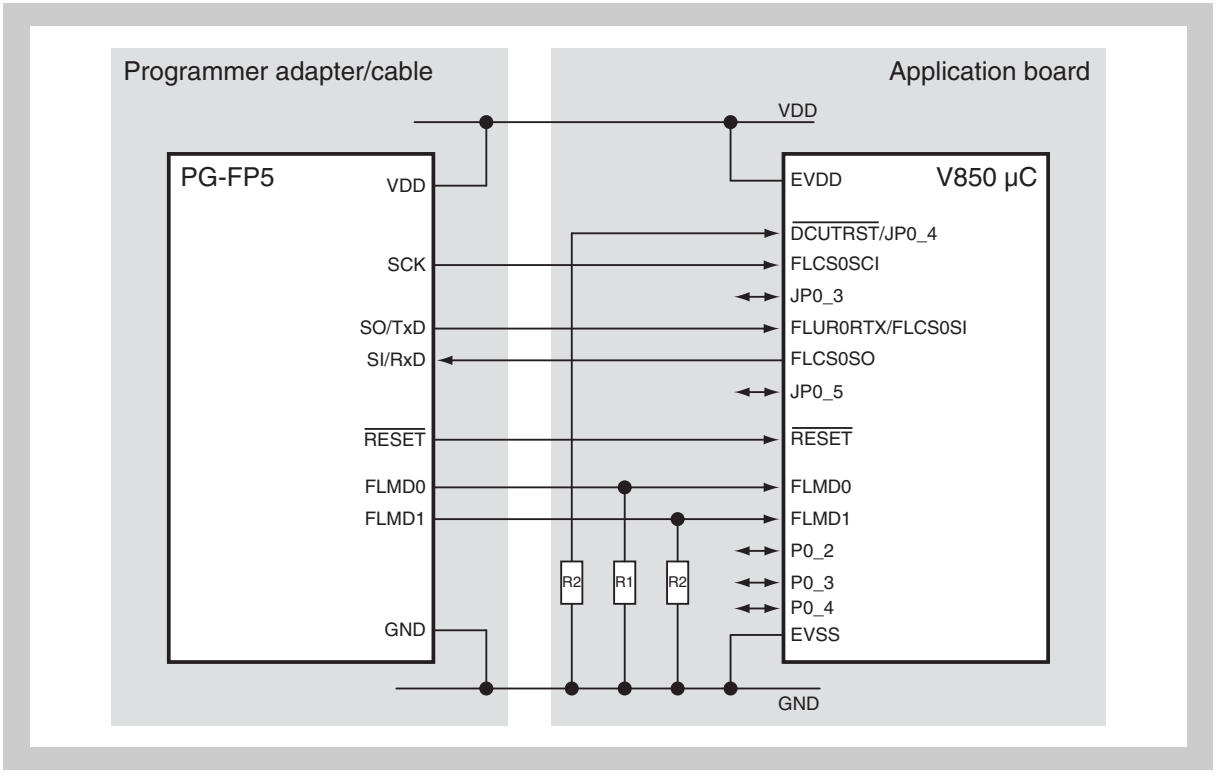


Figure 3-11 Pin connections in flash programming mode (FLMD1 by PG-FP5)

Further information For further information about the debugger refer to the chapter *"Flash Programming with Flash Programmer"* in the chapter *"Flash Memory"* and the PG-FP5 flash programmer documentation

"User's Manual PG-FP5",

document number R20UT0008EJxxxx (xxxx denotes the current version number).

3.10.4 Boundary scan mode

For boundary scan testing the tester is connected to the pins of the port group JP0, except JP0_5.
JP0_5 and P0_4 can be used for general port and alternative functions.

Boundary scan mode is selected by

FLMD0	FLMD1 (P0_1)	MODE0 (P0_2)	MODE1 (P0_3)	P0_4
high level	high level	low level	don't care	low level

Since FLMD0 and FLMD1 have to be set to high level for boundary scan testing, the resistors R5 and R6 must be properly dimensioned to ensure high level at FLMD0 and FLMD1 in the voltage divider R1 - R5 respectively R2 - R6.

The pull-up and -down resistors R4, R5, R6 can be mounted e.g. on the boundary scan adapter board or cable, as they are not required in normal operation.

Note In debug mode on-chip pull-up resistors are automatically connected to DCUTCK, DCUTMS and DCUTDI.

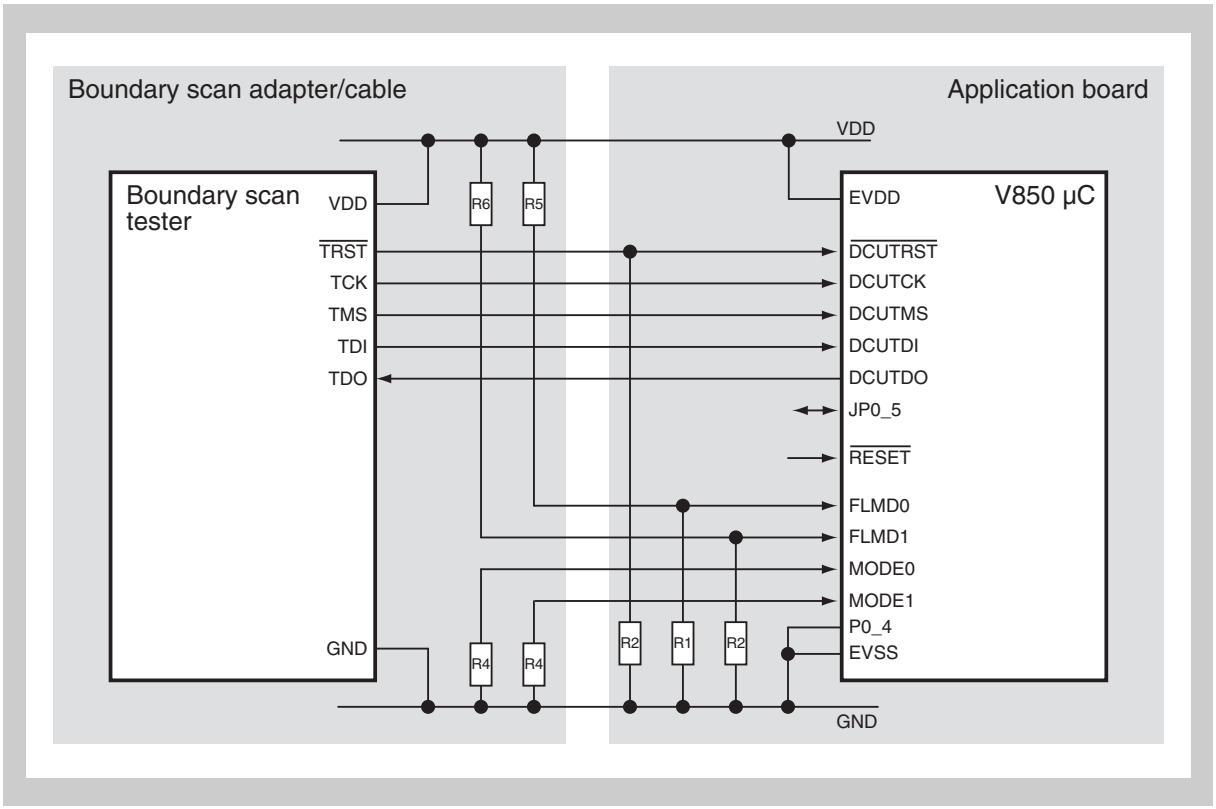


Figure 3-12 Pin connections in boundary scan mode

3.11 Address Space

In the following sections, the address space of the CPU is explained. Size and addresses of CPU address space and physical address space are explained. The address range of data space and program space together with their wrap-around properties are presented.

3.11.1 CPU data address and physical program address space

The CPU supports the following address space:

- 4 GB CPU data address space
  With the 32-bit general purpose registers, addresses for a 4 GB memory can be generated. This is the maximum address space supported by the CPU.
- 512 MB physical program address space
  The CPU provides 512 MB physical address space to access instruction opcodes in program memory. That means that a maximum of 512 MB internal or external program memory can be accessed.

3.11.2 Program and data space

The figure below shows the assignment of the CPU address space to data and program space.

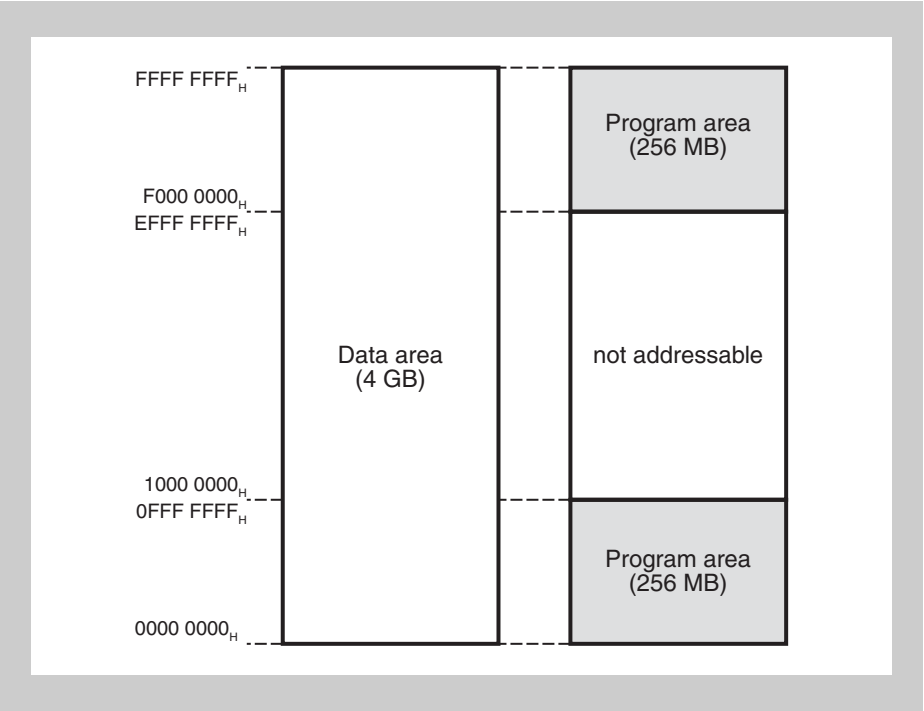


Figure 3-13 CPU address space

(1) Wrap-around of data space

If an operand address calculation exceeds 32 bits, only the lower 32 bits of the result are considered. Therefore, the addresses 0000 0000_H and FFFF $FFFF_H$ are contiguous addresses. This results in a wrap-around of the data space:

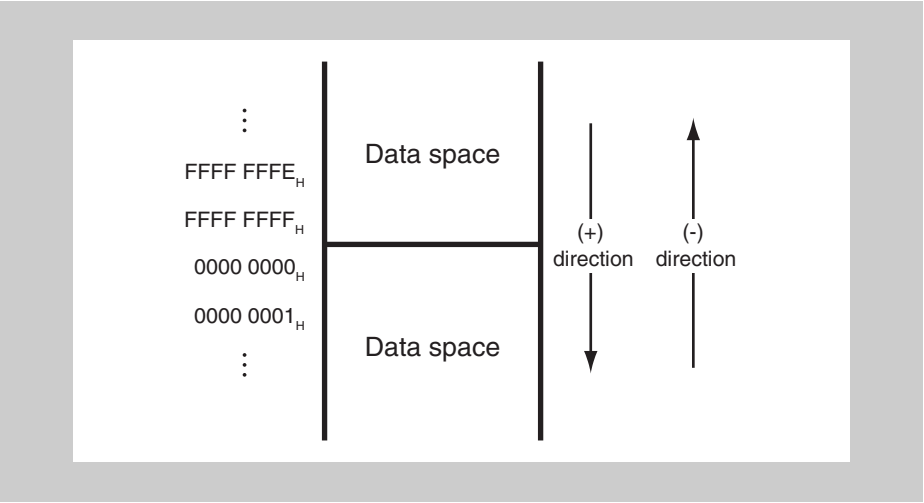


Figure 3-14 Wrap-around of data space

(2) Wrap-around of program space

If an instruction address calculation exceeds 28 bits, only the lower 28 bits of the result are considered. Therefore, the addresses 0000 0000_H and 0FFF $FFFF_H$ are contiguous addresses. This results in a wrap-around of the program space:

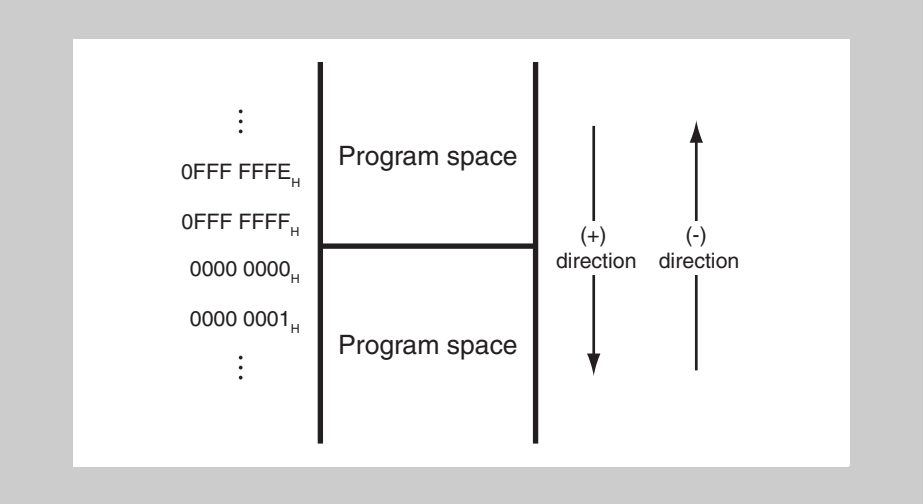


Figure 3-15 Wrap-around of program space

3.12 V850E2/Fx4-H CPU Address Map

In the following sections, the memory map of the CPU is introduced. Specific memory areas are described.

3.12.1 DMA address map

The DMA/DTS Controller can access all CPU address areas, except the address range

$FFFF\ 5000_H$ to $FFFF\ 7FFF_H$

This area can only be accessed by the CPU.

3.12.2 V850E2/Fx4-H memory map

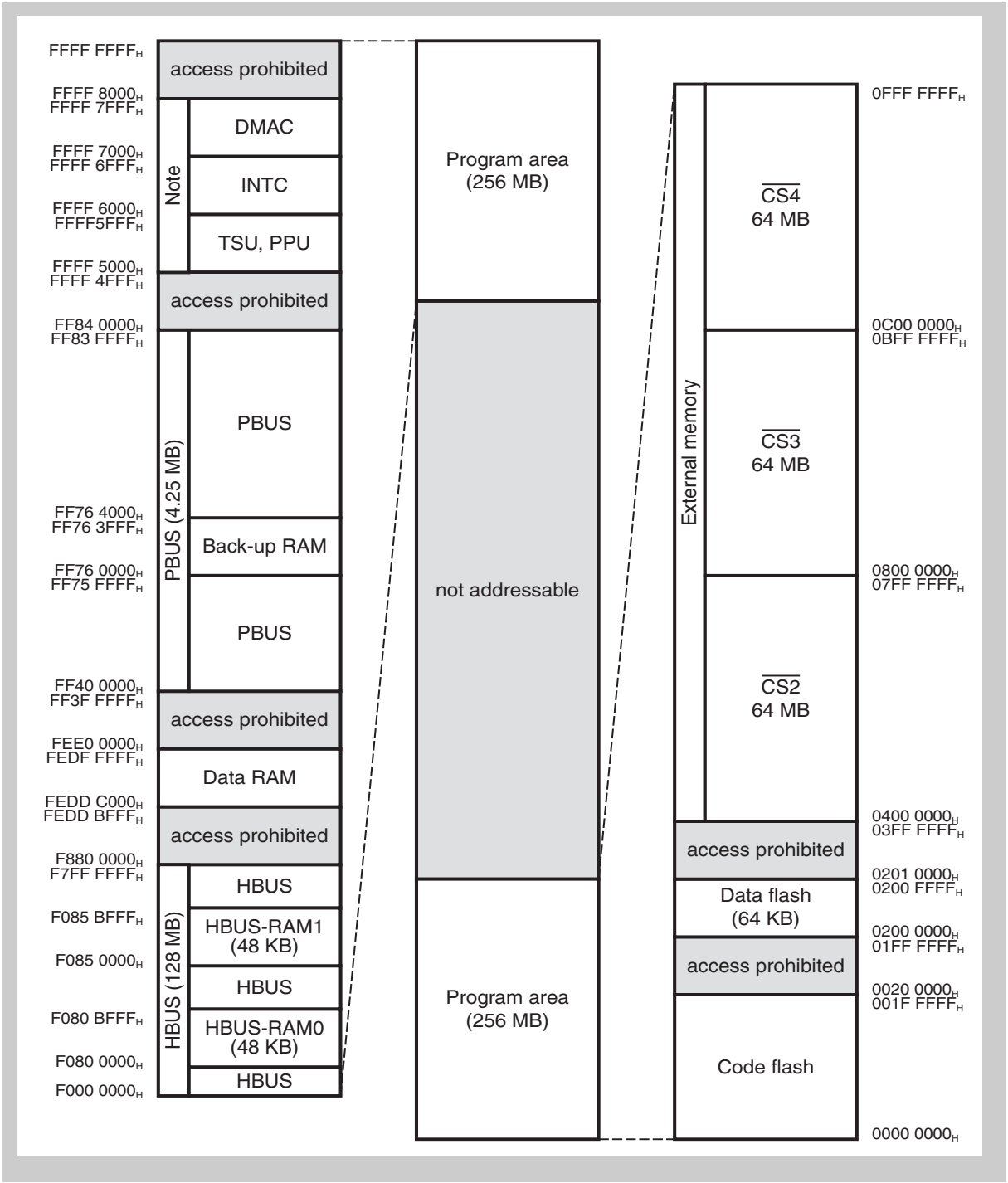


Figure 3-16 V850E2K/Fx4-H memory map

Note These areas are not accessible by the DMA/DTS Controller.

3.12.3 Memory areas

The internal memory of the V850E2/Fx4-H provides several areas:

- internal code flash area
- internal data flash area
- internal Data RAM area
- internal Back-up RAM area
- internal HBUS-RAM area
- external memory areas

The areas are briefly described below.

(1) Internal code flash areas

The following table summarizes the size and addresses of the code flash memories.

Table 3-23 Code flash memory areas

Series name	Product name	Size	Address range
FK4-H-2M	μPD70F3561	2 MB	0000 0000_H - 001F $FFFF_H$
FL4-H-2M	μPD70F3564		

(2) Internal data flash areas

The following table summarizes the size and addresses of the data flash memories.

Table 3-24 Internal data flash areas

Series name	Product name	Size	Address range
FK4-H-2M	μPD70F3561	64 KB	0200 0000_H - 0200 $FFFF_H$
FL4-H-2M	μPD70F3564		

(3) Internal Data RAM areas

The following table summarizes the size and addresses of the Data RAM memories.

Table 3-25 Data RAM memory areas

Series name	Product name	Size	Address range
FK4-H-2M	μPD70F3561	144 KB	FEDD $C000_H$ - FEDF $FFFF_H$
FL4-H-2M	μPD70F3564		

(4) Internal Back-up RAM areas

The following table summarizes the size and addresses of the Back-up RAM memories.

Table 3-26 Back-up RAM memory areas

Series name	Product name	Size	Address range
FK4-H-2M	µPD70F3561	16 KB	FF76 0000_H - FF76 $3FFF_H$
FL4-H-2M	µPD70F3564		

Refer to the section *"Back-up RAM (BURAM)"* below in this chapter for Back-up RAM details.

(5) Internal HBUS-RAM areas

The following table summarizes the size and addresses of the HBUS-RAM memories.

Table 3-27 HBUS-RAM memory areas

Series name	Product name	Size	Address range
FK4-H-2M	µPD70F3561	HBUS-RAM0: 48 KB	F080 0000_H - F080 $BFFF_H$
		HBUS-RAM1: 48 KB	F085 0000_H - F085 $BFFF_H$
FL4-H-2M	µPD70F3564	HBUS-RAM0: 48 KB	F080 0000_H - F080 $BFFF_H$
		HBUS-RAM1: 48 KB	F085 0000_H - F085 $BFFF_H$

(6) External memory areas

The following table summarizes the size and addresses of the external memories.

Table 3-28 External memory areas

Device	Size	Address range
V850E2K/FK4-H	64 MB	$\overline{CS0}$: 0000 0000_H - 01FF $FFFF_H$
	64 MB	$\overline{CS2}$: 0400 0000_H - 07FF $FFFF_H$
	64 MB	$\overline{CS3}$: 0800 0000_H - 0BFF $FFFF_H$
	64 MB	$\overline{CS4}$: 0C00 0000_H - 0FFF $FFFF_H$

3.13 Back-up RAM (BURAM)

The 16 KB Back-up RAM is a PBUS module, organized as 4 K x 32 bit.

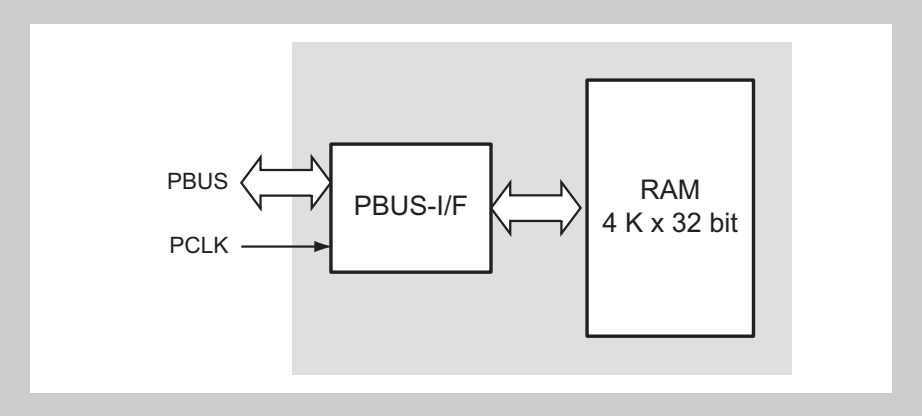


Figure 3-17 Block diagram of the Back-up RAM

Access The BURAM can be read/written in 32-bit units.

BURAM address The BURAM address is listed in the following table:

Table 3-29 BURAM address

BURAM	Address
BURAM	FF76 0000_H - FF76 $3FFF_H$

Clock supply The BURAM is supplied with the following clock.

Table 3-30 BURAM clock supply

BURAM	BURAM clock	Connected to
BURAM	PCLK	Clock Controller CKSCLK_A05

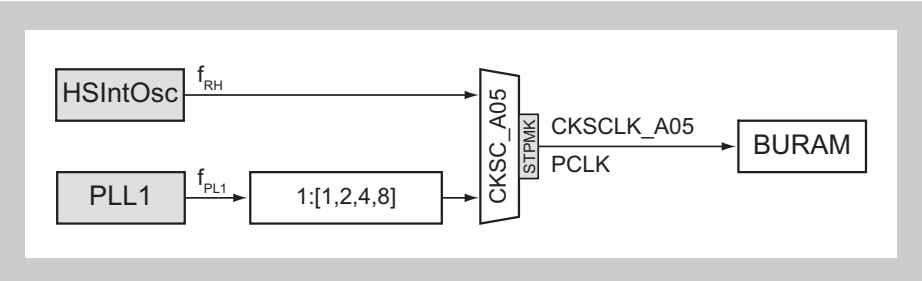


Figure 3-18 BURAM clock supply

Write permission Write access to the Back-up RAM must be explicitely enabled via the Back-up RAM control register BURC.

3.13.1 Back-up RAM protection

Write access to the Back-up RAM is disabled after reset release. Thus write access must be explicitly permitted by setting the write access permission bit BURC.BURWE = 1.

If a write access to the Back-up RAM occurs, while write is prohibited (BURC.BURWE = 0), the error bit BURAE.BURAERR is set.

The following registers control and monitor the write access to the Back-up RAM:

Table 3-31 Back-up RAM registers overview

Register name	Shortcut	Address
Back-up RAM control register	BURC	FF76 FE00$_H$
Back-up RAM access error register	BURAE	FF76 FE04$_H$
Back-up RAM access error clear register	BURAEC	FF76 FE08$_H$

(1) BURC – Back-up RAM control register

This register is used to permit respectively prohibit write access to the Back-up RAM.

Access This register can be read/written in 8-bit units.

Address FF76 FE00$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURWE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-32 BURC register contents

Bit position	Bit name	Function
0	BURWE	Back-up RAM write permission control: 0: write to Back-up RAM prohibited 1: write to Back-up RAM permitted

(2) BURAE – Back-up RAM access error register

This register reflects an erroneous write access to the Back-up RAM.

Access This register can be read in 8-bit units.

Address FF76 FE04$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURA ERR
R	R	R	R	R	R	R	R

Table 3-33 BURAE register contents

Bit position	Bit name	Function
0	BURA ERR	Back-up RAM write access error flag 0: no Back-up RAM write access error 1: Back-up RAM write access error has occurred

(3) BURAEC – Back-up RAM access error clear register

This register is used to clear the Back-up RAM write access error flag BURAERR.

Access This register can be read/written in 8-bit units.

Address FF76 FE08$_H$

Initial Value Reading this register returns always 00$_H$.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	BURA ERRC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-34 BURAEC register contents

Bit position	Bit name	Function
0	BURA ERRC	Back-up RAM write access error flag BURAERR clear 0: no function 1: clear BURAERR

3.14 CPU Subsystem - HBUS Bridge

The CPU Subsystem HBUS bridge provides interfaces between the CPU Subsystem internal resources and CPU Sybsystem external HBUS modules.

HBUS slave I/F The HBUS slave I/F connects to CPU Subsystem external HBUS masters and gives them access to CPU Subsystem internal resources.

HBUS master I/F The HBUS master I/F connects to CPU Subsystem external HBUS slaves and gives CPU Subsystem modules (e.g. CPU, DMAC) access to CPU Subsystem external HBUS resources. CPU Subsystem accesses to external HBUS slaves are done via buffers.

All CPU Subssytem external HBUS masters and slaves are connected to the Cross Connection Bus system XBUS.

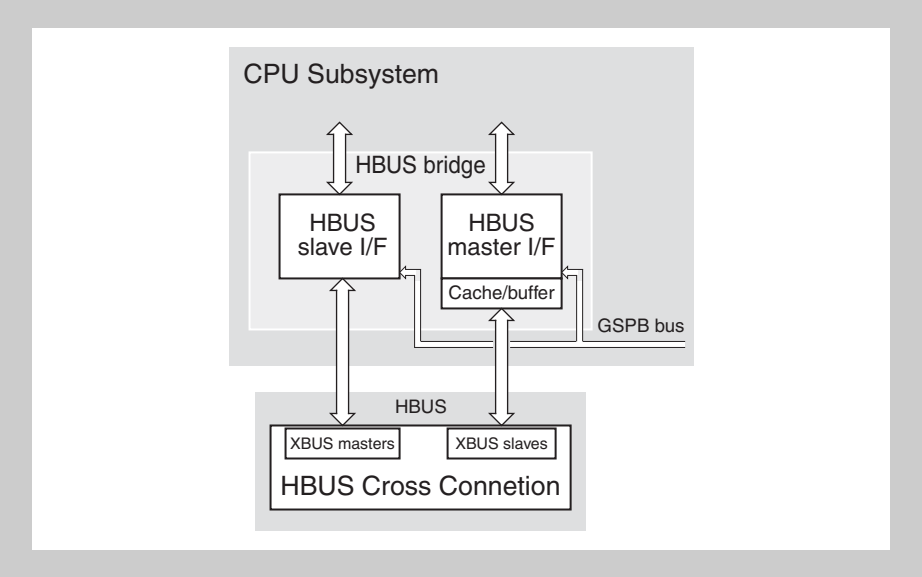


Figure 3-19 V850E2/Fx4-H CPU Subsystem HBUS bridge

XBUS HBUS master ID The XBUS is a HBUS master connected to the CPU Subsystem HBUS slave I/F. To allow access of the XBUS masters to the CPU Subsystem the HBUS bridge the access enable master ID register ATEAPMI has to be set correctly, as shown in the table.

Table 3-35 XBUS HBUS master ID

Bus master	XBUS HBUS master ID	ATEAPMI setting
XBUS	2	ATEAPMI = 0002_H (ATEAPMI.EN1 = 1)

3.14.1 Functional description

(1) HBUS master I/F - CPU Subsystem to HBUS transfers

Transfer types

Data accesses from the CPU Subsystem to the area, which is covered by the CPU Subsystem - HBUS bridge, can be handled in the following ways:

- direct access
- buffered access
- cached access

Address areas

Four address areas can be defined by base address and size. The used transfer type can be set for each of these areas. These setting are done by the ETACFG, ETACFGn and ETARADRSn registers.

Within the CPU Subsystem the CPU and the DMA Controller can act as initiators/masters of bus transfers. Note that on CPU Subsystem side only single data transfers are possible, i.e. neither CPU nor DMA Controller can issue burst transfers.

Note that the memory areas must lie in the HBUS area, refer to the section *“CPU address map”* in this chapter.

Direct access

Direct access will be transferred as is, i.e. single data transfers on CPU Subsystem side will be transferred to single data transfers on HBUS side. Direct write accesses of the CPU Subsystem (i.e. of the CPU or the DMA/DTS Controller) to the HBUS are performed as “posted write” accesses. That means, after the write the CPU or DMA/DTS Controller continues operation and the HBUS bridge takes care for transferring the data to its HBUS destination.

Buffered access

Main purpose of the buffered access is to convert consecutive single transfers on the CPU Subsystem side to burst transfers on the HBUS side. The implemented buffer size is 2 x 256 bits (i.e. two lines of 8 x 32 bit) for both read and write direction.

- Read access
  - buffer size: 2 x 256 bits (i.e. two lines of 8 x 32 bit)
  - buffer is refilled by burst transfer on HBUS side
  - refill can read critical word first or sequential (selectable by register setting)
  - prefetch mechanism can be configured (increment, decrement or off)
- Write access
  - buffer size: 2 x 256 bits (i.e. two lines of 8 x 32 bit)
  - buffer is used as FIFO
  - single transfer write on CPU Subsystem side will be combined to a burst write on HBUS side

Buffered write operation

The buffer uses two lines of eight words. For this reason the address space is distinguished in blocks with the size of eight 32-bit words.
The block start address is aligned to 8 words, thus the lower three address bits are assumed 000_B.

If a write access from the CPU Subsystem side to a buffered area is received, the write data is stored in the active buffer line.
The lower three bits of write address determine the position in the buffer line.

On a subsequent write access the write data is stored in the same active buffer line if the address matches the used address block of the active buffer. If the write access targets a different address block the other buffer line becomes the active line.

A buffer line is written to the HBUS side, if one of the following conditions is valid:

- the active line was changed (i.e. a write access to another address block was performed)
- a 16 bit or 8 bit access is performed
- a read access is made to an address in the block used by the active buffer line
- a buffer flush request is asserted

A buffer line write operation is performed as a write burst on the HBUS side only, if the buffer line is completely filled.

If not all entries of the buffer line contain valid data, several single write accesses on the HBUS side are performed.

Caution If eight or less subsequent write accesses are performed, the data will not be written to the HBUS side, unless the software ensures that either of the buffer line write conditions (as described above) are met.

Cached access

Cache size is 16 KB - four way associative cache is used.

- Cache features:
  - cache line size: 4 words (4 x 32 bits)
  - least recently used replacement algorithm
  - refill can read critical word first or sequential (selectable by register setting)
  - cache flush function
  - write-through policy (data is transferred as in direct mode)

(2) HBUS slave I/F - HBUS to CPU Subsystem access

The data access from the HBUS to the CPU Subsystem is always a direct access. However this bridge features a 4 word (16 byte) buffer which allows to handle burst access form HBUS side.

3.14.2 Register overview

The HBUS bridge is controlled and operated by the registers in the table below.

Table 3-36 HBUS bridge register overview

Register name	Shortcut	Address
Master I/F registers		
Configuration register	ETACFG	FFFF 7100$_H$
Command register	ETACMD	FFFF 7102$_H$
Wait insertion limit value register	ETAWRL	FFFF 7106$_H$
Master I/F error flag register	ETAERR	FFFF 7110$_H$
HBUS error response error address register	ETAEREA	FFFF 7120$_H$
Wait insertion limit error address register	ETAWLEA	FFFF 7128$_H$
Area 0 setting register	ETARCFG0	FFFF 7140$_H$
Area 1 setting register	ETARCFG1	FFFF 7142$_H$
Area 2 setting register	ETARCFG2	FFFF 7144$_H$
Area 3 setting register	ETARCFG3	FFFF 7146$_H$
Area 0 address register	ETARADRS0	FFFF 7150$_H$
Area 0 mask register	ETARMASK0	FFFF 7154$_H$
Area 1 address register	ETARADRS1	FFFF 7158$_H$
Area 1 mask register	ETARMASK1	FFFF 715C$_H$
Area 2 address register	ETARADRS2	FFFF 7160$_H$
Area 2 mask register	ETARMASK2	FFFF 7164$_H$
Area 3 address register	ETARADRS3	FFFF 7168$_H$
Area 3 mask register	ETARMASK3	FFFF 716C$_H$
Slave I/F registers		
CPU Subsystem error address register	ATEEEA	FFFF 7FC0$_H$
Lock hold time limit error address register	ATELEA	FFFF 7FC4$_H$
Slave I/F error flag register	ATESR	FFFF 7FD8$_H$
Buffer status display register	ATEBSR	FFFF 7FDA$_H$
Lock hold time limit register	ATESHL	FFFF 7FE2$_H$
Access enable master ID register	ATEAPMI	FFFF 7FE4$_H$

3.14.3 Master I/F registers details

(1) ETACFG - Configuration register

This register allows to define general configuration settings of the HBUS master I/F.

Access This register can be read/written in 16-bit units.

Address FFFF 7100_H

Initial Value 0980_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	BFM	0	BPM		CFM	0	0	0	0	0	0	0
R	R	R	R	R/W	R	R/W	R/W	R/W	R	R	R	R	R	R	R

Table 3-37 ETACFG register contents

Bit position	Bit name	Function
11	BFM	Buffer fill operation setting 0: Sequential 1: Critical word first
9, 8	BPM	Buffer prefetch operation setting 00_B: No prefetch 01_B: Prefetch with address increment 10_B: Prefetch with address decrement 11_B: Reserved
7	CFM	Cache fill operation setting 0: Sequential 1: Critical word first

(2) ETACMD - Command register

This register allows to initiate a flush operation of the HBUS master I/F buffer and cache.

Access This register can be read/written in 16-bit units.

Address FFFF 7102_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	BFL	CFL
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 3-38 ETACMD register contents

Bit position	Bit name	Function
1	BFL	Initiate flush operation of the buffer. If this bit is set to 1, the buffer flush operation starts. After the flush operation ends (from HBUS bridge point of view), this bit is automatically cleared to 0. 0: No flushing or stopped/flush operation ended 1: Init flushing / flush operation started.
0	CFL	Initiate flush operation of the cache. If this bit is set to 1, the cache flush operation starts. After the flush operation ends (from HBUS bridge point of view), this bit is automatically cleared to 0. 0: No flushing or stopped/flush operation ended 1: Init flushing / flush operation started.

(3) ETAWRL - Wait insertion limit register

This register specifies the upper limit of HBUS clocks that the HBUS may request a wait.

Caution It is strongly recommended to use this feature for debug purpose only and to disable the wait limit by setting this register to zero otherwise.

If a wait limit is set and the HBUS wait request exceeds the defined number of clocks a SYSERR exception will occur.

Access This register can be read/written in 16-bit units.

Address FFFF 7106_H

Initial Value $000F_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	ETAWRL[7:0]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-39 ETAWRL register contents

Bit position	Bit name	Function
7 to 0	ETAWRL[7:0]	These bits set the upper-limit value for a wait response from HBUS. 0: not limited / disabled other value n: limited to n bus clocks.

(4) ETAERR - Master I/F error flag register

This register informs about error occurrences during CPU and DMAC accesses to HBUS resources via the HBUS bridge master interface.

Access This register can be read/written in 16-bit units.

Address FFFF 7110_H

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

7	6	5	4	3	2	1	0
0	0	CDE	CTE	0	ERE	WLE	0
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

Table 3-40 ETAERR register contents

Bit position	Bit name	Function
5	CDE	This bit is set if a cache RAM error occurs. 0: no error 1: error was detected Clearing this bit after it was set to “1” due to an error detection is achieved by writing “1” to the bit.
4	CTE	This bit is set if a cache tag RAM error occurs. 0: no error 1: error was detected Clearing this bit after it was set to “1” due to an error detection is achieved by writing “1” to the bit.
2	ERE	This bit is set if an HBUS error response error occurs. In case of an error response error, the corresponding error address is stored in the ETAEREA register. 0: no error 1: error was detected Clearing this bit after was it set to “1” due to an error detection is achieved by writing “1” to the bit.
1	WLE	This bit is set if a HBUS wait limit error occurs. In case of a wait limit error, the corresponding error address is stored in the ETAWLEA register. 0: no error 1: error was detected Clearing this bit after it was set to “1” due to an error detection is achieved by writing “1” to the bit.

(5) ETAEREA - HBUS error response error address register

This register holds the address, at which a HBUS error response has occurred. Occurrence of a HBUS error response is indicated by ETAERR.ERE = 1.

ETAEREA always holds the address of the first error response. Following error responses do not overwrite the first address, until ETAERR.ERE has been cleared.

Access This register can be read/written in 32-bit units.

Address FFFF 7120$_H$

Initial Value This register is not initialized. Its value after power-up of the microcontroller is undefined.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETAEREA[31:16]															
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETAEREA[15:0]															
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

Table 3-41 ETAEREA register contents

Bit position	Bit name	Function
31 to 0	ETAEREA [31:0]	ETAEREA[31:0] is the address, at which an HBUS error response has occurred.

(6) ETAWLEA - Master I/F wait insertion limit error address register

This register holds the address, at which a wait insertion limit error has occurred. Occurrence of a wait insertion limit error is indicated by ETAERR.WLE = 1.

ETAWLEA always holds the address of the first wait insertion limit error. Following errors do not overwrite the first address, until ETAERR.WLE has been cleared.

Note The wait limit is specified in the EATWRL register.

Access This register can be read/written in 32-bit units.

Address FFFF 7128$_H$

Initial Value This register is not initialized. Its value after power-up of the microcontroller is undefined.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
B28[2:0]			ETAWLEA[28:16]												
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETAWLEA[15:0]															
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

Table 3-42 ETAWLEA register contents

Bit position	Bit name	Function
31 to 29	B28[2:0]	These bits always have the same value as the upper address bit ETAWLE[28].
28 to 0	ETAWLEA [28:0]	ETAWLEA[28:0] is address, at which the first wait insertion limit error has occurred.

(7) ETARCFGn - Area n setting register

This register enables the address area n and sets the transfer mode for this area.

Access This register can be read/written in 16-bit units.

Address ETARCFG0: FFFF 7140_H, ETARCFG1: FFFF 7142_H
ETARCFG2: FFFF 7144_H, ETARCFG3: FFFF 7146_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	MODE[3:0]				0	0	0	EN
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R	R	R	R/W

Table 3-43 ETARCFGn register contents

Bit position	Bit name	Function
7 to 4	MODE[3:0]	These bits set the operation mode for area n 0000_B: direct access 0001_B: use buffer 0010_B: use cache other: reserved These bits are only effective, if mode setting is enabled by EN = 1.
0	EN	Area n mode setting enable bit 0: mode setting via MODE[3:0] disable, direct access is used 1: mode setting via MODE[3:0] enable

Note In case the memory space of areas overlap the applied transfer type follow the following priority: use buffer > use cache > direct access

(8) ETARADRSn - Area n address register

This register is used to set the base address that specifies the memory area of area n.

Access This register can be read/written in 32-bit units.

Address ETARADRS0: FFFF 7150$_H$, ETARADRS1: FFFF 7158$_H$
ETARADRS2: FFFF 7160$_H$, ETARADRS3: FFFF 7168$_H$

Initial Value This register is not initialized. Its value after power-up of the microcontroller is undefined.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETARADRS[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETARADRS[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-44 ETARADRSn register contents

Bit position	Bit name	Function
31 to 0	ETAR ADRS[31:0]	These bits set the base address that specifies area n. Bit 28 is regarded as the sign bit for bits 31 to 29, in which sign extended values are stored. Be sure to set bits 11 to 0 to “0”.

(9) ETARMASKn - Area n mask register

This register is used to set the mask for the base addresses that specify areas 0 to 3. Starting with the lower bits, be sure to specify values with consecutive 1s for this register.

Access This register can be read/written in 32-bit units.

Address ETARMASK0: FFFF 7154_H, ETARMASK1: FFFF $715C_H$
ETARMASK2: FFFF 7164_H, ETARMASK3: FFFF $716C_H$

Initial Value This register is not initialized. Its value after power-up of the microcontroller is undefined.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ETARMASK[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETARMASK[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

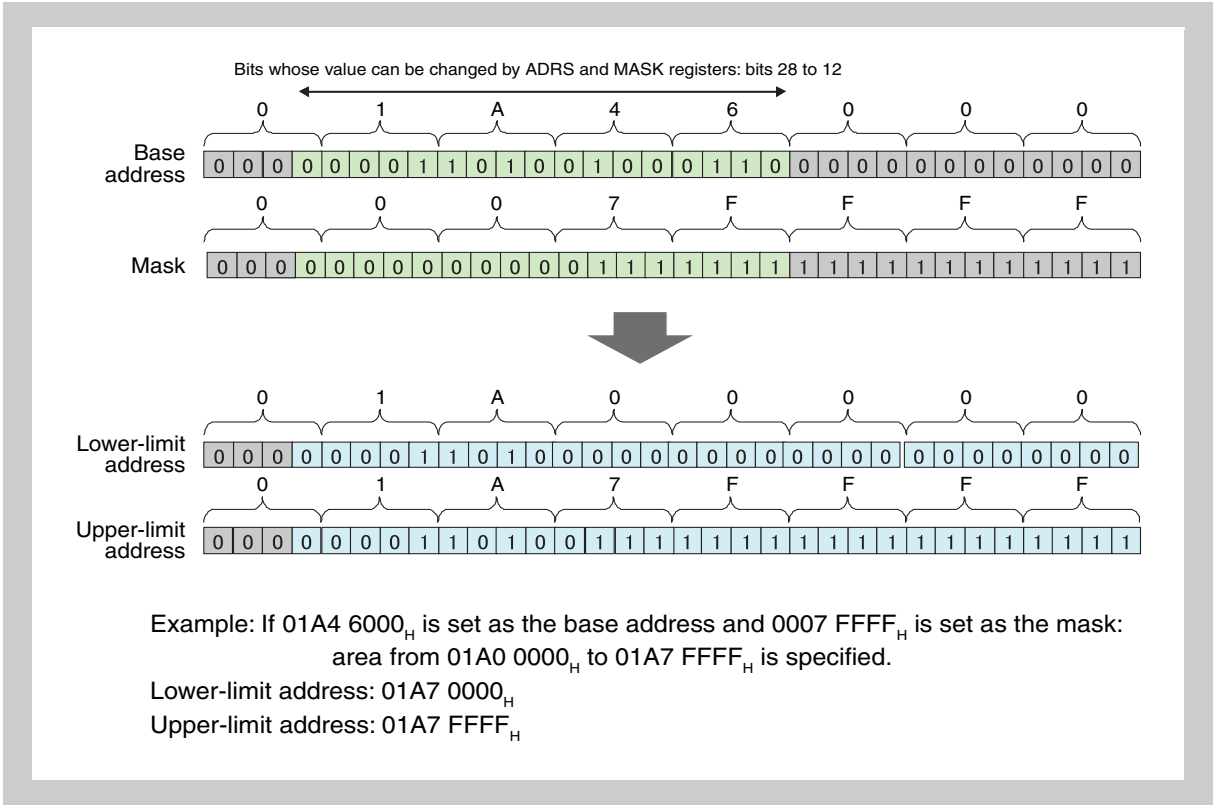
Table 3-45 ETARMASKn register contents

Bit position	Bit name	Function
31 to 0	ETAR MASK[31:0]	These bits are used to set the mask for the base address that specifies area n. Be sure to set bits 31 to 29 to “0”. Be sure to set bits 11 to 0 to 1.

(10) Calculation of area n address range

Each address area is set by using a base address and a mask. The lower-limit and upper-limit addresses of the address area are set by manipulating the base address with the mask. The address whose bits, which are set to 1 by the mask, are cleared to 0 is the lower-limit address and the address whose bits are set to 1 is the upper-limit address. Please refer to the figure below for an example.

Caution To specify a mask value, be sure to set a value of contiguous 1's from the lower bit.



Example: If 01A4 6000_H is set as the base address and 0007 $FFFF_H$ is set as the mask:
area from 01A0 0000_H to 01A7 $FFFF_H$ is specified.
Lower-limit address: 01A7 0000_H
Upper-limit address: 01A7 $FFFF_H$

Figure 3-20 Area n address range calculation example

3.14.4 Slave I/F registers details

(1) ATEEEA - Slave I/F CPU Subsystem error address register

This register holds the address, at which the CPU Subsystem has returned an error.

ATEEEA always holds the address of the first CPU Subsystem error. Following errors do not overwrite the first address.

A CPU Subsystem error response is indicated by the HBUS bridge slave interface error flag ATESR.CFL = 1. Clearing this flag also clears the address register ATEEEA.

Access This register can be read in 32-bit units.

Address FFFF 7FC0$_H$

Initial Value 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATEEEA[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATEEEA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 3-46 ATEEEA register contents

Bit position	Bit name	Function
31 to 0	ATEEEA [31:0]	ATEEEA[31:0] is address, where the CPU Subsystem has returned an error.

(2) ATELEA - Slave I/F lock hold time limit error address register

This register holds the address, at which lock hold time limit error has occurred.
ATELEA always holds the address of the first lock hold time limit error. Following errors do not overwrite the first address.

A lock hold time error is indicated by the HBUS bridge slave interface error flag ATESR.LE = 1. Clearing this flag also clears the address register ATELEA.

Note The wait limit is specified by the ATESHL register.

Access This register can be read in 32-bit units.

Address FFFF 7FC4$_H$

Initial Value 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ATELEA[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ATELEA[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 3-47 ATELEA register contents

Bit position	Bit name	Function
31 to 0	ATELEA [31:0]	ATELEA[31:0] is address, where the first lock hold time limit error has occurred.

(3) ATESR - Slave I/F error flag register

This register informs about error occurrences during HBUS master accesses to CPU Subsystem resources via the HBUS bridge slave interface.

Access This register can be read/written in 16-bit units.

Address FFFF 7FD8$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

7	6	5	4	3	2	1	0
0	0	0	0	PE	0	LE	CFL
R/W	R/W	R/W	RW	R/W	R/W	R/W	RW

Table 3-48 ATESR register contents

Bit position	Bit name	Function
3	PE	This bit indicates the occurrence of a HBUS bus access protection error. Occurrence of this error indicates a wrong setting of the access enable master ID register ATEAPMI. 0: no error 1: error was detected Clearing this bit to “0” after it was set to “1” due to an error detection is achieved by writing “1” to the bit.
1	LE	This bit indicates the occurrence of a lock hold limit error. The maximum lock hold time is specified via the ATESHL register. In case of a lock hold time error detection, the corresponding address is stored in the ATELEA register. 0: no error 1: error was detected Clearing this bit to “0” after it was set to “1” due to an error detection is achieved by writing “1” to the bit. This also clears the address register ATELEA.
0	CFL	This bit indicates the occurrence of an error response from the CPU Subsystem during an access to CPU Subsystem resources via the HBUS bridge slave I/F. Occurrence of this error indicates a erroneous access to CPU Subsystem resources, like e.g. a wrong address. In case of a CPU Subsystem error detection, the corresponding address is stored in the ATEEEA register. 0: no error 1: error was detected Clearing this bit to “0” after it was set to “1” due to an error detection is achieved by writing “1” to the bit. This also cleared the address register ATEEEA.

(4) ATEBSR - Buffer status display register

This register shows the status of the buffer of the HBUS slave I/F. The buffer is used to store data of a burst access on the HBUS side, i.e. if the buffer is empty no access is being executed from HBUS to the CPU Subsystem.

Access This register can be read in 16-bit units.

Address FFFF 7FDA$_H$

Initial Value 0001$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	E
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 3-49 ATEBSR register contents

Bit position	Bit name	Function
0	E	This bit displays the transfer buffer status. 0: There is data in the transfer buffer and the transfer is not complete. 1: There is no data in the transfer buffer and the transfer is complete.

(5) ATESHL - Slave I/F lock hold time limit register

This register specifies the lock hold time limit for accesses to the CPU Subsystem via the HBUS bridge slave I/F.

Access This register can be read/written in 16-bit units.

Address FFFF 7FE2$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	LIM[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-50 ATESHL register contents

Bit position	Bit name	Function
7 to 0	LIM[7:0]	LIM[7:0] specifies the lock hold time limit. 0: not limited (no lock hold time check) other value n: limited to n bus clocks

(6) ATEAPMI - Access enable master ID register

This register allows access of a HBUS master connected to the HBUS slave I/F.

Access This register can be read/written in 16-bit units.

Address FFFF 7FE4$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 3-51 ATEAPMI register contents

Bit position	Bit name	Function
15 to 0	EN15 to EN0	Allow access from HBUS master with ID 15 to 0 to the CPU Subsystem. 0: access not allowed 1: access allowed

Refer to the key word “XBUS HBUS master ID” at the beginning of this section for the correct ID.

3.15 Write protected Registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Writing to a write protected register requires a special register protection unlock sequence.

3.15.1 Register protection clusters

The protected registers are bundled in certain register protection clusters.

The protection mechanism treats all registers of the same cluster as a single protection unit.

If the protection unlock sequence for a register is initiated, no access to any other register of the same protection cluster is allowed. Otherwise the unlock sequence is disrupted and the register write fails.
The diagram below shows a disruption of the unlock sequence by an access to the same cluster within an interrupt service routine.

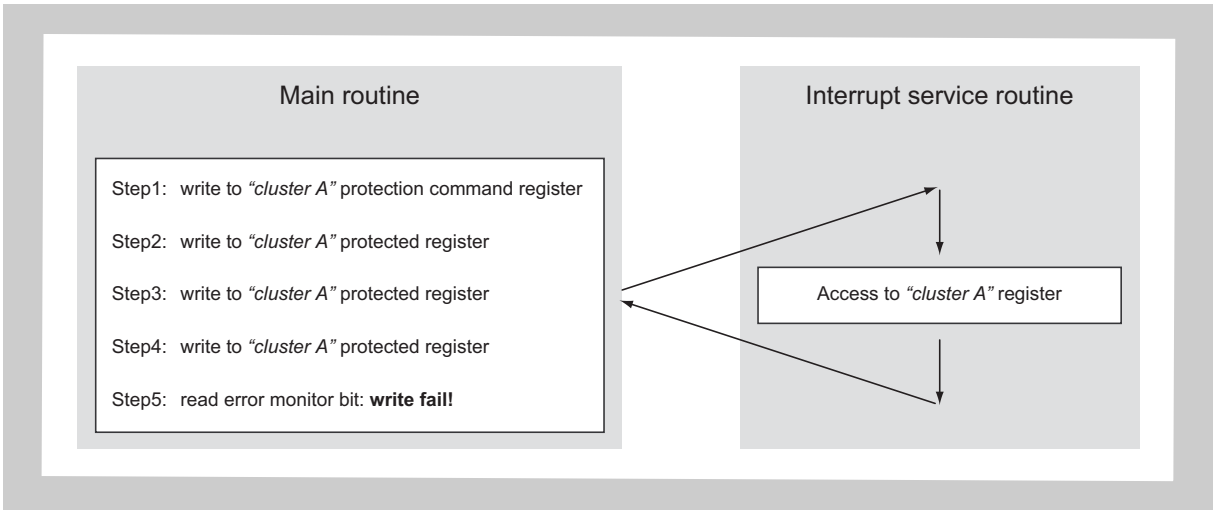


Figure 3-21 Disruption of register protection unlock sequence

Access to a register of another protection cluster during the unlock sequence does not disrupt the unlock sequence and the register write can be completed successfully.
The following diagram below shows such situation.

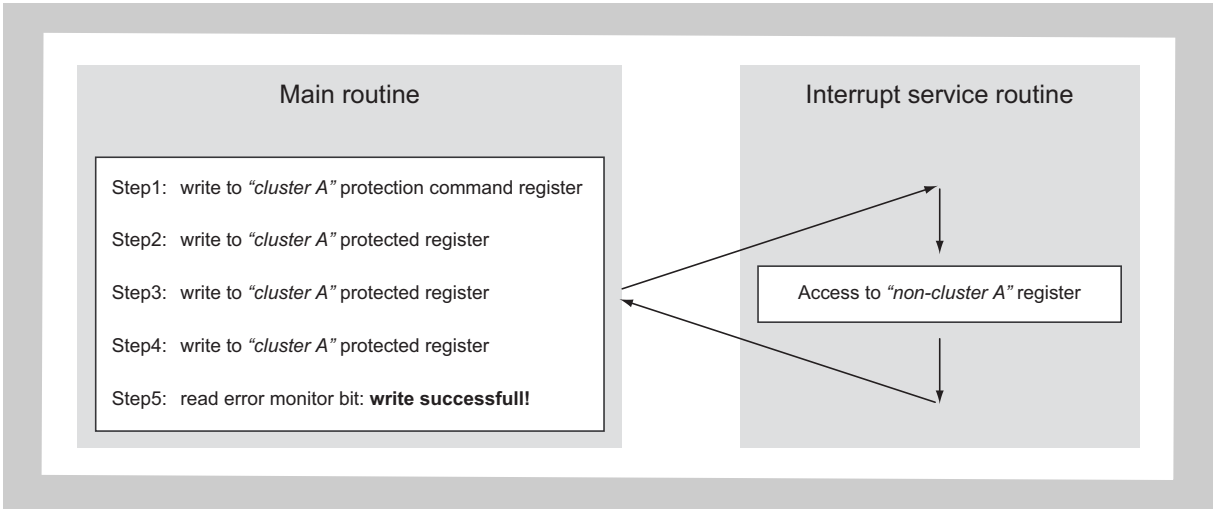


Figure 3-22 Successful register protection unlock sequence

The V850E2/Fx4-H register protection clusters are described in section *"V850E2/Fx4-H write protected registers"*.

3.15.2 Register protection unlock sequence

Write access to a write protected register is only possible within a special protection unlock sequence:

1. Write the fixed value $A5_H$ to the protection command register
2. Write the desired value to the protected register.
3. Write the bit-wise inversion of the desired value to the protected register.
4. Write the desired value to the protected register.
5. Verify successful write of the desired value to the protected register by verifying that the error monitor bit in the protection status register is "0". In case the write was not successful, indicated by the error monitor bit set to "1", the entire sequence has to be restarted at step 1.

In case of any access to another register between step 1 to step 4 of the above sequence, the protection mechanism behaves as follows:

- If the second register belongs to the same cluster, the write to the protected register fails (indicated by the error monitor bit set to "1"). The entire sequence has to be restarted at step 1.
- If the second register does not belong to the same cluster, the protection unlock sequence is not disrupted and the write to the first register can be completed successfully.

3.15.3 Register protection and interrupt/emulation break

If an interrupt/emulation break occurs during the protection unlock sequence, the protection mechanism behaves as follows:

(1) Interrupts during protection unlock sequence

If an interrupt is acknowledged during the above protection unlock sequence and the interrupt service routine does not access any register of the same register protection cluster, the protection unlock sequence is not disrupted and the write to the protected register can be successfully completed after returning from the interrupt service routine.

(2) Emulator break during protection sequence

If an emulation break occurs during the above protection unlock sequence, e.g. because of a breakpoint hit, the register protection is suspended until normal operation is resumed.
This means even if any register of the same cluster is accessed during the break, the protection unlock sequence is not disrupted and the error monitor bit is not set to “1”.

3.15.4 V850E2/Fx4-H write protected registers

The following table lists all V850E2/Fx4-H write protected registers:

Table 3-52 Write protected registers

Module	Protected register	Protection registers		Protection cluster
		Command	Status	
Clock Controller	CKSC_0n	PROTCMD0	PROTS0	Control protection cluster 0
Clock Controller	CKSC_1n	PROTCMD1	PROTS1	Control protection cluster 1
Clock Controller	PLLEk	PROTCMD2	PROTS2	Control protection cluster 2
	MOSCE			
	SOSCE			
	ROSCE			
	CKSC_An			
Stand-by Controller	PSC0			
	PSC1			
	PSC2			
Reset Controller	SWRESA			
	LVICNT			
On-Chip Debug control	IDMODEI	PROTCMD3	PROTS3	Control protection cluster 3
Clock Monitors	CLMAnCTL0	CLMAPCMD	CLMAnPS	Clock Monitor protection cluster
Port control[a]	PDSCn, JPDSCn PODCn, JPODCn	PPCMDn	PPROTSn	Port protection cluster 1 to 4
Self-programming control	FLMDCNT	FLMDPCMD	FLMDPS	Self-programming protection cluster

a) Each port group n has its own protection command and status register. Refer to *1 “Port protection clusters” on page 210* for details of port control registers protection.

(1) Port protection clusters

Following port registers feature write protection:

- Port drive strength control registers PDSCn, JPDSCn
- Port open drain control registers PODCn, JPODCn

The above listed port control registers of certain port groups n are assigned to four port protection clusters:

Table 3-53 Port protection clusters

Port protection cluster	Port groups
1	JP0
2	P0
3	P1 to P4, P10, P11
4	P12, P13, P21, P24 to P29

Note Each port group n has its own port protection command register PPCMDn and port protection status register PPROTSn.
However any port protection command register of the same port protection cluster can be used in the protection unlock sequence for enabling write access to all protected registers of the port protection cluster.

Register width The protected port control registers and the related protection command registers PPCMDn are 32-bit registers.

Caution All protected registers as well as the port protection command registers PPCMDn must be accessed by 32-bit accesses.

Thus the protection unlock sequence looks as follows:

1. Write the fixed value 0000 $00A5_H$ to the protection command register PPCMDn.
2. Write the desired value to the protected register with the upper 16 bit bit[31:16] set to “0” (0000 $xxxx_H$).
3. Write the bit-wise inversion of the desired value to the protected register, thus the upper 16 bit bit[31:16] are set to “1” (FFFF $xxxx_H$).
4. Write the desired value to the protected register with the upper 16 bit bit[31:16] set to “0” (0000 $xxxx_H$).
5. Verify successful write of the desired value to the protected register by verifying that PPROTSn.PPROTSnERR = 0.

3.15.5 V850E2/Fx4-H Protection registers overview

The register write protection is controlled and operated by the following registers:

Table 3-54 Protection command registers overview (1/2)

Register Name	Shortcut	Address
Control protection clusters:		
Protection command register 0	PROTCMD0	FF42 4000$_H$
Protection command register 1	PROTCMD1	FF42 8000$_H$
Protection command register 2	PROTCMD2	FF42 0300$_H$
Protection command register 3	PROTCMD3	FF42 0308$_H$
Protection status register 0	PROTS0	FF42 4004$_H$
Protection status register 1	PROTS1	FF42 8004$_H$
Protection status register 2	PROTS2	FF42 0304$_H$
Protection status register 3	PROTS3	FF42 030C$_H$
Clock monitors cluster:		
CLMA0:		
Protection command register	CLMA0PCMD	FF80 2010$_H$
Protection status register	CLMA0PS	FF80 2014$_H$
CLMA2:		
Protection command register	CLMA2PCMD	FF80 4010$_H$
Protection status register	CLMA2PS	FF80 4014$_H$
CLMA3:		
Protection command register	CLMA3PCMD	FF80 5010$_H$
Protection status register	CLMA3PS	FF80 5014$_H$
Port protection clusters:		
Port protection cluster 1:		
Protection command register	JPPCMD0	FF44 04C0$_H$
Protection status register	JPPROTS0	FF44 04B0$_H$
Port protection cluster 2:		
Protection command register	PPCMD0	FF40 4C00$_H$
Protection status register	PPROTS0	FF40 4B00$_H$
Port protection cluster 3:		
Protection command registers	PPCMD1	FF40 4C04$_H$
	PPCMD2	FF40 4C08$_H$
	PPCMD3	FF40 4C0C$_H$
	PPCMD4	FF40 4C10$_H$
	PPCMD10	FF40 4C28$_H$
	PPCMD11	FF40 4C2C$_H$

Table 3-54 Protection command registers overview (2/2)

Register Name	Shortcut	Address
Protection status registers	PPROTS1	FF40 4B04$_H$
	PPROTS2	FF40 4B08$_H$
	PPROTS3	FF40 4B0C$_H$
	PPROTS4	FF40 4B10$_H$
	PPROTS10	FF40 4B28$_H$
	PPROTS11	FF40 4B2C$_H$
Port protection cluster 4:		
Protection command registers	PPCMD12	FF40 4C30$_H$
	PPCMD13	FF40 4C34$_H$
	PPCMD21	FF40 4C54$_H$
	PPCMD22	FF40 4C58$_H$
	PPCMD24	FF40 4C60$_H$
	PPCMD25	FF40 4C64$_H$
	PPCMD26	FF40 4C68$_H$
	PPCMD27	FF40 4C6C$_H$
	PPCMD28	FF40 4C70$_H$
	PPCMD29	FF40 4C74$_H$
Protection status registers	PPROTS12	FF40 4B30$_H$
	PPROTS13	FF40 4B34$_H$
	PPROTS21	FF40 4B54$_H$
	PPROTS22	FF40 4B58$_H$
	PPROTS24	FF40 4B60$_H$
	PPROTS25	FF40 4B64$_H$
	PPROTS26	FF40 4B68$_H$
	PPROTS27	FF40 4B6C$_H$
	PPROTS28	FF40 4B70$_H$
	PPROTS29	FF40 4B74$_H$
Self-programming protection cluster:		
FLMD protection command register	FLMDPCMD	FF43 8004$_H$
FLMD protection error status register	FLMDPS	FF43 8008$_H$

3.15.6 Control protection clusters registers details

(1) PROTCMDn – Protection command registers

These registers are the protection command registers for initiating the write protection unlock sequence for write protected registers.

Index n "n" denotes the number of the protection command registers, refer to the table *"Protection command registers overview"* in the previous section.

Access This register can be written in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value Reading this register returns an undefined value.

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–
W	W	W	W	W	W	W	W

The usage of these registers is detailed in section *"Register protection unlock sequence"* above.

Table 3-55 PROTCMDn register contents

Bit position	Bit name	Function
7 to 0	–	Protection commands to enable writing to Isolated-Area-m registers

Caution If the Isolated-Area-1 is in DEEPSTOP mode, the protection command register PROTCMD1, that protects the Isolated-Area-1 clock selector registers CKSC_1n, is not accessible.
Any CPU attempt to access PROTCMD1 will lead to a microcontroller deadlock.

(2) PROTSn – Protection status registers

This registers shows the status of the protection unlock sequence operated via PROTCMDn.

Index n "n" denotes the number of the protection command registers, refer to the table *"Protection command registers overview"* in the previous section.

Caution This register must not be written.

Access This register can be read in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PROTSn ERR
R	R	R	R	R	R	R	R

Table 3-56 PROTSn register contents

Bit position	Bit name	Function
0	PROTSn ERR	Protected write sequence error monitor 0: no protection error 1: protection error occurred

Caution If the Isolated-Area-1 is in DEEPSTOP mode, the protection status register PROTS1 is not accessible.
Any CPU attempt to access PROTS1 will lead to a microcontroller deadlock.

3.15.7 Clock monitors protection cluster registers details

(1) CLMAnPCMD – CLMAn protection command register

This register is the protection command register for the CLMAnCTL0 register.

Index n "n" denotes the number of the Clock Monitor, refer to the table *"Protection command registers overview"* in the previous section.

Access This register can be written in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value Reading this register returns an undefined value.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W

Table 3-57 CLMAnPCMD register contents

Bit position	Bit name	Function
7 to 0	–	Protection commands to enable writing to CLMAnCTL0

(2) CLMAnPS – CLMAn protection status register

This register is used to verify whether the write protected register CLMAnCTL0 has been written successfully or not.

Index n "n" denotes the number of the Clock Monitor, refer to the table *"Protection command registers overview"* in the previous section.

Access This register can be read in 8-bit units.
Writing to this register is ignored.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CLMAn PRERR
R	R	R	R	R	R	R	R

Table 3-58 CLMAnPS register contents

Bit position	Bit name	Function
0	CLMAnPRERR	Indicates whether the write protected register CLMAnCTL0 has been written successfully: 0: Write operation successful 1: Write operation failed

3.15.8 Port protection clusters registers details

(1) PPCMDn – Port protection command register

PPCMDn is the protection command register for port group n.

Index n "n" denotes the port group, refer to the table *"Protection command registers overview"* in the previous section.

Access This register can be written in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value Reading this register returns an undefined value.

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–
W	W	W	W	W	W	W	W

Table 3-59 PPCMDn register contents

Bit position	Bit name	Function
7 to 0	–	Protection commands to enable writing to several port registers

(2) PPROTSn – Port protection status register

PPROTSn is the protection status registers for write protected registers of port group n. It shows the status of the protection sequence operated via PPCMDn.

Index n "n" denotes the port group, refer to the table *"Protection command registers overview"* in the previous section.

Caution This register must not be written.

Access This register can be read in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PPROTSn PRERR
R	R	R	R	R	R	R	R

Table 3-60 PPROTSn register contents

Bit position	Bit name	Function
0	PPROTSn PRERR	Protected write sequence error monitor 0: no protection error 1: protection error occured

3.15.9 Self-programming protection cluster registers details

(1) FLMDPCMD – FLMD protection command register

This register is the protection command register for the FLMDCNT register.

Access This register can be written in 8-bit units.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value Reading this register returns an undefined value.

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–
W	W	W	W	W	W	W	W

Table 3-61 FLMDPCMD register contents

Bit position	Bit name	Function
7 to 0	–	Protection commands to enable writing to FLMDCNT

(2) FLMDPS – FLMD protection error status register

This register is used to verify whether the write protected register FLMDCNT has been written successfully or not.

Access This register can be read in 8-bit units.
Writing to this register is ignored.

Address Refer to the table *"Protection command registers overview"* in the previous section.

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FLMD PRERR
R	R	R	R	R	R	R	R

Table 3-62 FLMDPS register contents

Bit position	Bit name	Function
0	FLMDPRERR	Indicates whether the write protected register FLMDCNT has been written successfully: 0: Write operation successful 1: Write operation failed

Chapter 4 External Memory Controller (MEMC)

This chapter contains a generic description of the Bus and Memory Controller.

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

4.1 V850E2/Fx4-H MEMC Features

The following external devices can be connected to the microcontroller device:

- SRAM

Features summary

The bus and memory control of the microcontroller device provides:

- Multiplexed bus mode with up to 32 address / data signals
- Selectable data bus width for each chip select area (8, 16 or 32 bits)
- 3 chip select signals externally available (CS2 to CS4)
- Address setup/hold wait state can be inserted for each chip select area
- The following wait functions can be set individually for each chip select region:
  - Programmable data wait
  - Data hold wait
  - Data setup wait
  - Address setup wait
  - Address hold wait
  - Idle cycle insertion
- The data endian format can be selected for each chip select region individually.

Clock supply

Since the External Memory Controller is part of the CPU Subsystem, it is supplied with the same clock CKSCLK_000 as the CPU Subsystem.
Refer to the section *"CPU Subsystem"* in chapter *"CPU System Functions"* for details.

MEMC H/W reset

The External Memory Controller and its registers are initialized by the following reset signal:

Table 4-1 MEMC reset signal

MEMC	Reset signal
MEMC	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

4.2 Overview

The external memory access function of this microcontroller provides an interface for connecting memory, ASICs, etc., outside this microcontroller. This microcontroller allows the connection of the following type of memory.

- SRAM

This microcontroller provides 3 chip select areas, and the bus size and wait time can be set independently for each chip select area.

In addition to an external wait function, this microcontroller has various programmable wait functions that can be set for each chip select area, allowing the connection of various types of memory.

4.2.1 Operation mode, connectable memory types

(1) Multiplexed bus mode

This is an operation mode that connects address output and data input/output to external memory using the same signal line. In this operation mode, the memory that can be connected is limited to SRAM for all chip select areas, but it is possible to reduce the number of pins required for external memory connection.

4.2.2 Chip select output function

The external bus area of the memory space is divided into three chip select areas, and a chip select signal can be output for each chip select area. The allocation of these chip select areas is fixed by the system and cannot be changed through programming.

4.2.3 Operation setting function

Operation via the BCT0 and BCT1 registers can be enabled/disabled for each chip select area.

4.2.4 Bus sizing function

The bus size can be selected from 8 bits or 16 bits for each chip select area. To execute access when the data size exceeds the selected bus width, divide the data into sizes smaller than the bus width by using the bus sizing function.

4.2.5 Data endian setting function

The data endian (little endian/big endian) can be specified for the chip select areas. However, since the software development tools made by Renesas Electronics (assemblers and debuggers) only support little endian, instruction fetch operations in big endian are not possible.

The initial status of all chip select areas is little endian.

4.2.6 Programmable wait setting functions

This microcontroller has the following wait functions, which can be set for each chip select area.

- Programmable data wait
- Data hold wait
- Data setup wait (in multiplexed bus mode, during write access)
- Address setup wait (in multiplexed bus mode)
- Address hold wait (in multiplexed bus mode)
- Idle cycle function

4.2.7 External wait function

When accessing SRAM data waits of any width can be inserted from outside from the WAITZ pin. The WAITZ pin is sampled just before the data output cycle, and the data latch timing can be delayed by any amount.

4.3 Registers

These registers are used to control the external memory controller (SRAM memory controller) of this microcontroller.

Note The clock counts shown in this section indicate the clock count when operating on the external bus clock unless otherwise specified.

Table 4-2 OS Timer registers overview

Register name	Shortcut	Address
Bus size configuration register	BSC	FFFF 7200$_H$
Data endian configuration register	DEC	FFFF 7202$_H$
Bus cycle type configuration register 0	BCT0	FFFF 7204$_H$
Bus cycle type configuration register 1	BCT1	FFFF 7206$_H$
Data wait configuration register 0	DWC0	FFFF 7208$_H$
Data wait configuration register 1	DWC1	FFFF 720A$_H$
Data hold wait configuration register	DHC	FFFF 720C$_H$
Data setup wait configuration register	DSC	FFFF 720E$_H$
Address wait configuration register 0	AWC0	FFFF 7210$_H$
Address wait configuration register 1	AWC1	FFFF 7212$_H$
Idle cycle configuration register 0	ICC0	FFFF 7214$_H$
Idle cycle configuration register 1	ICC1	FFFF 7216$_H$
External wait error configuration register	EWC	FFFF 721A$_H$

(1) BSC - Bus size configuration register

The BSC register is used to set the bus size of the external bus for each chip select area.

Access This register can be read or written in 16-bit units.

Address FFFF 7200_H

Initial Values 5555_H

15	14	13	12	11	10	9	8
BS71	BS70	BS61	BS60	BS51	BS50	BS41	BS40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
BS31	BS30	BS21	BS20	BS11	BS10	BS01	BS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-3 BSC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	BSn1, BSn0	Bus size setting bits These bits set the bus width of each chip select area.

BSn1	BSn0	Bus size
0	0	8 bits
0	1	16 bits
1	0	Setting prohibited
1	1	Setting prohibited

(2) DEC - Data endian configuration register

The DEC register is used to set the endianness of the external bus.

Access This register can be read or written in 16-bit units.

Address FFFF 7202_H

Initial Values 0000_H

15	14	13	12	11	10	9	8
0	DE7	0	DE6	0	DE5	0	DE4
R	R/W	R	R/W	R	R/W	R	R/W
7	6	5	4	3	2	1	0
0	DE3	0	DE2	0	DE1	0	DE0
R	R/W	R	R/W	R	R/W	R	R/W

Table 4-4 DEC register contents

Bit position	Bit name	Function
14, 12, 10, 8, 6, 4, 2, 0	DEn	Data endian setting bits These bits set the endian of each chip select area. 0 : Little endian 1 : Big endian

(3) BCT0, BCT1 - Bus cycle type configuration registers 0 and 1

The BCT0 and BCT1 registers are used to set a bus cycle type of the external bus for each chip select area.

Access These registers can be read or written in 16-bit units.

Address BCT0: FFFF 7204_H
BCT1: FFFF 7206_H

Initial Values BCT0: 8800_H
BCT1: 8008_H

BCT0

15	14	13	12	11	10	9	8
ME3	0	BCT31	BCT30	ME2	0	BCT21	BCT20
R/W	R	R/W	R/W	R/W	R	R/W	R/W
7	6	5	4	3	2	1	0
ME1	0	BCT11	BCT10	ME0	0	BCT01	BCT00
R/W	R	R/W	R/W	R/W	R	R/W	R/W

BCT1

15	14	13	12	11	10	9	8
ME7	0	BCT71	BCT70	ME6	0	BCT61	BCT60
R/W	R	R/W	R/W	R/W	R	R/W	R/W
7	6	5	4	3	2	1	0
ME5	0	BCT51	BCT50	ME4	0	BCT41	BCT40
R/W	R	R/W	R/W	R/W	R	R/W	R/W

Table 4-5 BCT0, BCT1 registers content

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8,
5:4,
1:0</td><td>BCTn1,
BCTn0</td><td>Bus cycle type setting bits
In the multiplexed bus mode, the set value of these bits is ignored, and a bus cycle of SRAM type is always generated (n = 0 to 7).
<table>
<tr><th>BCTn1</th><th>BCTn0</th><th>Memory type</th></tr>
<tr><td>0</td><td>0</td><td>SRAM</td></tr>
<tr><td colspan="2">other</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>15, 11,
7, 3</td><td>MEn</td><td>Memory controller operation enable bits
These bits enable or disable the operation of the internal memory controller of this microcontroller for each chip select area.
0 : Memory controller operation stopped (External bus cycle is not generated.)
1 : Memory controller operation enabled
If the operation is disabled by the MEn bit, this microcontroller does not generate an external bus cycle, and the ongoing reading/writing is completed (n = 0 to 7).</td></tr>
</table>

(4) DWC0, DWC1 - Data wait configuration registers 0 and 1

The DWC0 and DWC1 registers are used to set the number of data wait states of the external bus.

The value set to the DWC0 and DWC1 registers becomes valid in the following bus cycles.

- Data transfer cycle in multiplexed bus mode

Access These registers can be read or written in 16-bit units.

Address DWC0: FFFF 7208_H
DWC1: FFFF $720A_H$

Initial Values DWC0: $FFFF_H$
DWC1: $FFFF_H$

	15	14	13	12	11	10	9	8
DWC0	DW33	DW32	DW31	DW30	DW23	DW22	DW21	DW20
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	DW13	DW12	DW11	DW10	DW03	DW02	DW01	DW00
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	15	14	13	12	11	10	9	8
DWC1	DW73	DW72	DW71	DW70	DW63	DW62	DW61	DW60
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	DW53	DW52	DW51	DW50	DW43	DW42	DW41	DW40
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-6 DWC0, DWC1 registers content

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 12,
11 to 8,
7 to 4,
3 to 0</td><td>DWn3,
DWn2,
DWn1,
DWn0</td><td>Data wait setting bits
These bits set the number of data wait states for each chip select area.
<table>
<tr><th>DWn3</th><th>DWn2</th><th>DWn1</th><th>DWn0</th><th>Number of data wait states</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>No data wait</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1 clock</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>2 clocks</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>3 clocks</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>4 clocks</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>5 clocks</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>6 clocks</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>7 clocks</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>8 clocks</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>9 clocks</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>10 clocks</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>11 clocks</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>12 clocks</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>13 clocks</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>14 clocks</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>15 clocks</td></tr>
</table>
</td></tr>
</table>

(5) DHC - Data hold wait configuration register

The DHC register is used to set the number of extended data hold wait states for each chip select area in the write cycle of the external bus.

One data hold wait state is always inserted in a write cycle. By use of the DHC register extended data hold wait states can be inserted, thus the total number of data hold wait states is DHC + 1.

Access This register can be read or written in 16-bit units.

Address FFFF 720C$_H$

Initial Values 0000$_H$

15	14	13	12	11	10	9	8
DH71	DH70	DH61	DH60	DH51	DH50	DH41	DH40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DH31	DH30	DH21	DH20	DH11	DH10	DH01	DH00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-7 DHC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	DHn1, DHn0	Data hold wait setting bits These bits set the number of extended data hold wait states for each chip select area.

DHn1	DHn0	Number of extended data hold wait states
0	0	No extended data hold wait
0	1	1 clock
1	0	2 clocks
1	1	3 clocks

(6) DSC - Data setup wait configuration register

The DSC register is used to set the number of data setup wait states of the external bus for each chip select area in the multiplexed bus mode.

Access This register can be read or written in 16-bit units.

Address FFFF 720E$_H$

Initial Values 0000$_H$

15	14	13	12	11	10	9	8
DS71	DS70	DS61	DS60	DS51	DS50	DS41	DS40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
DS31	DS30	DS21	DS20	DS11	DS10	DS01	DS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-8 DSC register contents

Bit position	Bit name	Function
15:14, 13:12, 11:10, 9:8, 7:6, 5:4, 3:2, 1:0	DSn1, DSn0	Data setup wait setting bits These bits set the number of data setup wait states for each chip select area.

DSn1	DSn0	Number of data setup wait states
0	0	No data setup wait
0	1	1 clock
1	0	2 clocks
1	1	3 clocks

(7) AWC0, AWC1 - Address wait configuration registers 0 and 1

The AWC registers are used to set the address wait period of the external bus for each chip select area.

Setting the AWC register is enabled only in the multiplexed bus mode.

Access These registers can be read or written in 16-bit units.

Address AWOC0: FFFF 7210_H
AWOC1: FFFF 7212_H

Initial Values AWOC0: $FFFF_H$
AWOC1: $FFFF_H$

	15	14	13	12	11	10	9	8
AWC0	AHW31	AHW30	ASW31	ASW30	AHW21	AHW20	ASW21	ASW20
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	AHW11	AHW10	ASW11	ASW10	AHW01	AHW00	ASW01	ASW00
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

	15	14	13	12	11	10	9	8
AWC1	AHW71	AHW70	ASW71	ASW70	AHW61	AHW60	ASW61	ASW60
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	AHW51	AHW50	ASW51	ASW50	AHW41	AHW40	ASW41	ASW40
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-9 AWC0, AWC1 registers content

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8,
5:4,
1:0</td><td>ASWn1,
ASWn0</td><td>Address setup wait setting bits
These bits set the number of address setup wait states for each chip select area.
<table>
<tr><th>ASWn1</th><th>ASWn0</th><th>Number of address setup wait states</th></tr>
<tr><td>0</td><td>0</td><td>No address setup wait</td></tr>
<tr><td>0</td><td>1</td><td>1 clock</td></tr>
<tr><td>1</td><td>0</td><td>2 clocks</td></tr>
<tr><td>1</td><td>1</td><td>3 clocks</td></tr>
</table></td></tr>
<tr><td>15:14,
11:10,
7:6,
3:2</td><td>AHWn1,
AHWn0</td><td>Address hold wait setting bits
These bits set the number of address hold wait states for each chip select area.
<table>
<tr><th>AHWn1</th><th>AHWn0</th><th>Number of address hold wait states</th></tr>
<tr><td>0</td><td>0</td><td>No address hold wait</td></tr>
<tr><td>0</td><td>1</td><td>1 clock</td></tr>
<tr><td>1</td><td>0</td><td>2 clocks</td></tr>
<tr><td>1</td><td>1</td><td>3 clocks</td></tr>
</table></td></tr>
</table>

(8) ICC0, ICC1 - Idle cycle configuration registers 0 and 1

The ICC registers are used to set the number of idle cycles of the external bus. The number of idle cycles can be set for each chip select area and in the read cycle or write cycle.

Access These registers can be read or written in 16-bit units.

Address ICC0: FFFF 7214_H
ICC1: FFFF 7216_H

Initial Values ICC0: 3333_H
ICC1: 3333_H

Caution The number of idle cycles set by the ICCm register (m = 0, 1) is invalid during the burst read cycle and bus sizing cycle.

ICC0

15	14	13	12	11	10	9	8
WIC31	WIC30	RIC31	RIC30	WIC21	WIC20	RIC21	RIC20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
WIC11	WIC10	RIC11	RIC10	WIC01	WIC00	RIC01	RIC00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ICC1

15	14	13	12	11	10	9	8
WIC71	WIC70	RIC71	RIC70	WIC61	WIC60	RIC61	RIC60
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
WIC51	WIC50	RIC51	RIC50	WIC41	WIC40	RIC41	RIC40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 4-10 ICC0, ICC1 registers content

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13:12,
9:8
5:4
1:0</td><td>RICn1,
RICn0</td><td>Idle cycle setting bits after read cycle
These bits set the number of idle cycles for each chip select area after a read cycle.
<table>
<tr><th>RICn1</th><th>RICn0</th><th>Number of idle cycles</th></tr>
<tr><td>0</td><td>0</td><td>No idle cycle</td></tr>
<tr><td>0</td><td>1</td><td>1 clock</td></tr>
<tr><td>1</td><td>0</td><td>2 clocks</td></tr>
<tr><td>1</td><td>1</td><td>3 clocks</td></tr>
</table>
Setting the RICn1 and RICn0 bits is enabled for read accesses in all the bus modes and of all the bus cycle types.</td></tr>
<tr><td>15:14
11:10
7:6
3:2</td><td>WICn1
WICn0</td><td>Idle cycle setting bits after write cycle
These bits set the number of idle cycles for each chip select area after a write cycle.
<table>
<tr><th>WICn1</th><th>WICn0</th><th>Number of idle cycles</th></tr>
<tr><td>0</td><td>0</td><td>No idle cycle</td></tr>
<tr><td>0</td><td>1</td><td>1 clock</td></tr>
<tr><td>1</td><td>0</td><td>2 clocks</td></tr>
<tr><td>1</td><td>1</td><td>3 clocks</td></tr>
</table>
Setting the WICn1 and WICn0 bits is enabled for write accesses in all the bus modes and of all the bus cycle types.</td></tr>
</table>

(9) EWC - External wait error configuration register

The EWC register is used to enable or disable the external wait error function for each chip select area.

Setting the EWC register is enabled in the following bus cycles.

- Multiplexed bus mode (SRAM bus cycle type)

Access This register can be read or written in 16-bit units.

Address FFFF 721A$_H$

Initial Values 0000$_H$

15	14	13	12	11	10	9	8
0	EW7	0	EW6	0	EW5	0	EW4
R	R/W	R	R/W	R	R/W	R	R/W
7	6	5	4	3	2	1	0
0	EW3	0	EW2	0	EW1	0	EW0
R	R/W	R	R/W	R	R/W	R	R/W

Table 4-11 EWC register contents

Bit position	Bit name	Function
14, 12, 10, 8, 6, 4, 2, 0	EWn	External wait error setting bits These bits enable or disable an external wait error in each chip select area. 0: Disables external wait error. 1: Enables external wait error. When this function is enabled, this microcontroller forcibly cancels the wait state when it has detected the external wait signal for the duration of 128 consecutive clocks, and the CPU generates the SysError exception.

4.4 Bus cycle type setting function

The bus cycles listed below can be used. Whether or not to generate a bus cycle and its type can be set for each chip select area by setting the BCT0 and BCT1 registers.

- SRAM bus cycle type

Moreover, in the multiplexed bus mode, it is possible to generate a bus cycle using the same signal line for address output and data input/output.

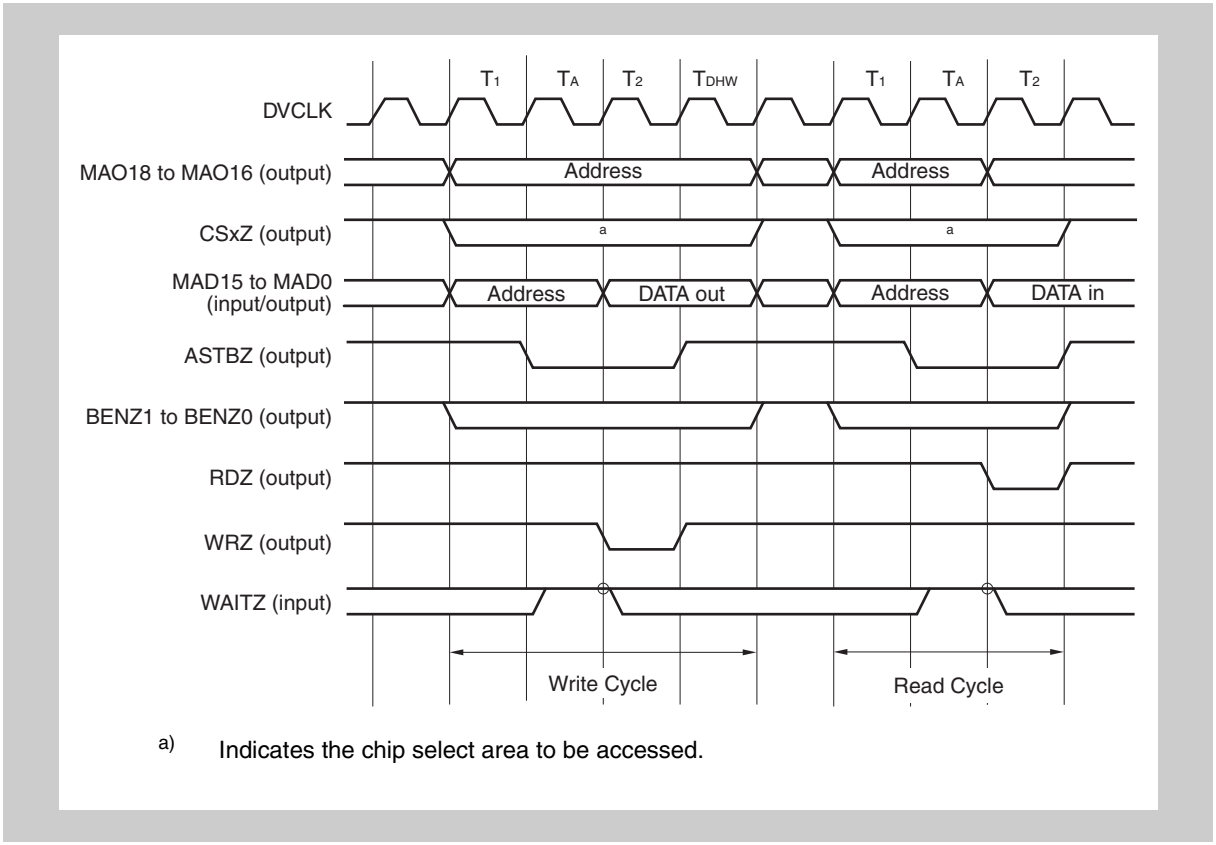
4.4.1 Multiplexed bus mode

The external bus access function can be used in the multiplexed bus mode.

In the multiplexed bus mode, external memory connection is possible through the use of address output and data input/output by switching the same signal line.

In addition to the address/data signals, external bus control is executed using the chip select signal, address strobe signal, read strobe signal, and write strobe signal.

The multiplexed bus mode is used when connecting an external ASIC in the multiplexed bus mode to reduce the number of external signals.



a) Indicates the chip select area to be accessed.

Figure 4-1 Multiplexed bus mode (read cycle/write cycle)

The valid wait functions in the multiplexed bus mode are listed below.

- Programmable data wait of 15 clocks max. through the DWC register
- Data wait through external pin
- Insertion of idle cycle of up to 3 clocks through the ICC register
- Data hold wait of up to 3 clocks through the DHC register
- Data setup wait of up to 3 clocks through the DSC register (write only)

Note When an external memory is connected using the multiplexed bus mode, an external data latch is required. For details, refer to *4.7 "Memory Connection Examples"*.

4.5 Bus control function

4.5.1 Chip select output function

The connected external memory area is managed divided into 3 chip select areas up to CSn (n = 2 to 4, as shown in *Figure 4-2 “External memory map” on page 235* .

When a bus cycle is generated for the external bus, This microcontroller makes the CSZn (n = 2 to 4) output pins corresponding to the access target address active (low level), along with outputting the access target address from the MAO[28:0] pins.

The various settings for the external bus, such as the bus size and number of wait/idle states, can all be made for each chip select area.

By using these functions, different types of memory can be connected for each chip select area.

The allocation of the chip select areas is fixed by the system and cannot be changed through programming.

The memory map is shown next.

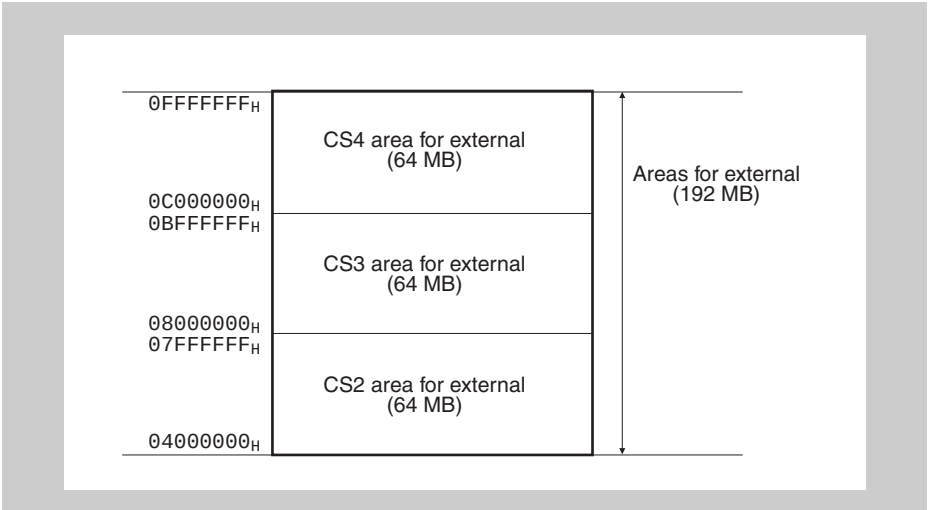


Figure 4-2 External memory map

4.5.2 Operation enable/operation disable setting function

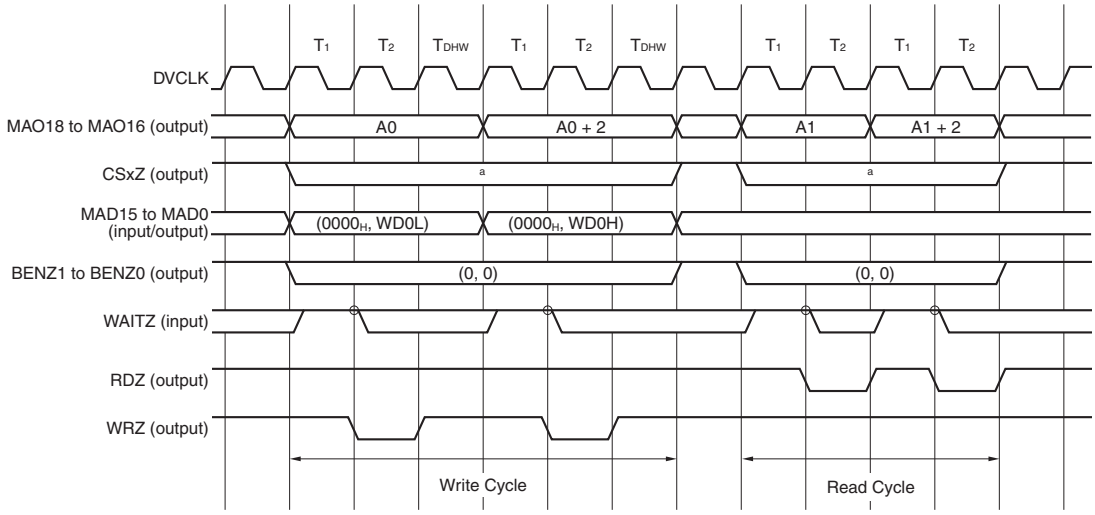
The operation can be enabled/disabled for each of the above-mentioned chip select areas shown in *4.5.1 "Chip select output function"* through the setting of the MEn (n = 0 to 7) bits of the BCT0 and BCT1 registers.

If an access request is issued from the CPU (or DMA) to a chip select area for which operation has been disabled with this function, no external bus cycle is generated, the write value is ignored, and the read value becomes 0000 0000_H.

4.5.3 Bus size setting function

Access requests from the CPU (or DMA) are executed after being divided in accordance with the bit width of the external bus of the access destination.

The bit width of the external bus can be selected from 16 and 8 bits for each chip select area by setting the BSC register.



a) Indicates the chip select area to be accessed.

Figure 4-3 SRAM cycle for 16-bit bus size

4.5.4 Data endian setting function

Either little endian or big endian can be selected as the data endian of the external bus interface. This setting can be made for each chip select area with the DEC register. Initial setting through input pins is not possible. The initial status is little endian for all the chip select areas.

Access to a chip select area for which big endian has been specified as the data endian is performed in big endian.

This function can be used only for the SRAM access type.

Caution In this microcontroller, instruction fetch operation with big endian is not supported.

Note For details on the data flow for each external bus size and data size, refer to *4.8 "Data Flow"*.

4.6 Wait Functions

Wait functions listed below.

Table 4-12 Wait functions

Wait Function		Data Wait		Data Hold Wait	Data Setup Wait	Address Wait	Idle
		Program-mable	External Wait				
Multiplexed bus mode	Read	√	√	-	-	√	√
	Write	√	√	√	√	√	√
Setting registers		DWC0 DWC1	-	DHC	DSC	AWC	ICC0 ICC1
Max. number of waits		15	-	3	3	3	3

4.6.1 Programmable data wait function

This wait function is for delaying the data latch timing by extending the read strobe and write strobe periods.

This function is enabled during all write accesses and at the first data transfer timing in the SRAM (multiplexed) modes.

Up to 15 cycles can be inserted.

Setting individual chip select areas with the DWC0 and DWC1 registers is possible.

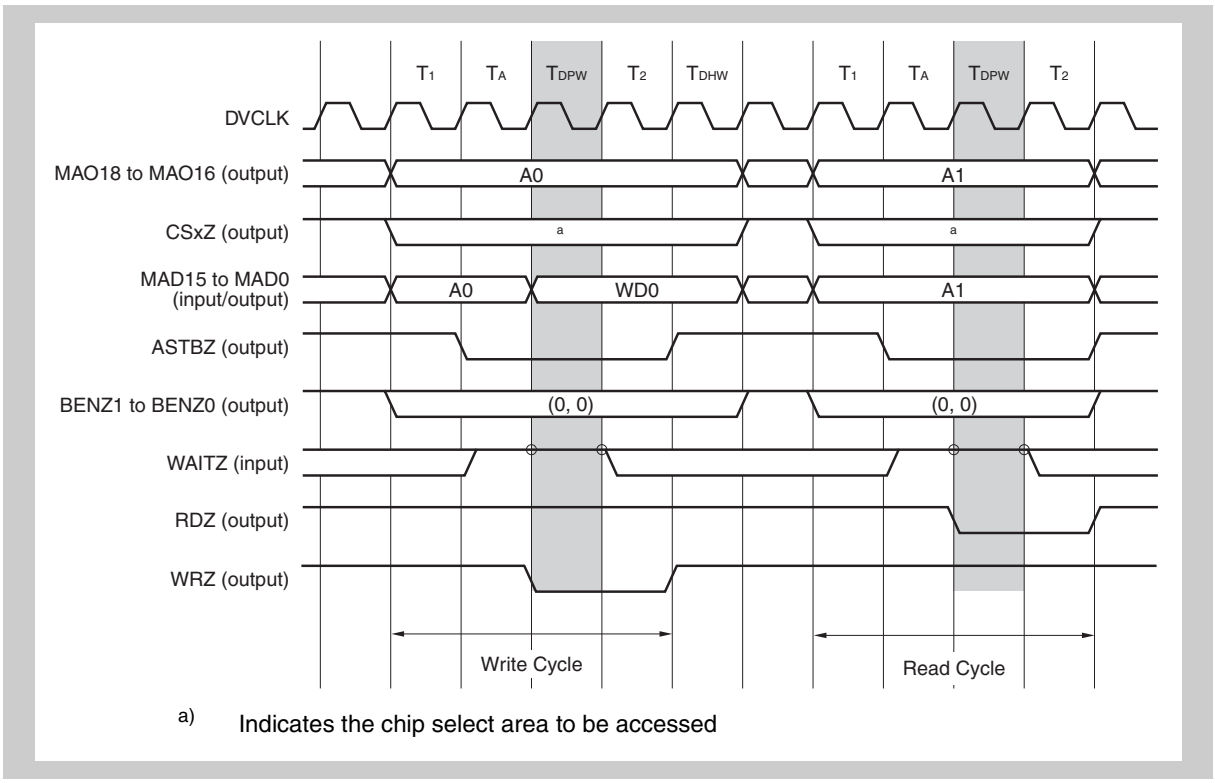


Figure 4-4 Programmable data wait for multiplexed bus mode

4.6.2 External wait function

Data waits of any length can be inserted from the WAITZ pin.

The WAITZ pin input level is sampled immediately after completion of the T_A, T_1 cycles and the T_{DPW}, T_{DEW} cycles.

Data wait cycles obtained by the programmable data wait set by data wait control registers 0 and 1 (DWC0, DWC1 registers) and the external wait specification set by the WAITZ pin input, are inserted.

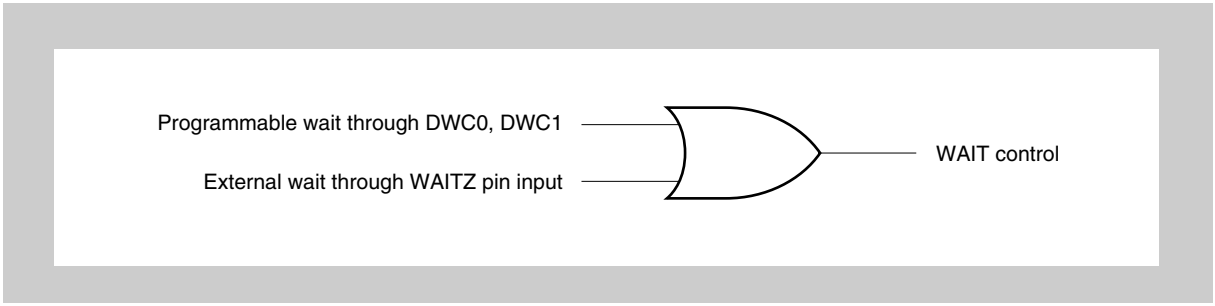
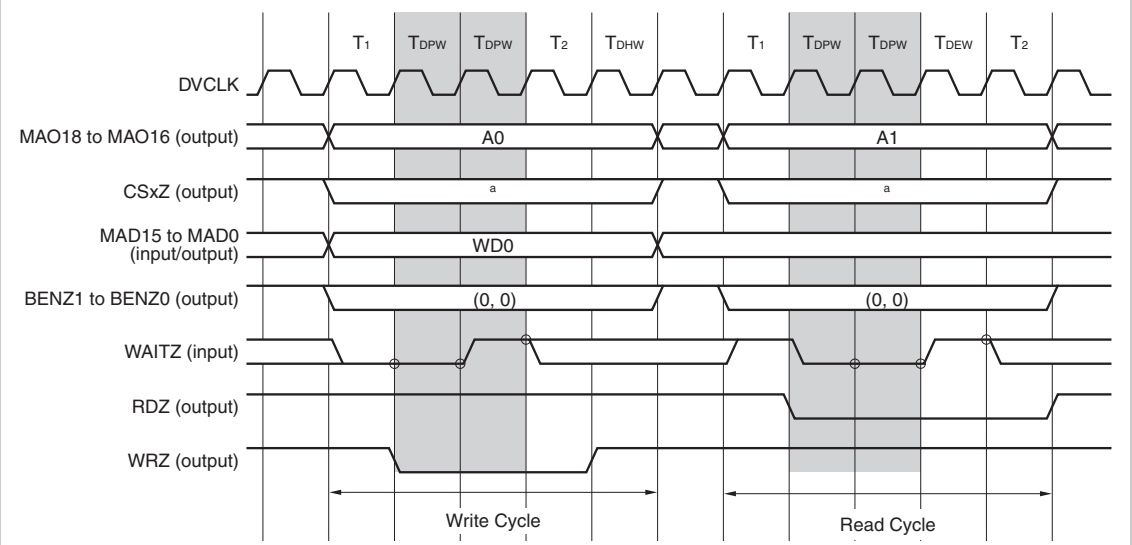


Figure 4-5 Internal data wait generator



a) Indicates the chip select area to be accessed.

Figure 4-6 Relationship between external data wait and programmable data wait (when DWC = 2)

4.6.3 Data setup wait function

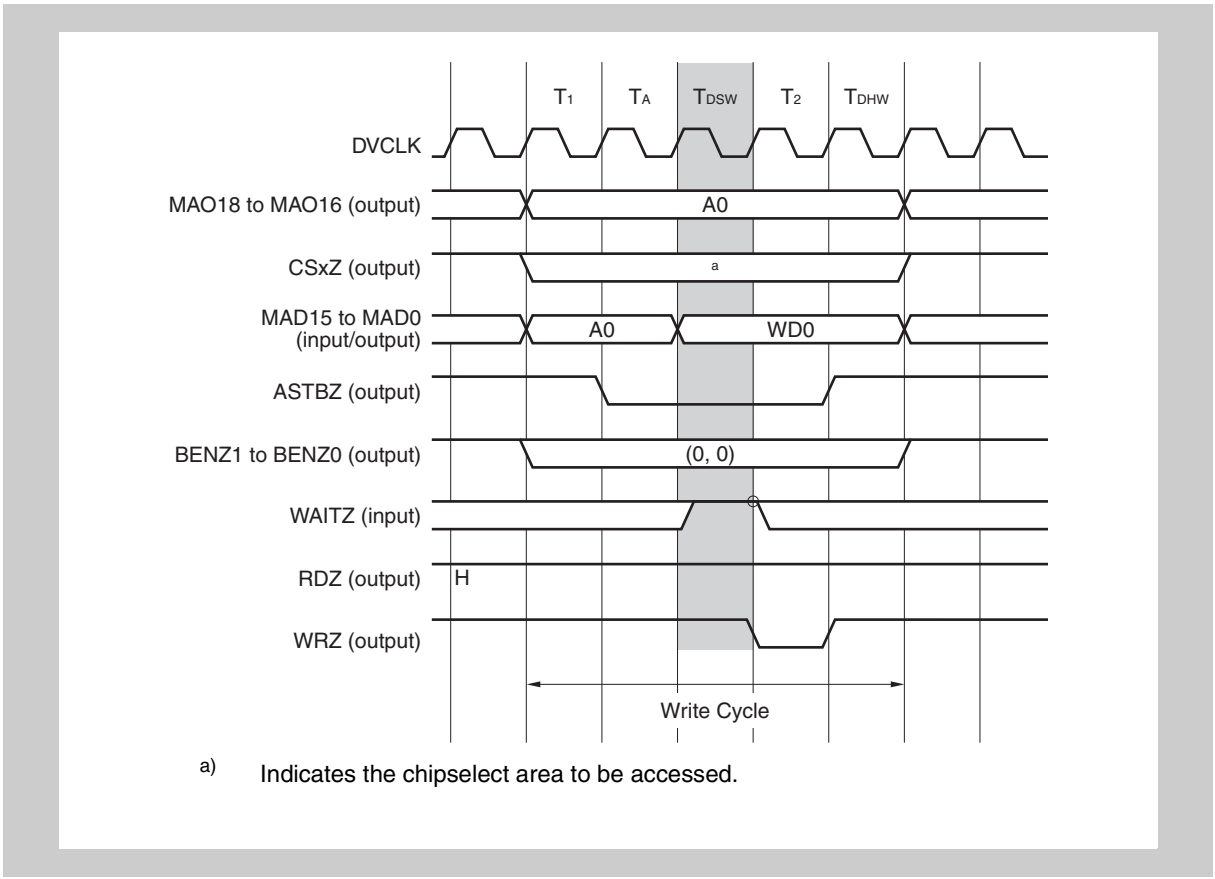
This function inserts a wait prior to the transfer state in order to secure the setup time for the data write strobe.

This function is enabled only during write cycles in the multiplexed bus mode.

Up to 3 cycles can be inserted.

The number of wait cycles to be inserted can be set for each chip select area with the DSC register.

The initial status is no wait for any of the chip select areas.



a) Indicates the chipselect area to be accessed.

Figure 4-7 Data setup wait

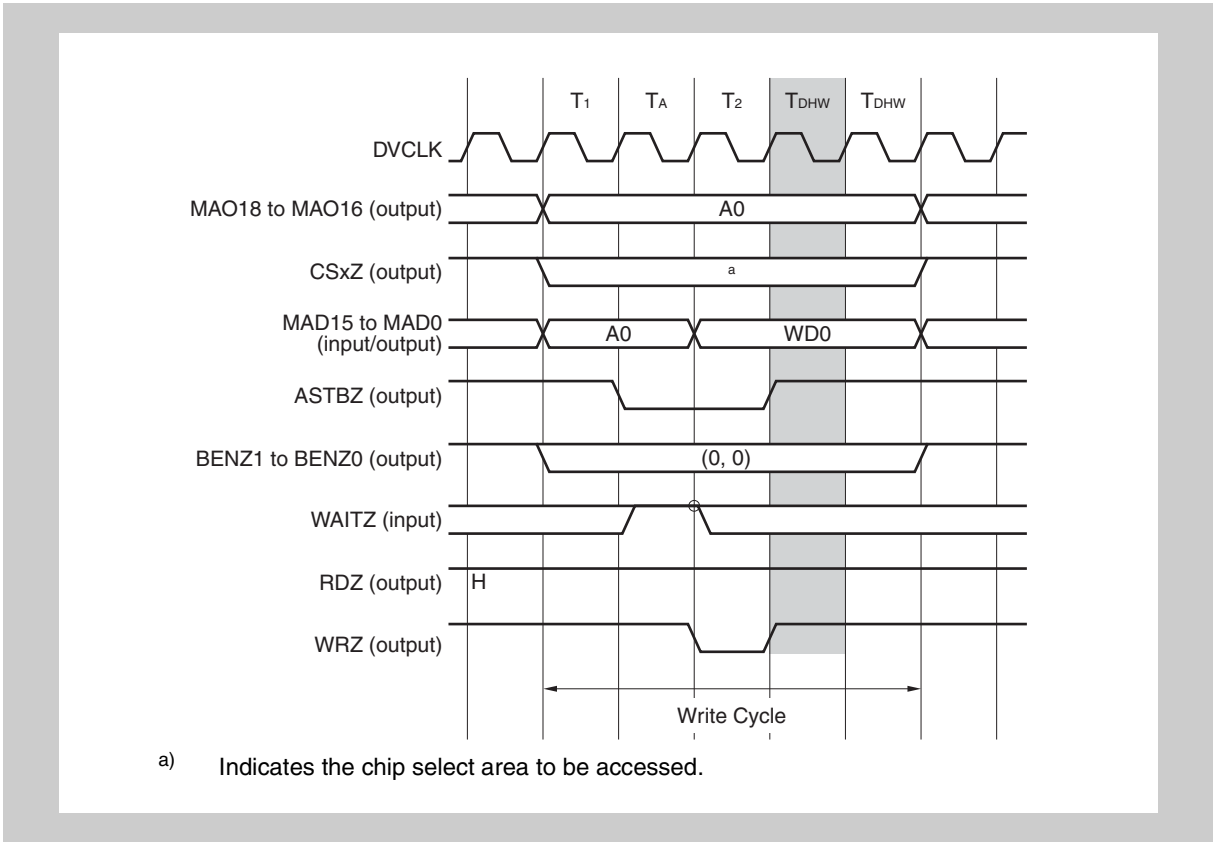
4.6.4 Data hold wait function

This function inserts a wait for the state following the rising edge of the write strobe signal in order to secure the hold time for the data write strobe.

This function is enabled only during the write cycle for all bus cycle types.

This microcontroller always inserts 1 data hold wait state upon occurrence of a write cycle. This data hold wait extends the DHC register setting by up to 3 cycles, allowing insertion of 4 cycles.

The number of data hold wait extensions can be set for each chip select area with the DHC register. The initial status is no wait extension for any of the chip select areas (1 data hold wait cycle).



a) Indicates the chip select area to be accessed.

Figure 4-8 Data hold wait (in multiplexed bus mode)

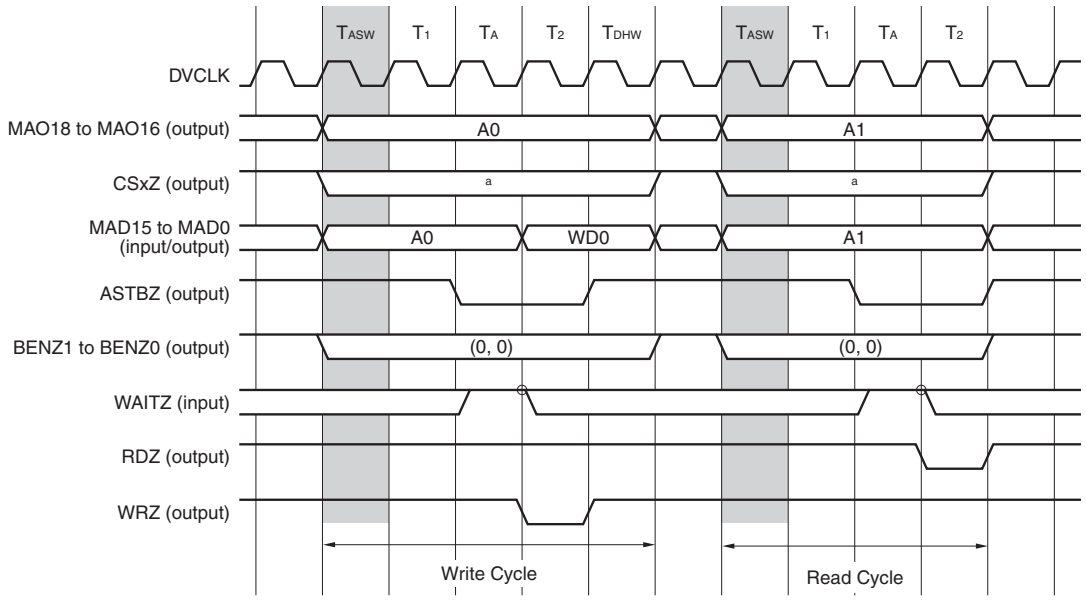
4.6.5 Address setup wait function

This function inserts a wait before the address transfer state in order to secure the setup time for the address strobe in the multiplexed bus mode.

This function is enabled only in the multiplexed bus mode.

Up to 3 cycles can be inserted.

Setting each chip select area with the AWC0 and AWC1 registers is possible.



a) Indicates the chip select area to be accessed.

Figure 4-9 Address setup wait

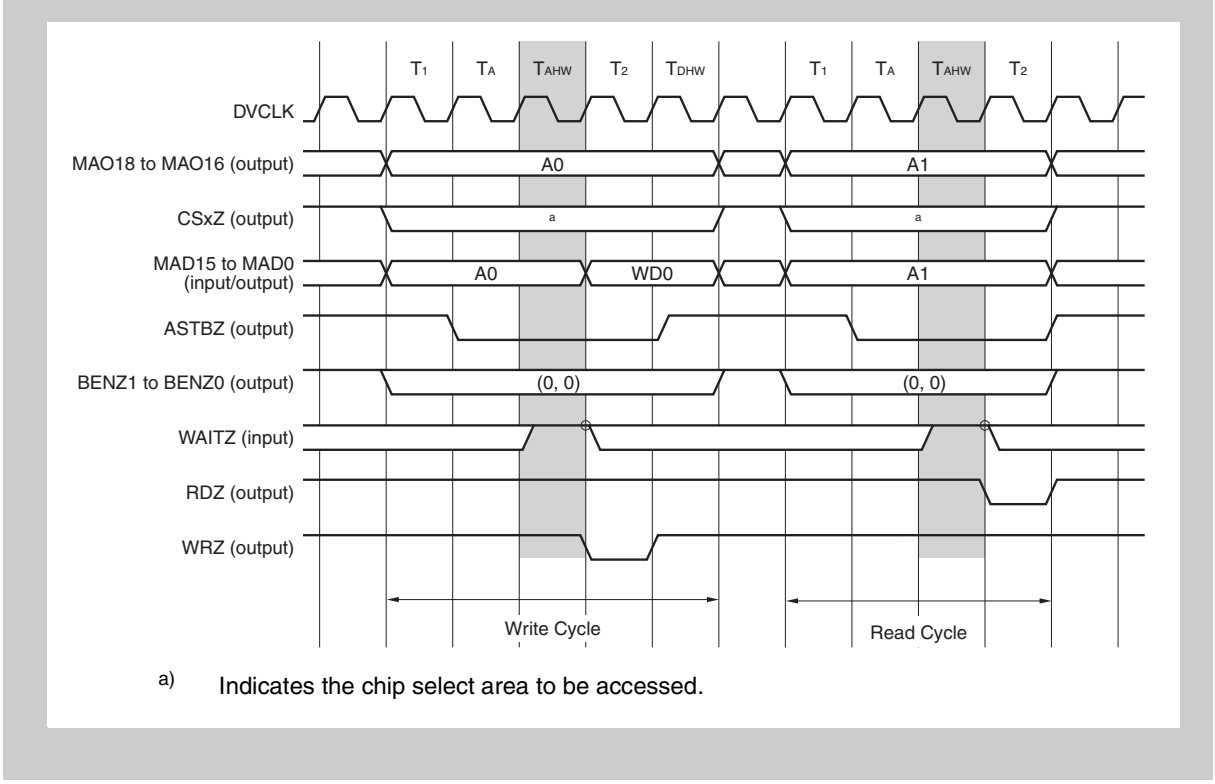
4.6.6 Address hold wait function

This function inserts a wait after the address transfer state in order to secure the hold time for the address strobe in the multiplexed bus mode.

This function is enabled only in the multiplexed bus mode.

Up to 3 cycles can be inserted.

Setting each chip select area with the AWC0 and AWC1 registers is possible.



a) Indicates the chip select area to be accessed.

Figure 4-10 Address hold wait

4.6.7 Idle insertion function

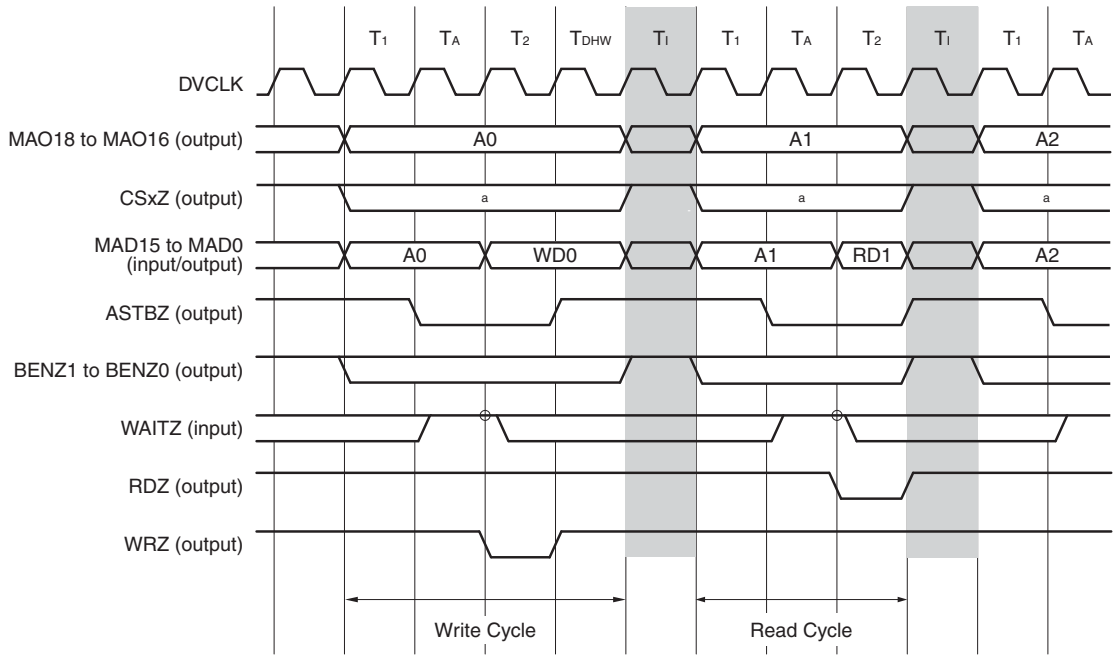
This function inserts an idle state after the last state of each cycle in order to prevent bus conflicts between cycles.

Up to 3 cycles can be inserted for all memory types.

This function can be set independently after a read cycle or after a write cycle for each chip select area by setting the ICC0 and ICC1 registers.

The initial status is no idle cycle for any of the chip select areas.

Caution The interval from completion of a bus cycle until the occurrence of subsequent bus cycles from the CPU (or DMA) lasts 1 cycle, regardless of the idle cycle setting. Therefore, a 1-cycle interval occurs between bus cycles even if the setting is no idle cycle.



a) Indicates the chip select area to be accessed.

Figure 4-11 Idle insertion

4.7 Memory Connection Examples

4.7.1 Multiplexed bus mode connection example

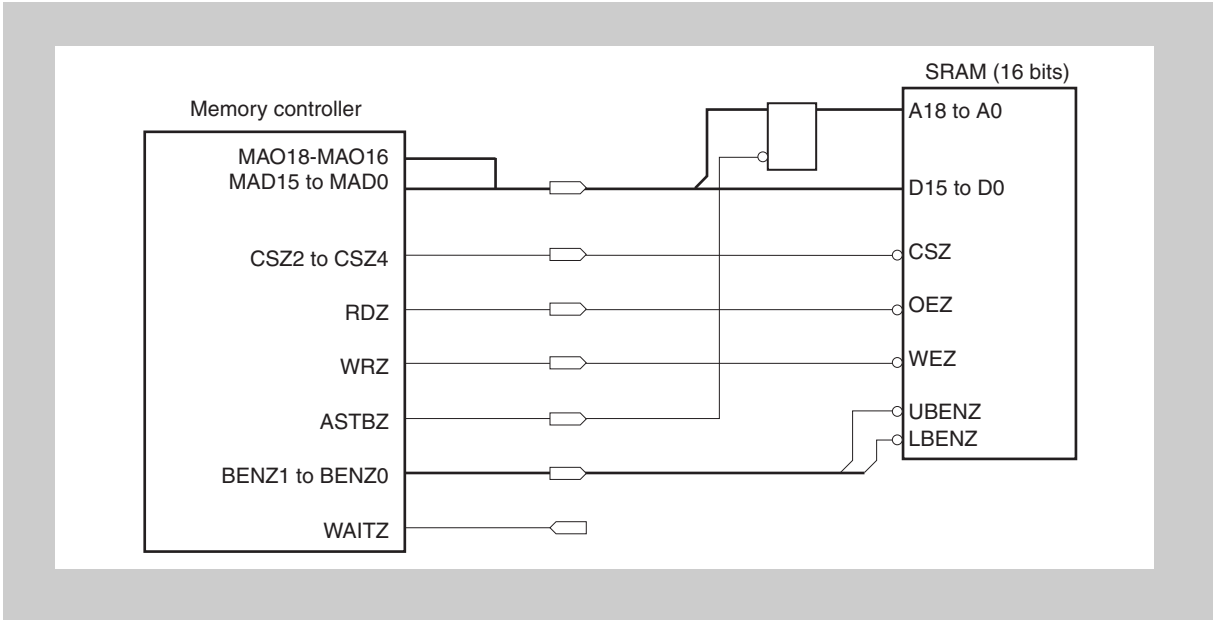
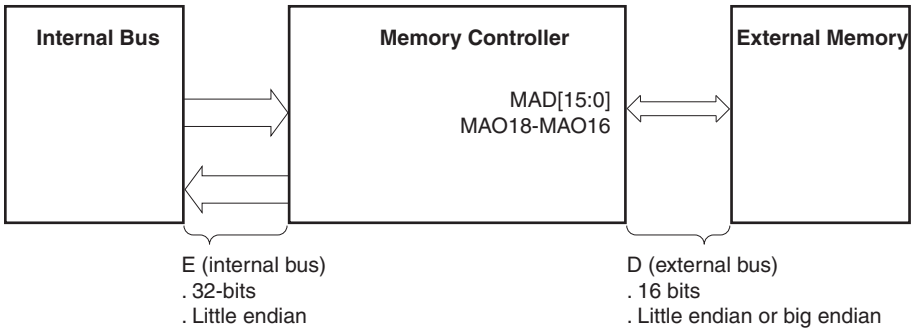


Figure 4-12 Multiplexed bus mode connection example

4.8 Data Flow

The data transfer flow to external memory differs according to factors such as the data width, endian specification, external bus width, and start address.



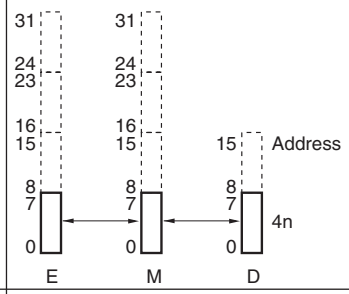
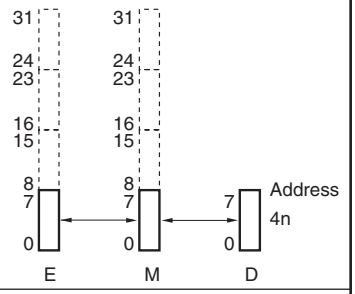
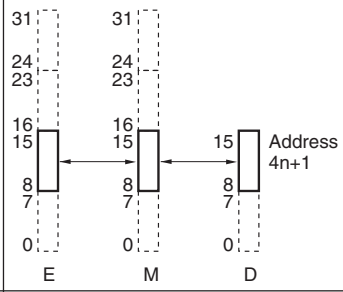
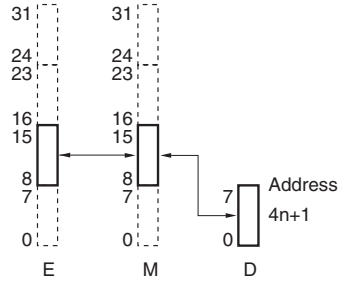
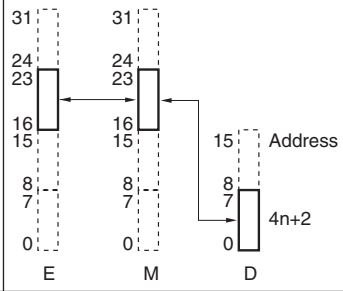
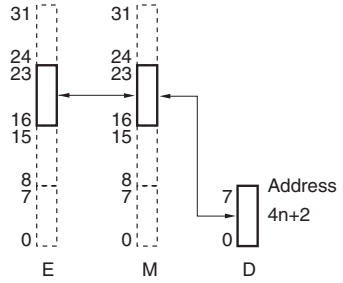
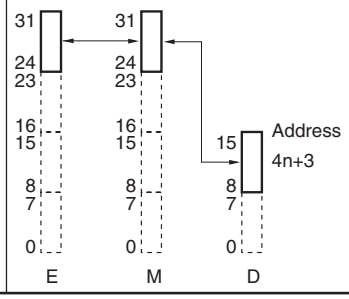
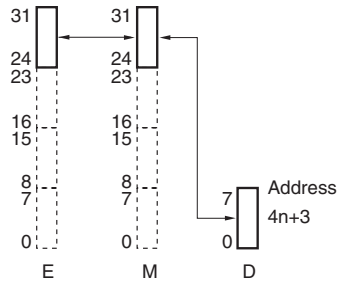
Note B:Byte data
HW: Halfword data
W: Word data

Figure 4-13 Internal Bus, Memory Controller, External Bus Data Flow

The data flows for various conditions are shown on the following pages.

4.8.1 Data flow during byte access

Table 4-13 Data Flow During Byte Access (Little Endian)

Accessed Address	Data Transfer Flow	
	External Data Bus: 16 Bits	External Data Bus: 8 Bits
4n		
4n+1		
4n+2		
4n+3		

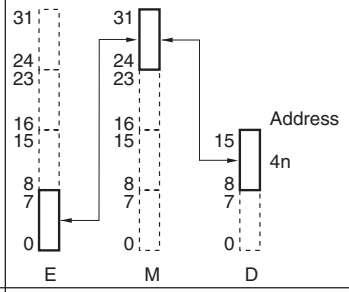
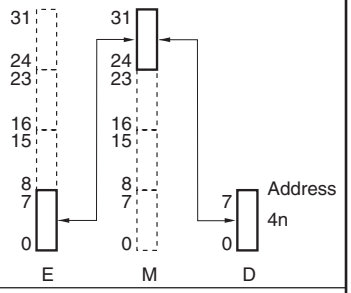
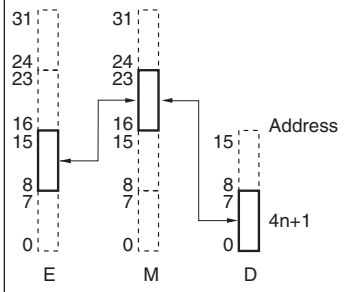
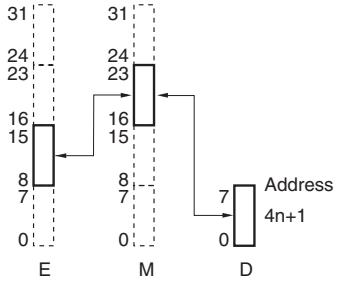
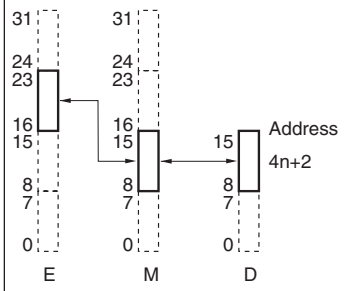
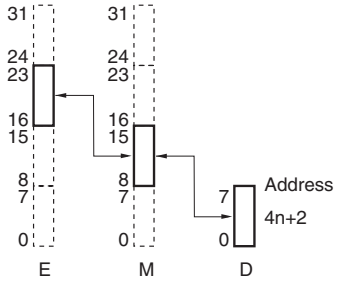
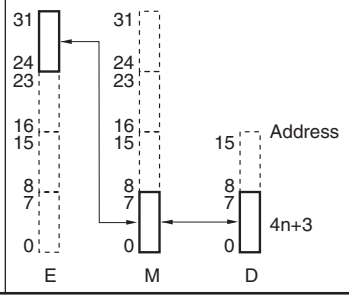
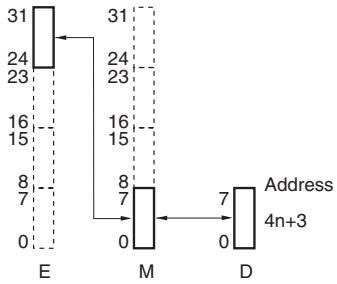
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-14 Data Flow During Byte Access (Big Endian)

Accessed Address	Data Transfer Flow	
	External Data Bus: 16 Bits	External Data Bus: 8 Bits
4n		
4n+1		
4n+2		
4n+3		

Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

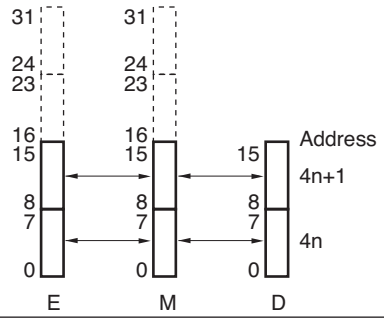
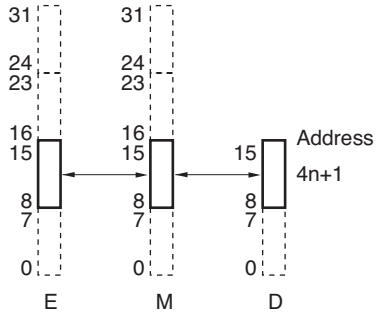
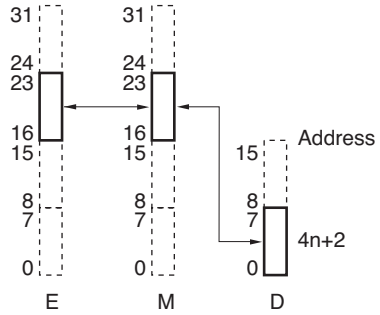
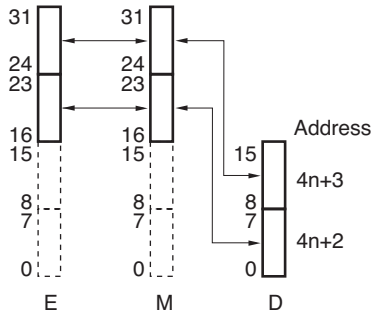
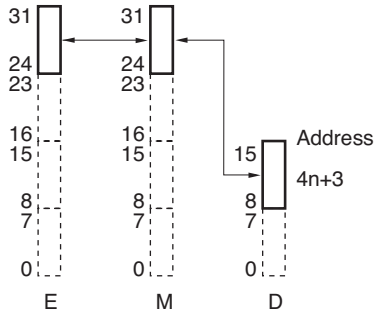
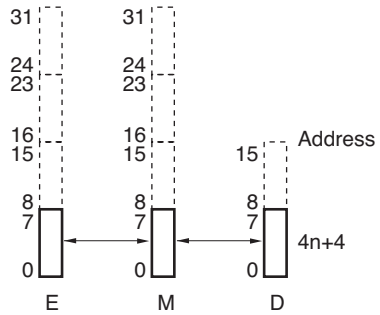
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-22 Data Flow During Halfword Access (Little Endian) (1/2)

Accessed Address	Data Transfer Flow External Data Bus: 16 Bits	
	1st	2nd
4n		-
4n+1		
4n+2		-
4n+3		

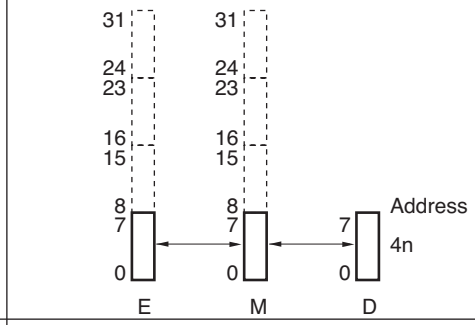
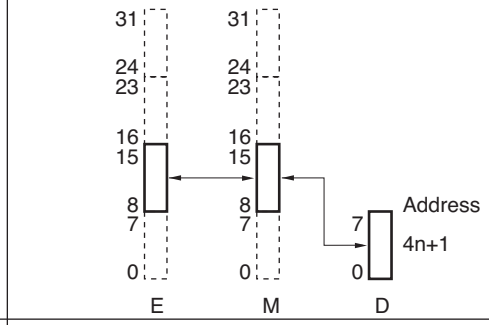
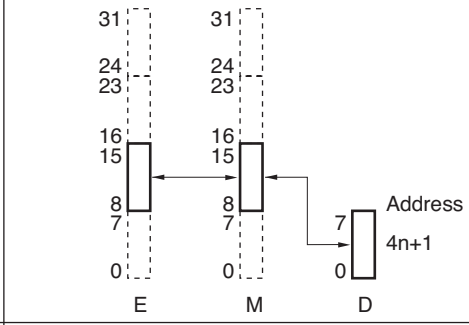
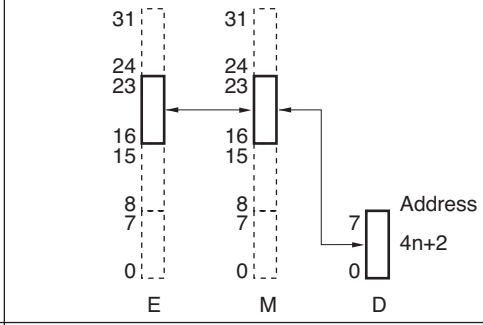
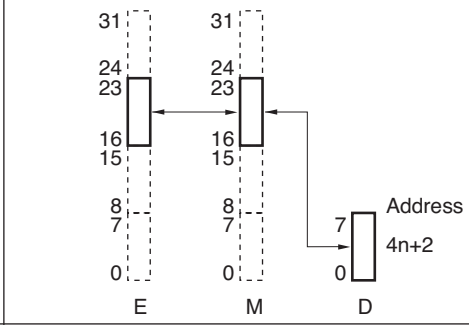
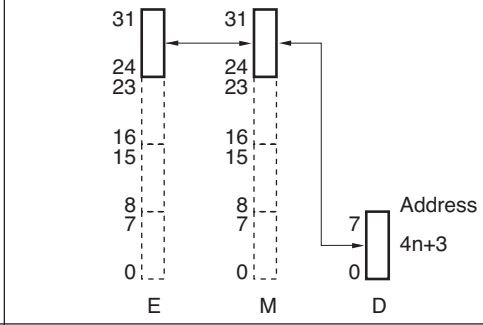
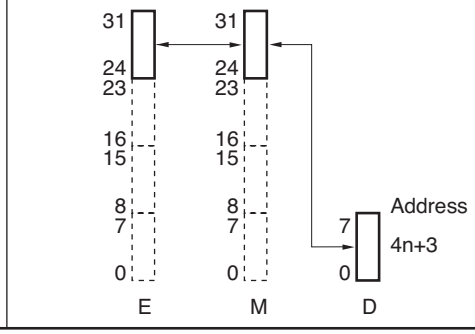
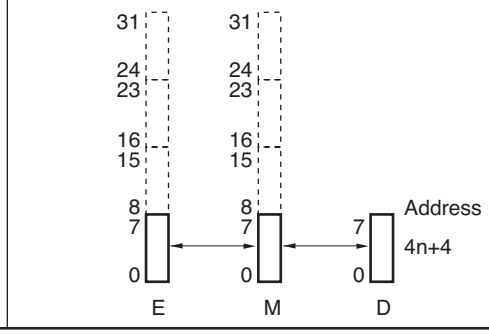
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-22 Data Flow During Halfword Access (Little Endian) (2/2)

Accessed Address	Data Transfer Flow	
	External Data Bus: 8 Bits	
	1st	2nd
4n		
4n+1		
4n+2		
4n+3		

Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

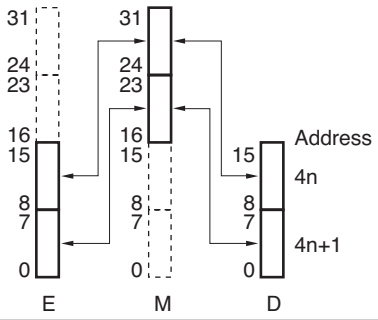
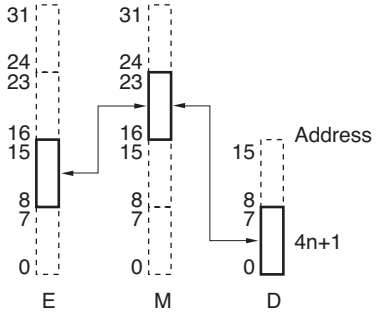
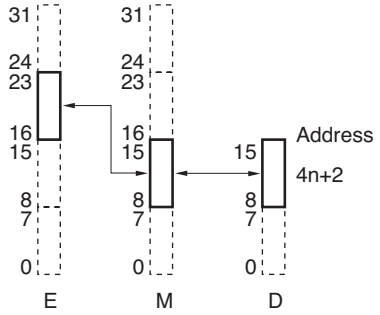
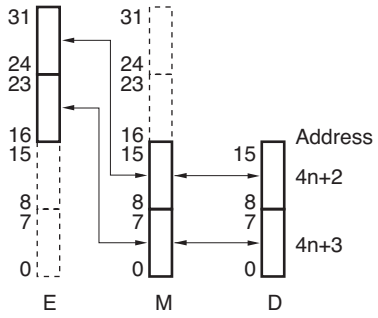
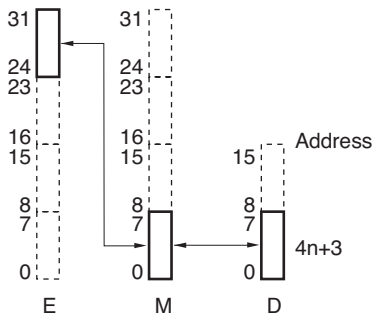
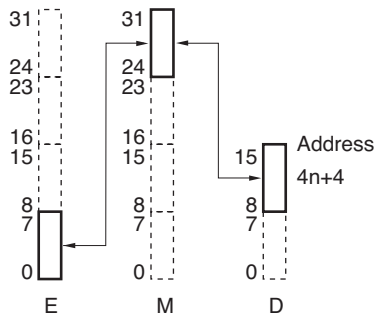
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-23 Data Flow During Halfword Access (Big Endian) (1/2)

Accessed Address	Data Transfer Flow External Data Bus: 16 Bits	
	1st	2nd
4n		-
4n+1		
4n+2		-
4n+3		

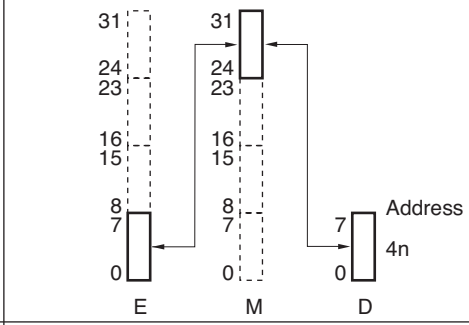
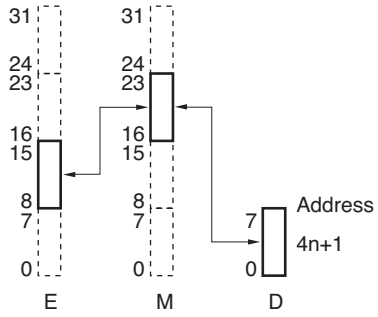
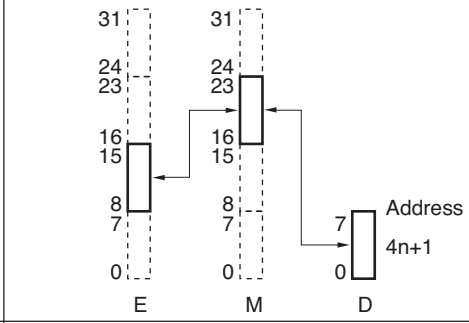
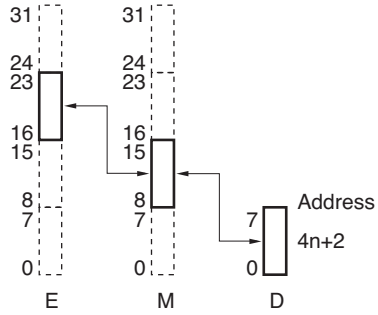
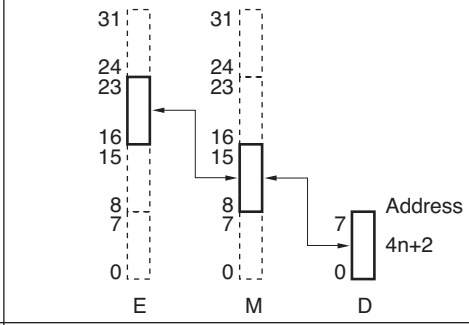
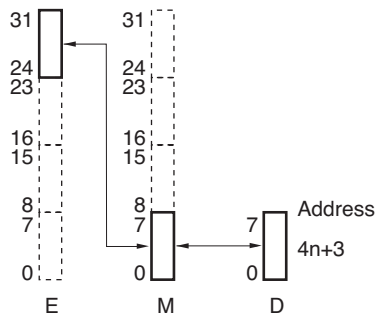
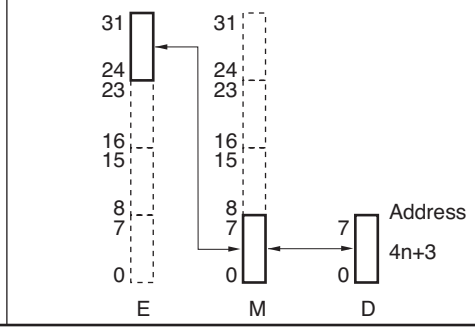
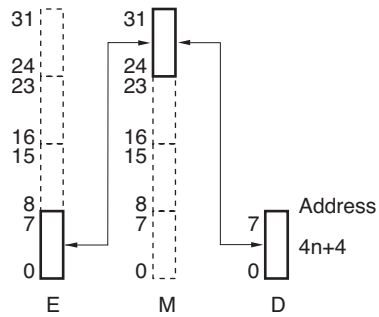
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-23 Data Flow During Halfword Access (Big Endian) (2/2)

Accessed Address	Data Transfer Flow External Data Bus: 8 Bits	
	1st	2nd
4n		
4n+1		
4n+2		
4n+3		

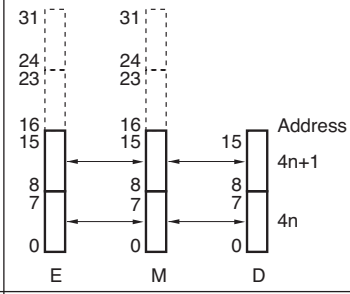
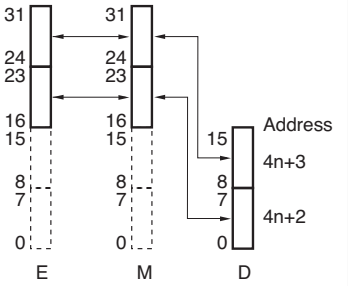
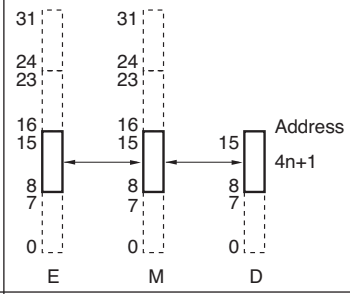
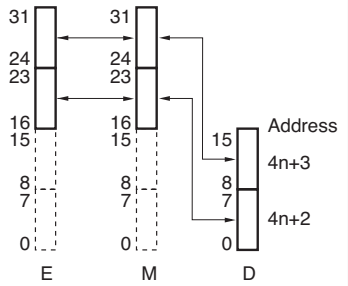
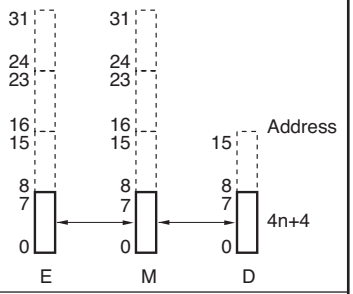
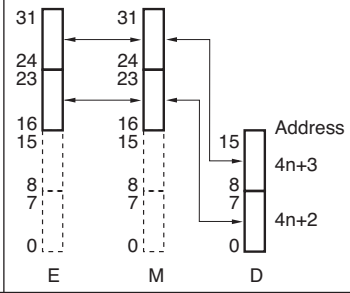
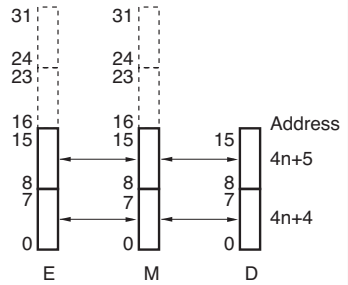
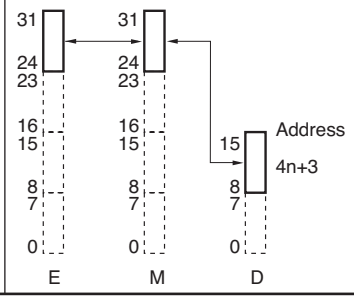
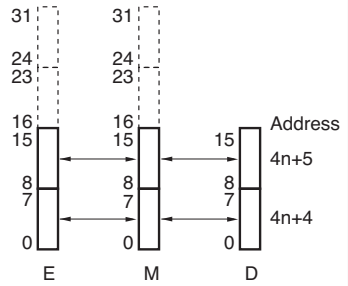
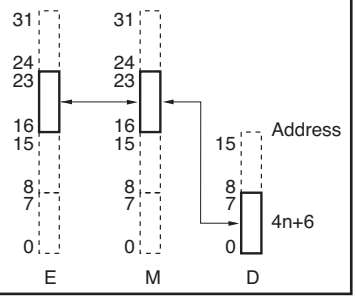
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-24 Data Flow During Word Access (Little Endian) (1/2)

Accessed Address	Data Transfer Flow External Data Bus: 16 Bits		
	1st	2nd	3rd
4n			-
4n+1			
4n+2			-
4n+3			

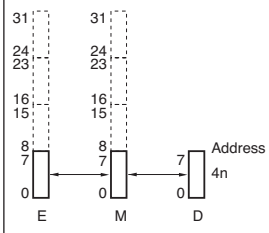
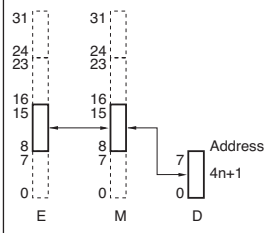
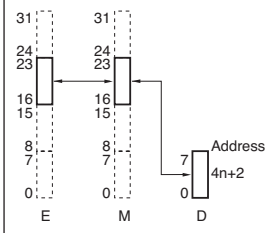
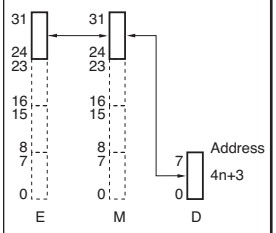
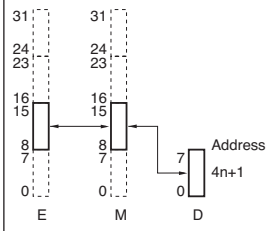
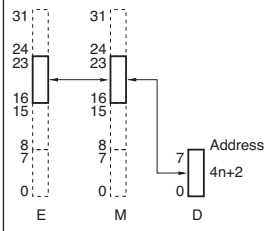
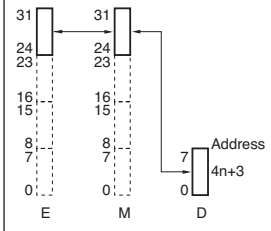
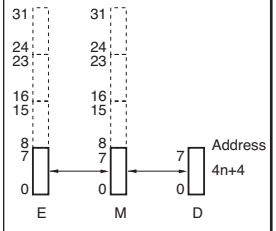
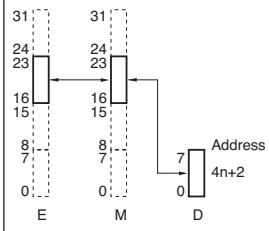
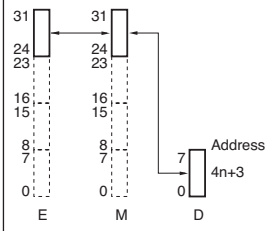
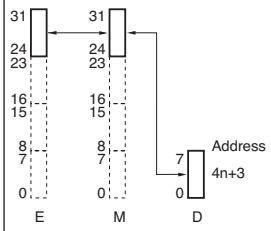
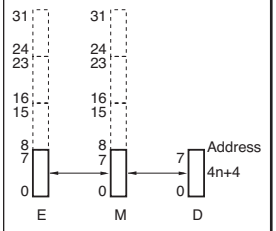
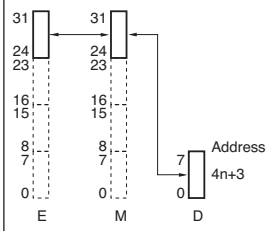
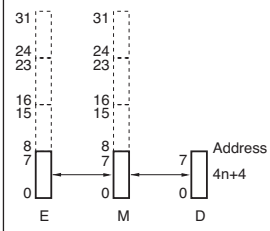
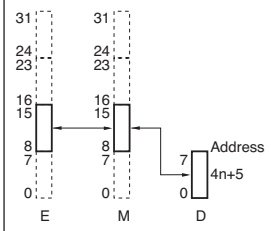
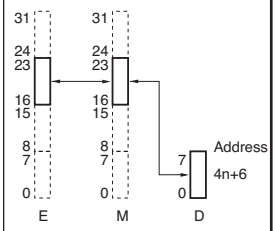
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-24 Data Flow During Word Access (Little Endian) (2/2)

Accessed Address	Data Transfer Flow			
	External Data Bus: 8 Bits			
	1st	2nd	3rd	4th
4n				
4n+1				
4n+2				
4n+3				

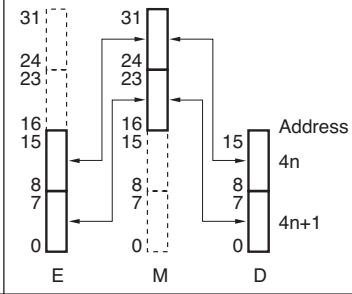
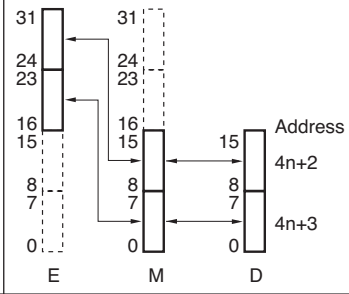
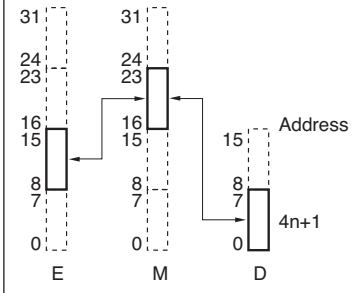
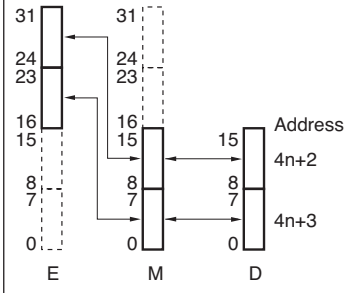
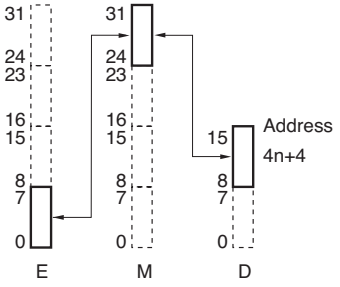
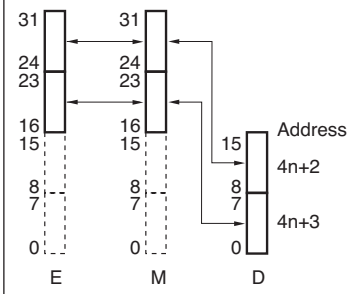
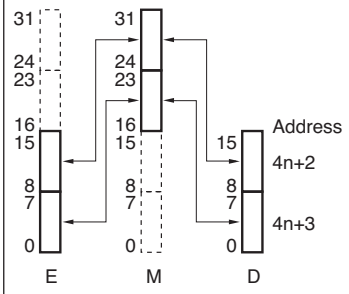
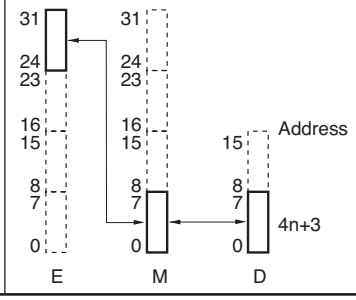
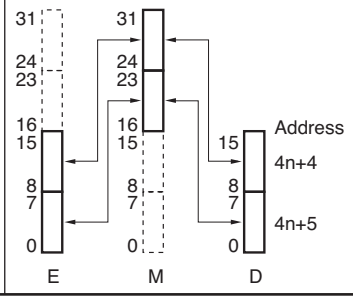
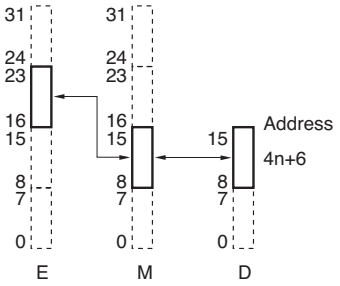
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-25 Data Flow During Word Access (Big Endian) (1/2)

Accessed Address	Data Transfer Flow External Data Bus: 16 Bits		
	1st	2nd	3rd
4n			-
4n+1			
4n+2			-
4n+3			

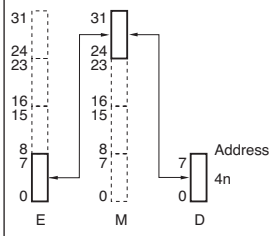
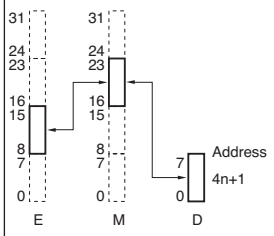
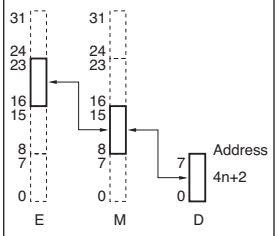
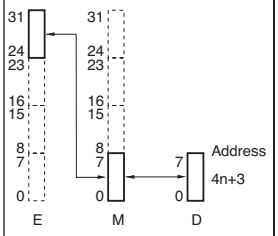
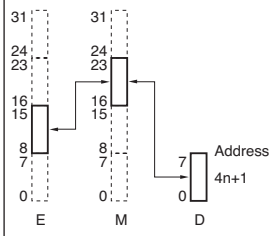
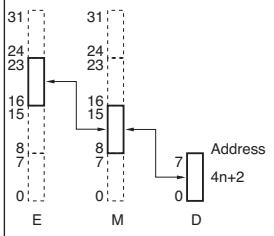
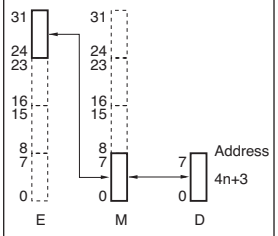
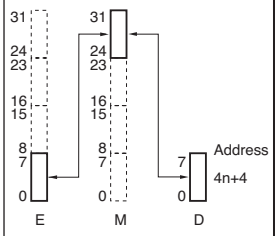
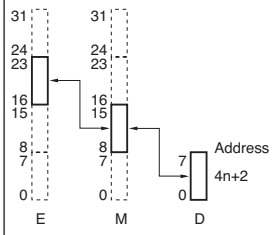
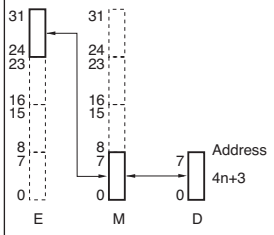
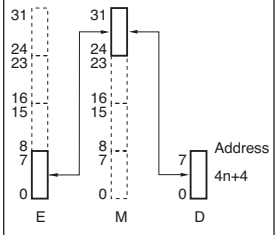
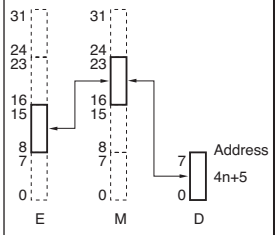
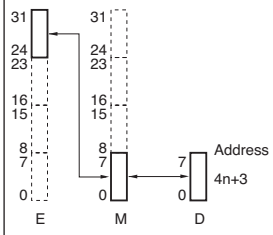
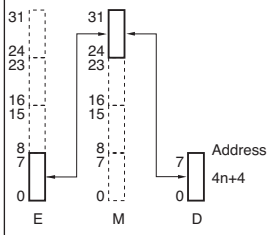
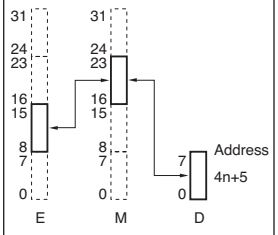
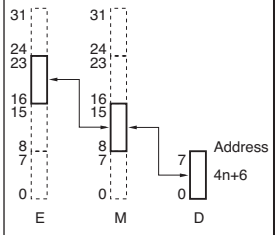
Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Table 4-25 Data Flow During Word Access (Big Endian) (2/2)

Accessed Address	Data Transfer Flow			
	External Data Bus: 8 Bits			
	1st	2nd	3rd	4th
4n				
4n+1				
4n+2				
4n+3				

Note E:Internal bus

M: MEMC data buffer

D: External data bus

n = 0, 1, 2, 3, …

Chapter 5 Interrupt Functions

This chapter describes the exception processing functions of this microcontroller.

At first an overview is given about all exception groups.

Afterwards all interrupt exceptions of this microcontroller are summarized and the interrupt control registers are described.

Finally the handling of the interrupts exceptions are detailed.

All other exceptions and their handling are described in the

V850E2M 32-bit Microcontroller Core Architecture
Document number R01US0001EJxxxx

where “xxxx” denotes the version of this document.

5.1 Exceptions and Interrupts

The phenomenon of forcing a branch operation from a currently running program to another program, due to a specific cause, is called an exception.

The exceptions are classified in exception groups. Each exception group is assigned to a certain priority P1 to P11, that determines in which order exceptions are processed, if they occur concurrently.

This microcontroller supports the following types of exceptions and interrupts:

Table 5-1 Exception and interrupt cause list (1/2)

Exception and interrupt name	Symbol	Cause group	Priority group	Exception level	Reference
CPU initialization	RESET	Reset input	P1	-	this manual chapter *“Reset Controller”*
FE level non-maskable interrupt[a]	FENMI	FENMI input	P2	FE	this chapter *“V850E2/Fx4-H Interrupt Requests”*
System error exception[a]	SYSERR	SYSERR input	P3	FE	this chapter *“V850E2/Fx4-H Exceptions”*
Peripheral device protection exception	PPI	Peripheral device protection violation	P4	FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *“Peripheral Device Protection”*
Timing supervision exception	TSI	Timing monitoring violation	P5	FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *“Timing Supervision Function”*
FE level maskable interrupt[a]	FEINT	FEINT input	P6	FE	this chapter *“V850E2/Fx4-H Interrupt Requests”*
Floating-point operation exception (imprecise)	FPI	FPU instruction	P7	EI	V850E2M 32-bit Microcontroller Core Architecture User Manual *“Floating-point Operation Function”*
EI level maskable interrupt[a]	INT	Maskable interrupt input	P8	EI	this chapter *“V850E2/Fx4-H Interrupt Requests”*

Table 5-1 Exception and interrupt cause list (2/2)

Exception and interrupt name	Symbol	Cause group	Priority group	Exception level	Reference
Execution protection exception	MIP	Execution protection violation	P9	FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Memory Protection"*
Memory error exception[a]	MEP	Instruction access error input	P10	FE	this chapter *"V850E2/Fx4-H Exceptions"*
Data protection exception	MDP	Data protection violation	P11	FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Memory Protection"*
Floating-point operation exception (precise)	FPP	FPU instruction		EI	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Floating-point Operation Function"*
Coprocessor unusable exception	UCPOP	Coprocessor instruction		FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Exceptions"*
Reserved instruction exception	RIEX	Reserved instruction		FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Exceptions"*
FE level software exception	FETRAPEX	FETRAP instruction (vector = 1_H to F_H)		FE	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Instructions"*
EI level software exception	EITRAP0	TRAP0n instruction (vector = 00_H to $0F_H$)		EI	
EI level software exception	EITRAP1	TRAP1n instruction (vector = 10_H to $1F_H$)		EI	
System call exception	SYSCALLEX	SYSCALL instruction (vector = 00_H to FF_H)		EI	V850E2M 32-bit Microcontroller Core Architecture User Manual *"Exceptions"*

a) The description of these exceptions and interrupts are subject to this chapter.

Priority order Priority group P1 has the highest, P11 the lowest priority.

Request Exception or interrupt request denotes the status, where an exception or interrupt event occurred and is registered in the CPU or Interrupt Controller respectively to be served.

Acknowledgement Exception or interrupt acknowledgement denotes the status, where the CPU branches to an exception or interrupt service or handler routine. Thus the current program flow is suspended.

Acknowledgement condition Before an exception or interrupt request is acknowledgement, certain acknowledgement conditions may have to be fulfilled. Note that certain exceptions are acknowledged unconditionally.

Resume Indicates whether execution restart from the last position at which program execution was interrupted - i.e. an exception or interrupt was acknowledged - is possible.

Restore Indicates whether restoring of the processor status (status of processor resources including general-purpose registers and system registers) at the time of program execution interruption is possible.

5.1.1 Exception handler switching

The initial values of registers for the exception handler switching functions are listed below:

- EH_CFG = $0000\ 0000_H$
- EH_BASE = $0000\ 0000_H$
- EH_RESET = $0000\ 0000_H$

For details refer to the section *"CPU Function Group/Exception Handler Address Switching Function Banks"* in the *"V850E2M 32-bit Microcontroller Core Architecture"* document.

5.2 V850E2/Fx4-H Exceptions

This section describes the V850E2/Fx4-H exceptions.

For detailed information about how to handle exceptions refer to the chapter *"Exceptions"* in the *"V850E2M 32-bit Microcontroller Core Architecture User's Manual"*.

5.2.1 Memory error exceptions MEP

A memory error exception MEP indicates an error, that occurred during a CPU instruction fetch:

- MEP exception level: FE level without acknowledgement conditions
- MEP exception code: FEIC = 0000 0330_H
- MEP exception handler offset: 0030_H
- MEP priority level: P10

MEP exceptions are not maskable.

The MEP exception is generated upon reading instruction code from memory. Note that this must not necessarily be the time the CPU is fetching the instruction for execution. Also a cache or buffer preload can cause an MEP exception.

MEP sources The source of an MEP exception can be one of the following:

- error detection by the HBUS bridge during instruction fetch
- uncorrectable ECC double-error detection during instruction fetch from the code flash
- uncorrectable ECC double-error detection during instruction fetch from the data RAM
- instruction fetch attempt from an undefined memory area (reserved area in the CPU address map)

MEP indicators The source of an MEP exception can be evaluated by the following:

- Error input from HBUS bridge: check the contents of the ETAERR register. Refer to the section *"CPU Subsystem - HBUS Bridge"* for further details.
- Code flash ECC double-error: CECCER.DEDFLG = 1
  Refer to the section *"Code flash error correction"* below for further details
- For data RAM ECC double-error detections and undefined memory area access there is no source indication other than the address of the program counter in the CPU's FEPC register, stored when the MEP occurred.

Note Single bit errors are corrected by the ECC and processing is continued using the fetched data after correction. No MEP is generated in this case.

MEP resume/restore

Since an MEP is acknowledged unconditionally at the time of reading an instruction code from a memory, which may be different to the time of CPU fetch of this instruction code, neither resuming to the application program nor recovering of the CPU's status after the MEP service routine is possible. Thus a system reset must be applied or a substitute program to recover the status must be executed.

When an additional MEP exception occurs during execution of an MEP exception handler program, the MEP exception is acknowledged again and it's handler routine re-executed.

Moreover, even during MEP exception processing, SYSERR exceptions and FE level non-maskable interrupts may occur. If this happens, SYSERR exception or FE level non-maskable interrupt processing starts at the time the error occurs.

5.2.2 System error exceptions SYSERR

In case of a serious system error, a system error exception SYSERR can be generated.

SYSERR exception can be enabled or disabled. A flag register is provided to determine the cause of such errors.

- SYSERR exception level: FE level without acknowledgement conditions
- SYSERR exception code: refer to the table *"System errors summary"* below
- SYSERR exception handler offset: 0030_H
- SYSERR priority level: P4

SYSERR exceptions are maskable.

(1) System error causes

Error conditions

The generation of a SYSERR exception can occur under following error conditions:

- error because of an uncorrectable ECC double-error detection during data read from the code flash
- error because of an uncorrectable ECC double-error detection during data read from the data RAM
- error detection by the HBUS bridge

These error conditions may be fulfilled during a CPU or DMA access.

SYSERR enable/disable

For CPU accesses any of the above error conditions can be separately enabled/disabled to generate a SYSERR exception by use of the system error control register SEG_CONT.

For DMA accesses a single control bit in SEG_CONT.DMAE is provided to enable SYSERR generation for any of the above error conditions all together. Refer to the section *"Error response support"* in the chapter *"DMA Controller (DMAC)"* for further information concerning DMAC errors.

SYSERR flags The source of a SYSERR exception can be evaluated by the system error flag register SEG_FLAG, that holds

- dedicated flags for each of the error conditions, if the error condition was met during a CPU access.
- a single bit that indicates that any of the error condition was met during a DMA access.

Notes

1. If an error occurs during a CPU access, the exact error cause can be evaluated by the dedicated SEG_FLAG flags.
2. Upon occurrence of an error caused by DMA access, SEG_FLAG.DMAF is set regardless of the exact cause of the error. Therefore, the CPU cannot determine the exact error cause from SEG_FLAG.
3. Setting of the flags in SEG_FLAG is independent of the SYSERR enable/ disable setting in the SEG_CONT register.

Table 5-2 System errors summary

Access by	System error cause	Error flag SEG_FLAG.	SYSERR exception if SEG_CONT.	SYSERR exception code in FEIC
CPU	data read from code flash	FCHF = 1	FCHE = 1	$0000\ 0231_H$
	data read from data RAM	RAMF = 1	RAME = 1	
	data access to HBUS area	EXTF = 1	EXTE = 1	
DMA	• data read from code flash • data read from data RAM • data access to HBUS area	DMAF = 1	DMAE = 1	$0000\ 0232_H$

(2) System error condition details

1. Error during data read from code flash
   This error occurs when two defective bits are detected by the code flash ECC, which can not be corrected, as the result of read access to the code flash area.

2. Error during data read from data RAM
   This error occurs when two defective bits are detected by the data RAM ECC, which can not be corrected, as the result of read access to the data RAM area.

3. Error during access to the HBUS area
   This error can occur during a read/write access to resources connected to the HBUS. Refer to the section *"CPU Subsystem - HBUS Bridge"* in the chapter *"CPU System Functions"* for further details.

4. Occurrence of error during DMA access
   Upon detection of one of the above errors during DMA read/write accesses, the DMA controller stops the transfer operations of all channels. Refer to the section *"Error response support"* in the chapter *"DMA Controller (DMAC)"* for further information concerning DMAC errors.

(3) SEG_CONT - System error control register

This register enables respectively disables system error causes to generate a SYSERR exception.

Access This register can be read/written

- 16-bit units via the register SEG_CONT
- the lower 8 bits in 8-bit units via the register SEG_CONTL
- the higher 8 bits in 8-bit units via the register SEG_CONTH

Address SEG_CONT: FFFF $64B0_H$
SEG_CONTL: FFFF $64B0_H$
SEG_CONTH: FFFF $64B1_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DMAE	0	0	RAME	0	EXTE	FCHE	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-3 SEG_CONT register contents

Bit position	Bit name	Function
7	DMAE	Enable SYSERR exception generation during DMA accesses 0: SYSERR generation disabled 1: SYSERR generation enabled
4	RAME	Enable SYSERR exception generation during CPU data read accesses to the data RAM 0: SYSERR generation disabled 1: SYSERR generation enabled
2	EXTE	Enable SYSERR exception generation during CPU HBUS accesses 0: SYSERR generation disabled 1: SYSERR generation enabled
1	FCHE	Enable SYSERR exception generation during CPU data read accesses to the code flash 0: SYSERR generation disabled 1: SYSERR generation enabled

(4) SEG_FLAG - System error flag register

This register informs about the occurrence of a SYSERR condition.

Note that the concerned flag is set upon a SYSERR condition, independent of SYSERR generation enabled/disabled via the SEG_CONT register.

Access This register can be read/written

- 16-bit units via the register SEG_FLAG
- the lower 8 bits in 8-bit units via the register SEG_FLAGL
- the higher 8 bits in 8-bit units via the register SEG_FLAGH

Address SEG_FLAG: FFFF 64B2$_H$
SEG_FLAGL: FFFF 64B2$_H$
SEG_FLAGH: FFFF 64B3$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DMAF	0	0	RAMF	0	EXTF	FCHF	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-4 SEG_FLAG register contents

Bit position	Bit name	Function
7	DMAF	SYSERR condition during DMA accesses 0: SYSERR condition has not occurred 1: SYSERR condition has occurred
4	RAMF	SYSERR condition during CPU data read accesses to the data RAM 0: SYSERR condition has not occurred 1: SYSERR condition has occurred
2	EXTF	SYSERR condition during CPU HBUS accesses 0: SYSERR condition has not occurred 1: SYSERR condition has occurred
1	FCHF	SYSERR condition during CPU data read accesses to the code flash 0: SYSERR condition has not occurred 1: SYSERR condition has occurred

5.2.3 Code flash error correction

The code flash memory is equipped with an Error Correction module ECC. When the CPU or the DMA accesses the code flash, the ECC module automatically encodes/decodes the ECC code.

Single-bit errors

In case of a single-bit error while reading from the code flash, the ECC module corrects this error automatically, i.e. the code flash ECC performs a single-error correction (SEC).

Detection and correction of a single-bit error is indicated by the SEC flag CECCER.SECFLG = 1.
The code flash address, where this single-bit error has occurred, is stored in the CSECADR register.

Double-bit errors

In case of double-bit errors while reading from the code flash, the ECC module detects this error, i.e. the code flash ECC performs a double-error detection (DED).

Detection of a double-bit error is indicated by the DED flag CECCER.DEDFLG = 1.
The code flash address, where this double-bit error has occurred, is stored in the CDEDADR register.

Upon detection of a double-bit error an exception can be generated.
Refer to the sections *"Memory error exceptions MEP"* and *"System error exceptions SYSERR"* about further information concerning exception processing of double-error detections.

Note Three or more erroneous bits may not detect.

(1) Code flash error correction registers overview

This section contains a description of all registers of the code flash error correction module.

Table 5-5 Code flash Error Correction module registers overview

Register name	Shortcut	Address
Code flash ECC error flag register	CECCER	FF43 2000$_H$
Code flash ECC error flag clear register	CECCERC	FF43 2004$_H$
Code flash ECC single-bit error correction address register	CSECADR	FF43 2008$_H$
Code flash ECC double-bit error detection address register	CDEDADR	FF43 200C$_H$

(2) CECCER - Code flash ECC error flag register

This register informs about a correction of a single-bit (SEC) error and detection of a double-bit (DED) error.

To clear an asserted bit in this register use the CECCERC register.

Access These registers can be read in 8-bit units.

Address FF43 2000$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	DEDFLG	SECFLG
R	R	R	R	R	R	R	R

Table 5-6 CECCER register contents

Bit position	Bit name	Function
1	DEDFLG	double-bit error detection flag 0: double-bit error was not detected 1: double-bit error was not detected
0	SECFLG	single-bit error correction flag 0: single-bit error was not detected 1: single-bit error was detected

(3) CECCERC - Code flash ECC error flag clear register

This register allows to clear the ECC error flags of CECCER.

If a 1 is written to a bit in this register, the corresponding bit in the CECCER register is cleared.

Access These registers can be read/written in 8-bit units.

Address FF43 2004$_H$

Initial Value Reading this registers returns always 00$_H$.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	DEDCLR	SEDCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-7 CECCERC register contents

Bit position	Bit name	Function
1	DEDCLR	Clear double-bit error detection flag 0: no function 1: clear CECCER.DEDFLG
0	SECCLR	Clear single-bit error detection flag 0: no function 1: clear CECCER.SECFLG

(4) CSECADR - Code flash ECC single-bit error correction address register

This register holds the address at which a single-bit error was detected and corrected.

Address storing in and clearing of this register depends on the single-bit error correction flag CECCER.SECFLG:

- The error address is only stored, if CECCER.SECFLG = 0.
- This register is cleared together with the flag CECCER.SECFLG by setting CECCERC.SECCLR = 1.

Thus only the address of the first detected error is stored. The addresses of all others which may follow, do not overwrite the first address, until the single-bit error flag - and thus this register - is cleared by CECCERC.SECLR = 1.

Access These registers can be read in 32-bit units.

Address FF43 2008$_H$

Initial Value 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	CSECADR[24:16]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSECADR[7:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 5-8 CSECADR register contents

Bit position	Bit name	Function
24 to 0	CSECADR [24:0]	single-bit error detection and correction address

(5) CDEDADR - Code flash ECC double-bit error detection address register

This register holds the address, at which a double-bit error was detected.

Address storing in and clearing of this register depends on the double-bit error detection flag CECCER.DEDFLG:

- The error address is only stored, if CECCER.DEDFLG = 0.
- This register is cleared together with the flag CECCER.DEDFLG by setting CECCERC.DEDCLR = 1.

Thus only the address of the first detected error is stored. The addresses of all others, which may follow, do not overwrite the first address, until the double-bit error flag - and thus this register - is cleared by CECCERC.DEDCLR = 1.

Access These registers can be read in 32-bit units.

Address FF43 200C$_H$

Initial Value 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	CDECADR[24:16]								
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDECADR[7:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 5-9 CDEDADR register contents

Bit position	Bit name	Function
24 to 0	CDECADR [24:0]	double-bit error detection address

5.3 V850E2/Fx4-H Interrupt Requests

Interrupt types The V850E2/Fx4-H nsupports following types of interrupts:

- FE level non-maskable interrupt (FENMI)
  - FENMI is served immediately, even if an FE level maskable (FEINT) or EI level maskable (EIINT) interrupt is in service (CPU system register PSW.NP = 1).
  - resume not possible, recover not possible
- FE level maskable interrupt (FEINT)
  - FEINT is served immediately, if no other FE level interrupt - FENMI or FEINT - is in service (CPU system register PSW.NP = 0). Thus PSW.NP = 1 would mask an FEINT.
  - resume possible, recover possible
  - highest priority maskable interrupt
- EI level maskable interrupt (EIINT)
  - EIINT may only be served, if no FE level interrupt - FENMI or FEINT - is in service (CPU system register PSW.NP = 0).
  - resume possible, recover possible
  - interrupt masking can be specified for each interrupt channel
  - 16 interrupt priority levels can be specified for each interrupt channel.

5.3.1 V850E2/Fx4-H interrupt sources

(1) FE level non-maskable interrupts

Priority group The FE level non-maskable interrupts have the priority P2.

Return PC An FE non-maskable interrupt does not allow to resume or recover.

Control register The control register of this interrupt is FNC.

Return instruction Since neither resume nor recover is possible, a reset must be applied.

Table 5-10 FE level non-maskable interrupt requests

Interrupt			Interrupt request		Unit	Priority group	Exception code	Handler address offset
Symbol	Control register		Name	Cause				
	Name	Address FFFF...						
FENMI	FNC	$645C_H$	NMI0	NMI0 input	Port	P2	0020_H	0020_H
			WDTA0NMI	Watchdog Timer 0 error NMI interrupt	WDTA0			
			WDTA1NMI	Watchdog Timer 1 error NMI interrupt	WDTA1			

FENMI sharing The source of the FENMI interrupt can be evaluated by a dedicated flag register. Refer to section *5.3.2 "V850E2/Fx4-H FE level non-maskable interrupt sharing" on page 290* for details.

(2) FE level maskable interrupts

Priority group The FE level maskable interrupts have the priority P6.

Return PC The program couter (PC) value, set after returning from any interrupt service routine by the RETI instruction is always the next address.

Control register The control register of this interrupt is FIC.

Return instruction The return instruction from FE level maskable interrupt service routines is FERET.

Table 5-11 FE level maskable interrupt requests

Interrupt			Interrupt request		Unit	Priority group	Exception code	Handler address offset
Symbol	Control register		Name	Cause				
	Name	Address FFFF...						
FEINT	FIC	$645E_H$	-	-	-	P6	0010_H	0010_H

(3) EI level maskable interrupts

Interrupt naming The composition of the interrupt request signal names, their assigned interrupt control registers and the bits in these registers follow special rules.

In the following the name of the interrupt request is represented by *<name>*.

- Interrupt request name: **INT***<name>*
  The prefix "**INT**" is put in front of *<name>*.
- Interrupt request control register: **IC***<name>*
  The prefix "**IC**" is put in front of *<name>*.
  The 16 bit of the 16-bit register **IC***<name>* can also be accessed byte-wise with the following names:
  - low byte (bits[7:0]): **IC***<name>***L** at the address of **IC***<name>*
    The suffix "**L**" is appended to the register name **IC***<name>*.
  - high byte (bits[15:8]): **IC***<name>***H** at the address of **IC***<name>* + 1
    The suffix "**H**" is appended to the register name **IC***<name>*.
- Interrupt control register bit names: **RF***<name>*, **MK***<name>*, **P[3:0]***<name>*
  The bit prefix "**RF**", "**MK**", "**P[3:0]**" prepends the interrupt *<name>*.
- Each interrupt request is assigned to a certain interrupt channel number n = 0 to 255.
  The functional description of the Interrupt Controller in this chapter refers to interrupt channel number *n* instead of to the interrupt name *<name>*.
  If **INT***<name>* is assigned to the interrupt channel number *n*, throughout this chapter
  - the interrupt request is named **INT***n*
  - the assigned interrupt control register is named **IC***n*
  - the interrupt control bits are named **RF***n*, **MK***n*, **P[3:0]***n*.

Example The interrupt request of the second TAUA0 channel (*<name>* = *TAUA0I2*) is named

INT*TAUA0I2*

The related interrupt control registers are

IC*TAUA0I2*, **IC***TAUA0I2L*, **IC***TAUA0I2H*

The bits in this register are

RF*TAUA0I2*, **MK***TAUA0I2*, **P[3:0]***TAUA0I2*

If, for instance, the interrupt channel for **INT***TAUA0I2* is n = *22*, the functional description of the Interrupt Controller refers to

INT*22*,

the related control register as

IC*22,*

and their control register bit names

RF*22*, **MK***22*, **P[3:0]***22*.

The following tables list the references between the interrupt channel number *n*, the assigned V850E2/Fx4-H interrupt requests and control register names.

Priority group The EI level maskable interrupts have the priority P8.

Return PC The program couter (PC) value, set after returning from any interrupt service routine by the EIRET instruction is always the next address.

Return instruction The return instruction from EI level maskable interrupt service routines is EIRET.

(4) V850E2K/FK4-H EI level maskable interrupts

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (1/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
0	ICWDTA0	6000_H	INTWDTA0	WDTA0 75% interrupt	WDTA0	1	0080_H	0080_H
1	ICWDTA1	6002_H	INTWDTA1	WDTA1 75% interrupt	WDTA1	2	0090_H	0090_H
2	ICLVI	6004_H	INTLVI	LVI interrupt	LVI	3	$00A0_H$	$00A0_H$
3	–	6006_H	reserved			4	$00B0_H$	$00B0_H$
4	ICVCPC0	6008_H	INTVCPC0	Voltage Comparator 0 interrupt	VCPC0	5	$00C0_H$	$00C0_H$
5	ICVCPC1	$600A_H$	INTVCPC1	Voltage Comparator 1 interrupt	VCPC1	6	$00D0_H$	$00D0_H$
6	ICRTCA01S	$600C_H$	INTRTCA01S	Real-Time Clock 1 second interrupt	RTCA0	7	$00E0_H$	$00E0_H$
7	ICRTCA0AL	$600E_H$	INTRTCA0AL	Alarm interrupt	RTCA0	8	$00F0_H$	$00F0_H$
8	ICRTCA0R	6010_H	INTRTCA0R	Fixed frequency interruption	RTCA0	9	0100_H	0100_H
9	ICP0	6012_H	INTP0	Edge detection interrupt	Port	10	0110_H	0110_H
			INTDTS0[a]	DTS channel 0 interrupt	DTS			
10	ICP1	6014_H	INTP1	Edge detection interrupt	Port	11	0120_H	0120_H
			INTDTS1[a]	DTS channel 1 interrupt	DTS			
11	ICP2	6016_H	INTP2	Edge detection interrupt	Port	12	0130_H	0130_H
			INTDTS2[a]	DTS channel 2 interrupt	DTS			
12	ICP3	6018_H	INTP3	Edge detection interrupt	Port	13	0140_H	0140_H
13	ICP4	$601A_H$	INTP4	Edge detection interrupt	Port	14	0150_H	0150_H
14	ICP5	$601C_H$	INTP5	Edge detection interrupt	Port	15	0160_H	0160_H
15	ICP6	$601E_H$	INTP6	Edge detection interrupt	Port	16	0170_H	0170_H
16	ICP7	6020_H	INTP7	Edge detection interrupt	Port	17	0180_H	0180_H
17	ICP8	6022_H	INTP8	Edge detection interrupt	Port	18	0190_H	0190_H
18	ICP9	6024_H	INTP9	Edge detection interrupt	Port	19	$01A0_H$	$01A0_H$
19	ICP10	6026_H	INTP10	Edge detection interrupt	Port	20	$01B0_H$	$01B0_H$
20	ICTAUA0I0	6028_H	INTTAUA0I0	Interrupt for channel 0	TAUA0	21	$01C0_H$	$01C0_H$
			INTTAPA0PEK0[b]	Peak interrupt 0	TAPA0			
21	ICTAUA0I1	$602A_H$	INTTAUA0I1	Interrupt for channel 1	TAUA0	22	$01D0_H$	$01D0_H$
			INTTAPAPEK1[b]	Peak interrupt 1	TAPA0			
22	ICTAUA0I2	$602C_H$	INTTAUA0I2	Interrupt for channel 2	TAUA0	23	$01E0_H$	$01E0_H$
23	ICTAUA0I3	$602E_H$	INTTAUA0I3	Interrupt for channel 3	TAUA0	24	$01F0_H$	$01F0_H$
24	ICTAUA0I4	6030_H	INTTAUA0I4	Interrupt for channel 4	TAUA0	25	0200_H	0200_H
			INTTAPA0VLY0[b]	Valley interrupt 0	TAPA0			
25	ICTAUA0I5	6032_H	INTTAUA0I5	Interrupt for channel 5	TAUA0	26	0210_H	0210_H
			INTTAPA0VLY1[b]	Valley interrupt 0	TAPA0			
26	ICTAUA0I6	6034_H	INTTAUA0I6	Interrupt for channel 6	TAUA0	27	0220_H	0220_H
27	ICTAUA0I7	6036_H	INTTAUA0I7	Interrupt for channel 7	TAUA0	28	0230_H	0230_H
28	ICTAUA0I8	6038_H	INTTAUA0I8	Interrupt for channel 8	TAUA0	29	0240_H	0240_H
29	ICTAUA0I9	$603A_H$	INTTAUA0I9	Interrupt for channel 9	TAUA0	30	0250_H	0250_H

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (2/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
30	ICTAUA0I10	$603C_H$	INTTAUA0I10	Interrupt for channel 10	TAUA0	31	0260_H	0260_H
31	ICTAUA0I11	$603E_H$	INTTAUA0I11	Interrupt for channel 11	TAUA0	32	0270_H	0270_H
32	ICTAUA0I12	6040_H	INTTAUA0I12	Interrupt for channel 12	TAUA0	33	0280_H	0280_H
33	ICTAUA0I13	6042_H	INTTAUA0I13	Interrupt for channel 13	TAUA0	34	0290_H	0290_H
34	ICTAUA0I14	6044_H	INTTAUA0I14	Interrupt for channel 14	TAUA0	35	$02A0_H$	$02A0_H$
35	ICTAUA0I15	6046_H	INTTAUA0I15	Interrupt for channel 15	TAUA0	36	$02B0_H$	$02B0_H$
36	ICTAUB1I0	6048_H	INTTAUB1I0	Interrupt for channel 0	TAUB1	37	$02C0_H$	$02C0_H$
			INTDTS3[a]	DTS channel 3 interrupt	DTS			
37	ICTAUB1I1	$604A_H$	INTTAUB1I1	Interrupt for channel 1	TAUB1	38	$02D0_H$	$02D0_H$
			INTDTS4[a]	DTS channel 4 interrupt	DTS			
38	ICTAUB1I2	$604C_H$	INTTAUB1I2	Interrupt for channel 2	TAUB1	39	$02E0_H$	$02E0_H$
39	ICTAUB1I3	$604E_H$	INTTAUB1I3	Interrupt for channel 3	TAUB1	40	$02F0_H$	$02F0_H$
40	ICTAUB1I4	6050_H	INTTAUB1I4	Interrupt for channel 4	TAUB1	41	0300_H	0300_H
41	ICTAUB1I5	6052_H	INTTAUB1I5	Interrupt for channel 5	TAUB1	42	0310_H	0310_H
42	ICTAUB1I6	6054_H	INTTAUB1I6	Interrupt for channel 6	TAUB1	43	0320_H	0320_H
43	ICTAUB1I7	6056_H	INTTAUB1I7	Interrupt for channel 7	TAUB1	44	0330_H	0330_H
44	ICTAUB1I8	6058_H	INTTAUB1I8	Interrupt for channel 8	TAUB1	45	0340_H	0340_H
45	ICTAUB1I9	$605A_H$	INTTAUB1I9	Interrupt for channel 9	TAUB1	46	0350_H	0350_H
46	ICTAUB1I10	$605C_H$	INTTAUB1I10	Interrupt for channel 10	TAUB1	47	0360_H	0360_H
47	ICTAUB1I11	$605E_H$	INTTAUB1I11	Interrupt for channel 11	TAUB1	48	0370_H	0370_H
48	ICTAUB1I12	6060_H	INTTAUB1I12	Interrupt for channel 12	TAUB1	49	0380_H	0380_H
49	ICTAUB1I13	6062_H	INTTAUB1I13	Interrupt for channel 13	TAUB1	50	0390_H	0390_H
50	ICTAUB1I14	6064_H	INTTAUB1I14	Interrupt for channel 14	TAUB1	51	$03A0_H$	$03A0_H$
51	ICTAUB1I15	6066_H	INTTAUB1I15	Interrupt for channel 15	TAUB1	52	$03B0_H$	$03B0_H$
52	ICTAUB2I0	6068_H	INTTAUB2I0	Interrupt for channel 0	TAUB2	53	$03C0_H$	$03C0_H$
			INTDTS5[a]	DTS channel 5 interrupt	DTS			
53	ICTAUB2I1	$606A_H$	INTTAUB2I1	Interrupt for channel 1	TAUB2	54	$03D0_H$	$03D0_H$
			INTDTS6[a]	DTS channel 6 interrupt	DTS			
54	ICTAUB2I2	$606C_H$	INTTAUB2I2	Interrupt for channel 2	TAUB2	55	$03E0_H$	$03E0_H$
55	ICTAUB2I3	$606E_H$	INTTAUB2I3	Interrupt for channel 3	TAUB2	56	$03F0_H$	$03F0_H$
56	ICTAUB2I4	6070_H	INTTAUB2I4	Interrupt for channel 4	TAUB2	57	0400_H	0400_H
57	ICTAUB2I5	6072_H	INTTAUB2I5	Interrupt for channel 5	TAUB2	58	0410_H	0410_H
58	ICTAUB2I6	6074_H	INTTAUB2I6	Interrupt for channel 6	TAUB2	59	0420_H	0420_H
59	ICTAUB2I7	6076_H	INTTAUB2I7	Interrupt for channel 7	TAUB2	60	0430_H	0430_H
60	ICTAUB2I8	6078_H	INTTAUB2I8	Interrupt for channel 8	TAUB2	61	0440_H	0440_H
61	ICTAUB2I9	$607A_H$	INTTAUB2I9	Interrupt for channel 9	TAUB2	62	0450_H	0450_H
62	ICTAUB2I10	$607C_H$	INTTAUB2I10	Interrupt for channel 10	TAUB2	63	0460_H	0460_H
63	ICTAUB2I11	$607E_H$	INTTAUB2I11	Interrupt for channel 11	TAUB2	64	0470_H	0470_H

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (3/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
64	ICTAUB2I12	6080_H	INTTAUB2I12	Interrupt for channel 12	TAUB2	65	0480_H	0480_H
65	ICTAUB2I13	6082_H	INTTAUB2I13	Interrupt for channel 13	TAUB2	66	0490_H	0490_H
66	ICTAUB2I14	6084_H	INTTAUB2I14	Interrupt for channel 14	TAUB2	67	$04A0_H$	$04A0_H$
67	ICTAUB2I15	6086_H	INTTAUB2I15	Interrupt for channel 15	TAUB2	68	$04B0_H$	$04B0_H$
68	ICTAUC3I0	6088_H	INTTAUC3I0	Interrupt for channel 0	TAUC3	69	$04C0_H$	$04C0_H$
69	ICTAUC3I1	$608A_H$	INTTAUC3I1	Interrupt for channel 1	TAUC3	70	$04D0_H$	$04D0_H$
70	ICTAUC3I2	$608C_H$	INTTAUC3I2	Interrupt for channel 2	TAUC3	71	$04E0_H$	$04E0_H$
71	ICTAUC3I3	$600E_H$	INTTAUC3I3	Interrupt for channel 3	TAUC3	72	$04F0_H$	$04F0_H$
72	ICTAUC3I4	6090_H	INTTAUC3I4	Interrupt for channel 4	TAUC3	73	0500_H	0500_H
73	ICTAUC3I5	6092_H	INTTAUC3I5	Interrupt for channel 5	TAUC3	74	0510_H	0510_H
74	ICTAUC3I6	6094_H	INTTAUC3I6	Interrupt for channel 6	TAUC3	75	0520_H	0520_H
75	ICTAUC3I7	6096_H	INTTAUC3I7	Interrupt for channel 7	TAUC3	76	0530_H	0530_H
76	ICTAUC3I8	6098_H	INTTAUC3I8	Interrupt for channel 8	TAUC3	77	0540_H	0540_H
77	ICTAUC3I9	$609A_H$	INTTAUC3I9	Interrupt for channel 9	TAUC3	78	0550_H	0550_H
78	ICTAUC3I10	$609C_H$	INTTAUC3I10	Interrupt for channel 10	TAUC3	79	0560_H	0560_H
79	ICTAUC3I11	$609E_H$	INTTAUC3I11	Interrupt for channel 11	TAUC3	80	0570_H	0570_H
80	ICTAUC3I12	$60A0_H$	INTTAUC3I12	Interrupt for channel 12	TAUC3	81	0580_H	0580_H
81	ICTAUC3I13	$60A2_H$	INTTAUC3I13	Interrupt for channel 13	TAUC3	82	0590_H	0590_H
82	ICTAUC3I14	$60A4_H$	INTTAUC3I14	Interrupt for channel 14	TAUC3	83	$05A0_H$	$05A0_H$
83	ICTAUC3I15	$60A6_H$	INTTAUC3I15	Interrupt for channel 15	TAUC3	84	$05B0_H$	$05B0_H$
84	ICTAUC4I0	$60A8_H$	INTTAUC4I0	Interrupt for channel 0	TAUC4	85	$05C0_H$	$05C0_H$
85	ICTAUC4I1	$60AA_H$	INTTAUC4I1	Interrupt for channel 1	TAUC4	86	$05D0_H$	$05D0_H$
86	ICTAUC4I2	$60AC_H$	INTTAUC4I2	Interrupt for channel 2	TAUC4	87	$05E0_H$	$05E0_H$
87	ICTAUC4I3	$60AE_H$	INTTAUC4I3	Interrupt for channel 3	TAUC4	88	$05F0_H$	$05F0_H$
88	ICTAUC4I4	$60B0_H$	INTTAUC4I4	Interrupt for channel 4	TAUC4	89	0600_H	0600_H
89	ICTAUC4I5	$60B2_H$	INTTAUC4I5	Interrupt for channel 5	TAUC4	90	0610_H	0610_H
90	ICTAUC4I6	$60B4_H$	INTTAUC4I6	Interrupt for channel 6	TAUC4	91	0620_H	0620_H
91	ICTAUC4I7	$60B6_H$	INTTAUC4I7	Interrupt for channel 7	TAUC4	92	0630_H	0630_H
92	ICTAUC4I8	$60B8_H$	INTTAUC4I8	Interrupt for channel 8	TAUC4	93	0640_H	0640_H
93	ICTAUC4I9	$60BA_H$	INTTAUC4I9	Interrupt for channel 9	TAUC4	94	0650_H	0650_H
94	ICTAUC4I10	$60BC_H$	INTTAUC4I10	Interrupt for channel 10	TAUC4	95	0660_H	0660_H
95	ICTAUC4I11	$60BE_H$	INTTAUC4I11	Interrupt for channel 11	TAUC4	96	0670_H	0670_H
96	ICTAUC4I12	$60C0_H$	INTTAUC4I12	Interrupt for channel 12	TAUC4	97	0680_H	0680_H
97	ICTAUC4I13	$60C2_H$	INTTAUC4I13	Interrupt for channel 13	TAUC4	98	0690_H	0690_H
98	ICTAUC4I14	$60C4_H$	INTTAUC4I14	Interrupt for channel 14	TAUC4	99	$06A0_H$	$06A0_H$
99	ICTAUC4I15	$60C6_H$	INTTAUC4I15	Interrupt for channel 15	TAUC4	100	$06B0_H$	$06B0_H$
100	ICADCA0ERR	$60C8_H$	INTADCA0ERR	Error interrupt	ADCA0	101	$06C0_H$	$06C0_H$

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (4/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
101	ICADCA0I0	$60CA_H$	INTADCA0I0	End of CG0 conversion	ADCA0	102	$06D0_H$	$06D0_H$
			INTDTS7[a]	DTS channel 7 interrupt	DTS			
102	ICADCA0I1	$60CC_H$	INTADCA0I1	End of CG1 conversion	ADCA0	103	$06E0_H$	$06E0_H$
			INTDTS8[a]	DTS channel 8 interrupt	DTS			
103	ICADCA0I2	$60CE_H$	INTADCA0I2	End of CG2 conversion	ADCA0	104	$06F0_H$	$06F0_H$
104	ICADCA0LLT	$60D0_H$	INTADCA0LLT	Conversion interrupt	ADCA0	105	0700_H	0700_H
105	ICFCNWUP	$60D2_H$	INTFCNWUP	Wake up interrupt	FCN[4:0]	106	0710_H	0710_H
106	ICFCN0ERR	$60D4_H$	INTFCN0ERR	Error interrupt	FCN0	107	0720_H	0720_H
107	ICFCN0REC	$60D6_H$	INTFCN0REC	Receive interrupt	FCN0	108	0730_H	0730_H
108	ICFCN0TRX	$60D8_H$	INTFCN0TRX	Transmit interrupt	FCN0	109	0740_H	0740_H
109	ICCSIG0IRE	$60DA_H$	INTCSIG0IRE	Reception error interrupt	CSIG0	110	0750_H	0750_H
110	ICCSIG0IR	$60DC_H$	INTCSIG0IR	Reception status interrupt	CSIG0	111	0760_H	0760_H
111	ICCSIG0IC	$60DE_H$	INTCSIG0IC	Communication status interrupt	CSIG0	112	0770_H	0770_H
112	ICLMA0IS	$60E0_H$	INTLMA0IS	Status interrupt	LMA0	113	0780_H	0780_H
113	ICLMA0IR	$60E2_H$	INTLMA0IR	Reception completion interrupt	LMA0	114	0790_H	0790_H
114	ICLMA0IT	$60E4_H$	INTLMA0IT	Transmission interrupt	LMA0	115	$07A0_H$	$07A0_H$
115	ICLMA1IS	$60E6_H$	INTLMA1IS	Status interrupt	LMA1	116	$07B0_H$	$07B0_H$
116	ICLMA1IR	$60E8_H$	INTLMA1IR	Reception completion interrupt	LMA1	117	$07C0_H$	$07C0_H$
117	ICLMA1IT	$60EA_H$	INTLMA1IT	Transmission interrupt	LMA1	118	$07D0_H$	$07D0_H$
118	ICADCA1ERR	$60EC_H$	INTADCA1ERR	Error interrupt	ADCA1	119	$07E0_H$	$07E0_H$
119	ICDMA0	$60EE_H$	INTDMA0	DMA channel 0 transfer completion (or count match interrupt INTCT0[c])	DMA	120	$07F0_H$	$07F0_H$
120	ICDMA1	$60F0_H$	INTDMA1	DMA channel 1 transfer completion (or count match interrupt INTCT1[c])	DMA	121	0800_H	0800_H
121	ICDMA2	$60F2_H$	INTDMA2	DMA channel 2 transfer completion (or count match interrupt INTCT2[c])	DMA	122	0810_H	0810_H
122	ICDMA3	$60F4_H$	INTDMA3	DMA channel 3 transfer completion (or count match interrupt INTCT3[c])	DMA	123	0820_H	0820_H
123	ICDMA4	$60F6_H$	INTDMA4	DMA channel 4 transfer completion (or count match interrupt INTCT4[c])	DMA	124	0830_H	0830_H
124	ICDMA5	$60F8_H$	INTDMA5	DMA channel 5 transfer completion (or count match interrupt INTCT5[c])	DMA	125	0840_H	0840_H
125	ICDMA6	$60FA_H$	INTDMA6	DMA channel 6 transfer completion (or count match interrupt INTCT6[c])	DMA	126	0850_H	0850_H
126	ICDMA7	$60FC_H$	INTDMA7	DMA channel 7 transfer completion (or count match interrupt INTCT7[c])	DMA	127	0860_H	0860_H
127	–	$60FE_H$	reserved			128	0870_H	0870_H
128	ICIICB0IS	6100_H	INTIICB0IS	Status interrupt	IICB0	129	0880_H	0880_H
129	ICIICB0IA	6102_H	INTIICB0IA	Data transmission/reception interrupt	IICB0	130	0890_H	0890_H
130	–	6104_H	reserved			131	$08A0_H$	$08A0_H$
131	–	6106_H	reserved			132	$08B0_H$	$08B0_H$

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (5/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
132	ICFCN1ERR	6108_H	INTFCN1ERR	Error interrupt	FCN1	133	$08C0_H$	$08C0_H$
133	ICFCN1REC	$610A_H$	INTFCN1REC	Receive interrupt	FCN1	134	$08D0_H$	$08D0_H$
134	ICFCN1TRX	$610C_H$	INTFCN1TRX	Transmit interrupt	FCN1	135	$08E0_H$	$08E0_H$
135	ICTAUJ0I0	$610E_H$	INTTAUJ0I0	Interrupt for channel 0	TAUJ0	136	$08F0_H$	$08F0_H$
136	ICTAUJ0I1	6110_H	INTTAUJ0I1	Interrupt for channel 1	TAUJ0	137	0900_H	0900_H
137	ICTAUJ0I2	6112_H	INTTAUJ0I2	Interrupt for channel 2	TAUJ0	138	0910_H	0910_H
138	ICTAUJ0I3	6114_H	INTTAUJ0I3	Interrupt for channel 3	TAUJ0	139	0920_H	0920_H
139	ICTAUJ1I0	6116_H	INTTAUJ1I0	Interrupt for channel 0	TAUJ1	140	0930_H	0930_H
140	ICTAUJ1I1	6118_H	INTTAUJ1I1	Interrupt for channel 1	TAUJ1	141	0940_H	0940_H
141	ICTAUJ1I2	$611A_H$	INTTAUJ1I2	Interrupt for channel 2	TAUJ1	142	0950_H	0950_H
142	ICTAUJ1I3	$611C_H$	INTTAUJ1I3	Interrupt for channel 3	TAUJ1	143	0960_H	0960_H
143	ICADCA1I0	$611E_H$	INTADCA1I0	End of CG0 conversion	ADCA1	144	0970_H	0970_H
			INTDTS9[a]	DTS channel 9 interrupt	DTS			
144	ICADCA1I1	6120_H	INTADCA1I1	End of CG1 conversion	ADCA1	145	0980_H	0980_H
			INTDTS10[a]	DTS channel 10 interrupt	DTS			
145	ICADCA1I2	6122_H	INTADCA1I2	End of CG2 conversion	ADCA1	146	0990_H	0990_H
			INTDTS11[a]	DTS channel 11 interrupt	DTS			
146	ICADCA1LLT	6124_H	INTADCA1LLT	Conversion interrupt	ADCA1	147	$09A0_H$	$09A0_H$
147	ICOSTM0	6126_H	INTOSTM0	OSTM0 interrupt	OSTM0	148	$09B0_H$	$09B0_H$
148 to 153	–	6128_H to 6132_H	reserved			149 to 154	$09C0_H$ to $0A10_H$	$09C0_H$ to $0A10_H$
154	ICFCN3ERR	6134_H	INTFCN3ERR	Error interrupt	FCN3	155	$0A20_H$	$0A20_H$
155	ICFCN3REC	6136_H	INTFCN3REC	Receive interrupt	FCN3	156	$0A30_H$	$0A30_H$
156	ICFCN3TRX	6138_H	INTFCN3TRX	Transmit interrupt	FCN3	157	$0A40_H$	$0A40_H$
157 to 159	–	$613A_H$ to $613E_H$	reserved			158 to 160	$0A50_H$ to $0A70_H$	$0A50_H$ to $0A70_H$
160	ICFCN2ERR[d]	6140_H	INTFCN2ERR	Error interrupt	FCN2	161	$0A80_H$	$0A80_H$
			INTETHA0SCTXTCH	Data calculation completion interrupt	ETHA0			
161	ICFCN2REC[d]	6142_H	INTFCN2REC	Receive interrupt	FCN2	162	$0A90_H$	$0A90_H$
			INTETHA0SCRXTCH	Checksum data transmission	ETHA0			
162	ICFCN2TRX	6144_H	INTFCN2TRX	Transmit interrupt	FCN2	163	$0AA0_H$	$0AA0_H$
163	ICCSIH0IC	6146_H	INTCSIH0IC	Communication status interrupt	CSIH0	164	$0AB0_H$	$0AB0_H$
			INTDTS12[a]	DTS channel 12 interrupt	DTS			
164	ICCSIH0IJC	6148_H	INTCSIH0IJC	Job completion interrupt	CSIH0	165	$0AC0_H$	$0AC0_H$
			INTDTS13[a]	DTS channel 13 interrupt	DTS			
165 to 167	–	$614A_H$ to $614E_H$	reserved			166 to 168	$0AD0_H$ to $0AF0_H$	$0AD0_H$ to $0AF0_H$

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (6/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
168	ICCSIH0IRE	6150_H	INTCSIH0IRE	Reception error interrupt	CSIH0	169	$0B00_H$	$0B00_H$
169	ICCSIH0IR	6152_H	INTCSIH0IR	Reception status interrupt	CSIH0	170	$0B10_H$	$0B10_H$
			INTDTS14[a]	DTS channel 14 interrupt	DTS			
170	ICCSIG4IRE	6154_H	INTCSIG4IRE	Reception error interrupt	CSIG4	171	$0B20_H$	$0B20_H$
171	ICCSIG4IR	6156_H	INTCSIG4IR	Reception status interrupt	CSIG4	172	$0B30_H$	$0B30_H$
172	ICCSIG4IC	6158_H	INTCSIG4IC	Communication status interrupt	CSIG4	173	$0B40_H$	$0B40_H$
173	INTDCN0ERR	$615A_H$	INTDCN0ERR	Error interrupt	DCN0	174	$0B50_H$	$0B50_H$
174	INTDCN0REC	$615C_H$	INTDCN0REC	Receive interrupt	DCN0	175	$0B60_H$	$0B60_H$
			INTDTS15[a]	DTS channel 15 interrupt	DTS			
175	INTDCN0TRX	$615E_H$	INTDCN0TRX	Transmit interrupt	DCN0	176	$0B70_H$	$0B70_H$
176	ICFLX0I0	6160_H	INTFLX0I0	Interrupt 0	FLXA0	177	$0B80_H$	$0B80_H$
			INTDTS16[a]	DTS channel 16 interrupt	DTS			
177	ICFLX0I1	6162_H	INTFLX0I1	Interrupt 1	FLXA0	178	$0B90_H$	$0B90_H$
			INTDTS17[a]	DTS channel 17 interrupt	DTS			
178	ICFLX0I2	6164_H	INTFLX0I2	Timer 0 interrupt	FLXA0	179	$0BA0_H$	$0BA0_H$
179	ICFLX0I3	6166_H	INTFLX0I3	Timer 1 interrupt	FLXA0	180	$0BB0_H$	$0BB0_H$
180	ICCSIH1IRE	6168_H	INTCSIH1IRE	Reception error interrupt	CSIH1	181	$0BC0_H$	$0BC0_H$
181	ICCSIH1IR	$616A_H$	INTCSIH1IR	Reception status interrupt	CSIH1	182	$0BD0_H$	$0BD0_H$
			INTDTS18[a]	DTS channel 18 interrupt	DTS			
182	ICCSIH1IC	$616C_H$	INTCSIH1IC	Communication status interrupt	CSIH1	183	$0BE0_H$	$0BE0_H$
			INTDTS19[a]	DTS channel 19 interrupt	DTS			
183	ICCSIH1IJC	$616E_H$	INTCSIH1IJC	Job completion interrupt	CSIH1	184	$0BF0_H$	$0BF0_H$
			INTDTS20[a]	DTS channel 20 interrupt	DTS			
184 to 186	–	6170_H to 6174_H	reserved			185 to 187	$0C00_H$ to $0C20_H$	$0C00_H$ to $0C20_H$
187	ICENCA0I0	6176_H	INTENCA0I0	Capture/compare match interrupt	ENCA0	188	$0C30_H$	$0C30_H$
188	ICENCA0I1	6178_H	INTENCA0I1	Capture/compare match interrupt	ENCA0	189	$0C40_H$	$0C40_H$
189	ICENCA0IUD	$617A_H$	INTENCA0IUD	Underflow interrupt	ENCA0	190	$0C50_H$	$0C50_H$
190	ICENCA0IEC	$617C_H$	INTENCA0IEC	Encoder clear interrupt	ENCA0	191	$0C60_H$	$0C60_H$
191	ICENCA0IOV	$617E_H$	INTENCA0IOV	Overflow interrupt	ENCA0	192	$0C70_H$	$0C70_H$
192	–	6180_H	reserved			193	$0C80_H$	$0C80_H$
193	ICLMA2IS	6182_H	INTLMA2IS	Status interrupt	LMA2	194	$0C90_H$	$0C90_H$
194	ICLMA2IR	6184_H	INTLMA2IR	Reception completion interrupt	LMA2	195	$0C8A_H$	$0C8A_H$
195	ICLMA2IT	6186_H	INTLMA2IT	Transmission interrupt	LMA2	196	$0C8B_H$	$0C8B_H$
196	ICLMA3IS	6188_H	INTLMA3IS	Status interrupt	LMA3	197	$0C8C_H$	$0C8C_H$
197	ICLMA3IR	$618A_H$	INTLMA3IR	Reception completion interrupt	LMA3	198	$0C8D_H$	$0C8D_H$
198	ICLMA3IT	$618C_H$	INTLMA3IT	Transmission interrupt	LMA3	199	$0C8E_H$	$0C8E_H$
199	ICLMA4IS	$618E_H$	INTLMA4IS	Status interrupt	LMA4	200	$0CF0_H$	$0CF0_H$

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (7/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
200	ICLMA4IR	6190_H	INTLMA4IR	Reception completion interrupt	LMA4	201	$0D00_H$	$0D00_H$
201	ICLMA4IT	6192_H	INTLMA4IT	Transmission interrupt	LMA4	202	$0D10_H$	$0D10_H$
202	ICCSIH2IRE	6194_H	INTCSIH2IRE	Reception error interrupt	CSIH2	203	$0D20_H$	$0D20_H$
203	ICCSIH2IR	6196_H	INTCSIH2IR	Reception status interrupt	CSIH2	204	$0D30_H$	$0D30_H$
			INTDTS21[a]	DTS channel 21 interrupt	DTS			
204	ICCSIH2IC	6198_H	INTCSIH2IC	Communication status interrupt	CSIH2	205	$0D40_H$	$0D40_H$
			INTDTS22[a]	DTS channel 22 interrupt	DTS			
205	ICCSIH2IJC	$619A_H$	INTCSIH2IJC	Job completion interrupt	CSIH2	206	$0D50_H$	$0D50_H$
			INTDTS23[a]	DTS channel 23 interrupt	DTS			
206	–	$619C_H$	reserved			207	$0D60_H$	$0D60_H$
	ICDTS24		INTDTS24[a]	DTS channel 24 interrupt	DTS			
207	–	$619E_H$	reserved			208	$0D70_H$	$0D70_H$
	ICDTS25		INTDTS25[aa]	DTS channel 25 interrupt	DTS			
208	ICP11	$61A0_H$	INTP11	Edge detection interrupt	Port	209	$0D80_H$	$0D80_H$
209	ICP12	$61A2_H$	INTP12	Edge detection interrupt	Port	210	$0D90_H$	$0D90_H$
210	ICP13	$61A4_H$	INTP13	Edge detection interrupt	Port	211	$0DA0_H$	$0DA0_H$
211	ICP14	$61A6_H$	INTP14	Edge detection interrupt	Port	212	$0DB0_H$	$0DB0_H$
212	ICP15	$61A8_H$	INTP15	Edge detection interrupt	Port	213	$0DC0_H$	$0DC0_H$
213	ICETHA0SRX	$61AA_H$	INTETHA0SRX	Receive data and ready	ETHA0	214	$0DD0_H$	$0DD0_H$
			INTDTS26[a]	DTS channel 26 interrupt	DTS			
214	ICETHA0SCRX	$61AC_H$	INTETHA0SCRX	Packet reception	ETHA0	215	$0DE0_H$	$0DE0_H$
			INTDTS27[a]	DTS channel 27 interrupt	DTS			
215	ICETHA0SCTX	$61AE_H$	INTETHA0SCTX	Packet transmission	ETHA0	216	$0DF0_H$	$0DF0_H$
			INTDTS28[a]	DTS channel 28 interrupt	DTS			
216	ICETHA0RS	$61B0_H$	INTETHA0RS	Reception status	ETHA0	217	$0E00_H$	$0E00_H$
			INTDTS29[a]	DTS channel 29 interrupt	DTS			
217	ICETHA0TS	$61B2_H$	INTETHA0TS	Transmission status	ETHA0	218	$0E10_H$	$0E10_H$
			INTDTS30[a]	DTS channel 30 interrupt	DTS			
218	ICETHA0FS	$61B4_H$	INTETHA0FS	FIFO status	ETHA0	219	$0E20_H$	$0E20_H$
			INTDTS31[a]	DTS channel 31 interrupt	DTS			
219	ICETHA0MAC	$61B6_H$	INTETHA0MAC	MAC core	ETHA0	220	$0E30_H$	$0E30_H$
220	ICKR0	$61B8_H$	INTKR0	Key return interrupt	KR0	221	$0E40_H$	$0E40_H$
221 to 224	–	$61BA_H$ to $61C0_H$	reserved			222 to 225	$0E50_H$ to $0E80_H$	$0E50_H$ to $0E80_H$
225	ICLMA5IS	$61C2_H$	INTLMA5IS	Status interrupt	LMA5	226	$0E90_H$	$0E90_H$
226	ICLMA5IR	$61C4_H$	INTLMA5IR	Reception completion interrupt	LMA5	227	$0EA0_H$	$0EA0_H$
227	ICLMA5IT	$61C6_H$	INTLMA5IT	Transmission interrupt	LMA5	228	$0EB0_H$	$0EB0_H$

Table 5-12 V850E2K/FK4-H EI level maskable interrupt requests (8/8)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
228	ICDMA8	$61C8_H$	INTDMA8	DMA channel 8 transfer completion (or count match interrupt INTCT8[c])	DMA	229	$0EC0_H$	$0EC0_H$
229	ICDMA9	$61CA_H$	INTDMA9	DMA channel 9 transfer completion (or count match interrupt INTCT9[c])	DMA	230	$0ED0_H$	$0ED0_H$
230	ICDMA10	$61CC_H$	INTDMA10	DMA channel 10 transfer completion (or count match interrupt INTCT10[c])	DMA	231	$0EE0_H$	$0EE0_H$
231	ICDMA11	$61CE_H$	INTDMA11	DMA channel 11 transfer completion (or count match interrupt INTCT11[c])	DMA	232	$0EF0_H$	$0EF0_H$
232	ICDMA12	$61D0_H$	INTDMA12	DMA channel 12 transfer completion (or count match interrupt INTCT12[c])	DMA	233	$0F00_H$	$0F00_H$
233	ICDMA13	$61D2_H$	INTDMA13	DMA channel 13 transfer completion (or count match interrupt INTCT13[c])	DMA	234	$0F10_H$	$0F10_H$
234	ICDMA14	$61D4_H$	INTDMA14	DMA channel 14 transfer completion (or count match interrupt INTCT14[c])	DMA	235	$0F20_H$	$0F20_H$
235	ICDMA15	$61D6_H$	INTDMA15	DMA channel 15 transfer completion (or count match interrupt INTCT15[c])	DMA	236	$0F39_H$	$0F39_H$
236	ICLMA6IS	$61D8_H$	INTLMA6IS	Status interrupt	LMA6	237	$0F40_H$	$0F40_H$
237	ICLMA6IR	$61DA_H$	INTLMA6IR	Reception completion interrupt	LMA6	238	$0F50_H$	$0F50_H$
238	ICLMA6IT	$61DC_H$	INTLMA6IT	Transmission interrupt	LMA6	239	$0F60_H$	$0F60_H$
239	ICLMA7IS	$61DE_H$	INTLMA7IS	Status interrupt	LMA7	240	$0F70_H$	$0F70_H$
240	ICLMA7IR	$61F0_H$	INTLMA7IR	Reception completion interrupt	LMA7	241	$0F80_H$	$0F80_H$
241	ICLMA7IT	$61F2_H$	INTLMA7IT	Transmission interrupt	LMA7	242	$0F90_H$	$0F90_H$
242	ICLMA8IS	$61F4_H$	INTLMA8IS	Status interrupt	LMA8	243	$0FA0_H$	$0FA0_H$
243	ICLMA8IR	$61F6_H$	INTLMA8IR	Reception completion interrupt	LMA8	244	$0FB0_H$	$0FB0_H$
244	ICLMA8IT	$61F8_H$	INTLMA8IT	Transmission interrupt	LMA8	245	$0FC0_H$	$0FC0_H$
245	ICLMA9IS	$61FA_H$	INTLMA9IS	Status interrupt	LMA9	246	$0FD9_H$	$0FD9_H$
246	ICLMA9IR	$61FC_H$	INTLMA9IR	Reception completion interrupt	LMA9	247	$0FE0_H$	$0FE0_H$
247	ICLMA9IT	$61FE_H$	INTLMA9IT	Transmission interrupt	LMA9	248	$0FF0_H$	$0FF0_H$
248	ICLMA10IS	6100_H	INTLMA10IS	Status interrupt	LMA10	249	1000_H	1000_H
249	ICLMA10IR	6102_H	INTLMA10IR	Reception completion interrupt	LMA10	250	1010_H	1010_H
250	ICLMA10IT	6104_H	INTLMA10IT	Transmission interrupt	LMA10	251	1020_H	1020_H
251	ICLMA11IS	6106_H	INTLMA11IS	Status interrupt	LMA11	252	1030_H	1030_H
252	ICLMA11IR	6108_H	INTLMA11IR	Reception completion interrupt	LMA11	253	1040_H	1040_H
253	ICLMA11IT	$610A_H$	INTLMA11IT	Transmission interrupt	LMA11	254	1050_H	1050_H
254	–	$610C_H$	reserved			255	1060_H	1060_H
255	–	$60E_H$	reserved			256	1070_H	1070_H

a) Selection of the DTS interrupts is achieved by setting the DTS path through mode, refer to *5.3.3 "V850E2/Fx4-H EI level maskable interrupt sharing" on page 292*

b) Selection of the shared TAPA interrupts is achieved by the TAPAINTSL0 register, refer to section *5.3.4 "V850E2/Fx4-H TAPA EI level maskable interrupt sharing" on page 295* .

c) Selection of the DMA interrupts is achieved by setting the DMA interrupt selection register refer to *5.3.5 "V850E2/Fx4-H DMA interrupt selection"*.

d) These interrupts are shared by several interrupt sources via a logical or-function, refer to *5.3.3 “V850E2/Fx4-H EI level maskable interrupt sharing” on page 292* .

(5) V850E2K/FL4-H EI level maskable interrupts

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (1/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
0	ICWDTA0	6000_H	INTWDTA0	WDTA0 75% interrupt	WDTA0	1	0080_H	0080_H
1	ICWDTA1	6002_H	INTWDTA1	WDTA1 75% interrupt	WDTA1	2	0090_H	0090_H
2	ICLVI	6004_H	INTLVI	LVI interrupt	LVI	3	$00A0_H$	$00A0_H$
3	–	6006_H	reserved			4	$00B0_H$	$00B0_H$
4	ICVCPC0	6008_H	INTVCPC0	Voltage Comparator 0 interrupt	VCPC0	5	$00C0_H$	$00C0_H$
5	ICVCPC1	$600A_H$	INTVCPC1	Voltage Comparator 1 interrupt	VCPC1	6	$00D0_H$	$00D0_H$
6	ICRTCA01S	$600C_H$	INTRTCA01S	Real-Time Clock 1 second interrupt	RTCA0	7	$00E0_H$	$00E0_H$
7	ICRTCA0AL	$600E_H$	INTRTCA0AL	Alarm interrupt	RTCA0	8	$00F0_H$	$00F0_H$
8	ICRTCA0R	6010_H	INTRTCA0R	Fixed frequency interruption	RTCA0	9	0100_H	0100_H
9	ICP0	6012_H	INTP0	Edge detection interrupt	Port	10	0110_H	0110_H
			INTDTS0[a]	DTS channel 0 interrupt	DTS			
10	ICP1	6014_H	INTP1	Edge detection interrupt	Port	11	0120_H	0120_H
			INTDTS1[a]	DTS channel 1 interrupt	DTS			
11	ICP2	6016_H	INTP2	Edge detection interrupt	Port	12	0130_H	0130_H
			INTDTS2[a]	DTS channel 2 interrupt	DTS			
12	ICP3	6018_H	INTP3	Edge detection interrupt	Port	13	0140_H	0140_H
13	ICP4	$601A_H$	INTP4	Edge detection interrupt	Port	14	0150_H	0150_H
14	ICP5	$601C_H$	INTP5	Edge detection interrupt	Port	15	0160_H	0160_H
15	ICP6	$601E_H$	INTP6	Edge detection interrupt	Port	16	0170_H	0170_H
16	ICP7	6020_H	INTP7	Edge detection interrupt	Port	17	0180_H	0180_H
17	ICP8	6022_H	INTP8	Edge detection interrupt	Port	18	0190_H	0190_H
18	ICP9	6024_H	INTP9	Edge detection interrupt	Port	19	$01A0_H$	$01A0_H$
19	ICP10	6026_H	INTP10	Edge detection interrupt	Port	20	$01B0_H$	$01B0_H$
20	ICTAUA0I0	6028_H	INTTAUA0I0	Interrupt for channel 0	TAUA0	21	$01C0_H$	$01C0_H$
			INTTAPA0PEK0[b]	Peak interrupt 0	TAPA0			
21	ICTAUA0I1	$602A_H$	INTTAUA0I1	Interrupt for channel 1	TAUA0	22	$01D0_H$	$01D0_H$
			INTTAPAPEK1[b]	Peak interrupt 1	TAPA0			
22	ICTAUA0I2	$602C_H$	INTTAUA0I2	Interrupt for channel 2	TAUA0	23	$01E0_H$	$01E0_H$
23	ICTAUA0I3	$602E_H$	INTTAUA0I3	Interrupt for channel 3	TAUA0	24	$01F0_H$	$01F0_H$
24	ICTAUA0I4	6030_H	INTTAUA0I4	Interrupt for channel 4	TAUA0	25	0200_H	0200_H
			INTTAPA0VLY0[b]	Valley interrupt 0	TAPA0			
25	ICTAUA0I5	6032_H	INTTAUA0I5	Interrupt for channel 5	TAUA0	26	0210_H	0210_H
			INTTAPA0VLY1[b]	Valley interrupt 0	TAPA0			
26	ICTAUA0I6	6034_H	INTTAUA0I6	Interrupt for channel 6	TAUA0	27	0220_H	0220_H
27	ICTAUA0I7	6036_H	INTTAUA0I7	Interrupt for channel 7	TAUA0	28	0230_H	0230_H
28	ICTAUA0I8	6038_H	INTTAUA0I8	Interrupt for channel 8	TAUA0	29	0240_H	0240_H
29	ICTAUA0I9	$603A_H$	INTTAUA0I9	Interrupt for channel 9	TAUA0	30	0250_H	0250_H

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (2/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
30	ICTAUA0I10	$603C_H$	INTTAUA0I10	Interrupt for channel 10	TAUA0	31	0260_H	0260_H
31	ICTAUA0I11	$603E_H$	INTTAUA0I11	Interrupt for channel 11	TAUA0	32	0270_H	0270_H
32	ICTAUA0I12	6040_H	INTTAUA0I12	Interrupt for channel 12	TAUA0	33	0280_H	0280_H
33	ICTAUA0I13	6042_H	INTTAUA0I13	Interrupt for channel 13	TAUA0	34	0290_H	0290_H
34	ICTAUA0I14	6044_H	INTTAUA0I14	Interrupt for channel 14	TAUA0	35	$02A0_H$	$02A0_H$
35	ICTAUA0I15	6046_H	INTTAUA0I15	Interrupt for channel 15	TAUA0	36	$02B0_H$	$02B0_H$
36	ICTAUB1I0	6048_H	INTTAUB1I0	Interrupt for channel 0	TAUB1	37	$02C0_H$	$02C0_H$
			INTDTS3[a]	DTS channel 3 interrupt	DTS			
37	ICTAUB1I1	$604A_H$	INTTAUB1I1	Interrupt for channel 1	TAUB1	38	$02D0_H$	$02D0_H$
			INTDTS4[a]	DTS channel 4 interrupt	DTS			
38	ICTAUB1I2	$604C_H$	INTTAUB1I2	Interrupt for channel 2	TAUB1	39	$02E0_H$	$02E0_H$
39	ICTAUB1I3	$604E_H$	INTTAUB1I3	Interrupt for channel 3	TAUB1	40	$02F0_H$	$02F0_H$
40	ICTAUB1I4	6050_H	INTTAUB1I4	Interrupt for channel 4	TAUB1	41	0300_H	0300_H
41	ICTAUB1I5	6052_H	INTTAUB1I5	Interrupt for channel 5	TAUB1	42	0310_H	0310_H
42	ICTAUB1I6	6054_H	INTTAUB1I6	Interrupt for channel 6	TAUB1	43	0320_H	0320_H
43	ICTAUB1I7	6056_H	INTTAUB1I7	Interrupt for channel 7	TAUB1	44	0330_H	0330_H
44	ICTAUB1I8	6058_H	INTTAUB1I8	Interrupt for channel 8	TAUB1	45	0340_H	0340_H
45	ICTAUB1I9	$605A_H$	INTTAUB1I9	Interrupt for channel 9	TAUB1	46	0350_H	0350_H
46	ICTAUB1I10	$605C_H$	INTTAUB1I10	Interrupt for channel 10	TAUB1	47	0360_H	0360_H
47	ICTAUB1I11	$605E_H$	INTTAUB1I11	Interrupt for channel 11	TAUB1	48	0370_H	0370_H
48	ICTAUB1I12	6060_H	INTTAUB1I12	Interrupt for channel 12	TAUB1	49	0380_H	0380_H
49	ICTAUB1I13	6062_H	INTTAUB1I13	Interrupt for channel 13	TAUB1	50	0390_H	0390_H
50	ICTAUB1I14	6064_H	INTTAUB1I14	Interrupt for channel 14	TAUB1	51	$03A0_H$	$03A0_H$
51	ICTAUB1I15	6066_H	INTTAUB1I15	Interrupt for channel 15	TAUB1	52	$03B0_H$	$03B0_H$
52	ICTAUB2I0	6068_H	INTTAUB2I0	Interrupt for channel 0	TAUB2	53	$03C0_H$	$03C0_H$
			INTDTS5[a]	DTS channel 5 interrupt	DTS			
53	ICTAUB2I1	$606A_H$	INTTAUB2I1	Interrupt for channel 1	TAUB2	54	$03D0_H$	$03D0_H$
			INTDTS6[a]	DTS channel 6 interrupt	DTS			
54	ICTAUB2I2	$606C_H$	INTTAUB2I2	Interrupt for channel 2	TAUB2	55	$03E0_H$	$03E0_H$
55	ICTAUB2I3	$606E_H$	INTTAUB2I3	Interrupt for channel 3	TAUB2	56	$03F0_H$	$03F0_H$
56	ICTAUB2I4	6070_H	INTTAUB2I4	Interrupt for channel 4	TAUB2	57	0400_H	0400_H
57	ICTAUB2I5	6072_H	INTTAUB2I5	Interrupt for channel 5	TAUB2	58	0410_H	0410_H
58	ICTAUB2I6	6074_H	INTTAUB2I6	Interrupt for channel 6	TAUB2	59	0420_H	0420_H
59	ICTAUB2I7	6076_H	INTTAUB2I7	Interrupt for channel 7	TAUB2	60	0430_H	0430_H
60	ICTAUB2I8	6078_H	INTTAUB2I8	Interrupt for channel 8	TAUB2	61	0440_H	0440_H
61	ICTAUB2I9	$607A_H$	INTTAUB2I9	Interrupt for channel 9	TAUB2	62	0450_H	0450_H
62	ICTAUB2I10	$607C_H$	INTTAUB2I10	Interrupt for channel 10	TAUB2	63	0460_H	0460_H
63	ICTAUB2I11	$607E_H$	INTTAUB2I11	Interrupt for channel 11	TAUB2	64	0470_H	0470_H

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (3/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
64	ICTAUB2I12	6080_H	INTTAUB2I12	Interrupt for channel 12	TAUB2	65	0480_H	0480_H
65	ICTAUB2I13	6082_H	INTTAUB2I13	Interrupt for channel 13	TAUB2	66	0490_H	0490_H
66	ICTAUB2I14	6084_H	INTTAUB2I14	Interrupt for channel 14	TAUB2	67	$04A0_H$	$04A0_H$
67	ICTAUB2I15	6086_H	INTTAUB2I15	Interrupt for channel 15	TAUB2	68	$04B0_H$	$04B0_H$
68	ICTAUC3I0	6088_H	INTTAUC3I0	Interrupt for channel 0	TAUC3	69	$04C0_H$	$04C0_H$
69	ICTAUC3I1	$608A_H$	INTTAUC3I1	Interrupt for channel 1	TAUC3	70	$04D0_H$	$04D0_H$
70	ICTAUC3I2	$608C_H$	INTTAUC3I2	Interrupt for channel 2	TAUC3	71	$04E0_H$	$04E0_H$
71	ICTAUC3I3	$600E_H$	INTTAUC3I3	Interrupt for channel 3	TAUC3	72	$04F0_H$	$04F0_H$
72	ICTAUC3I4	6090_H	INTTAUC3I4	Interrupt for channel 4	TAUC3	73	0500_H	0500_H
73	ICTAUC3I5	6092_H	INTTAUC3I5	Interrupt for channel 5	TAUC3	74	0510_H	0510_H
74	ICTAUC3I6	6094_H	INTTAUC3I6	Interrupt for channel 6	TAUC3	75	0520_H	0520_H
75	ICTAUC3I7	6096_H	INTTAUC3I7	Interrupt for channel 7	TAUC3	76	0530_H	0530_H
76	ICTAUC3I8	6098_H	INTTAUC3I8	Interrupt for channel 8	TAUC3	77	0540_H	0540_H
77	ICTAUC3I9	$609A_H$	INTTAUC3I9	Interrupt for channel 9	TAUC3	78	0550_H	0550_H
78	ICTAUC3I10	$609C_H$	INTTAUC3I10	Interrupt for channel 10	TAUC3	79	0560_H	0560_H
79	ICTAUC3I11	$609E_H$	INTTAUC3I11	Interrupt for channel 11	TAUC3	80	0570_H	0570_H
80	ICTAUC3I12	$60A0_H$	INTTAUC3I12	Interrupt for channel 12	TAUC3	81	0580_H	0580_H
81	ICTAUC3I13	$60A2_H$	INTTAUC3I13	Interrupt for channel 13	TAUC3	82	0590_H	0590_H
82	ICTAUC3I14	$60A4_H$	INTTAUC3I14	Interrupt for channel 14	TAUC3	83	$05A0_H$	$05A0_H$
83	ICTAUC3I15	$60A6_H$	INTTAUC3I15	Interrupt for channel 15	TAUC3	84	$05B0_H$	$05B0_H$
84	ICTAUC4I0	$60A8_H$	INTTAUC4I0	Interrupt for channel 0	TAUC4	85	$05C0_H$	$05C0_H$
85	ICTAUC4I1	$60AA_H$	INTTAUC4I1	Interrupt for channel 1	TAUC4	86	$05D0_H$	$05D0_H$
86	ICTAUC4I2	$60AC_H$	INTTAUC4I2	Interrupt for channel 2	TAUC4	87	$05E0_H$	$05E0_H$
87	ICTAUC4I3	$60AE_H$	INTTAUC4I3	Interrupt for channel 3	TAUC4	88	$05F0_H$	$05F0_H$
88	ICTAUC4I4	$60B0_H$	INTTAUC4I4	Interrupt for channel 4	TAUC4	89	0600_H	0600_H
89	ICTAUC4I5	$60B2_H$	INTTAUC4I5	Interrupt for channel 5	TAUC4	90	0610_H	0610_H
90	ICTAUC4I6	$60B4_H$	INTTAUC4I6	Interrupt for channel 6	TAUC4	91	0620_H	0620_H
91	ICTAUC4I7	$60B6_H$	INTTAUC4I7	Interrupt for channel 7	TAUC4	92	0630_H	0630_H
92	ICTAUC4I8	$60B8_H$	INTTAUC4I8	Interrupt for channel 8	TAUC4	93	0640_H	0640_H
93	ICTAUC4I9	$60BA_H$	INTTAUC4I9	Interrupt for channel 9	TAUC4	94	0650_H	0650_H
94	ICTAUC4I10	$60BC_H$	INTTAUC4I10	Interrupt for channel 10	TAUC4	95	0660_H	0660_H
95	ICTAUC4I11	$60BE_H$	INTTAUC4I11	Interrupt for channel 11	TAUC4	96	0670_H	0670_H
96	ICTAUC4I12	$60C0_H$	INTTAUC4I12	Interrupt for channel 12	TAUC4	97	0680_H	0680_H
97	ICTAUC4I13	$60C2_H$	INTTAUC4I13	Interrupt for channel 13	TAUC4	98	0690_H	0690_H
98	ICTAUC4I14	$60C4_H$	INTTAUC4I14	Interrupt for channel 14	TAUC4	99	$06A0_H$	$06A0_H$
99	ICTAUC4I15	$60C6_H$	INTTAUC4I15	Interrupt for channel 15	TAUC4	100	$06B0_H$	$06B0_H$
100	ICADCA0ERR	$60C8_H$	INTADCA0ERR	Error interrupt	ADCA0	101	$06C0_H$	$06C0_H$

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (4/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
101	ICADCA0I0	$60CA_H$	INTADCA0I0	End of CG0 conversion	ADCA0	102	$06D0_H$	$06D0_H$
			INTDTS7[a]	DTS channel 7 interrupt	DTS			
102	ICADCA0I1	$60CC_H$	INTADCA0I1	End of CG1 conversion	ADCA0	103	$06E0_H$	$06E0_H$
			INTDTS8[a]	DTS channel 8 interrupt	DTS			
103	ICADCA0I2	$60CE_H$	INTADCA0I2	End of CG2 conversion	ADCA0	104	$06F0_H$	$06F0_H$
104	ICADCA0LLT	$60D0_H$	INTADCA0LLT	Conversion interrupt	ADCA0	105	0700_H	0700_H
105	ICFCNWUP	$60D2_H$	INTFCNWUP	Wake up interrupt	FCN[4:0]	106	0710_H	0710_H
106	ICFCN0ERR	$60D4_H$	INTFCN0ERR	Error interrupt	FCN0	107	0720_H	0720_H
107	ICFCN0REC	$60D6_H$	INTFCN0REC	Receive interrupt	FCN0	108	0730_H	0730_H
108	ICFCN0TRX	$60D8_H$	INTFCN0TRX	Transmit interrupt	FCN0	109	0740_H	0740_H
109	ICCSIG0IRE	$60DA_H$	INTCSIG0IRE	Reception error interrupt	CSIG0	110	0750_H	0750_H
110	ICCSIG0IR	$60DC_H$	INTCSIG0IR	Reception status interrupt	CSIG0	111	0760_H	0760_H
111	ICCSIG0IC	$60DE_H$	INTCSIG0IC	Communication status interrupt	CSIG0	112	0770_H	0770_H
112	ICLMA0IS	$60E0_H$	INTLMA0IS	Status interrupt	LMA0	113	0780_H	0780_H
113	ICLMA0IR	$60E2_H$	INTLMA0IR	Reception completion interrupt	LMA0	114	0790_H	0790_H
114	ICLMA0IT	$60E4_H$	INTLMA0IT	Transmission interrupt	LMA0	115	$07A0_H$	$07A0_H$
115	ICLMA1IS	$60E6_H$	INTLMA1IS	Status interrupt	LMA1	116	$07B0_H$	$07B0_H$
116	ICLMA1IR	$60E8_H$	INTLMA1IR	Reception completion interrupt	LMA1	117	$07C0_H$	$07C0_H$
117	ICLMA1IT	$60EA_H$	INTLMA1IT	Transmission interrupt	LMA1	118	$07D0_H$	$07D0_H$
118	ICADCA1ERR	$60EC_H$	INTADCA1ERR	Error interrupt	ADCA1	119	$07E0_H$	$07E0_H$
119	ICDMA0	$60EE_H$	INTDMA0	DMA channel 0 transfer completion (or count match interrupt INTCT0[c])	DMA	120	$07F0_H$	$07F0_H$
120	ICDMA1	$60F0_H$	INTDMA1	DMA channel 1 transfer completion (or count match interrupt INTCT1[c])	DMA	121	0800_H	0800_H
121	ICDMA2	$60F2_H$	INTDMA2	DMA channel 2 transfer completion (or count match interrupt INTCT2[c])	DMA	122	0810_H	0810_H
122	ICDMA3	$60F4_H$	INTDMA3	DMA channel 3 transfer completion (or count match interrupt INTCT3[c])	DMA	123	0820_H	0820_H
123	ICDMA4	$60F6_H$	INTDMA4	DMA channel 4 transfer completion (or count match interrupt INTCT4[c])	DMA	124	0830_H	0830_H
124	ICDMA5	$60F8_H$	INTDMA5	DMA channel 5 transfer completion (or count match interrupt INTCT5[c])	DMA	125	0840_H	0840_H
125	ICDMA6	$60FA_H$	INTDMA6	DMA channel 6 transfer completion (or count match interrupt INTCT6[c])	DMA	126	0850_H	0850_H
126	ICDMA7	$60FC_H$	INTDMA7	DMA channel 7 transfer completion (or count match interrupt INTCT7[c])	DMA	127	0860_H	0860_H
127	–	$60FE_H$	reserved			128	0870_H	0870_H
128	ICIICB0IS	6100_H	INTIICB0IS	Status interrupt	IICB0	129	0880_H	0880_H
129	ICIICB0IA	6102_H	INTIICB0IA	Data transmission/reception interrupt	IICB0	130	0890_H	0890_H
130	–	6104_H	reserved			131	$08A0_H$	$08A0_H$
131	–	6106_H	reserved			132	$08B0_H$	$08B0_H$

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (5/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
132	ICFCN1ERR	6108_H	INTFCN1ERR	Error interrupt	FCN1	133	$08C0_H$	$08C0_H$
133	ICFCN1REC	$610A_H$	INTFCN1REC	Receive interrupt	FCN1	134	$08D0_H$	$08D0_H$
134	ICFCN1TRX	$610C_H$	INTFCN1TRX	Transmit interrupt	FCN1	135	$08E0_H$	$08E0_H$
135	ICTAUJ0I0	$610E_H$	INTTAUJ0I0	Interrupt for channel 0	TAUJ0	136	$08F0_H$	$08F0_H$
136	ICTAUJ0I1	6110_H	INTTAUJ0I1	Interrupt for channel 1	TAUJ0	137	0900_H	0900_H
137	ICTAUJ0I2	6112_H	INTTAUJ0I2	Interrupt for channel 2	TAUJ0	138	0910_H	0910_H
138	ICTAUJ0I3	6114_H	INTTAUJ0I3	Interrupt for channel 3	TAUJ0	139	0920_H	0920_H
139	ICTAUJ1I0	6116_H	INTTAUJ1I0	Interrupt for channel 0	TAUJ1	140	0930_H	0930_H
140	ICTAUJ1I1	6118_H	INTTAUJ1I1	Interrupt for channel 1	TAUJ1	141	0940_H	0940_H
141	ICTAUJ1I2	$611A_H$	INTTAUJ1I2	Interrupt for channel 2	TAUJ1	142	0950_H	0950_H
142	ICTAUJ1I3	$611C_H$	INTTAUJ1I3	Interrupt for channel 3	TAUJ1	143	0960_H	0960_H
143	ICADCA1I0	$611E_H$	INTADCA1I0	End of CG0 conversion	ADCA1	144	0970_H	0970_H
			INTDTS9[a]	DTS channel 9 interrupt	DTS			
144	ICADCA1I1	6120_H	INTADCA1I1	End of CG1 conversion	ADCA1	145	0980_H	0980_H
			INTDTS10[a]	DTS channel 10 interrupt	DTS			
145	ICADCA1I2	6122_H	INTADCA1I2	End of CG2 conversion	ADCA1	146	0990_H	0990_H
			INTDTS11[a]	DTS channel 11 interrupt	DTS			
146	ICADCA1LLT	6124_H	INTADCA1LLT	Conversion interrupt	ADCA1	147	$09A0_H$	$09A0_H$
147	ICOSTM0	6126_H	INTOSTM0	OSTM0 interrupt	OSTM0	148	$09B0_H$	$09B0_H$
148 to 150	–	6128_H to $612C_H$	reserved			149 to 151	$09C0_H$ to $09E0_H$	$09C0_H$ to $09E0_H$
151	ICCSIG2IRE	$612E_H$	INTCSIG2IRE	Reception error interrupt	CSIG2	152	$09F0_H$	$09F0_H$
152	ICCSIG2IR	6130_H	INTCSIG2IR	Reception status interrupt	CSIG2	153	$0A00_H$	$0A00_H$
153	ICCSIG2IC	6132_H	INTCSIG2IC	Communication status interrupt	CSIG2	154	$0A10_H$	$0A10_H$
154	ICFCN3ERR	6134_H	INTFCN3ERR	Error interrupt	FCN3	155	$0A20_H$	$0A20_H$
155	ICFCN3REC	6136_H	INTFCN3REC	Receive interrupt	FCN3	156	$0A30_H$	$0A30_H$
156	ICFCN3TRX	6138_H	INTFCN3TRX	Transmit interrupt	FCN3	157	$0A40_H$	$0A40_H$
157	ICFCN4ERR	$613A_H$	INTFCN4ERR	Error interrupt	FCN4	158	$0A50_H$	$0A50_H$
158	ICFCN4REC	$613C_H$	INTFCN4REC	Receive interrupt	FCN4	159	$0A60_H$	$0A60_H$
159	ICFCN4TRX	$613E_H$	INTFCN4TRX	Transmit interrupt	FCN4	160	$0A70_H$	$0A70_H$
160	ICFCN2ERR[d]	6140_H	INTFCN2ERR	Error interrupt	FCN2	161	$0A80_H$	$0A80_H$
			INTETHA0SCTXTCH	Data calculation completion interrupt	ETHA0			
161	ICFCN2REC[d]	6142_H	INTFCN2REC	Receive interrupt	FCN2	162	$0A90_H$	$0A90_H$
			INTETHA0SCRXTCH	Checksum data transmission	ETHA0			
162	ICFCN2TRX	6144_H	INTFCN2TRX	Transmit interrupt	FCN2	163	$0AA0_H$	$0AA0_H$
163	ICCSIH0IC	6146_H	INTCSIH0IC	Communication status interrupt	CSIH0	164	$0AB0_H$	$0AB0_H$
			INTDTS12[a]	DTS channel 12 interrupt	DTS			

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (6/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
164	ICCSIH0IJC	6148_H	INTCSIH0IJC	Job completion interrupt	CSIH0	165	$0AC0_H$	$0AC0_H$
			INTDTS13[a]	DTS channel 13 interrupt	DTS			
165 to 167	–	$614A_H$ to $614E_H$	reserved			166 to 168	$0AD0_H$ to $0AF0_H$	$0AD0_H$ to $0AF0_H$
168	ICCSIH0IRE	6150_H	INTCSIH0IRE	Reception error interrupt	CSIH0	169	$0B00_H$	$0B00_H$
169	ICCSIH0IR	6152_H	INTCSIH0IR	Reception status interrupt	CSIH0	170	$0B10_H$	$0B10_H$
			INTDTS14[a]	DTS channel 14 interrupt	DTS			
170	ICCSIG4IRE	6154_H	INTCSIG4IRE	Reception error interrupt	CSIG4	171	$0B20_H$	$0B20_H$
171	ICCSIG4IR	6156_H	INTCSIG4IR	Reception status interrupt	CSIG4	172	$0B30_H$	$0B30_H$
172	ICCSIG4IC	6158_H	INTCSIG4IC	Communication status interrupt	CSIG4	173	$0B40_H$	$0B40_H$
173	INTDCN0ERR	$615A_H$	INTDCN0ERR	Error interrupt	DCN0	174	$0B50_H$	$0B50_H$
174	INTDCN0REC	$615C_H$	INTDCN0REC	Receive interrupt	DCN0	175	$0B60_H$	$0B60_H$
			INTDTS15[a]	DTS channel 15 interrupt	DTS			
175	INTDCN0TRX	$615E_H$	INTDCN0TRX	Transmit interrupt	DCN0	176	$0B70_H$	$0B70_H$
176	ICFLX0I0	6160_H	INTFLX0I0	Interrupt 0	FLXA0	177	$0B80_H$	$0B80_H$
			INTDTS16[a]	DTS channel 16 interrupt	DTS			
177	ICFLX0I1	6162_H	INTFLX0I1	Interrupt 1	FLXA0	178	$0B90_H$	$0B90_H$
			INTDTS17[a]	DTS channel 17 interrupt	DTS			
178	ICFLX0I2	6164_H	INTFLX0I2	Timer 0 interrupt	FLXA0	179	$0BA0_H$	$0BA0_H$
179	ICFLX0I3	6166_H	INTFLX0I3	Timer 1 interrupt	FLXA0	180	$0BB0_H$	$0BB0_H$
180	ICCSIH1IRE	6168_H	INTCSIH1IRE	Reception error interrupt	CSIH1	181	$0BC0_H$	$0BC0_H$
181	ICCSIH1IR	$616A_H$	INTCSIH1IR	Reception status interrupt	CSIH1	182	$0BD0_H$	$0BD0_H$
			INTDTS18[a]	DTS channel 18 interrupt	DTS			
182	ICCSIH1IC	$616C_H$	INTCSIH1IC	Communication status interrupt	CSIH1	183	$0BE0_H$	$0BE0_H$
			INTDTS19[a]	DTS channel 19 interrupt	DTS			
183	ICCSIH1IJC	$616E_H$	INTCSIH1IJC	Job completion interrupt	CSIH1	184	$0BF0_H$	$0BF0_H$
			INTDTS20[a]	DTS channel 20 interrupt	DTS			
184 to 186	–	6170_H to 6174_H	reserved			185 to 187	$0C00_H$ to $0C20_H$	$0C00_H$ to $0C20_H$
187	ICENCA0I0	6176_H	INTENCA0I0	Capture/compare match interrupt	ENCA0	188	$0C30_H$	$0C30_H$
188	ICENCA0I1	6178_H	INTENCA0I1	Capture/compare match interrupt	ENCA0	189	$0C40_H$	$0C40_H$
189	ICENCA0IUD	$617A_H$	INTENCA0IUD	Underflow interrupt	ENCA0	190	$0C50_H$	$0C50_H$
190	ICENCA0IEC	$617C_H$	INTENCA0IEC	Encoder clear interrupt	ENCA0	191	$0C60_H$	$0C60_H$
191	ICENCA0IOV	$617E_H$	INTENCA0IOV	Overflow interrupt	ENCA0	192	$0C70_H$	$0C70_H$
192	–	6180_H	reserved			193	$0C80_H$	$0C80_H$
193	ICLMA2IS	6182_H	INTLMA2IS	Status interrupt	LMA2	194	$0C90_H$	$0C90_H$
194	ICLMA2IR	6184_H	INTLMA2IR	Reception completion interrupt	LMA2	195	$0C8A_H$	$0C8A_H$

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (7/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
195	ICLMA2IT	6186_H	INTLMA2IT	Transmission interrupt	LMA2	196	$0C8B_H$	$0C8B_H$
196	ICLMA3IS	6188_H	INTLMA3IS	Status interrupt	LMA3	197	$0C8C_H$	$0C8C_H$
197	ICLMA3IR	$618A_H$	INTLMA3IR	Reception completion interrupt	LMA3	198	$0C8D_H$	$0C8D_H$
198	ICLMA3IT	$618C_H$	INTLMA3IT	Transmission interrupt	LMA3	199	$0C8E_H$	$0C8E_H$
199	ICLMA4IS	$618E_H$	INTLMA4IS	Status interrupt	LMA4	200	$0CF0_H$	$0CF0_H$
200	ICLMA4IR	6190_H	INTLMA4IR	Reception completion interrupt	LMA4	201	$0D00_H$	$0D00_H$
201	ICLMA4IT	6192_H	INTLMA4IT	Transmission interrupt	LMA4	202	$0D10_H$	$0D10_H$
202	ICCSIH2IRE	6194_H	INTCSIH2IRE	Reception error interrupt	CSIH2	203	$0D20_H$	$0D20_H$
203	ICCSIH2IR	6196_H	INTCSIH2IR	Reception status interrupt	CSIH2	204	$0D30_H$	$0D30_H$
			INTDTS21[a]	DTS channel 21 interrupt	DTS			
204	ICCSIH2IC	6198_H	INTCSIH2IC	Communication status interrupt	CSIH2	205	$0D40_H$	$0D40_H$
			INTDTS22[a]	DTS channel 22 interrupt	DTS			
205	ICCSIH2IJC	$619A_H$	INTCSIH2IJC	Job completion interrupt	CSIH2	206	$0D50_H$	$0D50_H$
			INTDTS23[a]	DTS channel 23 interrupt	DTS			
206	–	$619C_H$	reserved			207	$0D60_H$	$0D60_H$
	ICDTS24		INTDTS24[a]	DTS channel 24 interrupt	DTS			
207	–	$619E_H$	reserved			208	$0D70_H$	$0D70_H$
	ICDTS25		INTDTS25[a]	DTS channel 25 interrupt	DTS			
208	ICP11	$61A0_H$	INTP11	Edge detection interrupt	Port	209	$0D80_H$	$0D80_H$
209	ICP12	$61A2_H$	INTP12	Edge detection interrupt	Port	210	$0D90_H$	$0D90_H$
210	ICP13	$61A4_H$	INTP13	Edge detection interrupt	Port	211	$0DA0_H$	$0DA0_H$
211	ICP14	$61A6_H$	INTP14	Edge detection interrupt	Port	212	$0DB0_H$	$0DB0_H$
212	ICP15	$61A8_H$	INTP15	Edge detection interrupt	Port	213	$0DC0_H$	$0DC0_H$
213	ICETHA0SRX	$61AA_H$	INTETHA0SRX	Receive data and ready	ETHA0	214	$0DD0_H$	$0DD0_H$
			INTDTS26[a]	DTS channel 26 interrupt	DTS			
214	ICETHA0SCRX	$61AC_H$	INTETHA0SCRX	Packet reception	ETHA0	215	$0DE0_H$	$0DE0_H$
			INTDTS27[a]	DTS channel 27 interrupt	DTS			
215	ICETHA0SCTX	$61AE_H$	INTETHA0SCTX	Packet transmission	ETHA0	216	$0DF0_H$	$0DF0_H$
			INTDTS28[a]	DTS channel 28 interrupt	DTS			
216	ICETHA0RS	$61B0_H$	INTETHA0RS	Reception status	ETHA0	217	$0E00_H$	$0E00_H$
			INTDTS29[a]	DTS channel 29 interrupt	DTS			
217	ICETHA0TS	$61B2_H$	INTETHA0TS	Transmission status	ETHA0	218	$0E10_H$	$0E10_H$
			INTDTS30[a]	DTS channel 30 interrupt	DTS			
218	ICETHA0FS	$61B4_H$	INTETHA0FS	FIFO status	ETHA0	219	$0E20_H$	$0E20_H$
			INTDTS31[a]	DTS channel 31 interrupt	DTS			
219	ICETHA0MAC	$61B6_H$	INTETHA0MAC	MAC core	ETHA0	220	$0E30_H$	$0E30_H$
220	ICKR0	$61B8_H$	INTKR0	Key return interrupt	KR0	221	$0E40_H$	$0E40_H$

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (8/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
221 to 224	–	$61BA_H$ to $61C0_H$	reserved			222 to 225	$0E50_H$ to $0E80_H$	$0E50_H$ to $0E80_H$
225	ICLMA5IS	$61C2_H$	INTLMA5IS	Status interrupt	LMA5	226	$0E90_H$	$0E90_H$
226	ICLMA5IR	$61C4_H$	INTLMA5IR	Reception completion interrupt	LMA5	227	$0EA0_H$	$0EA0_H$
227	ICLMA5IT	$61C6_H$	INTLMA5IT	Transmission interrupt	LMA5	228	$0EB0_H$	$0EB0_H$
228	ICDMA8	$61C8_H$	INTDMA8	DMA channel 8 transfer completion (or count match interrupt INTCT8[C])	DMA	229	$0EC0_H$	$0EC0_H$
229	ICDMA9	$61CA_H$	INTDMA9	DMA channel 9 transfer completion (or count match interrupt INTCT9[C])	DMA	230	$0ED0_H$	$0ED0_H$
230	ICDMA10	$61CC_H$	INTDMA10	DMA channel 10 transfer completion (or count match interrupt INTCT10[C])	DMA	231	$0EE0_H$	$0EE0_H$
231	ICDMA11	$61CE_H$	INTDMA11	DMA channel 11 transfer completion (or count match interrupt INTCT11[C])	DMA	232	$0EF0_H$	$0EF0_H$
232	ICDMA12	$61D0_H$	INTDMA12	DMA channel 12 transfer completion (or count match interrupt INTCT12[C])	DMA	233	$0F00_H$	$0F00_H$
233	ICDMA13	$61D2_H$	INTDMA13	DMA channel 13 transfer completion (or count match interrupt INTCT13[C])	DMA	234	$0F10_H$	$0F10_H$
234	ICDMA14	$61D4_H$	INTDMA14	DMA channel 14 transfer completion (or count match interrupt INTCT14[C])	DMA	235	$0F20_H$	$0F20_H$
235	ICDMA15	$61D6_H$	INTDMA15	DMA channel 15 transfer completion (or count match interrupt INTCT15[C])	DMA	236	$0F39_H$	$0F39_H$
236	ICLMA6IS	$61D8_H$	INTLMA6IS	Status interrupt	LMA6	237	$0F40_H$	$0F40_H$
237	ICLMA6IR	$61DA_H$	INTLMA6IR	Reception completion interrupt	LMA6	238	$0F50_H$	$0F50_H$
238	ICLMA6IT	$61DC_H$	INTLMA6IT	Transmission interrupt	LMA6	239	$0F60_H$	$0F60_H$
239	ICLMA7IS	$61DE_H$	INTLMA7IS	Status interrupt	LMA7	240	$0F70_H$	$0F70_H$
240	ICLMA7IR	$61F0_H$	INTLMA7IR	Reception completion interrupt	LMA7	241	$0F80_H$	$0F80_H$
241	ICLMA7IT	$61F2_H$	INTLMA7IT	Transmission interrupt	LMA7	242	$0F90_H$	$0F90_H$
242	ICLMA8IS	$61F4_H$	INTLMA8IS	Status interrupt	LMA8	243	$0FA0_H$	$0FA0_H$
243	ICLMA8IR	$61F6_H$	INTLMA8IR	Reception completion interrupt	LMA8	244	$0FB0_H$	$0FB0_H$
244	ICLMA8IT	$61F8_H$	INTLMA8IT	Transmission interrupt	LMA8	245	$0FC0_H$	$0FC0_H$
245	ICLMA9IS	$61FA_H$	INTLMA9IS	Status interrupt	LMA9	246	$0FD9_H$	$0FD9_H$
246	ICLMA9IR	$61FC_H$	INTLMA9IR	Reception completion interrupt	LMA9	247	$0FE0_H$	$0FE0_H$
247	ICLMA9IT	$61FE_H$	INTLMA9IT	Transmission interrupt	LMA9	248	$0FF0_H$	$0FF0_H$
248	ICLMA10IS	6100_H	INTLMA10IS	Status interrupt	LMA10	249	1000_H	1000_H
249	ICLMA10IR	6102_H	INTLMA10IR	Reception completion interrupt	LMA10	250	1010_H	1010_H
250	ICLMA10IT	6104_H	INTLMA10IT	Transmission interrupt	LMA10	251	1020_H	1020_H
251	ICLMA11IS	6106_H	INTLMA11IS	Status interrupt	LMA11	252	1030_H	1030_H
252	ICLMA11IR	6108_H	INTLMA11IR	Reception completion interrupt	LMA11	253	1040_H	1040_H
253	ICLMA11IT	$610A_H$	INTLMA11IT	Transmission interrupt	LMA11	254	1050_H	1050_H

Table 5-13 V850E2K/FL4-H EI level maskable interrupt requests (9/9)

Interrupt			Interrupt request			Default priority	Exception code	Handler address offset
Channel	Control register		Name	Cause	Unit			
	Name	Address FFFF...						
254	–	$610C_H$	reserved			255	1060_H	1060_H
255	–	$60E_H$	reserved			256	1070_H	1070_H

a) Selection of the DTS interrupts is achieved by setting the DTS path through mode, refer to *5.3.3 “V850E2/Fx4-H EI level maskable interrupt sharing” on page 292*

b) Selection of the shared TAPA interrupts is achieved by the TAPAINTSL0 register, refer to section *5.3.4 “V850E2/Fx4-H TAPA EI level maskable interrupt sharing” on page 295* .

c) Selection of the DMA interrupts is achieved by setting the DMA interrupt selection register refer to *5.3.5 “V850E2/Fx4-H DMA interrupt selection”*.

d) These interrupts are shared by several interrupt sources via a logical or-function, refer to *5.3.3 “V850E2/Fx4-H EI level maskable interrupt sharing” on page 292* .

5.3.2 V850E2/Fx4-H FE level non-maskable interrupt sharing

The FE level non-maskable interrupt FENMI is shared among several interrupt sources.

(1) FENMIF - FENMI factor register

This register contains information about which interrupt has generated the FE level non-maskable interrupt FENMI.

Access This register can be read in 32-bit units.

Address FF45 0000$_H$

Initial Value 0000 0000$_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	WDTA1 NMIF	WDTA0 NMIF	NMI0F
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 5-14 FENMIF register contents

Bit position	Bit name	Function
2	WDTA1NMIF	Watchdog Timer WDTA1NMIF flag 0: no WDTA1NMIF occured 1: WDTA1NMIF has occured
1	WDTA0NMIF	Watchdog Timer WDTA0NMIF flag 0: no WDTA0NMIF occured 1: WDTA0NMIF has occured
0	NMI0F	Port interrupt NMI0 flag 0: no NMI0 occured 1: NMI0 has occured

(2) FENMIFC - FENMI factor clear register

This register clears the FE level non-maskable flags of the FENMIF register.

Access This register can be written in 32-bit units.

Address FF45 0008$_H$

Initial Value Reading this registers returns always 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	WDTA1 NMIFC	WDTA0 NMIFC	NMI0FC
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 5-15 FENMIFC register contents

Bit position	Bit name	Function
2	WDTA1NMIFC	Watchdog Timer WDTA1NMIF flag clear 0: no function 1: clear FENMIF.WDTA1NMIF
1	WDTA0NMIFC	Watchdog Timer WDTA0NMIF flag clear 0: no function 1: clear FENMIF.WDTA0NMIF
0	NMI0FC	Port interrupt NMI0 flag clear 0: no function 1: clear FENMIF.NMI0F

5.3.3 V850E2/Fx4-H EI level maskable interrupt sharing

Several EI level maskable interrupts EIINTn are shared among different interrupt sources.

- “or-ed” interrupts: several interrupts share the same EI interrupt request by a logical “or” combination
- DTS interrupt path through: DTS interrupt requests can be applied to an EI level maskable interrupt by setting the respective pass through selection bit.

(1) “Or-ed” interrupts

Following interrupts are shared by several sources, which combined with a logical or-function. Thus any of the “or-ed” interrupts can request an interrupt.

Table 5-16 “Or-ed” interrupts

Interrupt	Default priority	Shared interrupts
EIINT160	161	INTFCN2ERR or INTETHA0SCTXTCH
EIINT161	162	INTFCN2REC or INTETHA0SCRXTCH

(2) DTS path through interrupts

For selecting INTDTSx as an interrupt source, the respective PTHMDx in the PTHMD0LL/LH/HL/HH registers has to be set to “1”.

The table below lists all of the above shared interrupt types.

Table 5-17 DTS path through interrupts (1/3)

Interrupt	Default priority	PTHMD bit		Interrupt source
EIINT9	10	PTHMD0LL.PTHMD00	0:	INTP0
			1:	INTDTS0
EIINT10	11	PTHMD0LL.PTHMD01	0:	INTP1
			1:	INTDTS1
EIINT11	12	PTHMD0LL.PTHMD02	0:	INTP2
			1:	INTDTS2
EIINT36	37	PTHMD0LL.PTHMD03	0:	INTTAUB1I0
			1:	INTDTS3
EIINT37	38	PTHMD0LL.PTHMD04	0:	INTTAUB1I1
			1:	INTDTS4
EIINT52	53	PTHMD0LL.PTHMD05	0:	INTTAUB2I0
			1:	INTDTS5
EIINT53	54	PTHMD0LL.PTHMD06	0:	NTTAUB2I1
			1:	INTDTS6
EIINT101	102	PTHMD0LL.PTHMD07	0:	INTADCA0I0
			1:	INTDTS7
EIINT102	103	PTHMD0LH.PTHMD08	0:	INTADCA0I1
			1:	INTDTS8

Table 5-17 DTS path through interrupts (2/3)

Interrupt	Default priority	PTHMD bit		Interrupt source
EIINT143	144	PTHMD0LH.PTHMD09	0:	INTADCA1I0
			1:	INTDTS9
EIINT144	145	PTHMD0LH.PTHMD10	0:	INTADCA1I1
			1:	INTDTS10
EIINT145	146	PTHMD0LH.PTHMD11	0:	INTADCA1I2
			1:	INTDTS11
EIINT163	164	PTHMD0LH.PTHMD12	0:	INTCSIH0IC
			1:	INTDTS12
EIINT164	165	PTHMD0LH.PTHMD13	0:	INTCSIH0IJC
			1:	INTDTS13
EIINT169	170	PTHMD0LH.PTHMD14	0:	INTCSIH0IR
			1:	INTDTS14
EIINT174	175	PTHMD0LH.PTHMD15	0:	INTDCN0REC
			1:	INTDTS15
EIINT176	177	PTHMD0HL.PTHMD16	0:	INTFLXA0I0
			1:	INTDTS16
EIINT177	178	PTHMD0HL.PTHMD17	0:	INTFLXA0I1
			1:	INTDTS17
EIINT181	182	PTHMD0HL.PTHMD18	0:	INTCSIH1IR
			1:	INTDTS18
EIINT182	183	PTHMD0HL.PTHMD19	0:	INTCSIH1IC
			1:	INTDTS19
EIINT183	184	PTHMD0HL.PTHMD20	0:	INTCSIH1IJC
			1:	INTDTS20
EIINT204	204	PTHMD0HL.PTHMD21	0:	INTCSIH2IR
			1:	INTDTS21
EIINT204	205	PTHMD0HL.PTHMD22	0:	INTCSIH2IC
			1:	INTDTS22
EIINT205	206	PTHMD0HL.PTHMD23	0:	INTCSIH2IJC
			1:	INTDTS23
EIINT206	207	PTHMD0HH.PTHMD24	0:	–
			1:	INTDTS24
EIINT207	208	PTHMD0HH.PTHMD25	0:	–
			1:	INTDTS25
EIINT213	214	PTHMD0HH.PTHMD26	0:	INTETHA0SRX
			1:	INTDTS26
EIINT214	215	PTHMD0HH.PTHMD27	0:	INTETHA0SCRX
			1:	INTDTS27
EIINT215	216	PTHMD0HH.PTHMD28	0:	INTETHA0SCTX
			1:	INTDTS28

Table 5-17 DTS path through interrupts (3/3)

Interrupt	Default priority	PTHMD bit		Interrupt source
EIINT216	217	PTHMD0HH.PTHMD29	0:	INTETHA0RS
			1:	INTDTS29
EIINT217	218	PTHMD0HH.PTHMD30	0:	INTETHA0TS
			1:	INTDTS30
EIINT218	219	PTHMD0HH.PTHMD31	0:	INTETHA0FS
			1:	INTDTS31

(3) PTHMD0 - Interrupt path through mode registers

These registers enable to pass several DTS interrupt requests through to Interrupt Controller interrupt requests.

Access These registers can be read/written in 8-bit units.

Address PTHMD0LL: FF77 0400_H; PTHMD0LH: FF77 0404_H
PTHMD0HL: FF77 0408_H; PTHMD0HH: FF77 $040C_H$

nitial Value 00_H. These registers are initialized by any reset.

PTHMD0HH:

7	6	5	4	3	2	1	0
PTHMD31	PTHMD30	PTHMD29	PTHMD28	PTHMD27	PTHMD26	PTHMD25	PTHMD24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PTHMD0HL:

7	6	5	4	3	2	1	0
PTHMD23	PTHMD22	PTHMD21	PTHMD20	PTHMD19	PTHMD18	PTHMD17	PTHMD16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PTHMD0LH:

7	6	5	4	3	2	1	0
PTHMD15	PTHMD14	PTHMD13	PTHMD12	PTHMD11	PTHMD10	PTHMD09	PTHMD08
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PTHMD0LL:

7	6	5	4	3	2	1	0
PTHMD07	PTHMD06	PTHMD05	PTHMD04	PTHMD03	PTHMD02	PTHMD01	PTHMD00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-18 PTHMD0 registers contents

Bit position	Bit name	Function
all	PTHMDx	DTS path through mode selection 0: DTS interrupt request is not selected 1: DTS interrupt request is selected as interrupt request

5.3.4 V850E2/Fx4-H TAPA EI level maskable interrupt sharing

Several EI level maskable interrupts EIINTn are shared among different interrupt sources. The interrupt is selected by the TAPA interrupt selection register TAPAINTSL0.

The table below lists all of the above shared interrupt types.

Table 5-19 TAPA EI level maskable interrupt sharing

Interrupt	Default priority	TAPAINTSL0 bit		Interrupt source
EIINT22	21	SLTA0PEK0	0:	INTTAUA0I0
			1:	INTTAPA0PEK0
EIINT23	22	SLTA0PEK1	0:	INTTAUA0I1
			1:	INTTAPA0PEK1
EIINT26	25	SLTA0VLY0	0:	INTTAUA0I4
			1:	INTTAPA0VLY0
EIINT27	26	SLTA0VLY1	0:	INTTAUA0I5
			1:	INTTAPA0VLY1

(1) TAPAINTSL0 - TAPA interrupt selector register

This register selects the TAPA interrupt requests.

Access This register can be read/written in 8-bit units.

Address FF77 0410$_H$

Initial Value 00$_H$. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	SLTA0VLY1	SLTA0VLY0	SLTA0PEK1	SLTA0PEK0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-20 TAPAINTSEL0 register contents

Bit position	Bit name	Function
3	SLTA0VLY1	INTTAPA0VLY1 selection 0: INTTAUA0I5 1: INTTAPA0VLY1
2	SLTA0VLY0	INTTAPA0VLY0 selection 0: INTTAUA0I4 1: INTTAPA0VLY0
1	SLTA0PEK1	INTTAPA0PEK1 selection 0: INTTAUA0I1 1: INTTAPA0PEK1
0	SLTA0PEK0	INTTAPA0PEK0 selection 0: INTTAUA0I0 1: INTTAPA0PEK0

5.3.5 V850E2/Fx4-H DMA interrupt selection

The DMA Controller generates two different interrupts for each DMA channel m:

- DMA channel m transfer completion interrupt INTDMAm
- DMA channel m count match interrupt INTCTm

By use of the DMA interrupt selection registers DMAINTSL0 and DMAINTSL0 one of the above interrupt can be selected as an interrupt request, separate for each DMA channel.

(1) DMAINTSL0 - DMA interrupt selection register 0

This register selects the DMA interrupt to be used as interrupt request for DMA channels 0 to 7.

Access This register can be read/written in 8-bit units.

Address FF77 0414$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
SLINTCT7	SLINTCT6	SLINTCT5	SLINTCT4	SLINTCT3	SLINTCT2	SLINTCT1	SLINTCT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-21 DMAINTSL0 register contents

Bit position	Bit name	Function
7 to 0	SLINTCTm	DMA channel m (m = 0 to 7) interrupt request 0: DMA channel m completion INTDMAm is used as interrupt request 1: DMA channel m count match INTCTm is used as interrupt request

(2) DMAINTSL1 - DMA interrupt selection register 1

This register selects the DMA interrupt to be used as interrupt request for DMA channels 8 to 15.

Access This register can be read/written in 8-bit units.

Address FF77 0418$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
SLINTCT15	SLINTCT14	SLINTCT13	SLINTCT12	SLINTCT11	SLINTCT10	SLINTCT8	SLINTCT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-22 DMAINTSL1 register contents

Bit position	Bit name	Function
7 to 0	SLINTCTm	DMA channel m (m = 8 to 15) interrupt request 0: DMA channel m completion INTDMAm is used as interrupt request 1: DMA channel m count match INTCTm is used as interrupt request

5.4 External Interrupts

5.4.1 Edge detection configuration

The external interrupts can be configured to generate an interrupt request upon a rising or falling edge or upon both edges of the external pin.

Following registers are used to specify the edge:

Table 5-23 External interrupts edge detection configuration registers

Interrupt	Register
INTP0	FCLA0CTL0
INTP1	FCLA0CTL1
INTP2	FCLA0CTL2
INTP3	FCLA0CTL3
INTP4	FCLA0CTL4
INTP5	FCLA0CTL5
INTP6	FCLA0CTL6
INTP7	FCLA0CTL7
INTP8	FCLA1CTL0
INTP9	FCLA1CTL1
INTP10	FCLA1CTL2
INTP11	FCLA1CTL3
INTP12	FCLA1CTL4
INTP13	FCLA1CTL5
INTP14	FCLA1CTL6
INTP15	FCLA1CTL7
NMI	FCLA2CTL0

Refer to the section *"Port filters assignment"* in the chapter *"Port Functions"* for details of these registers.

5.4.2 External interrupts as trigger and wake-up signals

The external interrupts can be used as

- DMA trigger factors
- DTS trigger factors
- A/D Converter H/W triggers
- stand-by wake-up factors

Table 5-24 External interrupts as triggers and wake-up

Interrupt	DMA trigger factor[a]	DTS channel[a]	ADCAn H/W trigger[b]	Stand-by wake-up factor[c]
INTP0	1	0	ADCA0TTIN201 ADCA1TTIN201	WUFLm07
INTP1	2	1	ADCA0TTIN101 ADCA1TTIN101	WUFLm08
INTP2	3	2	ADCA0TTIN001 ADCA1TTIN001	WUFLm09
INTP3	4	32	ADCA0TTIN202 ADCA1TTIN202	WUFLm10
INTP4	5	33	ADCA0TTIN102 ADCA1TTIN102	WUFLm11
INTP5	6	34	ADCA0TTIN002 ADCA1TTIN002	WUFLm12
INTP6	7	35	–	WUFLm13
INTP7	8	36	–	WUFLm14
INTP8	9	37	–	WUFLm15
INTP9	10	38	–	WUFLm16
INTP10	11	39	–	WUFLm17
INTP11	–	–	–	WUFLm18
INTP12	–	–	–	WUFLm19
INTP13	–	–	–	WUFLm20
INTP14	–	–	–	WUFLm21
INTP15	–	–	–	WUFLm22
NMI	–	–	–	WUFLm00

a) Refer to the chapter *"DMA/DTS Controller (DMAC)"* for details.
b) Refer to the chapter *"A/D Converter A (ADCA)"* for details.
c) The stand-by wake-up factors are indicated by their respective bit in the wake-up factor register WUFLm. refer to the chapter *"Stand-by Controller (STBC)"* for details.

Stand-by wake-up

To use an external interrupt as a wake-up factor from DEEPSTOP mode, its analog filter must be set to edge detection mode (FCLAnINTm = 0).
Refer to the section *"Port Filters Functional Description"* in the *"Port Functions"* chapter for details.

5.5 Interrupt Controller Control Registers

(1) ICn - EI level interrupt control registers (n = 0 to 255)

These registers, each of which is for a channel n of EI level maskable interrupt INTn, are used to set a condition to control each channel.

Caution Do not access ICn registers of interrupt channels, not listed in above interrupt request tables.

Access These registers can be read/written in

- 16-bit units
  – via 16-bit registers ICn
- 8-bit or 1-bit units
  – bits 7 to 0 via 8-bit registers ICnL
  – bits 15 to 8 via 8-bit registers ICnH.

Address ICn: FFFF 6000_H + 2n
ICnL: FFFF 6000_H + 2n, ICnH: FFFF 6001_H + 2n

Initial Value $008F_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	RFn	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
MKn	0	0	0	P[3:0]n			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-25 ICn register contents (1/2)

Bit position	Bit name	Function
12	RFn	Interrupt request flag RFn is set upon occurrence of an EI level maskable interrupt INTn. RFn can also be set by the application program, which generates also an EI level maskable interrupt. 0: no interrupt request 1: interrupt request RFn is reset if the interrupt is acknowledged, i.e. the interrupt service routine is started. RFn can also be reset by the application program.

Table 5-25 ICn register contents (2/2)

Bit position	Bit name	Function
7	MKn	Interrupt mask bit When the MKn bit is set, the interrupt request set to the interrupt request flag RFn is masked, so that the interrupt request is not issued from that channel to the CPU. From a channel with the MKn bit set, interrupt pending status is not displayed by the ICSR.PMF bit. The MKn bit does not mask a signal input from an interrupt input itself and, therefore, the corresponding interrupt request flag is set even when the MKn bit is set. The setting of the corresponding bit of the interrupt mask register IMR is also reflected. 0: enables interrupt servicing 1: disables interrupt servicing
3 to 0	P[3:0]n	These bits specify 16 levels of interrupt priorities. The highest priority is 0 and the lowest is 15. If two or more interrupt requests of EI level are generated at the same time, the interrupt source having the higher priority specified by these bits is selected and reported to the CPU. If the priority specified by the P[3:0]n bits is the same, the source having the lower channel number is selected by a fixed priority.

(2) IMRm - EI level interrupt mask registers (m = 0 to 15)

These registers are a collection of the MKn bits of the ICn registers. Each bit of IMRm reflects the setting of the corresponding ICn.MKn bit. Setting IMRm is reflected in the corresponding MKn bit.

Caution MKn bits for interrupt channels, not listed in above interrupt request tables, must be set to 1.

Access These registers can be read/written in:

- 16-bit units
  – via 16-bit registers IMRm
- 8-bit or 1-bit units
  – bits 7 to 0 via 8-bit registers IMRmL
  – bits 15 to 8 via 8-bit registers IMRmH.

Address

IMR0: FFFF 6400_H	IMR1: FFFF 6402_H
IMR2: FFFF 6404_H	IMR3: FFFF 6406_H
IMR4: FFFF 6408_H	IMR5: FFFF $640A_H$
IMR6: FFFF $640C_H$	IMR7: FFFF $640E_H$
IMR8: FFFF 6410_H	IMR9: FFFF 6412_H
IMR10: FFFF 6414_H	IMR11: FFFF 6416_H
IMR12: FFFF 6418_H	IMR13: FFFF $641A_H$
IMR14: FFFF $641C_H$	IMR15: FFFF $641E_H$
IMRmL: address of IMRm	IMRmH: address of IMRm + 1

Initial Value $FFFF_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
IMRmMK m x 16 + 15	IMRmMK m x 16 + 14	IMRmMK m x 16 + 13	IMRmMK m x 16 + 12	IMRmMK m x 16 + 11	IMRmMK m x 16 + 10	IMRmMK m x 16 + 9	IMRmMK m x 16 + 8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
IMRmMK m x 16 + 7	IMRmMK m x 16 + 6	IMRmMK m x 16 + 5	IMRmMK m x 16 + 4	IMRmMK m x 16 + 3	IMRmMK m x 16 + 2	IMRmMK m x 16 + 1	IMRmMK m x 16 + 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-26 IMRm registers contents

Bit position	Bit name	Function
15 to 0	IMRmMK255 to IMRmMK0	These bits mask an interrupt from channels n = 0 to 255 of EI level maskable interrupt INTn. 0: interrupt request issued 1: interrupt request not issued

(3) ISPR - In-service priority register

This register holds the interrupt priority of an EI level maskable interrupt INTn that is being processed by the CPU.

ISPR[15:0] setting

When the CPU enters an interrupt service routine, i.e. acknowledges the interrupt, with interrupt request priority ICn.P[3:0]n = 0 to 15, the bit ISPR[15:0] of this register, corresponding to this priority, is set.

ISPR[15:0] clearing

When the CPU has completed the interrupt service routine by the EIRET instruction, the ISPR[15:0] bit, that was set upon entering this interrupt service routine, is cleared.
Since the interrupt service routine with the highest priority is always completed first, the cleared bit is always the one, that corresponds to the highest priority.

Nested interrupt services

If multiple EI level maskable interrupts with different priorities ICn.P[3:0]n have been acknowledged by the CPU (lower prioritized interrupt service routines have been interrupted by higher prioritized), the corresponding ISPR[15:0] bits of all interrupts in service are sequentially set.

ISPR[15:0] clearing by software

Clearing this register is protected by a special sequence, involving the in-service priority clear register ISPC register.
All ISPR[15:0] bits can be cleared only all together by use of ISPC. For that purpose ISPC has to be written with $FFFF_H$, followed by writing 0000_H to ISPR.
Note that it is not possible to clear single ISPR[15:0] bits.
Refer also to *5 “ISPC - In-service priority clear register” on page 304* .

Access

This register can be accessed as follows:

- writing 0000_H via the 16-bit register ISPR for clearing all ISPR[15:0], after ISPC has been written with $FFFF_H$
- reading in 8-bit or 16-bit units
  – ISPR[15:0] via 16-bit register ISPR
  – ISPR[7:0] via 8-bit register ISPRL
  – ISPR[15:8] via 8-bit register ISPRH

Address

ISPR: $FFFF\ 6440_H$
ISPRL: $FFFF\ 6440_H$, ISPRH: $FFFF\ 6441_H$

Initial Value

0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
ISPR15	ISPR14	ISPR13	ISPR12	ISPR11	ISPR10	ISPR9	ISPR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
ISPR7	ISPR6	ISPR5	ISPR4	ISPR3	ISPR2	ISPR1	ISPR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-27 ISPR register contents

Bit position	Bit name	Function
15 to 0	ISPR15 to ISPR0	These bits indicate the priority of the interrupt being acknowledged. 0: Interrupt request of the priority corresponding to the bit position is not acknowledged. 1: Interrupt request of the priority corresponding to the bit position is being processed by the CPU.

(4) PMR - Priority mask register

This register specifies an interrupt priority by which an interrupt request flag of EI level maskable interrupt INTn is to be masked. It disables all at once the interrupt requests from the INTn channel for which the interrupt priority specified by this register is set.

The position of each bit of this register corresponds to an interrupt priority. For example, if 1 is set to bit 0, channel of interrupt priority 0 can be masked.

Access This register can be read/written in

- in 16-bit units
  – PMR[15:0] via 16-bit register PMR
- in 8-bit or 1-bit units
  – PMR[7:0] via 8-bit register PMRL
  – PMR[15:8] via 8-bit register PMRH.

Address PMR: FFFF 6448$_H$
PMRL: FFFF 6448$_H$, PMRH: FFFF 6449$_H$

Access This register can be read/written in 16-bit or 8-bit or 1-bit units.

Address FFFF 6448$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
PMR15	PMR14	PMR13	PMR12	PMR11	PMR10	PMR9	PMR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PMR7	PMR6	PMR5	PMR4	PMR3	PMR2	PMR1	PMR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-28 PMR register contents

Bit position	Bit name	Function
15 to 0	PMR15 to PMR0	These bits specify an interrupt priority by which an interrupt request flag 0: Enables interrupt servicing of the priority corresponding to a specified bit position. 1: Disables interrupt servicing of the priority corresponding to a specified bit position.

(5) ISPC - In-service priority clear register

This register prepares to clear the in-service priority register ISPR.

For clearing ISPR proceed as follows:

1. write $FFFF_H$ to ISPC.ISCPC[15:0]
2. write 0000_H to ISPR.ISPR[15:0]

As a result, the internal mode registers of the Interrupt Controller for interrupt servicing, which indicate that an interrupt request is being processed by the CPU, are cleared.

At the same time, all the processing modes of FE level NMI, FE level maskable interrupt FEINT, and EI level maskable interrupts INTn of the ICSR register are canceled.

After writing $FFFF_H$ to ISPC[15:0], $FFFF_H$ is also read back until ISPR has been cleared by writing ISPR.ISPR[15:0] = 0000_H. Afterwards ISPC is automatically cleared to 0000_H.
Before writing ISPR.ISPR[15:0] = 0000_H, ISPC can be reset by writing 0000_H, and thus disabling any ISPR clear attempt.

ISPC can only be written with $FFFF_H$ or 0000_H. Writing any other value does not change the register content.

Refer also to *3 “ISPR - In-service priority register” on page 302* .

Caution Writing $FFFF_H$ to ISPC and 0000_H to ISPR in order to clear ISPR does not need to maintain any time relation to each other. In particular this means, that any kind of other processes may take place beween both write accesses.

Access This register can be read/written in 16-bit units.

Address FFFF 6450_H

Initial Value 0000_H. This register is initialized by any reset.

15	14	13	12	11	10	9	8
ISPC[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
ISPC[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 5-29 ISPC register contents

Bit position	Bit name	Function
15 to 0	ISPC[15:0]	ISPR clear preparation • writing $FFFF_H$: enables ISPR to be cleared by ISPR.ISPR[15:0] = 0000_H • writing 0000_H: disables ISPR to be cleared If ISPR has been cleared after ISPC.ISPC[15:0] = $FFFF_H$, ISPC returns to 0000_H automatically.

(6) SCR - Selected channel hold register

This register holds the channel number n of the EI level maskable interrupt INTn, acknowledged by the CPU.

Access This register can be read only in

- in 16-bit units:
  – via 16-bit register SCR
- in 8-bit units:
  – SCR[7:0] via 8-bit register SCRL.

Address SCR: FFFF 6458$_H$
SCRL: FFFF 6458$_H$

Access This register can be read only in 16-bit or 8-bit units.

Address FFFF 6458$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
SCR[7:0]							
R	R	R	R	R	R	R	R

Table 5-30 SCR register contents

Bit position	Bit name	Function
7 to 0	SCR[7:0]	Holds the channel number n of the maskable interrupt that has been acknowledged by the CPU. The value of these bits is updated when an interrupt vector is reported to the CPU. It is overwritten when multiple interrupts of EI level maskable interrupt (INTn) are acknowledged. These bits are not updated when an FE level interrupt is acknowledged. Writing to this register is ignored.

(7) ICSR - Interrupt controller status register

This register indicates the operation status of the Interrupt Controller. Especially the FNE, FIE and EIE of this register serve as a mode register of interrupt servicing.

Access This register can be read only in

- 16-bit units:
  – via 16-bit register ICSR
- 8-bit or 1-bit units:
  – bits 7 to 0 via 8-bit register ICSRL
  – bits 15 to 8 via 8-bit register ICSRL

Address ICSR: FFFF 645A$_H$
ICSRL: FFFF 645A$_H$, ICSRH: FFFF 645B$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	PMF
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	FNR	FIR	EIR	0	FNE	FIE	EIE
R	R	R	R	R	R	R	R

Table 5-31 ICSR register contents

Bit position	Bit name	Function
8	PMF	Indicates 1 if the request flag of a channel n of EI level interrupt INTn, that has the interrupt priority prohibited by the setting of PMR from being serviced, is set.
6	FNR	Indicates 1 if an FE level non-maskable interrupt FENMI has been issued to the CPU.
5	FIR	Indicates 1 if an FE level maskable interrupt FEINT has been issued to the CPU.
4	EIR	Indicates 1 if an EI level maskable interrupt INTn has been issued to the CPU.
2	FNE	Indicates 1 if the CPU is processing the FE level non-maskable interrupt FENMI.
1	FIE	Indicates 1 if the CPU is processing the FE level maskable interrupt FEINT.
0	EIE	Indicates 1 if the CPU is processing the EI level maskable interrupt INTn.

(8) FNC - FE level NMI status register

This register informs about the occurence of a FE level non-maskable interrupt FENMI.

Access This register can be read only in

- 16-bit units:
  – via 16-bit register FNC
- 8-bit or 1-bit units:
  – bits 15 to 8 via 8-bit register FNCH

Address FNC: FFFF 645C$_H$
FNCH: FFFF 645D$_H$

Initial Value 0000$_H$. This register is initialized by any reset.

15	14	13	12	11	10	9	8
0	0	0	FNRF	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 5-32 FNC register contents

Bit position	Bit name	Function
12	FNRF	FENMI interrupt request flag 0: no interrupt request FENMI 1: FENMI interrupt request occured

(9) FIC - FE level INT status register

This register informs about the occurence of a FE level maskable interrupt FEINT.

Access This register can be read only in

- 16-bit units:
  – via 16-bit register FIC
- 8-bit or 1-bit units:
  – bits 15 to 8 via 8-bit register FICH

Address FiC: FFFF 645E$_H$
FICH: FFFF 645F$_H$

Initial Value 0000$_H$. This register is initialized by any reset

15	14	13	12	11	10	9	8
0	0	0	FIRF	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

Table 5-33 FIC register contents

Bit position	Bit name	Function
12	FIRF	FEINT interrupt request flag 0: no FEINT interrupt request 1: FEINT interrupt request occured

5.6 Interrupt Acknowledgment and Restoring

This section describes the operation during interrupt acknowledgment and restoring from interrupt servicing.

CPU registers This section frequently refers to registers of the CPU. For a detailed description of these registers refer to the respective “V850E2M 32-bit Microcontroller Core Architecture” user manual.

5. 6. 1 FE level non-maskable interrupt FENMI

When an FENMI interrupt is requested, an FE level non-maskable interrupt is generated in the CPU. This FE level non-maskable interrupt is used when a fatal system error occurs.

Caution Upon acknowledgment of the FENMI interrupt, generation of the next FENMI, FEINT, or INTn interrupt is pended until the FERET instruction is executed (interrupt request is acknowledged and held.)
FENMI can be acknowledged even when the NP bit of the program status word CPU register PSW is set to 1. Therefore, if the FENMI interrupt occurs during the processing of an FEINT exception, PPI exception, or other FE level exceptions, the save address is lost and cannot be restored. After a FENMI interrupt is requested and the required processing has been completed, execute a system reset, etc. Return to the original processing is not possible.

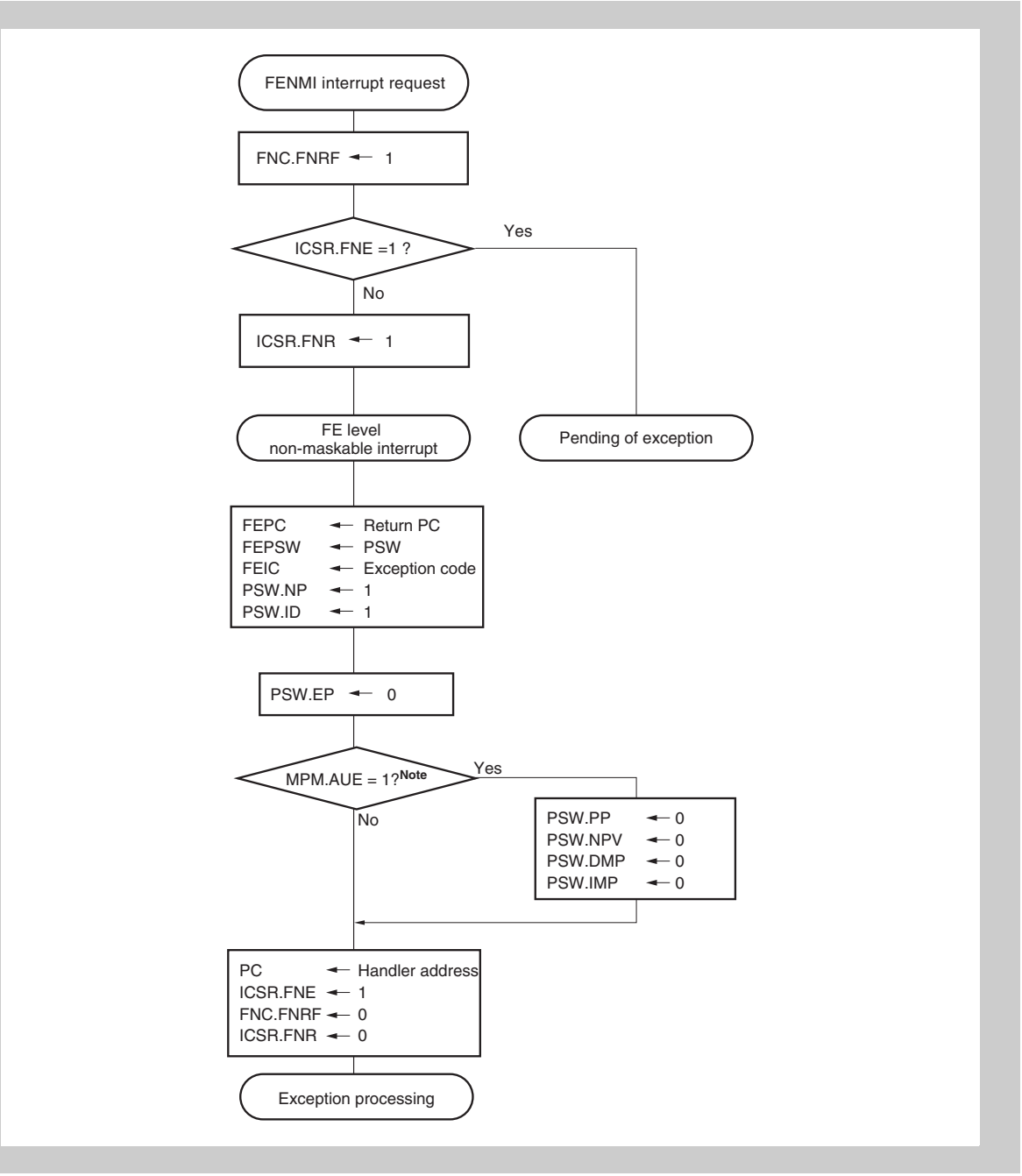


Figure 5-1 Processing upon occurrence of FENMI interrupt request

Note If a processor protection exception (MDP or MIP) occurs, the PSW.PP, NPV, DMP, and IMP bits are always cleared, regardless of the status of MPM.AUE. MPM.AUE controls the execution level auto transition function, refer to the "V850E2M 32-bit Microcontroller Core Architecture" (Document number R01US0001EJxxxx) for details.

FENMI restore An FE level non-maskable interrupt FENMI cannot be restored since such interrupt indicates a fatal system error.

Execute a system reset after exception processing.

5. 6. 2 FE level maskable interrupt FEINT

Caution A FE level maskable interrupt FEINT is recoverable.

Upon acknowledgment of the FEINT interrupt, generation of the next FEINT or INTn interrupt is kept pending until the FERET instruction is executed. Interrupt request is acknowledged and held.

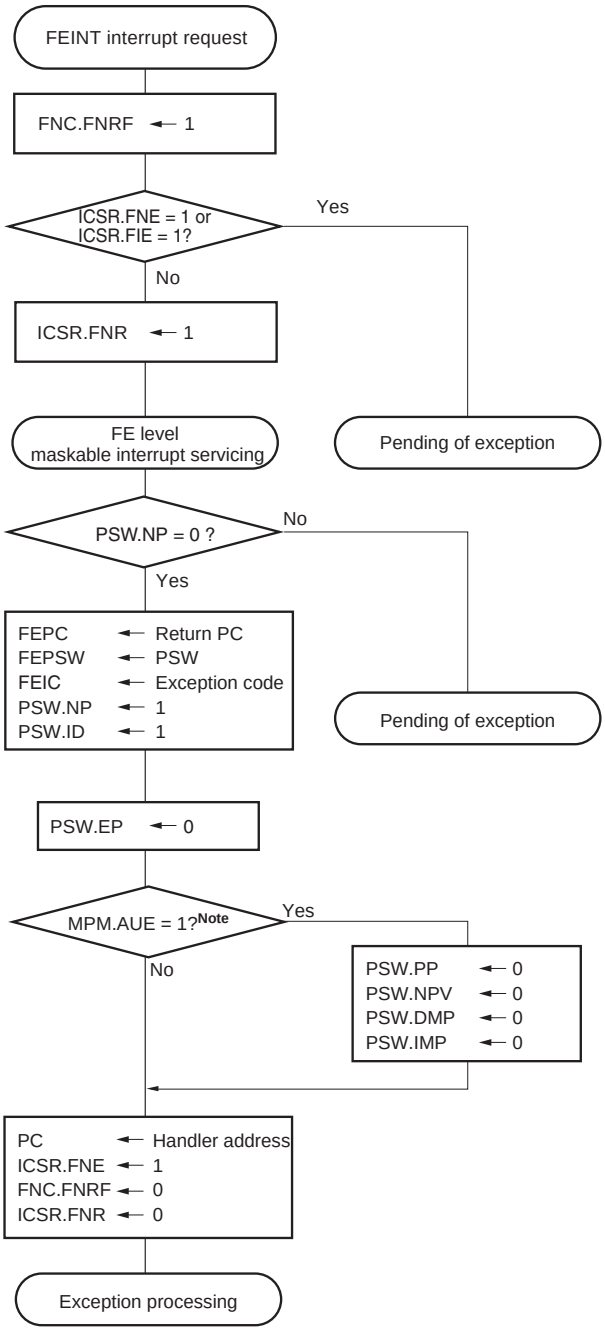


Figure 5-2 Processing upon occurrence of FEINT interrupt request

Note If a processor protection exception (PPI or TSI) occurs, the PSW.PP, NPV, DMP, and IMP bits are always cleared, regardless of the status of MPM.AUE. MPM.AUE controls the execution level auto transition function, refer to the “V850E2M 32-bit Microcontroller Core Architecture” (Document number R01US0001EJxxxx) for details.

FEINT restore Restore from FE level maskable interrupt FEINT servicing is performed using the FERET instructions. Execution of the FERET instruction while the PSW.EP bit status is cleared launches restore processing from the FE level maskable interrupt interrupt FEINT. Completely restoring from interrupt servicing when the PSW.EP bit is "1" is not possible (clearing of the ICSR, ISPR, and other registers is not performed). For return from FE level maskable interrupts, execute the FERET instruction with the PSW.EP bit always cleared.

Caution In the V850E2M CPU, although RETI instructions are provided for backward compatibility with V850E1 and V850E2 architectures, but their use is, in principle, prohibited. Replace all RETI instructions other than existing programs that cannot be modified with EIRET or FERET instructions.

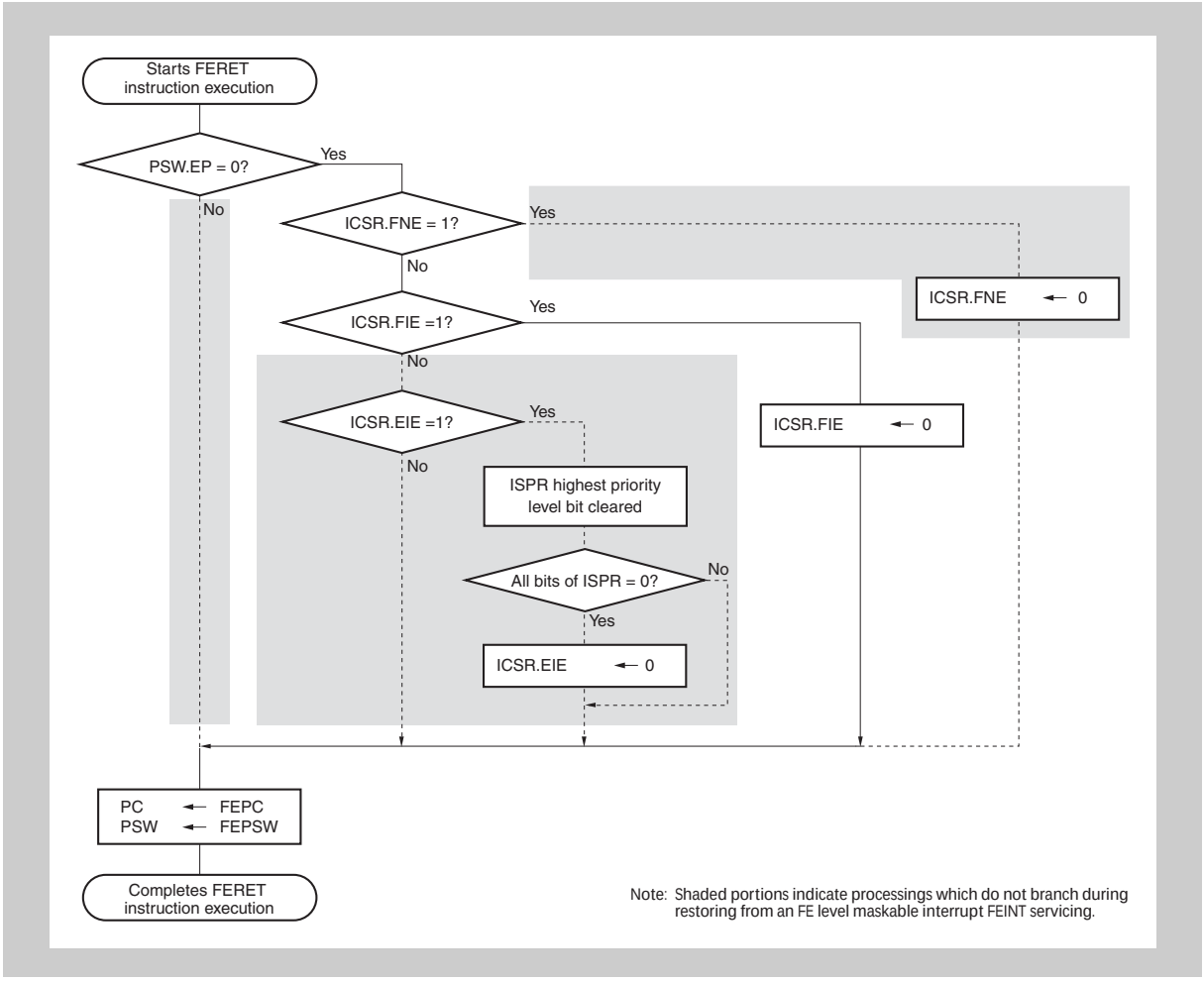


Figure 5-3 Restore from FE level maskable interrupt (FEINT) servicing

5. 6. 3 EI level maskable interrupt INTn

When an EI level maskable interrupt is requested, an INTn interrupt is requested to the CPU (the transition to the interrupt handler occurs from the setting of the IMR register of the Interrupt Controller). This interrupt is a recoverable EI level interrupt.

In the case of an INTn interrupt, its channel number n is set to the selected channel hold register SCR. As a result, the channel number can be easily known when wishing to share the same interrupt vectors among several channels.

Caution Upon acknowledgment of the EI level interrupt, the priority level of the currently acknowledged interrupt is registered to the in-service priority register ISPR. Then, until execution of the EIRET instruction, interrupt with a priority level lower than that of this ISPR register are not generated. Interrupt request is acknowledged and held.
Registration of the priority level of the currently acknowledged interrupt and deletion of the priority level of the interrupt during EIRET to/from the ISPR register are automatically performed by the hardware. Write to the ISPR register cannot be performed by software. Write operations are ignored.

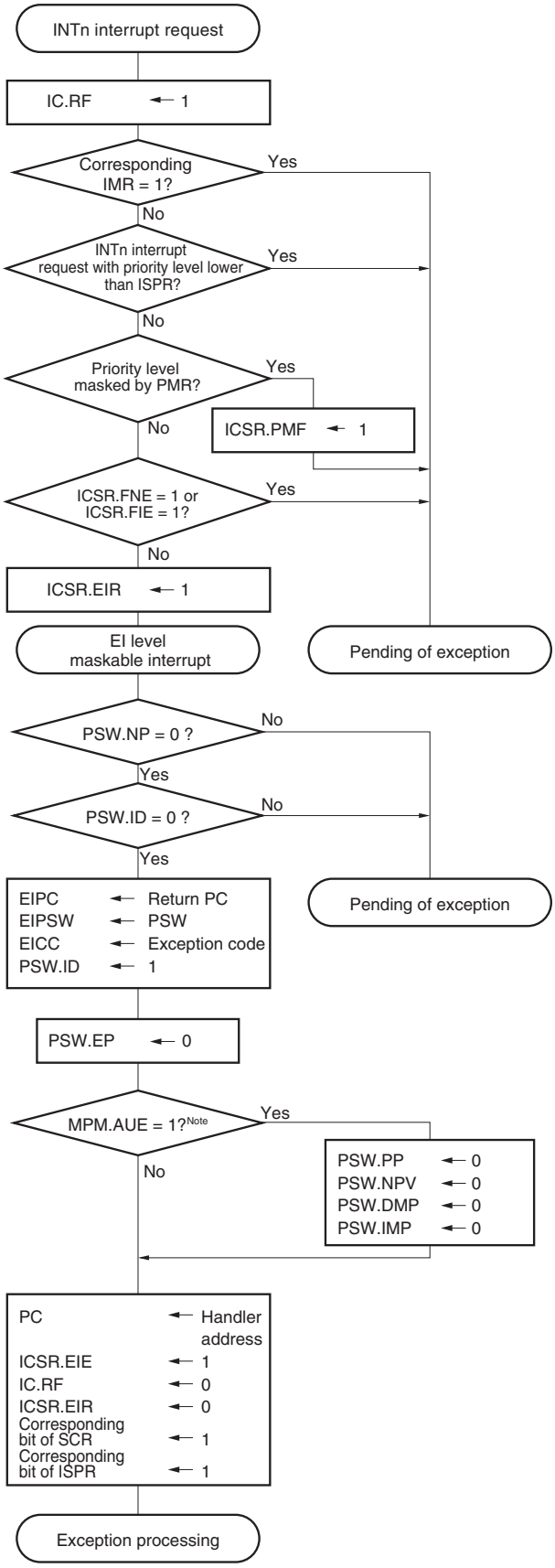


Figure 5-4 Processing upon occurrence of INTn interrupt request

Note MPM.AUE controls the execution level auto transition function, refer to the “V850E2M 32-bit Microcontroller Core Architecture” (Document number R01US0001EJxxxx) for details.

INTn restore Restore from EI level maskable interrupt INTn is performed using the EIRET instruction. Execution of the EIRET instruction while the PSW.EP bit status is cleared launches restore processing from the interrupt. Completely restoring from interrupt servicing when the PSW.EP bit is "1" is not possible (clearing of the ICSR, ISPR, and other registers is not performed). For return from EI level maskable interrupt, execute the EIRET instruction with the PSW.EP bit always cleared.

Caution In the V850E2M CPU, although RETI instructions are provided for backward compatibility with V850E1 and V850E2 architectures, but their use is, in principle, prohibited. Replace all RETI instructions other than existing programs that cannot be modified with EIRET or FERET instructions.

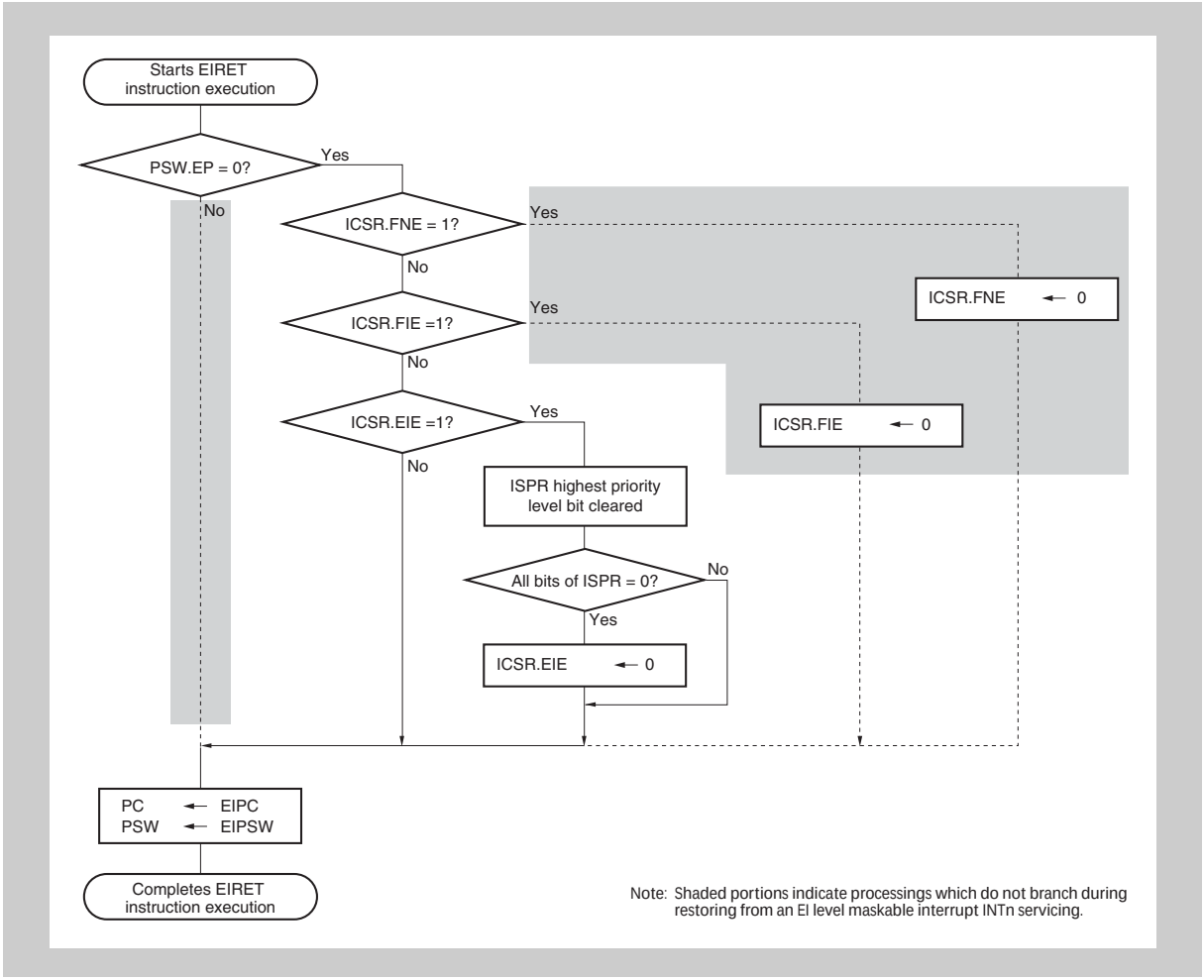


Figure 5-5 Restore from EI level maskable interrupt INTn

5.7 Interrupt Operation

5. 7. 1 Mask function of EI level maskable interrupt INTn

Interrupt masking can be specified for each respective interrupt channel of INTn. Interrupt masking is performed by the following register settings.

ICn.MKn	Operation
1	Masks interrupt
0	Enables interrupt

The ICn.MKn bits can also be read and written via the corresponding IMRmEIMKn bits of the EI level interrupt mask registers IMRm.

Operation example

3. When "1" is written to IMRm.IMRmEIMKn bits, interrupts are prohibited for the corresponding channels.
4. When the ICn.MKn bits are read, "1" is read.

Caution For MKn bit, the processing after interrupt hold is masked. Even if the MKn bit is set to 1, interrupt request acknowledgment and hold are performed. Therefore, even if software interrupts are requested for interrupts for which interrupt prohibit has been specified with MKn, no interrupt occurs. Moreover, when MKn bit is again set to 0 while an interrupt request is held, that interrupts occur at that timing. To delete an interrupt request that is already being held, clear the corresponding RFn bit.

5. 7. 2 Interrupt priority level judgment

When FENMI, FEINT, and INTn interrupts are input, priorities including other exceptions are determined, and the exception with the highest priority (including interrupts) is requested. Exceptions requested at the same time (including interrupts) are processed in a pre-allocated priority order (the default priority order). The priority orders of FENMI, FEINT, and INTn interrupts are as follows.

FENMI > FEINT > INTn

See *"V850E2/Fx4-H Interrupt Requests"* and the *"V850E2M 32-bit Microcontroller Core Architecture User Manual"* for other exceptions.

For INTn interrupts, the priority level can be set independently for each interrupt source. The priority level is specified with ICn.P[3:0]n. Priority levels from 0 to 15 can be set. 0 is the highest priority level, and 15 the lowest.
In the case of multiple INTn interrupts with the same priority level, the interrupt with the lowest interrupt channel number, i.e. with the highest default priority, has priority.

Table 5-34 Example of INTn interrupt priority level settings and priority levels

INTn[a]	ICn.P[3:0] setting[b]	Priority level during operation[c]
INT0	3	10
INT1	4	11
INT2	0	1
INT3	0	2
INT4	1	3
INT5	2	6
INT6	2	7
INT7	1	4
INT8	1	5
INT9	2	8
INT10	2	9

a) n = 0: highest default priority
b) 0: highest priority
c) 1: highest priority

During interrupt servicing, the Interrupt Controller also processes multiple interrupts acknowledging other interrupts. When multiple INTn interrupts are requested at the same time, the interrupt to be acknowledged is determined by the following procedure.

(1) Comparison with the priority level as the interrupt currently being serviced

Interrupts with the same or lower priority level as the interrupt currently being serviced are held pending.

The priority level of the interrupt currently being serviced is shown in the ISPR register.

Interrupts with a higher priority level than the interrupt currently being serviced proceed to the next priority judgment stage.

(2) Masking through priority mask register (PMR)

Only interrupts enabled by the PMR register proceed to the next priority judgment stage.

(3) The requested interrupt source with the highest priority level is selected

When interrupts are being simultaneously requested from multiple sources, the interrupt source from the highest priority level, determined by ICn.P[3:0]n, with the smallest interrupt channel number, i.e. highest default priority, is selected.

(4) Interrupt hold by CPU

Interrupt acknowledgment is pending according to the state of the NP and ID bits of the PSW register.

At this time, priority judgment

- among INTn interrupts, according to their assigned ICn.P[3:0] and their default priority (interrupt channel number n)
- and priority judgment among INTn, FEINT and FENMI interrupts

is performed even while interrupt acknowledgment is pending, and the interrupt with the highest priority is selected upon realization of the acknowledgment condition.

Example When a priority level ICn.P[3:0]n = 5 INTn interrupt has already been requested and interrupt generation is pending because the value of the PSW.ID bit is "1", a subsequent priority level ICn.P[3:0]n = 3 INTn interrupt is requested. Then, if the PSW.ID bit is cleared, the priority level 3 INTn interrupt is generated.

Multiple interrupt servicing in which an interrupt is acknowledged while another interrupt is being serviced is shown below.

When an interrupt request is acknowledged, the PSW.ID flag is automatically set to 1. Therefore, when multiple interrupts are to be used, clear the ID flag to 0 beforehand (for example, by placing the EI instruction in the interrupt service program) to set the interrupt enable mode.

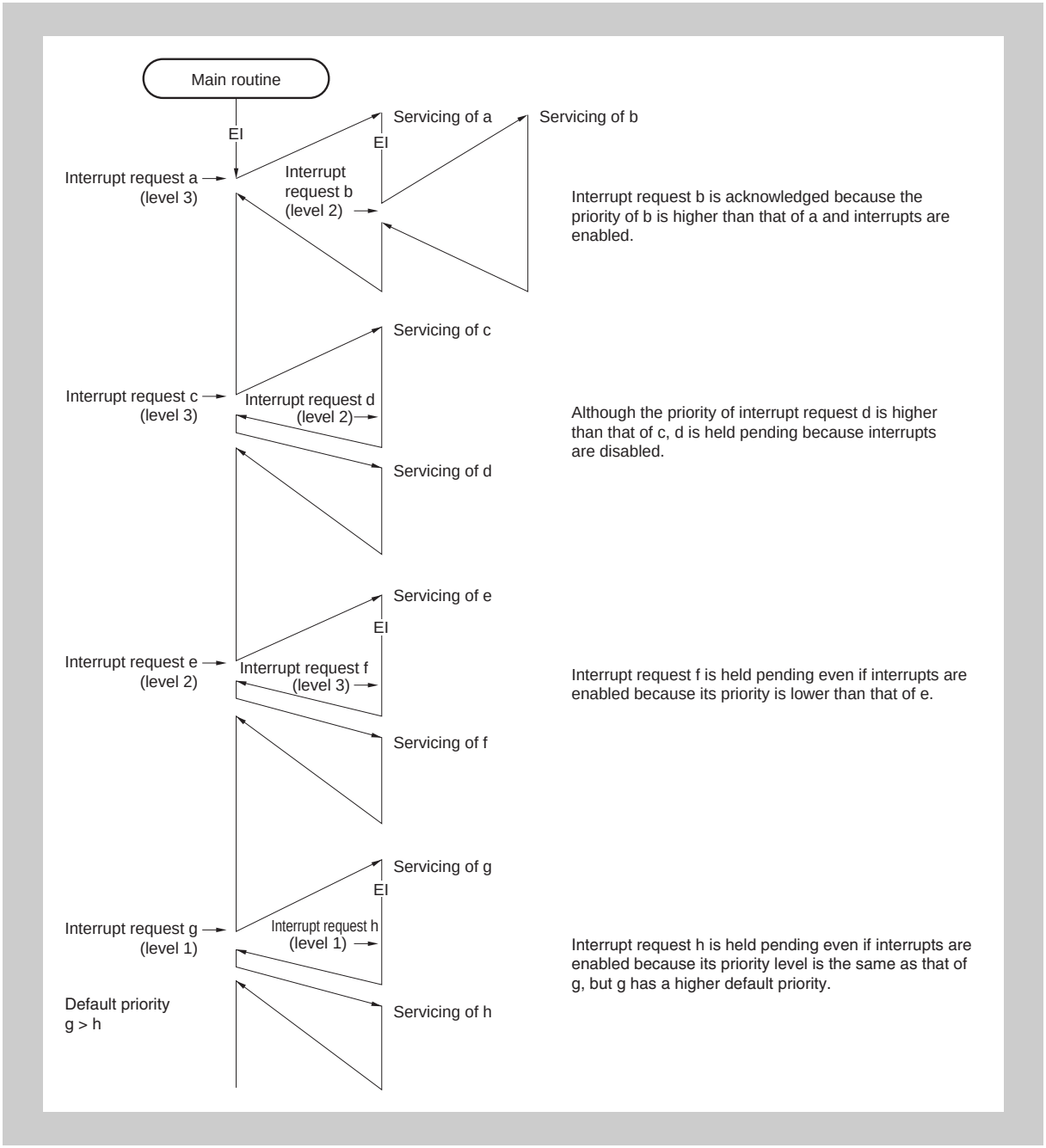


Figure 5-6 Example of processing in which an interrupt request is issued while another interrupt is being serviced (1/2)

Notes

1. "a" to "h" in the figure are the temporary names of interrupt request signals shown for the sake of explanation.
2. The default priority in the figure indicates the relative priority between two interrupt request signals.

Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

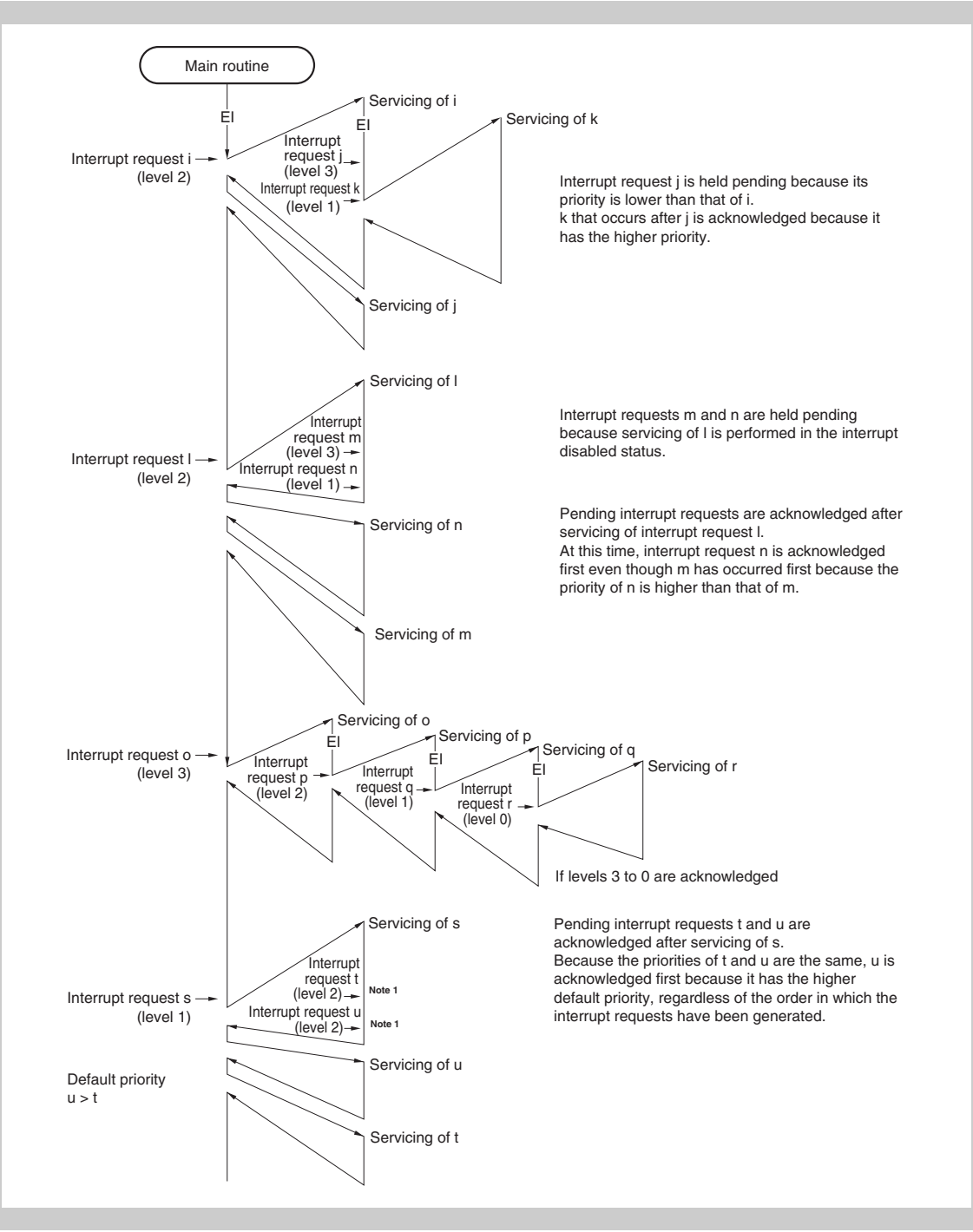


Figure 5-7 Example of processing in which an interrupt request signal is issued while another interrupt is being serviced (2/2)

Notes
1. Interrupt request “t” has lower default priority than interrupt request “u”.
2. “i” to “u” in the figure are the temporary names of interrupt request signals shown for the sake of explanation.

Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

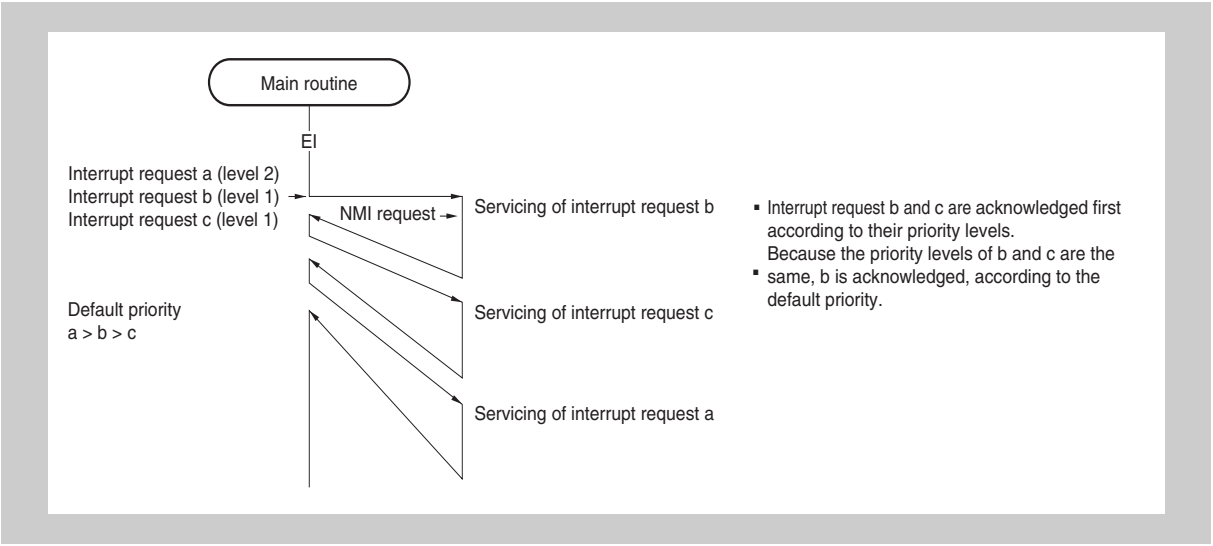


Figure 5-8 Example of servicing simultaneously generated interrupt requests

Notes
1. “a to “c” in the figure are the temporary names of interrupt request signals shown for the sake of explanation.
2. The default priority in the figure indicates the relative priority between two interrupt request signals.

Caution To perform multiple interrupt servicing, the values of the EIPC and EIPSW registers must be saved before executing the EI instruction. When returning from multiple interrupt servicing, restore the values of EIPC and EIPSW after executing the DI instruction.

5. 7. 3 Priority mask function

The priority mask function prohibits in batch INTn interrupts of the specified interrupt priority level.

The interrupt masking priority level is specified with the PMR register. Masking and acknowledgment can be set for each priority level.

The following operations are possible using this function:

- Temporary prohibition of interrupts that have a priority level that is lower than a given priority level
- Temporary prohibition of interrupts that have a given priority level

PMR.PMRm	Operation
0	Acknowledges requests from priority level m interrupt source.
1	Masks requests from priority level m interrupt source.

Note m = 0 to 15

The PMR register prohibits interrupt occurrence. Interrupt request is acknowledged and held even while the interrupt occurrence is prohibited.

The presence of INTn interrupts held pending with this function can be checked with the next section.

5. 7. 4 Pending interrupt report function

The state of the currently pending interrupt can be checked with the pending interrupt report function.

This function allows checking of the following states:

- When interrupts that are masked only by the priority mask function (PMR) exist
  The ICSR.PMF bit is set to 1.
  The ICSR.PMF bit is not set to 1 only when interrupts that are priority masked through ISPR register or interrupts masked through MKn bit exist. Thus, the existence of priority requests pending through the priority mask function can be checked while interrupts are prohibited through priority masking.
- When EI level maskable interrupt request is not output to the CPU
  The ICSR.EIR bit is set to 1.
  By looking at the ICSR.EIR bit in the interval during which PSW.ID = 1, it is possible to check whether an INTn interrupt request exists.
- When FE level maskable interrupt request is not output to the CPU
  ICSR.FIR bit is set to 1.
  By looking at the ICSR.FIR bit in the interval during which PSW.NP = 1, it is possible to check whether a FEINT interrupt request exists.

5. 7. 5 In-service priority clear function

This function initializes the internal status of the Interrupt Controller. It operates when the ISPC register is accessed. The following operations are possible using this function.

- Clear all contents of ISPR register
- Clear ICSR.EIE, FIE, and FNE bits

All the bits of ISPR register can be cleared to 0 by writing "1" to all the bits of this register and then writing "0" to all the bits of ISPR register. Moreover, the ICSR.EIE, FIE, and FNE bits, which all indicates the state in which an interrupt request is being processed in the CPU, are all be cleared.

The value of this register is automatically cleared to 0 by writing 0 to all the bits of ISPR. The values of the bits of ISPR remain unchanged in the case of write access that is not performed simultaneously to all the bits.

5.8 Exception Handler Address Switching Function

Interrupt handler addresses can be switched by software.

For details, refer to the chapter *“Exception Handler Address Switching Functio*n” in the “V850E2M 32-bit Microcontroller Core Architecture” user manual.

Chapter 6 DMA/DTS Controller (DMAC)

This chapter contains a generic description of the DMA/DTS Controller (DMA).

The first section describes all V850E2/Fx4-H specific properties.

6.1 V850E2/Fx4-H DMA Features

DMA channels

This microcontroller provides 16 DMA channels.

Channel index n

Throughout this chapter, the index n is used for two purposes:

- DMA function: n indentifies an individual DMA channel, thus n = 0 to 15
- DTS function: n indentifies an individual DTS trigger, thus n = 0 to 127

DMA address map

Refer to the section *"V850E2/Fx4-H CPU Address Map"* in chapter *"CPU System Functions"* for details.

Clock supply

Since the DMA Controller is part of the CPU Subsystem, it is supplied with the same clock CKSCLC_000 as the CPU Subsystem.
Refer to the section *"CPU Subsystem"* in chapter *"CPU System Functions"* for details.

DMAC H/W reset

The DMA/DTS Controller and its registers are initialized by the following reset signal:

Table 6-1 DMAC reset signal

DMAC	Reset signal
DMAC	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

(1) DMA trigger factors

The assignment of a DMA trigger source to the DMA channel n is done by the DTFNRn.IFC[6:0].

The following table lists all DMA triggers and how to select them by setting the DMA trigger factor register DTFRn.

Table 6-2 DMA trigger factors 0 to 63 selection

DTFRn.IFCn[6:0]	DMA trigger interrupt
0	no DMA
1	INTP0
2	INTP1
3	INTP2
4	INTP3
5	INTP4
6	INTP5
7	INTP6
8	INTP7
9	INTP8
10	INTP9
11	INTP10
12	INTTAUA0I8
13	INTTAUA0I9
14	INTTAUA0I10
15	INTTAUA0I11
16	INTTAUA0I12
17	INTTAUA0I13
18	INTTAUA0I14
19	INTTAUA0I15
20	INTTAUB1I0
21	INTTAUB1I1
22	INTTAUB1I4
23	INTTAUB1I5
24	INTTAUB1I8
25	INTTAUB1I9
26	INTTAUB1I12
27	INTTAUB1I13
28	INTTAUB2I0
29	INTTAUB2I1
30	INTTAUB2I4
31	INTTAUB2I5
32	INTTAUB2I8
33	INTTAUB2I9
34	INTTAUB2I12
35	INTTAUB2I13
36	reserved
37	reserved
38	INTADCA0I0
39	INTADCA0I1
40	INTADCA0I2
41	INTADCA0LLT
42	INTCSIG0IR
43	INTCSIG0IC
44	INTLMA0IR
45	INTLMA0IT
46	INTLMA1IR
47	INTLMA1IT
48	INTDMA0 / INTCT0[a]
49	INTDMA1 / INTCT1[a]
50	INTDMA2 / INTCT2[a]
51	INTDMA3 / INTCT3[a]
52	INTDMA4 / INTCT4[a]
53	INTDMA5 / INTCT5[a]
54	INTDMA6 / INTCT6[a]
55	INTDMA7 / INTCT7[a]
56	reserved
57	INTIICB0IA
58	reserved
59	INTTAUJ0I0
60	INTTAUJ0I1
61	INTTAUJ0I2
62	INTTAUJ0I3
63	INTTAUJ1I0

a) Selection of the DMA interrupts is achieved by setting the DMA interrupt selection register refer to *5.3.5 “V850E2/Fx4-H DMA interrupt selection” on page 296* .

Table 6-3 DMA trigger factors 64 to 127 selection

DTFRn.IFCn[6:0]	DMA trigger interrupt
64	INTTAUJ1I1
65	INTTAUJ1I2
66	INTTAUJ1I3
67	INTADCA1I0
68	INTADCA1I1
69	INTADCA1I2
70	reserved
71	reserved
72	INTCSIG2IR
73	INTCSIG2IC
74	INTFCN3REC
75	INTFCN4ERR
76	INTFCN4TRX
77	INTCSIH0IC
78	INTCSIH0IJC
79	reserved
80	reserved
81	INTCSIH0IR
82	INTCSIG4IR
83	INTCSIG4IC
84	INTDCN0REC
85	INTFLX0I0
86	INTFLX0I1
87	INTCSIH1IR
88	INTCSIH1IC
89	INTCSIH1IJC
90	reserved
91	reserved
92	INTLMA2IT
93	INTLMA3IT
94	INTLMA4IR
95	INTLMA4IT

DTFRn.IFCn[6:0]	DMA trigger interrupt
96	INTCSIH2IR
97	INTCSIH2IC
98	INTCSIH2IJC
99	reserved
100	reserved
101	INTETHA0SRX
102	INTETHA0SCRX
103	INTETHA0SCTX
104	INTETHA0RS
105	INTETHA0TS
106	INTETHA0FS
107	INTLMA5IR
108	INTLMA5IT
109	INTLMA6IR
110	INTLMA6IT
111	INTLMA7IR
112	INTLMA7IT
113	INTLMA8IR
114	INTLMA8IT
115	INTLMA9IR
116	INTLMA9IT
117	INTLMA10IR
118	INTLMA10IT
119	INTLMA11IR
120	INTLMA11IT
121	INTDTS121
122	INTDTS122
123	INTDTS123
124	INTDTS124
125	INTDTS125
126	INTDTS126
127	INTDTS127

(2) DTS trigger factors

The following table lists all DTS triggers.
The assignment of a DTS trigger signal to a DTS channel is fixed.

Table 6-4 DTS trigger factors 0 to 63 selection

DTS channel	DTS trigger interrupt
0	INTP0
1	INTP1
2	INTP2
3	INTTAUB1I0
4	INTTAUB1I1
5	INTTAUB2I0
6	INTTAUB2I1
7	INTADCA0I0
8	INTADCA0I1
9	INTADCA1I0
10	INTADCA1I1
11	INTADCA1I2
12	INTCSIH0IC
13	INTCSIH0IJC
14	INTCSIH0IR
15	INTDCN0REC
16	INTFLXA0I0
17	INTFLXA0I1
18	INTCSIH1IR
19	INTCSIH1IC
20	INTCSIH1IJC
21	INTCSIH2IR
22	INTCSIH2IC
23	INTCSIH2IJC
24	reserved
25	reserved
26	INTETHA0SRX
27	INTETHA0SCRX
28	INTETHA0SCTX
29	INTETHA0RS
30	INTETHA0TS
31	INTETHA0FS

DTS channel	DTS trigger interrupt
32	INTP3
33	INTP4
34	INTP5
35	INTP6
36	INTP7
37	INTP8
38	INTP9
39	INTP10
40	INTTAUA0I10
41	INTTAUA0I11
42	INTTAUA0I12
43	INTTAUA0I13
44	INTTAUA0I14
45	INTTAUA0I15
46	INTTAUB1I8
47	INTTAUB1I9
48	INTTAUB1I12
49	INTTAUB1I13
50	INTTAUB2I4
51	INTTAUB2I5
52	INTTAUB2I8
53	INTTAUB2I9
54	INTTAUB2I12
55	INTTAUB2I13
56	INTADCA0I2
57	INTADCA0LLT
58	INTLMA0IR
59	INTLMA0IT
60	INTLMA1IR
61	INTLMA1IT
62	reserved
63	INTTAUJ0I2

Table 6-5 DTS trigger factors 64 to 127 selection

DTS channel	DTS trigger interrupt
64	INTTAUJ0I3
65	INTTAUJ1I0
66	INTTAUJ1I1
67	INTTAUJ1I2
68	INTTAUJ1I3
69	INTCSIG2IR
70	INTCSIG2IC
71	INTLMA2IT
72	INTLMA3IT
73	INTTAUB1I4
74	INTTAUB1I5
75	INTTAUA0I8
76	INTTAUA0I9
77	reserved
78	reserved
79	INTCSIG0IR
80	INTCSIG0IC
81	INTDMA0 / INTCT0[a]
82	INTDMA1 / INTCT1[a]
83	INTDMA2 / INTCT2[a]
84	INTDMA3 / INTCT3[a]
85	INTDMA4 / INTCT4[a]
86	INTDMA5 / INTCT5[a]
87	INTDMA6 / INTCT6[a]
88	INTDMA7 / INTCT7[a]
89	INTIICB0IA
90	reserved
91	INTTAUJ0I1
92	INTTAUJ0I2
93	reserved
94	reserved
95	INTFCN3REC

DTS channel	DTS trigger interrupt
96	INTFCN4ERR
97	INTFCN4TRX
98	reserved
99	reserved
100	INTCSIG4IR
101	INTCSIG4IC
102	reserved
103	reserved
104	INTLMA4IR
105	INTLMA4IT
106	INTLMA5IR
107	INTLMA5IT
108	INTLMA6IR
109	INTLMA6IT
110	INTLMA7IR
111	INTLMA7IT
112	INTLMA8IR
113	INTLMA8IT
114	INTLMA9IR
115	INTLMA9IT
116	INTLMA10IR
117	INTLMA10IT
118	INTLMA11IR
119	INTLMA11IT
120	INTDMA0 / INTCT0[a]
121	INTDMA1 / INTCT1[a]
122	INTDMA2 / INTCT2[a]
123	INTDMA3 / INTCT3[a]
124	INTDMA4 / INTCT4[a]
125	INTDMA5 / INTCT5[a]
126	INTDMA6 / INTCT6[a]
127	INTDMA7 / INTCT7[a]

a) Selection of the DMA interrupts is achieved by setting the DMA interrupt selection register refer to *5.3.5 “V850E2/Fx4-H DMA interrupt selection” on page 296* .

6.2 Definition of Terms

The terms used in this document are defined as follows.

Table 6-6 Definition of terms

Term	Function
DMA transfer	Period from the start of the first DMA cycle to assertion of INTDMA
DMA cycle	Period of transferring one unit of data (since a read cycle of DMA data bus has been started and until a write cycle is completed. In the case of 128-bit transfer, until the read cycle is completed four times and the write cycle is completed four times)
Hardware DMA transfer request	256 hardware DMA transfer request (INTIN[255:0])
Software DMA transfer request	DMA transfer request by internal register (SR bit of DTS register)
DMA transfer request	Hardware DMA transfer request or software DMA transfer request
Transfer information (TI)	Information required for DMA transfer, such as transfer address, transfer data size, and transfer count. In particular, the DTS's transfer information is called TI (Transfer Information).
TI fetch	When DTS reads transfer information from Data RAM
TI write back	When DTS writes transfer information back to Data RAM
TI fetch cycle	Period during which DTS reads transfer information from Data RAM
DTS cycle	Period during which DTS performs a transfer according to the TI
TI write back cycle	Period during which DTS writes transfer information back to Data RAM
Single transfer	In the case of DMAC, one DMA cycle is executed per transfer request.
Single step transfer	This function is used only with DMAC. The number of transfers set in the transfer count setting register (DTC) is executed per software DMA transfer request. Since the bus is released for each transfer, the CPU is able to generate interrupts. If a higher-priority transfer request occurs during execution of a single step transfer, the single step transfer is aborted while the higher-priority transfer request is executed.
Block transfer	This function is used only with DTS. For each transfer request, one TI fetch cycle, the DTS cycles for the number specified in the transfer count setting register (TI-A's bits [31:16]), and one TI write back cycle are executed. Since the bus is released for each transfer, the CPU is able to generate interrupts. If a higher-priority DTS request occurs during execution of a block transfer, DTS ignores (cannot acknowledge) the request until the block transfer is completed.

6.3 General

Direct memory access (DMA) function is used to access data without going via the CPU.

The DMA/DTS Controller incorporates following units:

- DMA trigger factor register (DTFR)
- DMA controller 0 (DMAC0) for DMA channels n = 0 to 7
- DMA controller 1 (DMAC1) for DMA channels n = 8 to 15
- DTS controller (DTS)
- DMA/DTS data transfer unit

Note The number of DMA channels, expressed by the index n, is defined in the first section of this chapter under the key words *"DMA channels"* and *"Channel index n"*.
In case the supported number of DMA channels is maximum 8, i.e. n ≤ 7, DMAC1 is not existing.

6.3.1 DMA controller (DMAC) function

- Registers to store transfer information (transfer address, transfer size, etc.) and registers to control DMAC are included.
- When a DMA transfer request is accepted, a transfer request is output to DMAT, according to the contained transfer information.
- Hardware DMA transfer requests, DMA acknowledge signals, and DMA transfer completion interrupts are input and output.
- Write back information is written back to registers.

6.3.2 DMA trigger factor register (DTFR) function

- DMA transfer factors are selected from among interrupt signals. (16 channels are selected from 128-channel interrupt signals.)

6.3.3 Data transfer service (DTS) function

- A register that controls DTS is included (transfer information is set in Data RAM).
- When a DTS transfer request is accepted, transfer information set in Data RAM is read, and a transfer request is output to DMAT, according to the transfer information.
- Hardware DTS transfer requests, DTS acknowledge signals, and DTS transfer completion interrupts are input and output.
- Write back information (next time's transfer information) is written back to Data RAM.

6.3.4 DTS factor selector (DTSFSL) function

- DTS transfer factors are selected from among interrupt signals. (Up to 128 transfer factors can be assigned.)
- Up to four priority levels can be set.
- Various interrupt sources from the DTS and interrupt signals from peripherals are selected.

Table 6-7 Transfer target spaces

Transfer destination / Transfer source	Peripheral I/O	External memory	Data RAM	Internal Code flash	Internal Data-Flash
Peripheral I/O	√	√	√	x	x
External memory	√	√	√	x	x
Data RAM	√	√	√	x	x
Internal Code flash	√	√	√	x	x
Internal Data-Flash	√	√	√	x	x

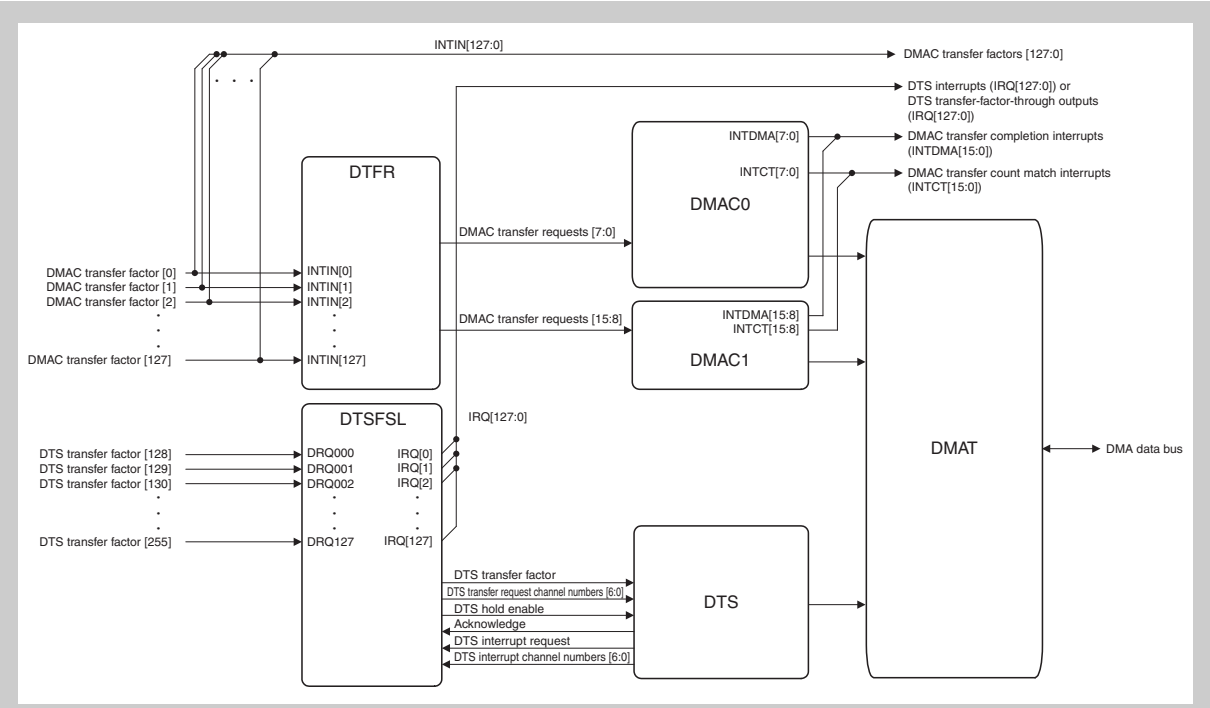


Figure 6-1 DMA/DTS Controller block diagram

6.3.5 DMA access memory map

Refer to the section *"V850E2/Fx4-H Address Map"* in chapter *"CPU System Functions"* for details.

6.3.6 Prioritization of channels

The following describes how the prioritization of the DMA subsystem's various transfer channels is determined. Prioritization is determined via two stages: the first stage is when the priority level in each group among DMAC0, DMAC1, and DTS is determined separately.

Among DMAC0, the priority is set as CH0 > CH1 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7, in which CH0 has the highest priority.

Among DMAC1, the priority is set as CH8 > CH9 > CH10 > CH11 > CH12 > CH13 > CH14 > CH15, in which CH8 has the highest priority.

Among DTS, where four levels of priority settings are possible, the priority is determined using the default priority channels: CH000 > CH001 > CH002... > CH127.

6.3.7 Arbitration of transfer requests

Since the DMA subsystem includes three transfer controllers (DMAC0, DMAC1, and DTS), these transfer requests are arbitrated internally. This arbitration is always performed using the fixed prioritization: DMAC0 > DMAC1 > DTS. DMAC0's priority level is the highest.

However, when DMAC0, DMAC1, or DTS completes one transfer request, the request is temporarily withdrawn, so the following phenomena occur.

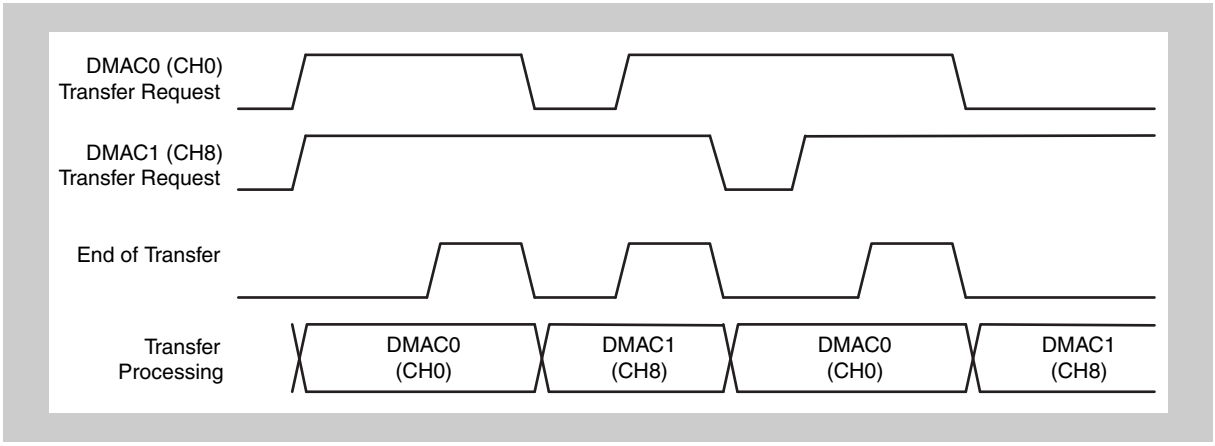


Figure 6-2 Alternation of DMAC0 and DMAC1

Even when CH0's transfer requests occur continuously, the DMAC temporarily withdraws transfer requests. As a result, DMAC1's requests take priority.

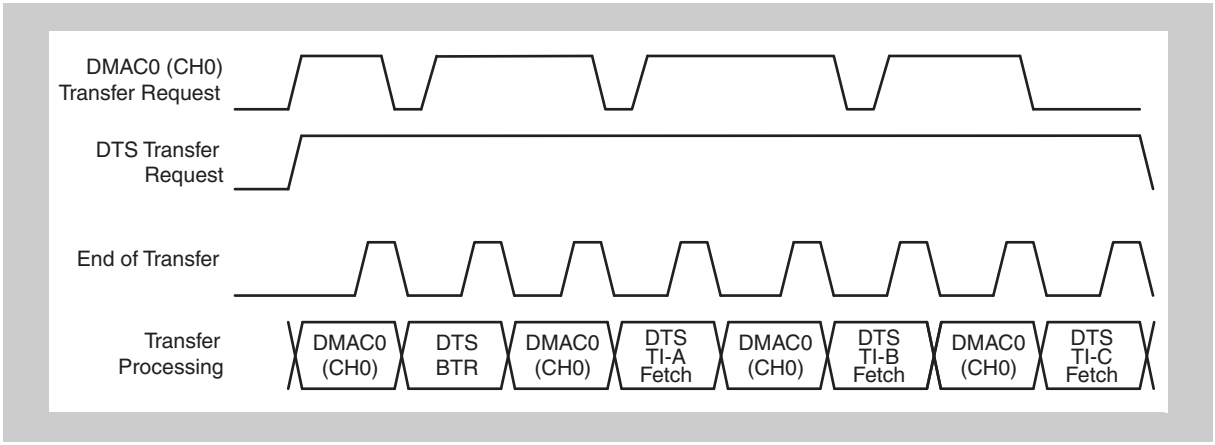


Figure 6-3 Alternation of DMAC0 and DTS

Even when CH0's transfer requests occur continuously, the DMAC0 temporarily withdraws transfer requests. As a result, DTS's requests take priority. After a base table read (BTR) cycle, DTS continues to output TI-A fetch requests, but since DMAC0 has a higher priority level than DTS, DMAC0 takes priority as a result of arbitration. When DMAC0's transfer is completed, DMAC0 temporarily withdraws requests, and DTS's request is executed.

6.3.8 Stand-by function

A stop acknowledge signal is returned following a stop request when transfer of the DMA or DTS transfer unit is completed.

The transfer unit for DMA is one DMA cycle (from when read starts until write ends), and for DTS it is a series of the TI fetch cycle, DTS cycle, or TI write back cycle.

Since DMAC0, DMAC1, and DTS each have a transfer unit, when there are three requests, a stop acknowledge is returned only after all transfer units have been transferred.

Note With DTS, due to the relation with the hold buffer, exceptions to the above may occur. Refer to *6.9 "DTS Function"* for details.

6.4 DMAC Function

6.4.1 Features

Transfer data size

- 8 bits
- 16 bits
- 32 bits
- 128 bits

Transfer data

Fixed to little endian

Misaligned data not supported

Maximum transfer count

32,768 (2^{15}) times (MSB of the 16-bit register is used for the next address function.)

Channel priority control

Fixed priority (highest priority (CH0) → lowest priority (CH15))

Subject to transfer

- Code flash
- Data RAM
- Data-Flash
- External memory area
- Peripheral I/O area

Transfer type

2-cycle transfer (dual address transfer)

The address at both the transfer source and destination is accessed. Two bus cycles are required to execute transfer once (read cycle + write cycle). Because the bus is not locked between the read cycle and write cycle, the CPU cycle may interrupt. When a 128-bit access is made, the write cycle is executed four times after the read cycle has been executed four times. Because the bus is not locked between the read cycles and between the write cycles, the CPU cycle may interrupt.

Transfer mode

- Single transfer mode (when hardware DMA transfer request is generated)

  When a hardware DMA transfer request is generated, the bus mastership is acquired, and the bus is always released after transfer has been executed once. If another hardware DMA transfer request is generated after that, transfer is executed once again. This operation is repeated until transfer has been executed the number of times specified by the transfer count register (DTC).

- Single-step transfer mode (when software DMA transfer request is generated)

  When a software DMA transfer request is generated, the bus mastership is acquired, and the bus is released each time transfer has been executed once. Once a software DMA transfer request has been acknowledged, this operation is repeated until transfer has been executed the number of times specified by the transfer count register (DTC).

Transfer address control

- Incremental
- Decremental
- Fixed

Transfer error support

When the data from the transfer source contains an error, or if an error occurs at the transfer destination, DMA transfer is aborted, and a SYSERR exception is output for the CPU.

DMA transfer request

A hardware DMA transfer request or a software DMA transfer request can be selected for each channel (by setting the DTRS register). The software DMA transfer request can be set by software (by setting the DTS register). This register also has a status bit (DTS register) that indicates that a hardware DMA transfer request has been generated.

Transfer count match interrupt output function

This function has a transfer count compare register (DTCC) for each channel and outputs an interrupt signal (INTCT15 to INTCT0) upon a match with the transfer count register (DTC) of each channel.

Transfer completion interrupt output function

This function outputs a transfer completion interrupt signal (INTDMA15 to INTDMA0) when DMA transfer of each channel has been completed the number of times specified by the transfer count register (DTC).

Next address setting function

This function has a register for each channel for setting the transfer address and transfer count (Current) of the DMA transfer currently being executed, and a register for setting the transfer address and transfer count (Next) of the DMA transfer to be executed next, following completion of the DMA transfer currently being executed. This function also has a bit for each register for setting whether to copy Next to Current upon completion of DMA transfer.

Stand-by support

When a stop mode request is generated, DMA transfer is aborted momentarily and the stop mode is entered.

DMA transfer abort function

This function supports aborting DMA transfer by software.

6.4.2 DMAC setting registers

Table 6-8 DMAC setting registers (1/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF7300H	DTRC0	DMA transfer request control register 0	R/W	√	√			00H
FFFF7310H	DTRS0	DMA transfer request select register CH0				√		0000H
FFFF7314H	DSA0	DMA source address register CH0					√	00000000H
FFFF7314H	DSA0L	DMA source address register LCH0				√		0000H
FFFF7316H	DSA0H	DMA source address register HCH0				√		0000H
FFFF7318H	DSC0	DMA source chip select register CH0				√		0001H
FFFF731CH	DNSA0	DMA next source address register CH0					√	00000000H
FFFF731CH	DNSA0L	DMA next source address register LCH0				√		0000H
FFFF731EH	DNSA0H	DMA next source address register HCH0				√		0000H
FFFF7320H	DNSC0	DMA next source chip select register CH0				√		0001H
FFFF7324H	DDA0	DMA destination address register CH0					√	00000000H
FFFF7324H	DDA0L	DMA destination address register LCH0				√		0000H
FFFF7326H	DDA0H	DMA destination address register HCH0				√		0000H
FFFF7328H	DDC0	DMA destination chip select register CH0				√		0001H
FFFF732CH	DNDA0	DMA next destination address register CH0					√	00000000H
FFFF732CH	DNDA0L	DMA next destination address register LCH0				√		0000H
FFFF732EH	DNDA0H	DMA next destination address register HCH0				√		0000H
FFFF7330H	DNDC0	DMA next destination chip select register CH0				√		0001H
FFFF7332H	DTC0	DMA transfer count register CH0				√		0000H
FFFF7334H	DNTC0	DMA next transfer count register CH0				√		0000H
FFFF7336H	DTCC0	DMA transfer count compare register CH0				√		0000H
FFFF7338H	DTCT0	DMA transfer control register CH0				√		0000H
FFFF733AH	DTS0	DMA transfer status register CH0		√	√			00H
FFFF7340H	DTRS1	DMA transfer request select register CH1				√		0000H
FFFF7344H	DSA1	DMA source address register CH1					√	00000000H
FFFF7344H	DSA1L	DMA source address register LCH1				√		0000H
FFFF7346H	DSA1H	DMA source address register HCH1				√		0000H
FFFF7348H	DSC1	DMA source chip select register CH1				√		0001H
FFFF734CH	DNSA1	DMA next source address register CH1					√	00000000H
FFFF734CH	DNSA1L	DMA next source address register LCH1				√		0000H
FFFF734EH	DNSA1H	DMA next source address register HCH1				√		0000H
FFFF7350H	DNSC1	DMA next source chip select register CH1				√		0001H
FFFF7354H	DDA1	DMA destination address register CH1					√	00000000H
FFFF7354H	DDA1L	DMA destination address register LCH1				√		0000H
FFFF7356H	DDA1H	DMA destination address register HCH1				√		0000H
FFFF7358H	DDC1	DMA destination chip select register CH1				√		0001H

Table 6-8 DMAC setting registers (2/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF735CH	DNDA1	DMA next destination address register CH1	R/W				√	00000000H
FFFF735CH	DNDA1L	DMA next destination address register LCH1				√		0000H
FFFF735EH	DNDA1H	DMA next destination address register HCH1				√		0000H
FFFF7360H	DNDC1	DMA next destination chip select register CH1				√		0001H
FFFF7362H	DTC1	DMA transfer count register CH1				√		0000H
FFFF7364H	DNTC1	DMA next transfer count register CH1				√		0000H
FFFF7366H	DTCC1	DMA transfer count compare register CH1				√		0000H
FFFF7368H	DTCT1	DMA transfer control register CH1				√		0000H
FFFF736AH	DTS1	DMA transfer status register CH1		√	√			00H
FFFF7370H	DTRS2	DMA transfer request select register CH2				√		0000H
FFFF7374H	DSA2	DMA source address register CH2					√	00000000H
FFFF7374H	DSA2L	DMA source address register LCH2				√		0000H
FFFF7376H	DSA2H	DMA source address register HCH2				√		0000H
FFFF7378H	DSC2	DMA source chip select register CH2				√		0001H
FFFF737CH	DNSA2	DMA next source address register CH2					√	00000000H
FFFF737CH	DNSA2L	DMA next source address register LCH2				√		0000H
FFFF737EH	DNSA2H	DMA next source address register HCH2				√		0000H
FFFF7380H	DNSC2	DMA next source chip select register CH2				√		0001H
FFFF7384H	DDA2	DMA destination address register CH2					√	00000000H
FFFF7384H	DDA2L	DMA destination address register LCH2				√		0000H
FFFF7386H	DDA2H	DMA destination address register HCH2				√		0000H
FFFF7388H	DDC2	DMA destination chip select register CH2				√		0001H
FFFF738CH	DNDA2	DMA next destination address register CH2					√	00000000H
FFFF738CH	DNDA2L	DMA next destination address register LCH2				√		0000H
FFFF738EH	DNDA2H	DMA next destination address register HCH2				√		0000H
FFFF7390H	DNDC2	DMA next destination chip select register CH2				√		0001H
FFFF7392H	DTC2	DMA transfer count register CH2				√		0000H
FFFF7394H	DNTC2	DMA next transfer count register CH2				√		0000H
FFFF7396H	DTCC2	DMA transfer count compare register CH2				√		0000H
FFFF7398H	DTCT2	DMA transfer control register CH2				√		0000H
FFFF739AH	DTS2	DMA transfer status register CH2		√	√			00H
FFFF73A0H	DTRS3	DMA transfer request select register CH3				√		0000H
FFFF73A4H	DSA3	DMA source address register CH3					√	00000000H
FFFF73A4H	DSA3L	DMA source address register LCH3				√		0000H
FFFF73A6H	DSA3H	DMA source address register HCH3				√		0000H
FFFF73A8H	DSC3	DMA source chip select register CH3				√		0001H

Table 6-8 DMAC setting registers (3/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF73ACH	DNSA3	DMA next source address register CH3	R/W				√	00000000H
FFFF73ACH	DNSA3L	DMA next source address register LCH3				√		0000H
FFFF73AEH	DNSA3H	DMA next source address register HCH3				√		0000H
FFFF73B0H	DNSC3	DMA next source chip select register CH3				√		0001H
FFFF73B4H	DDA3	DMA destination address register CH3					√	00000000H
FFFF73B4H	DDA3L	DMA destination address register LCH3				√		0000H
FFFF73B6H	DDA3H	DMA destination address register HCH3				√		0000H
FFFF73B8H	DDC3	DMA destination chip select register CH3				√		0001H
FFFF73BCH	DNDA3	DMA next destination address register CH3					√	00000000H
FFFF73BCH	DNDA3L	DMA next destination address register LCH3				√		0000H
FFFF73BEH	DNDA3H	DMA next destination address register HCH3				√		0000H
FFFF73C0H	DNDC3	DMA next destination chip select register CH3				√		0001H
FFFF73C2H	DTC3	DMA transfer count register CH3				√		0000H
FFFF73C4H	DNTC3	DMA next transfer count register CH3				√		0000H
FFFF73C6H	DTCC3	DMA transfer count compare register CH3				√		0000H
FFFF73C8H	DTCT3	DMA transfer control register CH3				√		0000H
FFFF73CAH	DTS3	DMA transfer status register CH3		√	√			00H
FFFF73D0H	DTRS4	DMA transfer request select register CH4				√		0000H
FFFF73D4H	DSA4	DMA source address register CH4					√	00000000H
FFFF73D4H	DSA4L	DMA source address register LCH4				√		0000H
FFFF73D6H	DSA4H	DMA source address register HCH4				√		0000H
FFFF73D8H	DSC4	DMA source chip select register CH4				√		0001H
FFFF73DCH	DNSA4	DMA next source address register CH4					√	00000000H
FFFF73DCH	DNSA4L	DMA next source address register LCH4				√		0000H
FFFF73DEH	DNSA4H	DMA next source address register HCH4				√		0000H
FFFF73E0H	DNSC4	DMA next source chip select register CH4				√		0001H
FFFF73E4H	DDA4	DMA destination address register CH4					√	00000000H
FFFF73E4H	DDA4L	DMA destination address register LCH4				√		0000H
FFFF73E6H	DDA4H	DMA destination address register HCH4				√		0000H
FFFF73E8H	DDC4	DMA destination chip select register CH4				√		0001H
FFFF73ECH	DNDA4	DMA next destination address register CH4					√	00000000H
FFFF73ECH	DNDA4L	DMA next destination address register LCH4				√		0000H
FFFF73EEH	DNDA4H	DMA next destination address register HCH4				√		0000H
FFFF73F0H	DNDC4	DMA next destination chip select register CH4				√		0001H
FFFF73F2H	DTC4	DMA transfer count register CH4				√		0000H
FFFF73F4H	DNTC4	DMA next transfer count register CH4				√		0000H

Table 6-8 DMAC setting registers (4/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF73F6H	DTCC4	DMA transfer count compare register CH4	R/W			√		0000H
FFFF73F8H	DTCT4	DMA transfer control register CH4				√		0000H
FFFF73FAH	DTS4	DMA transfer status register CH4		√	√			00H
FFFF7400H	DTRS5	DMA transfer request select register CH5				√		0000H
FFFF7404H	DSA5	DMA source address register CH5					√	00000000H
FFFF7404H	DSA5L	DMA source address register LCH5				√		0000H
FFFF7406H	DSA5H	DMA source address register HCH5				√		0000H
FFFF7408H	DSC5	DMA source chip select register CH5				√		0001H
FFFF740CH	DNSA5	DMA next source address register CH5					√	00000000H
FFFF740CH	DNSA5L	DMA next source address register LCH5				√		0000H
FFFF740EH	DNSA5H	DMA next source address register HCH5				√		0000H
FFFF7410H	DNSC5	DMA next source chip select register CH5				√		0001H
FFFF7414H	DDA5	DMA destination address register CH5					√	00000000H
FFFF7414H	DDA5L	DMA destination address register LCH5				√		0000H
FFFF7416H	DDA5H	DMA destination address register HCH5				√		0000H
FFFF7418H	DDC5	DMA destination chip select register CH5				√		0001H
FFFF741CH	DNDA5	DMA next destination address register CH5					√	00000000H
FFFF741CH	DNDA5L	DMA next destination address register LCH5				√		0000H
FFFF741EH	DNDA5H	DMA next destination address register HCH5				√		0000H
FFFF7420H	DNDC5	DMA next destination chip select register CH5				√		0001H
FFFF7422H	DTC5	DMA transfer count register CH5				√		0000H
FFFF7424H	DNTC5	DMA next transfer count register CH5				√		0000H
FFFF7426H	DTCC5	DMA transfer count compare register CH5				√		0000H
FFFF7428H	DTCT5	DMA transfer control register CH5				√		0000H
FFFF742AH	DTS5	DMA transfer status register CH5		√	√			00H
FFFF7430H	DTRS6	DMA transfer request select register CH6				√		0000H
FFFF7434H	DSA6	DMA source address register CH6					√	00000000H
FFFF7434H	DSA6L	DMA source address register LCH6				√		0000H
FFFF7436H	DSA6H	DMA source address register HCH6				√		0000H
FFFF7438H	DSC6	DMA source chip select register CH6				√		0001H
FFFF743CH	DNSA6	DMA next source address register CH6					√	00000000H
FFFF743CH	DNSA6L	DMA next source address register LCH6				√		0000H
FFFF743EH	DNSA6H	DMA next source address register HCH6				√		0000H
FFFF7440H	DNSC6	DMA next source chip select register CH6				√		0001H
FFFF7444H	DDA6	DMA destination address register CH6					√	00000000H
FFFF7444H	DDA6L	DMA destination address register LCH6				√		0000H
FFFF7446H	DDA6H	DMA destination address register HCH6				√		0000H
FFFF7448H	DDC6	DMA destination chip select register CH6				√		0001H

Table 6-8 DMAC setting registers (5/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF744CH	DNDA6	DMA next destination address register CH6	R/W				√	00000000H
FFFF744CH	DNDA6L	DMA next destination address register LCH6				√		0000H
FFFF744EH	DNDA6H	DMA next destination address register HCH6				√		0000H
FFFF7450H	DNDC6	DMA next destination chip select register CH6				√		0001H
FFFF7452H	DTC6	DMA transfer count register CH6				√		0000H
FFFF7454H	DNTC6	DMA next transfer count register CH6				√		0000H
FFFF7456H	DTCC6	DMA transfer count compare register CH6				√		0000H
FFFF7458H	DTCT6	DMA transfer control register CH6				√		0000H
FFFF745AH	DTS6	DMA transfer status register CH6		√	√			00H
FFFF7460H	DTRS7	DMA transfer request select register CH7				√		0000H
FFFF7464H	DSA7	DMA source address register CH7					√	00000000H
FFFF7464H	DSA7L	DMA source address register LCH7				√		0000H
FFFF7466H	DSA7H	DMA source address register HCH7				√		0000H
FFFF7468H	DSC7	DMA source chip select register CH7				√		0001H
FFFF746CH	DNSA7	DMA next source address register CH7					√	00000000H
FFFF746CH	DNSA7L	DMA next source address register LCH7				√		0000H
FFFF746EH	DNSA7H	DMA next source address register HCH7				√		0000H
FFFF7470H	DNSC7	DMA next source chip select register CH7				√		0001H
FFFF7474H	DDA7	DMA destination address register CH7					√	00000000H
FFFF7474H	DDA7L	DMA destination address register LCH7				√		0000H
FFFF7476H	DDA7H	DMA destination address register HCH7				√		0000H
FFFF7478H	DDC7	DMA destination chip select register CH7				√		0001H
FFFF747CH	DNDA7	DMA next destination address register CH7					√	00000000H
FFFF747CH	DNDA7L	DMA next destination address register LCH7				√		0000H
FFFF747EH	DNDA7H	DMA next destination address register HCH7				√		0000H
FFFF7480H	DNDC7	DMA next destination chip select register CH7				√		0001H
FFFF7482H	DTC7	DMA transfer count register CH7				√		0000H
FFFF7484H	DNTC7	DMA next transfer count register CH7				√		0000H
FFFF7486H	DTCC7	DMA transfer count compare register CH7				√		0000H
FFFF7488H	DTCT7	DMA transfer control register CH7				√		0000H
FFFF748AH	DTS7	DMA transfer status register CH7		√	√			00H
FFFF7500H	DTRC1	DMA transfer request control register 1		√	√			00H
FFFF7510H	DTRS8	DMA transfer request select register CH8				√		0000H
FFFF7514H	DSA8	DMA source address register CH8					√	00000000H
FFFF7514H	DSA8L	DMA source address register LCH8				√		0000H
FFFF7516H	DSA8H	DMA source address register HCH8				√		0000H

Table 6-8 DMAC setting registers (6/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF7518H	DSC8	DMA source chip select register CH8	R/W			√		0001H
FFFF751CH	DNSA8	DMA next source address register CH8					√	00000000H
FFFF751CH	DNSA8L	DMA next source address register LCH8				√		0000H
FFFF751EH	DNSA8H	DMA next source address register HCH8				√		0000H
FFFF7520H	DNSC8	DMA next source chip select register CH8				√		0001H
FFFF7524H	DDA8	DMA destination address register CH8					√	00000000H
FFFF7524H	DDA8L	DMA destination address register LCH8				√		0000H
FFFF7526H	DDA8H	DMA destination address register HCH8				√		0000H
FFFF7528H	DDC8	DMA destination chip select register CH8				√		0001H
FFFF752CH	DNDA8	DMA next destination address register CH8					√	00000000H
FFFF752CH	DNDA8L	DMA next destination address register LCH8				√		0000H
FFFF752EH	DNDA8H	DMA next destination address register HCH8				√		0000H
FFFF7530H	DNDC8	DMA next destination chip select register CH8				√		0001H
FFFF7532H	DTC8	DMA transfer count register CH8				√		0000H
FFFF7534H	DNTC8	DMA next transfer count register CH8				√		0000H
FFFF7536H	DTCC8	DMA transfer count compare register CH8				√		0000H
FFFF7538H	DTCT8	DMA transfer control register CH8				√		0000H
FFFF753AH	DTS8	DMA transfer status register CH8		√	√			00H
FFFF7540H	DTRS9	DMA transfer request select register CH9				√		0000H
FFFF7544H	DSA9	DMA source address register CH9					√	00000000H
FFFF7544H	DSA9L	DMA source address register LCH9				√		0000H
FFFF7546H	DSA9H	DMA source address register HCH9				√		0000H
FFFF7548H	DSC9	DMA source chip select register CH9				√		0001H
FFFF754CH	DNSA9	DMA next source address register CH9					√	00000000H
FFFF754CH	DNSA9L	DMA next source address register LCH9				√		0000H
FFFF754EH	DNSA9H	DMA next source address register HCH9				√		0000H
FFFF7550H	DNSC9	DMA next source chip select register CH9				√		0001H
FFFF7554H	DDA9	DMA destination address register CH9					√	00000000H
FFFF7554H	DDA9L	DMA destination address register LCH9				√		0000H
FFFF7556H	DDA9H	DMA destination address register HCH9				√		0000H
FFFF7558H	DDC9	DMA destination chip select register CH9				√		0001H
FFFF755CH	DNDA9	DMA next destination address register CH9					√	00000000H
FFFF755CH	DNDA9L	DMA next destination address register LCH9				√		0000H
FFFF755EH	DNDA9H	DMA next destination address register HCH9				√		0000H
FFFF7560H	DNDC9	DMA next destination chip select register CH9				√		0001H
FFFF7562H	DTC9	DMA transfer count register CH9				√		0000H

Table 6-8 DMAC setting registers (7/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF7564H	DNTC9	DMA next transfer count register CH9	R/W			√		0000H
FFFF7566H	DTCC9	DMA transfer count compare register CH9				√		0000H
FFFF7568H	DTCT9	DMA transfer control register CH9				√		0000H
FFFF756AH	DTS9	DMA transfer status register CH9		√	√			00H
FFFF7570H	DTRS10	DMA transfer request select register CH10				√		0000H
FFFF7574H	DSA10	DMA source address register CH10					√	00000000H
FFFF7574H	DSA10L	DMA source address register LCH10				√		0000H
FFFF7576H	DSA10H	DMA source address register HCH10				√		0000H
FFFF7578H	DSC10	DMA source chip select register CH10				√		0001H
FFFF757CH	DNSA10	DMA next source address register CH10					√	00000000H
FFFF757CH	DNSA10L	DMA next source address register LCH10				√		0000H
FFFF757EH	DNSA10H	DMA next source address register HCH10				√		0000H
FFFF7580H	DNSC10	DMA next source chip select register CH10				√		0001H
FFFF7584H	DDA10	DMA destination address register CH10					√	00000000H
FFFF7584H	DDA10L	DMA destination address register LCH10				√		0000H
FFFF7586H	DDA10H	DMA destination address register HCH10				√		0000H
FFFF7588H	DDC10	DMA destination chip select register CH10				√		0001H
FFFF758CH	DNDA10	DMA next destination address register CH10					√	00000000H
FFFF758CH	DNDA10L	DMA next destination address register LCH10				√		0000H
FFFF758EH	DNDA10H	DMA next destination address register HCH10				√		0000H
FFFF7590H	DNDC10	DMA next destination chip select register CH10				√		0001H
FFFF7592H	DTC10	DMA transfer count register CH10				√		0000H
FFFF7594H	DNTC10	DMA next transfer count register CH10				√		0000H
FFFF7596H	DTCC10	DMA transfer count compare register CH10				√		0000H
FFFF7598H	DTCT10	DMA transfer control register CH10				√		0000H
FFFF759AH	DTS10	DMA transfer status register CH10		√	√			00H
FFFF75A0H	DTRS11	DMA transfer request select register CH11				√		0000H
FFFF75A4H	DSA11	DMA source address register CH11					√	00000000H
FFFF75A4H	DSA11L	DMA source address register LCH11				√		0000H
FFFF75A6H	DSA11H	DMA source address register HCH11				√		0000H
FFFF75A8H	DSC11	DMA source chip select register CH11				√		0001H
FFFF75ACH	DNSA11	DMA next source address register CH11					√	00000000H
FFFF75ACH	DNSA11L	DMA next source address register LCH11				√		0000H
FFFF75AEH	DNSA11H	DMA next source address register HCH11				√		0000H
FFFF75B0H	DNSC11	DMA next source chip select register CH11				√		0001H
FFFF75B4H	DDA11	DMA destination address register CH11					√	00000000H
FFFF75B4H	DDA11L	DMA destination address register LCH11				√		0000H
FFFF75B6H	DDA11H	DMA destination address register HCH11				√		0000H

Table 6-8 DMAC setting registers (8/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF75B8H	DDC11	DMA destination chip select register CH11	R/W			√		0001H
FFFF75BCH	DNDA11	DMA next destination address register CH11					√	00000000H
FFFF75BCH	DNDA11L	DMA next destination address register LCH11				√		0000H
FFFF75BEH	DNDA11H	DMA next destination address register HCH11				√		0000H
FFFF75C0H	DNDC11	DMA next destination chip select register CH11				√		0001H
FFFF75C2H	DTC11	DMA transfer count register CH11				√		0000H
FFFF75C4H	DNTC11	DMA next transfer count register CH11				√		0000H
FFFF75C6H	DTCC11	DMA transfer count compare register CH11				√		0000H
FFFF75C8H	DTCT11	DMA transfer control register CH11				√		0000H
FFFF75CAH	DTS11	DMA transfer status register CH11		√	√			00H
FFFF75D0H	DTRS12	DMA transfer request select register CH12				√		0000H
FFFF75D4H	DSA12	DMA source address register CH12					√	00000000H
FFFF75D4H	DSA12L	DMA source address register LCH12				√		0000H
FFFF75D6H	DSA12H	DMA source address register HCH12				√		0000H
FFFF75D8H	DSC12	DMA source chip select register CH12				√		0001H
FFFF75DCH	DNSA12	DMA next source address register CH12					√	00000000H
FFFF75DCH	DNSA12L	DMA next source address register LCH12				√		0000H
FFFF75DEH	DNSA12H	DMA next source address register HCH12				√		0000H
FFFF75E0H	DNSC12	DMA next source chip select register CH12				√		0001H
FFFF75E4H	DDA12	DMA destination address register CH12					√	00000000H
FFFF75E4H	DDA12L	DMA destination address register LCH12				√		0000H
FFFF75E6H	DDA12H	DMA destination address register HCH12				√		0000H
FFFF75E8H	DDC12	DMA destination chip select register CH12				√		0001H
FFFF75ECH	DNDA12	DMA next destination address register CH12					√	00000000H
FFFF75ECH	DNDA12L	DMA next destination address register LCH12				√		0000H
FFFF75EEH	DNDA12H	DMA next destination address register HCH12				√		0000H
FFFF75F0H	DNDC12	DMA next destination chip select register CH12				√		0001H
FFFF75F2H	DTC12	DMA transfer count register CH12				√		0000H
FFFF75F4H	DNTC12	DMA next transfer count register CH12				√		0000H
FFFF75F6H	DTCC12	DMA transfer count compare register CH12				√		0000H
FFFF75F8H	DTCT12	DMA transfer control register CH12				√		0000H
FFFF75FAH	DTS12	DMA transfer status register CH12		√	√			00H
FFFF7600H	DTRS13	DMA transfer request select register CH13				√		0000H

Table 6-8 DMAC setting registers (9/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF7604H	DSA13	DMA source address register CH13	R/W				√	00000000H
FFFF7604H	DSA13L	DMA source address register LCH13				√		0000H
FFFF7606H	DSA13H	DMA source address register HCH13				√		0000H
FFFF7608H	DSC13	DMA source chip select register CH13				√		0001H
FFFF760CH	DNSA13	DMA next source address register CH13					√	00000000H
FFFF760CH	DNSA13L	DMA next source address register LCH13				√		0000H
FFFF760EH	DNSA13H	DMA next source address register HCH13				√		0000H
FFFF7610H	DNSC13	DMA next source chip select register CH13				√		0001H
FFFF7614H	DDA13	DMA destination address register CH13					√	00000000H
FFFF7614H	DDA13L	DMA destination address register LCH13				√		0000H
FFFF7616H	DDA13H	DMA destination address register HCH13				√		0000H
FFFF7618H	DDC13	DMA destination chip select register CH13				√		0001H
FFFF761CH	DNDA13	DMA next destination address register CH13					√	00000000H
FFFF761CH	DNDA13L	DMA next destination address register LCH13				√		0000H
FFFF761EH	DNDA13H	DMA next destination address register HCH13				√		0000H
FFFF7620H	DNDC13	DMA next destination chip select register CH13				√		0001H
FFFF7622H	DTC13	DMA transfer count register CH13				√		0000H
FFFF7624H	DNTC13	DMA next transfer count register CH13				√		0000H
FFFF7626H	DTCC13	DMA transfer count compare register CH13				√		0000H
FFFF7628H	DTCT13	DMA transfer control register CH13				√		0000H
FFFF762AH	DTS13	DMA transfer status register CH13		√	√			00H
FFFF7630H	DTRS14	DMA transfer request select register CH14				√		0000H
FFFF7634H	DSA14	DMA source address register CH14					√	00000000H
FFFF7634H	DSA14L	DMA source address register LCH14				√		0000H
FFFF7636H	DSA14H	DMA source address register HCH14				√		0000H
FFFF7638H	DSC14	DMA source chip select register CH14				√		0001H
FFFF763CH	DNSA14	DMA next source address register CH14					√	00000000H
FFFF763CH	DNSA14L	DMA next source address register LCH14				√		0000H
FFFF763EH	DNSA14H	DMA next source address register HCH14				√		0000H
FFFF7640H	DNSC14	DMA next source chip select register CH14				√		0001H
FFFF7644H	DDA14	DMA destination address register CH14					√	00000000H
FFFF7644H	DDA14L	DMA destination address register LCH14				√		0000H
FFFF7646H	DDA14H	DMA destination address register HCH14				√		0000H
FFFF7648H	DDC14	DMA destination chip select register CH14				√		0001H

Table 6-8 DMAC setting registers (10/10)

Address	Symbol	Function register name	R/W	Operable bits				Initial value
				1	8	16	32	
FFFF764CH	DNDA14	DMA next destination address register CH14	R/W				√	00000000H
FFFF764CH	DNDA14L	DMA next destination address register LCH14				√		0000H
FFFF764EH	DNDA14H	DMA next destination address register HCH14				√		0000H
FFFF7650H	DNDC14	DMA next destination chip select register CH14				√		0001H
FFFF7652H	DTC14	DMA transfer count register CH14				√		0000H
FFFF7654H	DNTC14	DMA next transfer count register CH14				√		0000H
FFFF7656H	DTCC14	DMA transfer count compare register CH14				√		0000H
FFFF7658H	DTCT14	DMA transfer control register CH14				√		0000H
FFFF765AH	DTS14	DMA transfer status register CH14		√	√			00H
FFFF7660H	DTRS15	DMA transfer request select register CH15				√		0000H
FFFF7664H	DSA15	DMA source address register CH15					√	00000000H
FFFF7664H	DSA15L	DMA source address register LCH15				√		0000H
FFFF7666H	DSA15H	DMA source address register HCH15				√		0000H
FFFF7668H	DSC15	DMA source chip select register CH15				√		0001H
FFFF766CH	DNSA15	DMA next source address register CH15					√	00000000H
FFFF766CH	DNSA15L	DMA next source address register LCH15				√		0000H
FFFF766EH	DNSA15H	DMA next source address register HCH15				√		0000H
FFFF7670H	DNSC15	DMA next source chip select register CH15				√		0001H
FFFF7674H	DDA15	DMA destination address register CH15					√	00000000H
FFFF7674H	DDA15L	DMA destination address register LCH15				√		0000H
FFFF7676H	DDA15H	DMA destination address register HCH15				√		0000H
FFFF7678H	DDC15	DMA destination chip select register CH15				√		0001H
FFFF767CH	DNDA15	DMA next destination address register CH15					√	00000000H
FFFF767CH	DNDA15L	DMA next destination address register LCH15				√		0000H
FFFF767EH	DNDA15H	DMA next destination address register HCH15				√		0000H
FFFF7680H	DNDC15	DMA next destination chip select register CH15				√		0001H
FFFF7682H	DTC15	DMA transfer count register CH15				√		0000H
FFFF7684H	DNTC15	DMA next transfer count register CH15				√		0000H
FFFF7686H	DTCC15	DMA transfer count compare register CH15				√		0000H
FFFF7688H	DTCT15	DMA transfer control register CH15				√		0000H
FFFF768AH	DTS15	DMA transfer status register CH15		√	√			00H

Caution If an unmapped address is accessed, a write access is ignored and “0” is returned in response to a read access.

6.4.3 Enabling or disabling writing control registers

The following control registers cannot be written while DMA transfer is enabled. All these registers can always be read, however.

Table 6-9 Enabling/disabling writing control registers

Always writable	DTRC, DNSAnL, DNSAnH, DNSCn, DNDAnL, DNDAnH, DNDCn, DNTCn, DTSn
Writing prohibited while DMA transfer is enabled (DTE = 1) (Operation is not guaranteed if these registers are written.)	DTRSn, DSAnL, DSAnH, DSCn, DDAnL, DDAnH, DDCn, DTCn, DTCCn, DTCTn

6.5 DMA Control Registers

6.5.1 DTRCx – DMA transfer request control register (x = 0, 1)

Access This register can be read or written in 8-bit units.

Address DTRC0: $FFFF7300_H$, DTRC1: $FFFF7500_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
DTRCx ERR	0	0	0	0	0	0	DTRCx ADS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-10 DTRCx registers contents

Bit position	Bit name	Function
7	DTRCx ERR	DMA transfer error status This bit indicates that an error response has been received from the transfer target during DMA transfer. If an error response is received, the DTRCxERR and DTRCxADS bits are set and a system error exception SYSERR is generated. To clear this bit, write “0” to it. 0: No DMA transfer error 1: DMA transfer error
0	DTRCx ADS	DMA transfer aborted This bit indicates that DMA transfer has been aborted by a transfer stop request. In addition, the current DMA transfer can be aborted if the user writes “1” to this bit. 0: DMA transfer not aborted 1: DMA transfer aborted/DMA transfer abort request

6.5.2 DTRSn – DMA transfer request select register

Access This register can be read or written in 16-bit units.

Address DTRS15: FFFF7660$_H$, DTRS14: FFFF7630$_H$, DTRS13: FFFF7600$_H$, DTRS12: FFFF75D0$_H$, DTRS11: FFFF75A0$_H$, DTRS10: FFFF7570$_H$, DTRS9: FFFF7540$_H$, DTRS8: FFFF7510$_H$, DTRS7: FFFF7460$_H$, DTRS6: FFFF7430$_H$, DTRS5: FFFF7400$_H$, DTRS4: FFFF73D0$_H$, DTRS3: FFFF73A0$_H$, DTRS2: FFFF7370$_H$, DTRS1: FFFF7340$_H$, DTRS0: FFFF7310$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	DTR3	DTR2	DTR1	DTR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-11 DTRSn registers contents

Bit position	Bit name	Function
3 to 0	DTR3 to DTR0	DMA transfer request assignment These bits specify assignment of a DMA transfer request to channel n.

DTR3	DTR2	DTR1	DTR0	DMA transfer request
0	0	0	0	Software DMA transfer request
0	0	0	1	Hardware DMA transfer request
Other than above				Setting prohibited

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.
2. The operation is not guaranteed if DTR[3:0] are set to a prohibited status.

6.5.3 DSAnL – DMA source address register L

These registers hold the lower 16 bits of the transfer source address. The higher 16 bits of the transfer source address are accessible via the DSAnH register.

32-bit access The entire 29-bit address is also accessible through the 32-bit register DSAn. DSAn has the same address as the DSAnL register.

If the entire address shall be read, while the DMA transfer of channel is enables (DTSn.DTSnDTE = 1), proceed as follows in order to acquire the correct address:

- Read DSAn twice consecutively and compare the upper 13 address bits SA[28:16] of both read accesses.
- In case the upper 13 address bits SA[28:16] are identical between both read accesses the address SA[28:0] of the second read access represents the correct address.
- In case the upper 13 address bits SA[28:16] are different between both read accesses the address SA[28:0] of the first read access represents the correct address.

Access This register can be read or written in 16-bit units.

Address DSA15L: FFFF7664$_H$, DSA14L: FFFF7634$_H$, DSA13L: FFFF7604$_H$, DSA12L: FFFF75D4$_H$, DSA11L: FFFF75A4$_H$, DSA10L: FFFF7574$_H$, DSA9L: FFFF7544$_H$, DSA8L: FFFF7514$_H$, DSA7L: FFFF7464$_H$, DSA6L: FFFF7434$_H$, DSA5L: FFFF7404$_H$, DSA4L: FFFF73D4$_H$, DSA3L: FFFF73A4$_H$, DSA2L: FFFF7374$_H$, DSA1L: FFFF7344$_H$, DSA0L: FFFF7314$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
SA15	SA14	SA13	SA12	SA11	SA10	SA9	SA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-12 DSAnL registers contents

Bit position	Bit name	Function
15 to 0	SA15 to SA0	DMA source address lower 16 bits If this register is referenced during DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with DSAnH via the 32-bit register DSAn. If the NSAV bit of the DNSAnH register is not set (to “1”) when DMA transfer has been completed, the values of these bits return to the values when DMA transfer was started.

Cautions

1. Writing this register is prohibited while DMA transfer is enabled (DTSn.DTSnDTE = 1). If it is written, correct DMA operation is not guaranteed.
2. Set an address by accessing in 32-bit units while the DTSnDTE bit is “0” in order to avoid data being transferred from an address that has not been completely set.
3. DMA transfer of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as follows (x indicates any bit).

   The operation is not guaranteed if a setting other than the following is made.

Data size	SA3	SA2	SA1	SA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.4 DSAnH – DMA source address register H

These registers hold the higher 13 bits of the transfer source address.
The lower 16 bits of the transfer source address are accessible via the DSAnL register.

Note The entire 29-bit transfer source address is accessible via the 32-bit register DSAn. Refer to the description of the DSAnL for details about the 32-bit access.

Access This register can be read or written in 16-bit units.

Address DSA15H: $FFFF7666_H$, DSA14H: $FFFF7636_H$, DSA13H: $FFFF7606_H$, DSA12H: $FFFF75D6_H$, DSA11H: $FFFF75A6_H$, DSA10H: $FFFF7576_H$, DSA9H: $FFFF7546_H$, DSA8H: $FFFF7516_H$, DSA7H: $FFFF7466_H$, DSA6H: $FFFF7436_H$, DSA5H: $FFFF7406_H$, DSA4H: $FFFF73D6_H$, DSA3H: $FFFF73A6_H$, DSA2H: $FFFF7376_H$, DSA1H: $FFFF7346_H$, DSA0H: $FFFF7316_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	SA28	SA27	SA26	SA25	SA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-13 DSAnH registers contents

Bit position	Bit name	Function
12 to 0	SA28 to SA16	DMA source address If this register is referenced during DMA transfer, the address from which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with DSAnL in 32-bit units. If the NSAV bit of the DNSAnH register is not set (to “1”) when DMA transfer has been completed, the values of these bits return to the values when DMA transfer was started.

Cautions

1. Writing this register is prohibited while DMA transfer is enabled (DTSn.DTSnDTE = 1). If it is written, correct DMA operation is not guaranteed.
2. Set an address by accessing in 32-bit units while the DTSnDTE bit is “0” in order to avoid data being transferred from an address that has not been completely set.

6.5.5 DSCn – DMA source chip select register

Access This register can be read or written in 16-bit units.

Address DSC15: FFFF7668$_H$, DSC14: FFFF7638$_H$, DSC13: FFFF7608$_H$, DSC12: FFFF75D8$_H$, DSC11: FFFF75A8$_H$, DSC10: FFFF7578$_H$, DSC9: FFFF7548$_H$, DSC8: FFFF7518$_H$, DSC7: FFFF7468$_H$, DSC6: FFFF7438$_H$, DSC5: FFFF7408$_H$, DSC4: FFFF73D8$_H$, DSC3: FFFF73A8$_H$, DSC2: FFFF7378$_H$, DSC1: FFFF7348$_H$, DSC0: FFFF7318$_H$

Initial Value 0001$_H$

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	SCS1	SCS0	SCSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-14 DSCn registers contents

Bit position	Bit name	Function
2 1 0	SCS1 SCS0 SCSE	DMA source chip select These bits specify an area to be selected as the transfer source of channel n.

SCS1	SCS0	SCSE	Selected area
0	0	1	Data-Flash, external memory area, peripheral I/O area
0	1	0	Code flash, Data RAM
Other than above			Setting prohibited

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.
2. Set the SCS0 and SCSE bits so that only one of them is “1”. If both of these bits are set to 1, the operation is not guaranteed.
3. Be sure to set the SCS1 bit to “0”.

6.5.6 DNSAnL – DMA next source address register L

Access This register can be read or written in 16-bit units.

Address DNSA15L: FFFF766C$_H$, DNSA14L: FFFF763C$_H$, DNSA13L: FFFF760C$_H$, DNSA12L: FFFF75DC$_H$, DNSA11L: FFFF75AC$_H$, DNSA10L: FFFF757C$_H$, DNSA9L: FFFF754C$_H$, DNSA8L: FFFF751C$_H$, DNSA7L: FFFF746C$_H$, DNSA6L: FFFF743C$_H$, DNSA5L: FFFF740C$_H$, DNSA4L: FFFF73DC$_H$, DNSA3L: FFFF73AC$_H$, DNSA2L: FFFF737C$_H$, DNSA1L: FFFF734C$_H$, DNSA0L: FFFF731C$_H$

Access 0000$_H$

15	14	13	12	11	10	9	8
NAS15	NAS14	NAS13	NAS12	NAS11	NAS10	NAS9	NAS8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
NAS7	NAS6	NAS5	NAS4	NAS3	NAS2	NAS1	NAS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-15 DNSAnL registers contents

Bit position	Bit name	Function
15 to 0	NSA15 to NSA0	DMA next source address These bits set the lower 16 bits of the transfer source address when data is next transferred from channel n.

Caution DMA transfer of misaligned data is not supported. The lower 4 bits of the address corresponding to data size are as follows (x indicates any bit).

The operation is not guaranteed if a setting other than the following is made.

Data size	NSA3	NSA2	NSA1	NSA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.7 DNSAnH – DMA next source address register H

Access This register can be read or written in 16-bit units.

Address DNSA15H: FFFF766E$_H$, DNSA14H: FFFF763E$_H$, DNSA13H: FFFF760E$_H$, DNSA12H: FFFF75DE$_H$, DNSA11H: FFFF75AE$_H$, DNSA10H: FFFF757E$_H$, DNSA9H: FFFF754E$_H$, DNSA8H: FFFF751E$_H$, DNSA7H: FFFF746E$_H$, DNSA6H: FFFF743E$_H$, DNSA5H: FFFF740E$_H$, DNSA4H: FFFF73DE$_H$, DNSA3H: FFFF73AE$_H$, DNSA2H: FFFF737E$_H$, DNSA1H: FFFF734E$_H$, DNSA0H: FFFF731E$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
NSAV	0	0	NSA28	NSA27	NSA26	NSA25	NSA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
NSA23	NSA22	NSA21	NSA20	NSA19	NSA18	NSA17	NSA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-16 DNSAnH registers contents

Bit position	Bit name	Function
15	NSAV	DMA next source address valid This bit controls whether to copy an address from the DMA next source address register to the DMA source address register when DMA transfer has been completed. It is cleared when the address has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
12 to 0	NSA28 to NSA16	DMA next source address These bits specify the higher 13 bits of the transfer source address for the next transfer of channel n.

6.5.8 DNSCn – DMA next source chip select register

Access This register can be read or written in 16-bit units.

Address DNSC15: $FFFF7670_H$, DNSC14: $FFFF7640_H$, DNSC13: $FFFF7610_H$, DNSC12: $FFFF75E0_H$, DNSC11: $FFFF75B0_H$, DNSC10: $FFFF7580_H$, DNSC9: $FFFF7550_H$, DNSC8: $FFFF7520_H$, DNSC7: $FFFF7470_H$, DNSC6: $FFFF7440_H$, DNSC5: $FFFF7410_H$, DNSC4: $FFFF73E0_H$, DNSC3: $FFFF73B0_H$, DNSC2: $FFFF7380_H$, DNSC1: $FFFF7350_H$, DNSC0: $FFFF7320_H$

Initial Value 0001_H

15	14	13	12	11	10	9	8
NSCV	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	NSCS1	NSCS0	NSCSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-17 DNSCn registers contents

Bit position	Bit name	Function
15	NSCV	DMA next source address select valid This bit controls whether to copy a chip select signal from the DMA next source chip select register to the DMA source chip select register when DMA transfer has been completed. It is cleared when the chip select signal has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
2 1 0	NSCS1 NSCS0 NSCSE	DMA next source chip select These bits specify an area to be selected as the transfer source for the next transfer of channel n.

NSCS1	NSCS0	NSCSE	Selected area
0	0	1	Data-Flash, external memory area, peripheral I/O area
0	1	0	Code flash, Data RAM
Other than above			Setting prohibited

Cautions

1. Set the NSCS0 and NSCSE bits so that only one of them is 1. If both of these bits are set to 1, the operation is not guaranteed.
2. Be sure to set the NCSC1 bit to “0”.

6.5.9 DDAnL – DMA destination address register L

These registers hold the lower 16 bits of the transfer destination address. The higher 16 bits of the transfer destination address are accessible via the DDAnH register.

32-bit access

The entire 29-bit address is also accessible through the 32-bit register DDAn. DDAn has the same address as the DDAnL register.

If the entire address shall be read, while the DMA transfer of channel is enables (DTSn.DTSnDTE = 1), proceed as follows in order to acquire the correct address:

- Read DDAn twice consecutively and compare the upper 13 address bits DA[28:16] of both read accesses.
- In case the upper 13 address bits DA[28:16] are identical between both read accesses the address DA[28:0] of the second read access represents the correct address.
- In case the upper 13 address bits DA[28:16] are different between both read accesses the address DA[28:0] of the first read access represents the correct address.

Access

This register can be read or written in 16-bit units.

Address

DDA15L: FFFF7674$_H$, DDA14L: FFFF7644$_H$, DDA13L: FFFF7614$_H$, DDA12L: FFFF75E4$_H$, DDA11L: FFFF75B4$_H$, DDA10L: FFFF7584$_H$, DDA9L: FFFF7554$_H$, DDA8L: FFFF7524$_H$, DDA7L: FFFF7474$_H$, DDA6L: FFFF7444$_H$, DDA5L: FFFF7414$_H$, DDA4L: FFFF73E4$_H$, DDA3L: FFFF73B4$_H$, DDA2L: FFFF7384$_H$, DDA1L: FFFF7354$_H$, DDA0L: FFFF7324$_H$

Initial Value

0000_H

15	14	13	12	11	10	9	8
DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-18 DDAnL registers contents

Bit position	Bit name	Function
15 to 0	DA15 to DA0	DMA destination address If this register is referenced during DMA transfer, the address to which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with DDAnH in 32-bit units. If the NDAV bit of the DNDAnH register is not set (to “1”) when DMA transfer has been completed, the values of these bits return to the values when DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTSn.DTSnDTE = 1). If they are written, the operation is not guaranteed.
2. Set an address by accessing in 32-bit units while the DTSnDTE bit is “0” in order to avoid data being transferred from an address that has not been completely set.
3. If an error occurs in the transfer target in the read cycle of DMA transfer, the write cycle is not executed but the destination address is updated.
4. DMA transfer of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as follows (x indicates any bit).

   The operation is not guaranteed if a setting other than the following is made.

Data size	DA3	DA2	DA1	DA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.10 DDAnH – DMA destination address register H

These registers hold the higher 13 bits of the transfer destination address. The lower 16 bits of the transfer destination address are accessible via the DDAnL register.

Note The entire 29-bit transfer destination address is accessible via the 32-bit register DDAn. Refer to the description of the DDAnL for details about the 32-bit access.

Access This register can be read or written in 16-bit units.

Address DDA15H: $FFFF7676_H$, DDA14H: $FFFF7646_H$, DDA13H: $FFFF7616_H$, DDA12H: $FFFF75E6_H$, DDA11H: $FFFF75B6_H$, DDA10H: $FFFF7586_H$, DDA9H: $FFFF7556_H$, DDA8H: $FFFF7526_H$, DDA7H: $FFFF7476_H$, DDA6H: $FFFF7446_H$, DDA5H: $FFFF7416_H$, DDA4H: $FFFF73E6_H$, DDA3H: $FFFF73B6_H$, DDA2H: $FFFF7386_H$, DDA1H: $FFFF7356_H$, DDA0H: $FFFF7326_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	DA28	DA27	DA26	DA25	DA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-19 DDAnH registers contents

Bit position	Bit name	Function
12 to 0	DA28 to DA16	DMA destination address If this register is referenced during DMA transfer, the address to which data is to be transferred next can be read. When referencing this register, it is recommended to access this register together with DDAnL in 32-bit units. If the NDAV bit of the DNDAnH register is not set (to “1”) when DMA transfer has been completed, the values of these bits return to the values when DMA transfer was started.

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTSn.DTSnDTE bit = 1). If they are written, the operation is not guaranteed.
2. Set an address by accessing in 32-bit units while the DTSnDTE bit is “0” in order to avoid data being transferred from an address that has not been completely set.
3. If an error occurs in the transfer target in the read cycle of DMA transfer, the write cycle is not executed but the destination address is updated.

6.5.11 DDCn – DMA destination chip select register

Access This register can be read or written in 16-bit units.

Address DDC15: $FFFF7678_H$, DDC14: $FFFF7648_H$, DDC13: $FFFF7618_H$, DDC12: $FFFF75E8_H$, DDC11: $FFFF75B8_H$, DDC10: $FFFF7588_H$, DDC9: $FFFF7558_H$, DDC8: $FFFF7528_H$, DDC7: $FFFF7478_H$, DDC6: $FFFF7448_H$, DDC5: $FFFF7418_H$, DDC4: $FFFF73E8_H$, DDC3: $FFFF73B8_H$, DDC2: $FFFF7388_H$, DDC1: $FFFF7358_H$, DDC0: $FFFF7328_H$

Initial Value 0001_H

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	DCS1	DCS0	DCSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-20 DDCn registers contents

Bit position	Bit name	Function
2 1 0	DCS1 DCS0 DCSE	DMA destination chip select These bits specify an area to be selected as the transfer destination of channel n.

DCS1	DCS0	DCSE	Selected area
0	0	1	Data-Flash, external memory area, peripheral I/O area
0	1	0	Code flash, Data RAM
Other than above			Setting prohibited

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.
2. Set the DCS0 and DCSE bits so that only one of them is “1”. If both of these bits are set to 1, the operation is not guaranteed.
3. Be sure to set the DCS1 bit to “0”.

6.5.12 DNDAnL – DMA next destination address register L

Access This register can be read or written in 16-bit units.

Address DNDA15L: FFFF767C$_H$, DNDA14L: FFFF764C$_H$, DNDA13L: FFFF761C$_H$, DNDA12L: FFFF75EC$_H$, DNDA11L: FFFF75BC$_H$, DNDA10L: FFFF758C$_H$, DNDA9L: FFFF755C$_H$, DNDA8L: FFFF752C$_H$, DNDA7L: FFFF747C$_H$, DNDA6L: FFFF744C$_H$, DNDA5L: FFFF741C$_H$, DNDA4L: FFFF73EC$_H$, DNDA3L: FFFF73BC$_H$, DNDA2L: FFFF738C$_H$, DNDA1L: FFFF735C$_H$, DNDA0L: FFFF732C$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
NDA15	NDA14	NDA13	NDA12	NDA11	NDA10	NDA9	NDA8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
NDA7	NDA6	NDA5	NDA4	NDA3	NDA2	NDA1	NDA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-21 DNDAnL registers contents

Bit position	Bit name	Function
15 to 0	NDA15 to NDA0	DMA next destination address These bits specify the lower 16 bits of the transfer destination address when the next transfer of channel n is executed.

Caution DMA transfer of misaligned data is not supported. The lower 4 bits of an address corresponding to the transfer data size are as follows (x indicates any bit).

The operation is not guaranteed if a setting other than the following is made.

Data size	NDA3	NDA2	NDA1	NDA0
8 bits	x	x	x	x
16 bits	x	x	x	0
32 bits	x	x	0	0
128 bits	0	0	0	0

6.5.13 DNDAnH – DMA next destination address register H

Access This register can be read or written in 16-bit units.

Address DNDA15H: FFFF767E$_H$, DNDA14H: FFFF764E$_H$, DNDA13H: FFFF761E$_H$, DNDA12H: FFFF75EE$_H$, DNDA11H: FFFF75BE$_H$, DNDA10H: FFFF758E$_H$, DNDA9H: FFFF755E$_H$, DNDA8H: FFFF752E$_H$, DNDA7H: FFFF747E$_H$, DNDA6H: FFFF744E$_H$, DNDA5H: FFFF741E$_H$, DNDA4H: FFFF73EE$_H$, DNDA3H: FFFF73BE$_H$, DNDA2H: FFFF738E$_H$, DNDA1H: FFFF735E$_H$, DNDA0H: FFFF732E$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
NDAV	0	0	NDA28	NDA27	NDA26	NDA25	NDA24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
NDA23	NDA22	NDA21	NDA20	NDA19	NDA18	NDA17	NDA16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-22 DNDAnH registers contents

Bit position	Bit name	Function
15	NDAV	DMA next destination address valid This bit specifies whether to copy an address from the DMA next destination address register to the DMA destination address register when DMA transfer has been completed. It is cleared when the address has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
12 to 0	NDA28 to NDA16	DMA next destination address These bits specify the higher 13 bits of the transfer destination address when the next transfer of channel n is executed.

6.5.14 DNDCn – DMA next destination chip select register

Access This register can be read or written in 16-bit units.

Address DNDC15: $FFFF7680_H$, DNDC14: $FFFF7650_H$, DNDC13: $FFFF7620_H$, DNDC12: $FFFF75F0_H$, DNDC11: $FFFF75C0_H$, DNDC10: $FFFF7590_H$, DNDC9: $FFFF7560_H$, DNDC8: $FFFF7530_H$, DNDC7: $FFFF7480_H$, DNDC6: $FFFF7450_H$, DNDC5: $FFFF7420_H$, DNDC4: $FFFF73F0_H$, DNDC3: $FFFF73C0_H$, DNDC2: $FFFF7390_H$, DNDC1: $FFFF7360_H$, DNDC0: $FFFF7330_H$

Initial Value 0001_H

15	14	13	12	11	10	9	8
NDCV	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	NDCS1	NDCS0	NDCSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-23 DNDCn registers contents

Bit position	Bit name	Function
15	NDCV	DMA next destination chip select valid This bit specifies whether to copy a chip select signal from the DMA next destination chip select register to the DMA destination chip select register when DMA transfer is completed. It is cleared when the chip select signal is copied. 0: Does not copy/copying completed 1: Copies/copying not completed
2 1 0	NDCS1 NDCS0 NDCSE	DMA next destination chip select These bits specify an area to be selected as the transfer destination for the next transfer of channel n.

NDCS1	NDCS0	NDCSE	Selected area
0	0	1	Data-Flash, external memory area, peripheral I/O area
0	1	0	Code flash, Data RAM
Other than above			Setting prohibited

Cautions

1. Set the NDCS0 and NDCSE bits so that only one of them is “1”. If both of these bits are set to 1, the operation is not guaranteed.
2. Be sure to set the NDCS1 bit to “0”.

6.5.15 DTCn – DMA transfer count register

Access This register can be read or written in 16-bit units.

Address DTC15: FFFF7682$_H$, DTC14: FFFF7652$_H$, DTC13: FFFF7622$_H$, DTC12: FFFF75F2$_H$, DTC11: FFFF75C2$_H$, DTC10: FFFF7592$_H$, DTC9: FFFF7562$_H$, DTC8: FFFF7532$_H$, DTC7: FFFF7482$_H$, DTC6: FFFF7452$_H$, DTC5: FFFF7422$_H$, DTC4: FFFF73F2$_H$, DTC3: FFFF73C2$_H$, DTC2: FFFF7392$_H$, DTC1: FFFF7362$_H$, DTC0: FFFF7332$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
0	DTC14	DTC13	DTC12	DTC11	DTC10	DTC9	DTC8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DTC7	DTC6	DTC5	DTC4	DTC3	DTC2	DTC1	DTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-24 DTCn registers contents

Bit position	Bit name	Function
14 to 0	DTC14 to DTC0	DMA transfer count These bits specify the number of times of DMA transfers (DMA transfer count) for channel n. When this register is referenced during DMA transfer, the remaining number of times DMA transfer to be executed can be read. If the NTCV bit of the DNTCn register is not set (to “1”), these bits hold the values when DMA transfer has been completed (0000H).

DTC[14:0]	The operation
0000H	Transfer executed 32,768 times or until completion of transfer
0001H	Transfer executed once or transfer to be executed once
:	:
7FFFH	Transfer executed 32,767 times or 32,767 times of transfer to be executed

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.
2. If an error occurs in the transfer target in the read cycle of DMA transfer, the write cycle is not executed but the destination address is updated.

6.5.16 DNTCn – DMA next transfer count register

Access This register can be read or written in 16-bit units.

Address DNCT15: $FFFF7684_H$, DNCT14: $FFFF7654_H$, DNCT13: $FFFF7624_H$, DNCT12: $FFFF75F4_H$, DNCT11: $FFFF75C4_H$, DNCT10: $FFFF7594_H$, DNCT9: $FFFF7564_H$, DNCT8: $FFFF7534_H$, DNCT7: $FFFF7484_H$, DNCT6: $FFFF7454_H$, DNCT5: $FFFF7424_H$, DNCT4: $FFFF73F4_H$, DNCT3: $FFFF73C4_H$, DNCT2: $FFFF7394_H$, DNCT1: $FFFF7364_H$, DNCT0: $FFFF7334_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
NTCV	NDTC14	NDTC13	NDTC12	NDTC11	NDTC10	NDTC9	NDTC8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
NDTC7	NDTC6	NDTC5	NDTC4	NDTC3	NDTC2	NDTC1	NDTC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-25 DNTCn registers contents

Bit position	Bit name	Function
15	NTCV	DMA next transfer count valid This bit controls whether to copy the DMA transfer count from the DMA next transfer count register to the DMA count register when DMA transfer has been completed. It is cleared when the DMA transfer count has been copied. 0: Does not copy/copying completed 1: Copies/copying not completed
14 to 0	NDTC14 to NDTC0	DMA next transfer count These bits specify the DMA next transfer count for channel n.

DTC[14:0]	The operation
0000H	Transfer executed 32,768 times
0001H	Transfer executed once
:	:
7FFFH	Transfer executed 32,767 times

6.5.17 DTCCn – DMA transfer count compare register

Access This register can be read or written in 16-bit units.

Address DTCC15: $FFFF7686_H$, DTCC14: $FFFF7656_H$, DTCC13: $FFFF7626_H$, DTCC12: $FFFF75F6_H$, DTCC11: $FFFF75C6_H$, DTCC10: $FFFF7596_H$, DTCC9: $FFFF7566_H$, DTCC8: $FFFF7536_H$, DTCC7: $FFFF7486_H$, DTCC6: $FFFF7456_H$, DTCC5: $FFFF7426_H$, DTCC4: $FFFF73F6_H$, DTCC3: $FFFF73C6_H$, DTCC2: $FFFF7396_H$, DTCC1: $FFFF7366_H$, DTCC0: $FFFF7336_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	DTCC[14:8]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DTCC[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-26 DTCCn registers contents

Bit position	Bit name	Function
14 to 0	DTCC[14:0]	DMA transfer count comparison DTCC[14:0] specify a value to be compared with the value of the DMA transfer count register of channel n DTCn. If DTCn and DTCCn match, an interrupt is generated. DTCCn can be used to set a trigger for setting the next address.

DTCC[14:0]	The operation
0000H	Not compared
0001H	Interrupt is generated when DTC = 0001H.
:	:
7FFFH	Interrupt is generated when DTC = 7FFFH.

Caution Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.

6.5.18 DTCTn – DMA transfer control register

Access This register can be read or written in 16-bit units.

Address DTCT15: FFFF7688$_H$, DTCT14: FFFF7658$_H$, DTCT13: FFFF7628$_H$, DTCT12: FFFF75F8$_H$, DTCT11: FFFF75C8$_H$, DTCT10: FFFF7598$_H$, DTCT9: FFFF7568$_H$, DTCT8: FFFF7538$_H$, DTCT7: FFFF7488$_H$, DTCT6: FFFF7458$_H$, DTCT5: FFFF7428$_H$, DTCT4: FFFF73F8$_H$, DTCT3: FFFF73C8$_H$, DTCT2: FFFF7398$_H$, DTCT1: FFFF7368$_H$, DTCT0: FFFF7338$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
0	DS1	DS0	MLE	INF	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
SACM1	SACM0	DACM1	DACM0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-27 DTCTn registers contents (1/2)

Bit position	Bit name	Function
14 13	DS1 DS0	DMA transfer data size These bits specify the DMA transfer data size of channel n. **DS1 / DS0 / Transfer data size** 0 / 0 / 8 bits 0 / 1 / 16 bits 1 / 0 / 32 bits 1 / 1 / 128 bits
11	INF	When this bit is set to “1”, the next register valid bit (NSAV, NSCV, NDAV, NDCV, or NTCV) is not cleared even when the destination, source, or transfer count is copied from the DMA next register to the DMA current register.
12	MLE	Multi-link enable This bit specifies whether to acknowledge the next DMA transfer request, even if the TC bit is not cleared (to “0”) after DMA transfer has been completed. If this bit is set (to “1”), the DTE bit is not cleared upon completion of DMA transfer. Even if the TC bit is not cleared, DMA transfer is executed if a DMA transfer request is issued. 0: Clears DTE bit upon completion of DMA transfer. 1: Does not clear DTE bit upon completion of DMA transfer.
7 6	SACM1 SACM0	DMA transfer source address counting direction These bits specify the direction in which the transfer source address of channel n is to be counted. **SACM1 / SACM0 / Counting direction** 0 / 0 / Incremented 0 / 1 / Decremented 1 / 0 / Fixed 1 / 1 / Setting prohibited

Table 6-27 DTCTn registers contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>5
4</td><td>DACM1
DACM0</td><td>DMA transfer destination address counting direction
These bits specify the direction in which the transfer destination address of channel n is to be counted.
<table>
<tr><th>DACM1</th><th>DACM0</th><th>Counting direction</th></tr>
<tr><td>0</td><td>0</td><td>Incremented</td></tr>
<tr><td>0</td><td>1</td><td>Decremented</td></tr>
<tr><td>1</td><td>0</td><td>Fixed</td></tr>
<tr><td>1</td><td>1</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
</table>

Cautions

1. Writing these bits is prohibited while DMA transfer is enabled (DTE bit = 1). If they are written, the operation is not guaranteed.
2. The operation cannot be guaranteed if the SACM[1:0] and DACM[1:0] bits are set to a prohibited status
3. Be sure to set bits 11 and 0 to “0”.

6.5.19 DTSn – DMA transfer status register

Access This register can be read or written in 8- or 1-bit units.

Address DTS15: FFFF768A$_H$, DTS14: FFFF765A$_H$, DTS13: FFFF762A$_H$, DTS12: FFFF75FA$_H$, DTS11: FFFF75CA$_H$, DTS10: FFFF759A$_H$, DTS9: FFFF756A$_H$, DTS8: FFFF753A$_H$, DTS7: FFFF748A$_H$, DTS6: FFFF745A$_H$, DTS5: FFFF742A$_H$, DTS4: FFFF73FA$_H$, DTS3: FFFF73CA$_H$, DTS2: FFFF739A$_H$, DTS1: FFFF736A$_H$, DTS0: FFFF733A$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
DTSnTC	DTSnDT	0	0	DTSnER	DTSnDR	DTSnSR	DTSnDTE
R/W	R/W	R	R	R	R	R/W	R/W

Table 6-28 DTSn registers contents (1/2)

Bit position	Bit name	Function
7	DTSnTC	DMA transfer end status This bit indicates that DMA transfer has been completed. Write “0” to this bit to clear it after reading “1” from it. It is recommended to write this bit using bit manipulation such as CLR1. 0: DMA transfer not completed 1: DMA transfer completed
6	DTSnDT	DT DMA transfer status This bit indicates that a DMA transfer request has been acknowledged and that DMA transfer is in progress. It is not set (to “1”) when only a DMA transfer request is issued. This bit is cleared (to “0”) when DMA transfer has been completed. If the DTE bit is “0”, this bit can be cleared by the user. (It can also be written at the same time as the DTE bit.) 0: DMA transfer request acknowledged 1: DMA transfer in progress
3	DTSnER	DMA transfer error flag This bit indicates that a DMA transfer error has occurred in channel n. It is cleared (to “0”) when the ERR bit of the DTRC register is cleared. Note that this bit is read-only. 0: No DMA transfer error 1: DMA transfer error
2	DTSnDR	Hardware DMA transfer request flag This bit indicates that channel n has a hardware DMA transfer request. It is cleared (to “0”) when the hardware DMA transfer request is deasserted. This bit operates regardless of the status of the DTE bit. It is not set (to “1”) by a software DMA transfer request, or by a hardware DMA transfer request when a software DMA transfer request is selected by the DMA transfer request select register. Note that this bit is read-only. 0: No hardware DMA transfer request 1: Hardware DMA transfer request
1	DTSnSR	Software DMA transfer request This bit selects a software DMA transfer request. If a software DMA transfer request is selected by the DMA transfer request select register, writing “1” to this bit and the DTE bit starts DMA transfer. This bit is automatically cleared (to “0”) when DMA transfer has been completed. Writing “0” to this bit aborts DMA transfer. 0: No software DMA transfer request 1: Software DMA transfer request

Table 6-28 DTSn registers contents (2/2)

Bit position	Bit name	Function
0	DTSnDTE	DMA transfer enable This bit enables or disables DMA transfer. DMA transfer is executed if “1” is written to this bit and a DMA transfer request is issued. This bit is automatically cleared (to “0”) if the MLE bit is “0” when DMA transfer has been completed. DMA transfer is aborted if “0” is written to this bit during DMA transfer. 0: Disables DMA transfer 1: Enables DMA transfer

6.6 DMAC Function Details

6.6.1 DMAC transfer setting flow

Figure 6-4 “DMAC transfer setting flow” shows the flow for setting DMAC transfer.

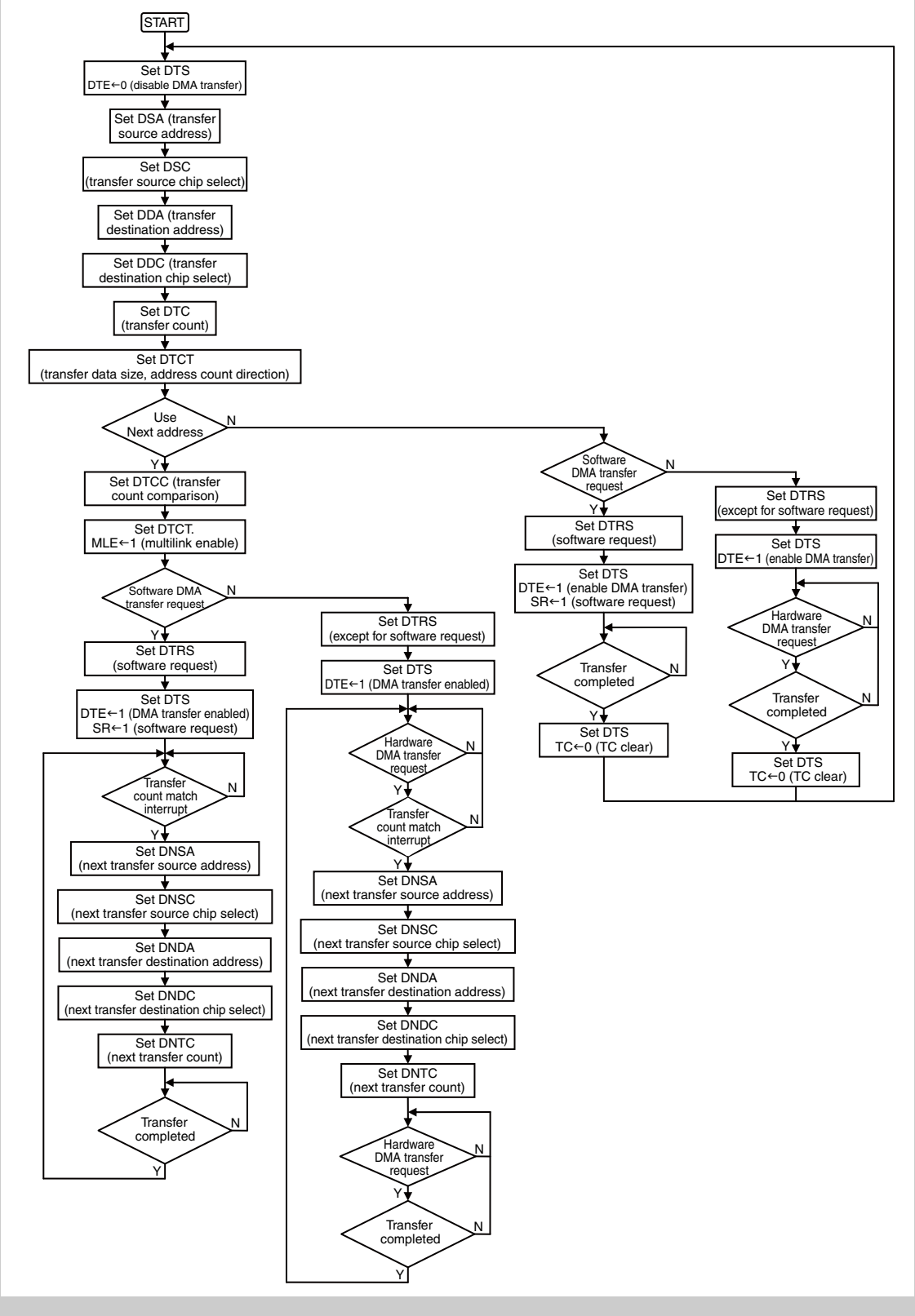


Figure 6-4 DMAC transfer setting flow

6.6.2 DMAC transfer modes

A single-transfer mode and a single-step transfer mode are supported as transfer modes.

In either mode, transfer is executed in two cycles (dual address transfer) and therefore, a read cycle and a write cycle are generated each time transfer is executed. In the case of 128-bit transfer, the read cycle is generated four times and the write cycle is generated four times, in that order.

Note that the bus is not locked. Consequently, a CPU cycle may interrupt between the read and write cycles, and between the four read cycles and four write cycles during 128-bit transfer.

(1) Single transfer mode (when hardware DMA transfer request is generated)

When a hardware DMA transfer request is acknowledged, data of the transfer data size (8, 16, 32, or 128 bits) is transferred. Each time transfer has been executed, the bus is released and the DMA controller waits for a DMA transfer request. At this time, the acknowledge n signal that indicates that the hardware DMA transfer request has been acknowledged is also output.

Each time a hardware DMA transfer request has been acknowledged, transfer is executed once. This operation is repeated the number of times specified by the DMA transfer count register n (DTCn).

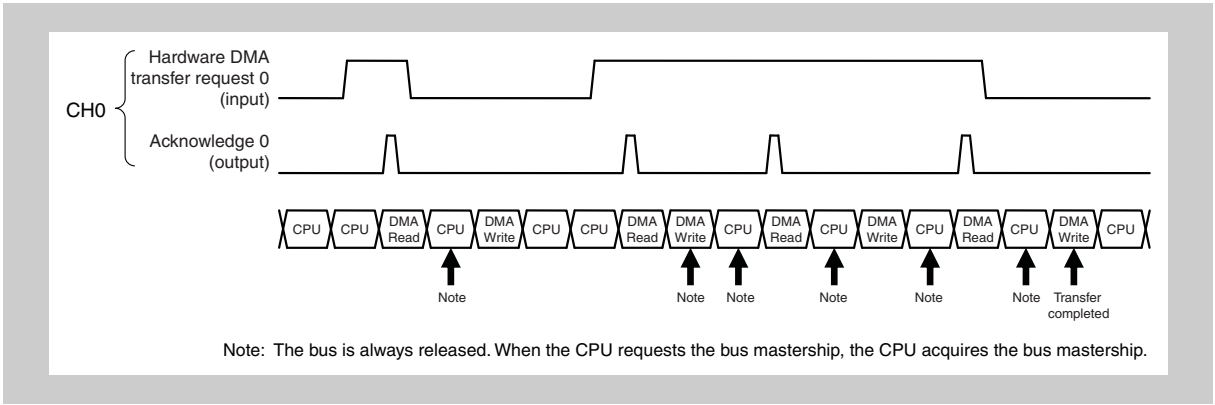
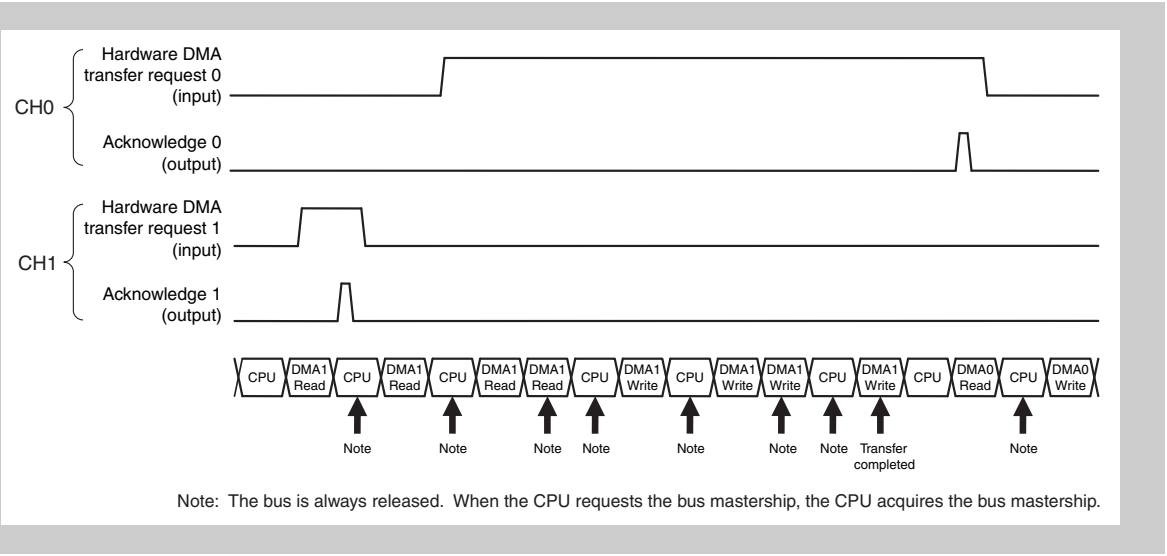


Figure 6-5 Example of single transfer (8/16/32 bits)



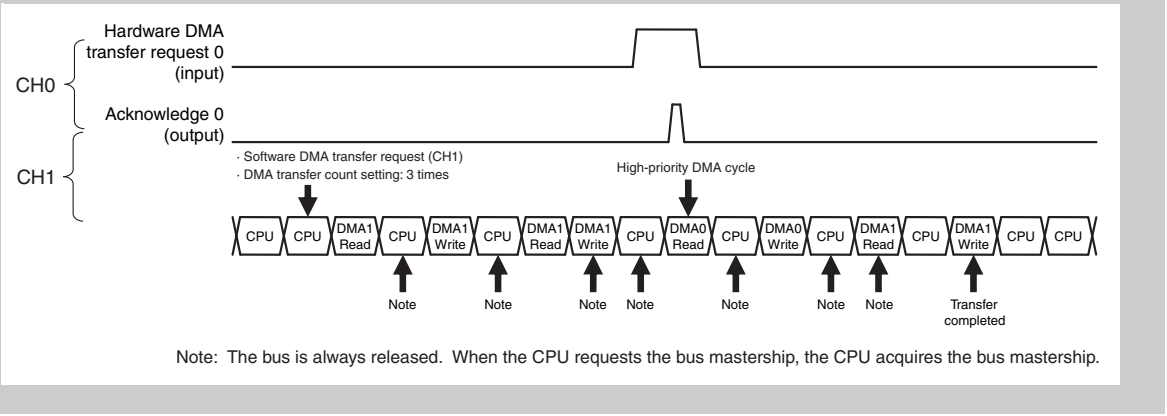
Note: The bus is always released. When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-6 Example of single transfer (128 bits, DMA channel priority: CH0 (high) > CH1 (low))

(2) Single-step transfer mode (when software DMA transfer request is generated)

When a software DMA transfer request is acknowledged, data of the transfer data size (8, 16, 32, or 128 bits) is transferred. Each time transfer has been executed, the bus is released. At this time, the acknowledge n signal that indicates that a hardware DMA transfer request has been acknowledged is not output.

Once a software DMA transfer request has been acknowledged, this operation is repeated the number of times specified by the DMA transfer count register n (DTCn). Because the priority is identified each time transfer is executed, the DMA cycle of a channel having the higher priority may interrupt.



Note: The bus is always released. When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-7 Example of single-step transfer (8/16/32 bits, DMA channel priority: CH0 (high) > CH1 (low))

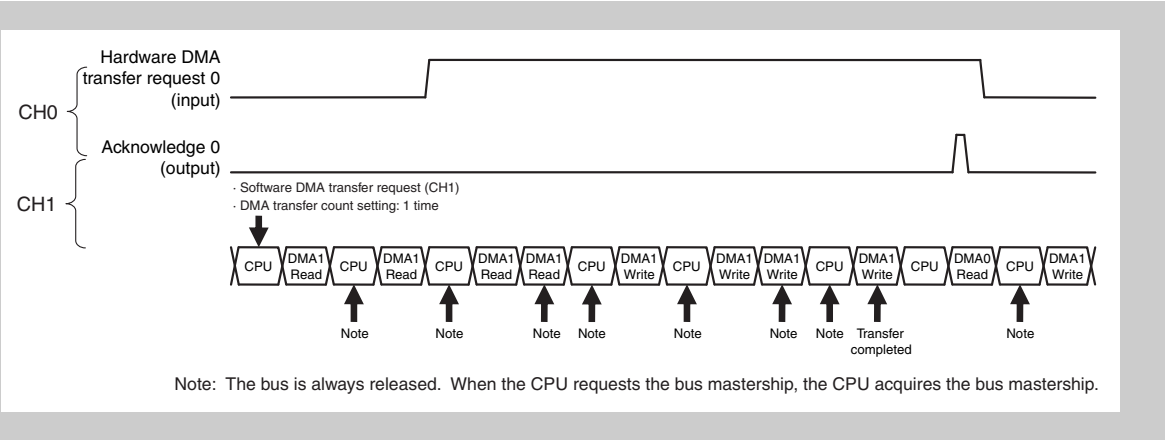


Figure 6-8 Example of single-step transfer (128 bits, DMA channel priority: CH0 (high) > CH1 (low))

6.6.3 DMAC channel priority control

The priority of each channel is fixed and is as follows.

DMAC0 > DMAC1

DMAC0: CH0 > CH1 > CH2 > CH3 > CH4 > CH5 > CH6 > CH7

DMAC1: CH8 > CH9 > CH10 > CH11 > CH12 > CH13 > CH14 > CH15

If another DMA transfer request with a high priority is generated, the DMA transfer request with the higher priority always takes precedence. When a software DMA transfer request is generated, the bus is also released each time a DMA cycle has been completed. If a DMA transfer request with a high priority is generated, therefore, the DMA transfer request with the higher priority always takes precedence.

An example where another DMA transfer request with a high priority is generated when DMA transfer is executed is shown below.

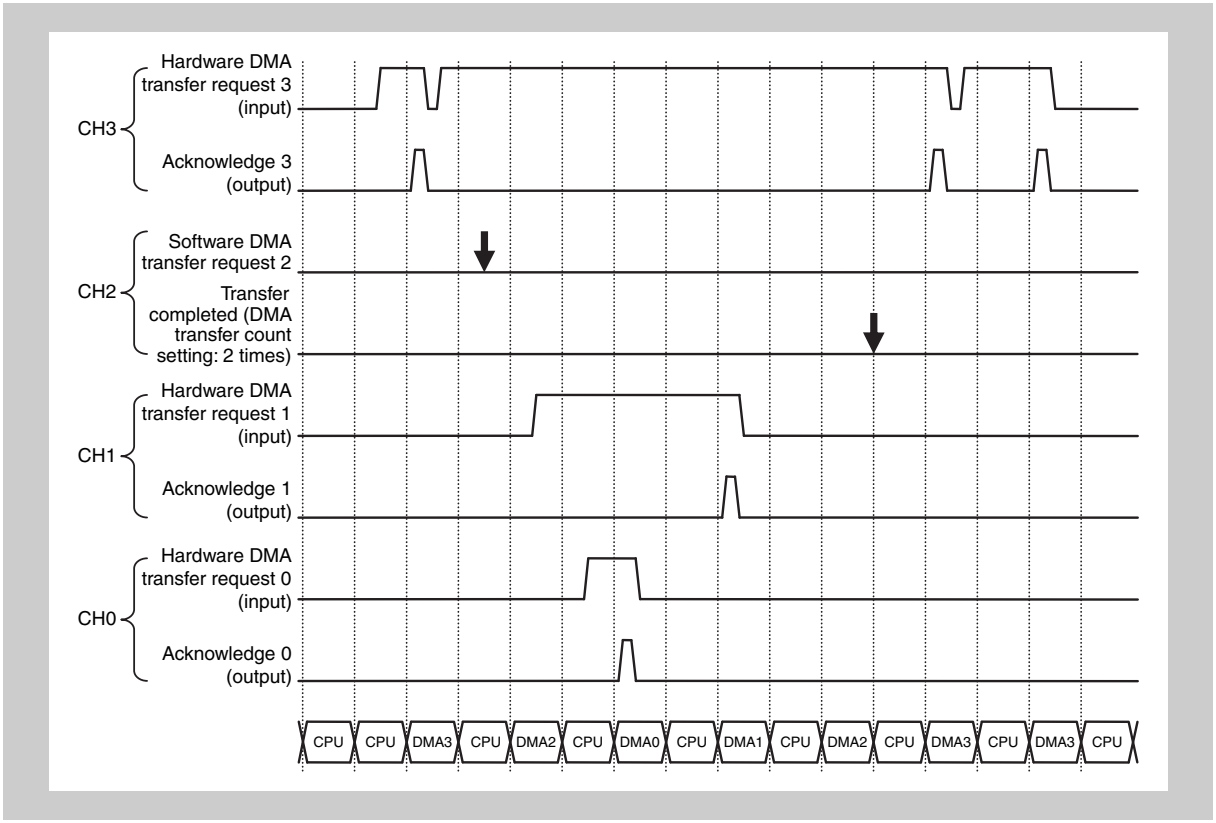


Figure 6-9 Example of priority control

6.6.4 Valid DMA transfer request conditions

Whether a DMA transfer request of channel n is acknowledged depends on the setting of the ERR and ADS bits of the DMA transfer request control register (DTRC), the MLE bit of the DMA transfer control register (DTCTn), and the TC and DTE bits of the DMA transfer status register (DTSn). *Table 6-29 "Valid DMA transfer request conditions of channel n"* shows the relationship between the setting of each of the above bits and whether a DMA transfer request is acknowledged.

Table 6-29 Valid DMA transfer request conditions of channel n

Register.bit name	DTSn.DTE	DTSn.TC	DTCTn.MLE	DTRC.ERR	DTRC.ADS	DMA transfer request
When DMA transfer is disabled	0	X	X	X	X	Invalid
When DMA transfer error occurs	X	X	X	1	X	Invalid
When DMA transfer is aborted	X	X	X	X	1	Invalid
When DMA transfer is completed (multilink disabled)	X	1	0	X	X	Invalid
When DMA transfer is completed/not completed (multilink enabled)	1	X	1	0	0	Valid
When DMA transfer is enabled	1	0	0	0	0	Valid

6.6.5 Next address function

(1) Next address setting register

This register is used to set beforehand the transfer information to be transferred next during DMA transfer. This information is copied to the corresponding register at the start of the last DMA cycle. The following registers are provided.

- DMA next source address register (DNSAnH/DNSAnL)
- DMA next source chip select register (DNSCn)
- DMA next destination address register (DNDAnH/DNDAnL)
- DMA next destination chip select register (DNDCn)
- DMA next transfer count register (DNTCn)

Each one of these registers has a valid bit in the most significant position (address register: most significant bit on H side), and whether to copy the transfer information to the current register at the start of the last DMA cycle can be selected. When the transfer information to be transferred next is copied to the current register, the valid bit is cleared.

(2) Processing upon DMA transfer completion when using next address function

Normally, upon completion of DMA transfer, the DMA transfer enable bit (DTE) is cleared at the same time the DMA transfer completion status bit (TC) of the DMA transfer status register (DTSn) is set, and subsequent DMA transfer requests are no longer acknowledged. However, if the multilink enable bit (MLE) is set, DTE is not cleared and DMA transfer requests can be acknowledged even if TC is set. Therefore, when using the next address function, it is possible to eliminate the need to clear TC upon DMA transfer completion and set DTE by setting MLE.

(3) Timing at which next address is set

The next address setting register can always be rewritten. However, to prevent a conflict between copying to the current register and a write operation by the user, complete setting the next address setting register before the last DMA cycle starts.

Use of the DMA transfer count match interrupt is recommended as the trigger for setting the next address setting register. In this case, set the DMA transfer count compare register (DTCCn) so as to secure the time required for setting the next address setting register.

6.6.6 Aborting/resuming DMA transfer

(1) Aborting or resuming DMA transfer for all channels through software

By setting the DMA transfer abort bit (ADS) of the DMA transfer request control register (DTRC), the next DMA transfer and those that follow can be aborted. During a DMA cycle, the next DMA transfer is aborted after the ongoing DMA cycle has been completed. Note that the DMA transfer enable bit (DTE) and the software DMA transfer request bit (SR) of the DMA transfer status register (DTS) are not cleared.

To resume the aborted DMA transfer, clear the ADS bit. If a DMA transfer is requested at that point, the transfer of the channel having the highest priority at that time is executed. To end DMA transfer, clear the DMA transfer request with the DTE bit cleared.

(2) Aborting or resuming DMA transfer by using DMA transfer enable bit (DTE)

By clearing the DMA transfer enable bit (DTE) of the DMA transfer status register (DTS), the next DMA transfer and those that follow can be aborted. During a DMA cycle, the next DMA transfer is aborted after the ongoing DMA cycle is completed. Note that the software DMA transfer request bit (SR) of DTS is not cleared.

To resume the aborted DMA transfer, set the DTE bit. If another channel is not executing DMA transfer at that point, the priority is identified as usual. If another channel is executing DMA transfer, the priority is identified after that transfer has been completed. To end DMA transfer, clear the DMA transfer request with the DTE bit cleared.

(3) Aborting or resuming DMA transfer by using software DMA transfer request bit (SR)

By clearing the software DMA transfer request bit (SR) of the DMA transfer status register, the next DMA transfer and those that follow can be aborted (DTS). During a DMA cycle, the next DMA transfer is aborted after the ongoing DMA cycle has been completed.

To resume the aborted DMA transfer, set the SR bit. If another channel is not executing DMA transfer at that point, the priority is identified as usual. If another channel is executing DMA transfer, the priority is identified after that transfer has been completed.

6.6.7 Error response support

(1) Aborting DMA transfer by error response

When an error occurs at the DMA transfer source or transfer destination, the DMA transfer abort bit (ADS) of the DMA transfer request control register (DTRC) to abort subsequent DMA transfers. At the same time, the DMA transfer error status bit (ERR) is set and a SYSERR exception is generated towards the CPU. The user can evaluate in which channel the error has occurred, by using the DMA transfer error flag (ER) of the DMA transfer status register (DTSn), when the user has confirmed that ERR has been set.

In this case, note that, if an error response is acknowledged in the read cycle, the write cycle is not executed, but the transfer address and the transfer count are updated.

(2) Canceling transfer abort by error response

DMA transfer abort can be canceled by clearing the DMA transfer abort bit (ADS) and DMA transfer error status bit (ERR) of the DMA transfer request control register (DTRC).

Clear the DMA transfer enable bit (DTE) of the DMA transfer status register (DTSn) in advance, so that DMA transfer is not resumed after its abort has been canceled. In the case of a software DMA transfer request, also clear the software DMA transfer request bit (SR).

6.6.8 Stand-by support

When a stop request is generated, DMA transfer stops until completion of the 2 DMA cycles currently being executed. Unlike DMA transfer abort caused by software, this does not affect the DMA control register. DMA transfer resumes upon cancellation of the stop request, and if a DMA request is already retained, that DMA transfer starts.

6.7 DTFR Function

The DMA trigger factor register (DTFR) selects DMA trigger factors from among interrupt signals, and requests DMAC for DMA transfer. DTFRn registers are included for selecting the signals to be used for DMA transfer requests from among the 128 input interrupt signals.

6.7.1 Features

Number of transfer factors

DMA transfer requests (for 16 channels) are selected from among 128 interrupt signals.

DMAC interface

The DMA transfer request signal n is output.

The DMA transfer request signal n is cleared by an acknowledge signal from DMA.

CPU interface

The last transfer signal from DMA is output as a CPU interrupt signal.

Clearing of transfer request

A function that clears transfer request signals sent to DMA through register access is provided.

Confirmation of transfer request

A function that checks transfer request signals sent to DMA through register access is provided.

6.8 DTFR Control Registers

6.8.1 DTFRn – DTFRn register

Access This register can be read or written in 16-bit units.

Address DTFR0: FFFF7B00$_H$, DTFR1: FFFF7B02$_H$, DTFR2: FFFF7B04$_H$, DTFR3: FFFF7B06$_H$, DTFR4: FFFF7B08$_H$, DTFR5: FFFF7B0A$_H$, DTFR6: FFFF7B0C$_H$, DTFR7: FFFF7B0E$_H$, DTFR8: FFFF7B10$_H$, DTFR9: FFFF7B12$_H$, DTFR10: FFFF7B14$_H$, DTFR11: FFFF7B16$_H$, DTFR12: FFFF7B18$_H$, DTFR13: FFFF7B1A$_H$, DTFR14: FFFF7B1C$_H$, DTFR15: FFFF7B1E$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
REQEN	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	IFCn6-0						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-30 DTFRn registers contents

Bit position	Bit name	Function
15	REQEN	This bit enables or disables operation of the DMA source selector of channel n. 1: Enables operation of source selector 0: Stops operation of source selector. Does not issue DMA transfer request (DMARQ). The settings of IFC6 to IFC0 are valid. Requests are always sampled.
6 to 0	IFCn6 to IFCn0	These bits select the transfer source. The set values are shown in the table in the first section of this chapter.

Caution Stopping DMA channel n by DTFRn.REQEN = 0 does not clear any pending DMA request for that channel. In order to clear also a pending DMA request, set also DRQCLR.RQCRn = 1.

6.8.2 DRQCLR – DMA request clear register

Access This register can be read or written in 16-bit units.

Address $FFFF7B40_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
RQCR15	RQCR14	RQCR13	RQCR12	RQCR11	RQCR10	RQCR9	RQCR8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
RQCR7	RQCR6	RQCR5	RQCR4	RQCR3	RQCR2	RQCR1	RQCR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-31 DRQCLR register contents

Bit position	Bit name	Function
15	RQCR15	Setting “1” to this bit clears transfer request held in channel 15 to “0”.
14	RQCR14	Setting “1” to this bit clears transfer request held in channel 14 to “0”.
13	RQCR13	Setting “1” to this bit clears transfer request held in channel 13 to “0”.
12	RQCR12	Setting “1” to this bit clears transfer request held in channel 12 to “0”.
11	RQCR11	Setting “1” to this bit clears transfer request held in channel 11 to “0”.
10	RQCR10	Setting “1” to this bit clears transfer request held in channel 10 to “0”.
9	RQCR9	Setting “1” to this bit clears transfer request held in channel 9 to “0”.
8	RQCR8	Setting “1” to this bit clears transfer request held in channel 8 to “0”.
7	RQCR7	Setting “1” to this bit clears transfer request held in channel 7 to “0”.
6	RQCR6	Setting “1” to this bit clears transfer request held in channel 6 to “0”.
5	RQCR5	Setting “1” to this bit clears transfer request held in channel 5 to “0”.
4	RQCR4	Setting “1” to this bit clears transfer request held in channel 4 to “0”.
3	RQCR3	Setting “1” to this bit clears transfer request held in channel 3 to “0”.
2	RQCR2	Setting “1” to this bit clears transfer request held in channel 2 to “0”.
1	RQCR1	Setting “1” to this bit clears transfer request held in channel 1 to “0”.
0	RQCR0	Setting “1” to this bit clears transfer request held in channel 0 to “0”.

Note Writing “0” to bits 15 to 0 is ignored.

6.8.3 DRQSTR – DMA request check register

Access This register is read-only, in 16-bit units.

Address $FFFF7B44_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
RQST15	RQST14	RQST13	RQST12	RQST11	RQST10	RQST9	RQST8
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
RQST7	RQST6	RQST5	RQST4	RQST3	RQST2	RQST1	RQST0
R	R	R	R	R	R	R	R

Table 6-32 DRQSTR register contents

Bit position	Bit name	Function
15	RQST15	1: Request issued (DMA transfer request signal 15 is “1”), 0: No request (DMA transfer request signal 15 is “0”)
14	RQST14	1: Request issued (DMA transfer request signal 14 is “1”), 0: No request (DMA transfer request signal 14 is “0”)
13	RQST13	1: Request issued (DMA transfer request signal 13 is “1”), 0: No request (DMA transfer request signal 13 is “0”)
12	RQST12	1: Request issued (DMA transfer request signal 12 is “1”), 0: No request (DMA transfer request signal 12 is “0”)
11	RQST11	1: Request issued (DMA transfer request signal 11 is “1”), 0: No request (DMA transfer request signal 11 is “0”)
10	RQST10	1: Request issued (DMA transfer request signal 10 is “1”), 0: No request (DMA transfer request signal 10 is “0”)
9	RQST9	1: Request issued (DMA transfer request signal 9 is “1”), 0: No request (DMA transfer request signal 9 is “0”)
8	RQST8	1: Request issued (DMA transfer request signal 8 is “1”), 0: No request (DMA transfer request signal 8 is “0”)
7	RQST7	1: Request issued (DMA transfer request signal 7 is “1”), 0: No request (DMA transfer request signal 7 is “0”)
6	RQST6	1: Request issued (DMA transfer request signal 6 is “1”), 0: No request (DMA transfer request signal 6 is “0”)
5	RQST5	1: Request issued (DMA transfer request signal 5 is “1”), 0: No request (DMA transfer request signal 5 is “0”)
4	RQST4	1: Request issued (DMA transfer request signal 4 is “1”), 0: No request (DMA transfer request signal 4 is “0”)
3	RQST3	1: Request issued (DMA transfer request signal 3 is “1”), 0: No request (DMA transfer request signal 3 is “0”)
2	RQST2	1: Request issued (DMA transfer request signal 2 is “1”), 0: No request (DMA transfer request signal 2 is “0”)
1	RQST1	1: Request issued (DMA transfer request signal 1 is “1”), 0: No request (DMA transfer request signal 1 is “0”)
0	RQST0	1: Request issued (DMA transfer request signal 0 is “1”), 0: No request (DMA transfer request signal 0 is “0”)

6.9 DTS Function

The data transfer service (DTS) does not include registers for setting transfer information (TI). Consequently, each time a transfer request is generated, TI corresponding to the request number is fetched (read from Data RAM), and the transfer request for the TI is then sent to DMAT to execute data transfers, etc.

6.9.1 Features

Number of channels

Number of channels is freely set (channels 1 to 128)

Transfer data size

8 bits

16 bits

32 bits

Transfer data

Fixed to little endian

Misaligned data not supported

Maximum transfer count

When the DTS transfer count register is used as a 16-bit register: 65535 times

When the DTS transfer count register is split into two 8-bit registers: 255 x n: (n indicates no limitation)

Therefore, the number of transfers is not limited.

Channel priority control

No priority control function is provided (control must be done outside of DTS).

Subject to transfer

Code flash

Data RAM

Data-Flash

External memory area

Peripheral I/O area

Chain function

This function enables DTS to be started multiple times by one request. After writing back TI, other TI is continuously fetched, so that data transfer (etc.) can be executed.

Transfer type

- 2-cycle transfer (dual address transfer) function

  After TI is fetched, data is read from the transfer source, and then the read data is written to the transfer destination. Afterward, updated TI is written back. This operation is the same as a DMA transfer, except for the TI fetch and write back. Since the bus remains unlocked during all cycles, a CPU cycle may interrupt.

- Compare function

  After TI is fetched, data is read from the comparison source and the comparison destination, and these two data are then compared. Afterward, updated TI is written back. A true/false judgment of the compare results can then be used to select whether to output an interrupt to the CPU or to continue processing using the chain function.

- Flag check function (bit check function)

  This function is used to check whether any data bit is “1” or “0”.

  After TI is fetched, mask data for flag checking and flagged check data are read, then any bit is determined. Afterward, the updated TI is written back. A true/false judgment of the flag check results can then be used to select whether to output an interrupt to the CPU or to continue processing using the chain function.

Transfer count register split mode

- Split

  In this mode, the 16-bit transfer count register is split into upper (8 bits) and lower (8 bits) sides which are used separately.

  The lower side is used as a transfer count register and the upper side is used as a transfer count hold register.

  Although the maximum transfer count is 255 times, when the lower side is zero, the upper side's values can be copied to the lower side, so transfers can be performed continuously without resetting the transfer count.

- Combined

  The entire 16 bits of the transfer count register are used. The maximum transfer count is 65535 times.

Transfer mode

- Single transfer mode

  When a DTS transfer request is generated, one DTS cycle is performed. Afterward, when a DTS transfer request is generated, another DTS cycle is performed. This operation is repeated until the transfer count register value reaches “0”.

- Block transfer mode

  When a DTS transfer request is generated, the number of DTS cycles specified in the transfer count register is performed.

  Other DTS transfer requests are not accepted until the number of DTS cycles specified in the transfer count register is completed. However, because the bus is not locked, a CPU cycle may interrupt.

Transfer address control

Incremental

Decremental

Fixed

Transfer error support

When an error response from the DMA data bus occurs, the DTS cycle is aborted and an error flag is set.

A SYSERR exception is also generated.

Interrupt output function	An interrupt source signal is output (by the DTSFSL function) when the number of DTS transfers specified in the bus cycle count register has been completed, or depending on a flag check result and a compare result.
DTS transfer abort function	This function supports aborting DTS transfer by software, but not by hardware.
Stand-by support	The stand-by request is used for stand-by support.

6.10 DTS Control Registers

DTS uses the following control registers to control operation settings.

The DTS's registers include registers that can be accessed from the CPU (register group A) and internal registers that cannot be accessed from the CPU (register group B). Transfer information read from the Data RAM is stored in the DTS internal register.

Register group B is comprised of DTS0SAR, DTS0DAR, DTS0TCEA, DTS0CIR, DTS0SCS, DTS0DCS, DTS0ESCS, and DTS0EDCS.

Table 6-33 List of DTS control registers

	Address[Note]	Symbol	Function register name	R/W	Operable bits			Initial value
					8	16	32	
Register group A	FFFF7C00H	DTS0TSR	DTS transfer status register	R	√	–	–	00H
	FFFF7C04H	DTS0TRC	DTS transfer request control register	R/W	√	–	–	00H
	FFFF7C08H	DTS0ICR	DTS initialization control register	R/W	–	√	–	0000H
	FFFF7C08H	DTS0ICH	DTS initialization channel number register	R/W	√	–	–	00H
	FFFF7C09H	DTS0ITR	DTS initialization channel trigger register	R/W	√	–	–	00H
	FFFF7C0CH	DTS0BTR	DTS base table register	R/W	–	–	√	00000000H
	FFFF7C10H	DTS0BVR	DTS base vector register	R/W	–	–	√	00000000H
	FFFF7C14H	DTS0ACR	DTS active channel register	R	–	√	–	0000H
	FFFF7C18H	DTS0TST	DTS TI hold status register	R	√	–	–	00H
	FFFF7C20H	DTS0HC	DTS TI hold channel number register	R	–	–	√	00000000H
	FFFF7C20H	DTS0HC0	DTS hold channel register 0	R	√	–	–	00H
	FFFF7C21H	DTS0HC1	DTS hold channel register 1	R	√	–	–	00H
	FFFF7C22H	DTS0HC2	DTS hold channel register 2	R	√	–	–	00H
	FFFF7C23H	DTS0HC3	DTS hold channel register 3	R	√	–	–	00H
Register group B	–	DTS0SAR	DTS source address register	–	–	–	–	00000000H
	–	DTS0DAR	DTS destination address register	–	–	–	–	00000000H
	–	DTS0TCEA	DTS transfer counter or else address register	–	–	–	–	0000H
	–	DTS0CIR	DTS control information register	–	–	–	–	0000H
	–	DTS0SCS	DTS source address count size register	–	–	–	–	00H
	–	DTS0DCS	DTS destination address count size register	–	–	–	–	00H
	–	DTS0ECSRA	DTS extension address count size/ repeat address register	–	–	–	–	00000000H

Note If an address that is not mapped to DTS is accessed, write operations are ignored and a zero is returned when read.

6.10.1 DTS0TSR – DTS transfer status register [register group A]

Access This register is read-only, in 8-bit units.

Address $FFFF7C00_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	DTS0STPE	DTS0STPU
R	R	R	R	R	R	R	R

Table 6-34 DTS0TSR register contents

Bit position	Bit name	Function
1	DTS0STPE	DTS STоPped by Error response (DTS transfer abort by error response) This bit indicates whether a DTS transfer is aborted by an error response received from the transfer target. When an error response is received, “1” is set to this bit and a SysError exception is generated to abort the DTS transfer. When this bit is “1”, no transfers are accepted. This is a read-only bit. To clear this bit, write “1” to the DTS0ECL bit in the DTS0TRC register.
0	DTS0STPU	DTS STоPped by User request (DTS transfer abort by user request) This bit indicates whether a DTS transfer is aborted by a user request. “1” is set to this bit by writing “1” to the DTS0UST bit in the DTS0TRC register, and transfer is aborted. When this bit is “1”, no transfers are accepted. This is a read-only bit. To clear this bit, write “1” to the DTS0UCL bit in the DTS0TRC register.

6.10.2 DTS0TRC – DTS transfer request control register [register group A]

Access This register can be read or written in 8-bit units.

Address $FFFF7C04_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
DTS0UST	0	0	0	0	0	DTS0ECL	DTS0UCL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-35 DTS0TRC register contents

Bit position	Bit name	Function
7	DTS0UST	DTS User request STop Trigger (DTS abort request trigger bit) This is the transfer interrupt request trigger for all DTS channels. When “1” is written to this bit, the DTS0STPU bit in DTS0TSR is set to “1” and DTS transfer is aborted. This bit is always “0” when read, and writing “0” to this bit is ignored. When this bit and DTS0UCL are set to “1” at the same time, this bit takes priority.
1	DTS0ECL	DTS dts0stpE CLear Trigger (DTS0STPE clear request trigger bit) This is the clear request trigger bit for the DTS0STPE bit in DTS0TSR. When this bit is set to “1”, the DTS0STPE bit in DTS0TSR is cleared to “0”, and DTS transfers can be accepted. This bit is always “0” when read, and writing “0” to this bit is ignored.
0	DTS0UCL	DTS dts0stpU CLear Trigger (DTS0STPU clear request trigger bit) This is the clear request trigger bit for the DTS0STPU bit in DTS0TSR. When this bit is set to “1”, the DTS0STPU bit in DTS0TSR is cleared to “0”, and DTS transfers can be accepted. If a transfer has been aborted, the aborted transfer is restarted. This bit is always “0” when read, and writing “0” to this bit is ignored.

6.10.3 DTS0ICR – DTS initialization control register [register group A]

Access This register can be read or written in 16-bit units.
However, when the higher 8 bits and lower 8 bits of the DTS0ICR register are used as the DTS0ITR register and DTS0ICH register, respectively, this register can be read or written in 8-bit units.

Address DTS0ICR: $FFFF7C08_H$

DTS0ITR: $FFFF7C09_H$, DTS0IC$_H$: $FFFF7C08_H$

Initial Value 0000_H

- DTS0ICR

15	14	13	12	11	10	9	8
0	0	0	0	0	DTS0ICS	DTS0HIT	DTS0TIT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	DTS0ICH6-0						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- DTS0ITR

7	6	5	4	3	2	1	0
0	0	0	0	0	DTS0ICS	DTS0HIT	DTS0TIT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- DTS0ICH

7	6	5	4	3	2	1	0
0	DTS0ICH6-0						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-36 DTS0ICR register contents

Bit position	Bit name	Function
10	DTS0ICS	DTS Initialization Continuous Status (transfer abort status flag) This status flag indicates whether the DTS transfer is being aborted. When this flag is “1”, writing to this register is prohibited (writing to this register is ignored). When DTS0HIT or DTS0TIT in this register is set to “1”, this bit is set to “1”. When the transfer abort processing requested by using DTS0HIT or DTS0TIT in this register is completed, this bit is cleared. 0: Normal status 1: Transfer is being aborted.
9	DTS0HIT	DTS Hold Initialization Trigger (transfer abort and TI hold buffer initialization trigger bit) This is the trigger bit used to request aborting the current DTS transfer and clearing of all TI held in DTS. When “1” is written to this bit, all the held TI is cleared once the DTS cycle is completed. While waiting for the DTS cycle to be completed, the DTS0ICS bit is set to “1”. This bit is always “0” when read.

Table 6-36 DTS0ICR register contents

Bit position	Bit name	Function
8	DTS0TIT	DTS Transfer Initialization Trigger (transfer abort trigger bit) This is the trigger bit used to request aborting the DTS transfer for the channel specified by the DTS0ICH bit in this register. When “1” is written to this bit the channel for current DTS transfer is compared with the channel specified by the DTS0ICH bit, once the DTS cycle is completed. If the two match, all subsequent transfers are stopped. While waiting for the DTS cycle to be completed, the DTS0ICS bit is set to “1”. If TI hold is set for the specified channel, the TI hold is also cleared. This bit is always “0” when read.
6 to 0	DTS0ICH6 to DTS0ICH0	DTS Initialization CHannel (initialization stop request channel) This bit specifies the channel for which initialization is to be aborted by the DTS0TIT bit. “1” can be written to this bit at the same time as when writing “1” to the DTS0HIT and DTS0TIT bits in this register.

6.10.4 DTS0BTR – DTS base table register [register group A]

Access This register can be read or written in 32-bit units.

Address $FFFF7C0C_H$

Initial Value 00000000_H

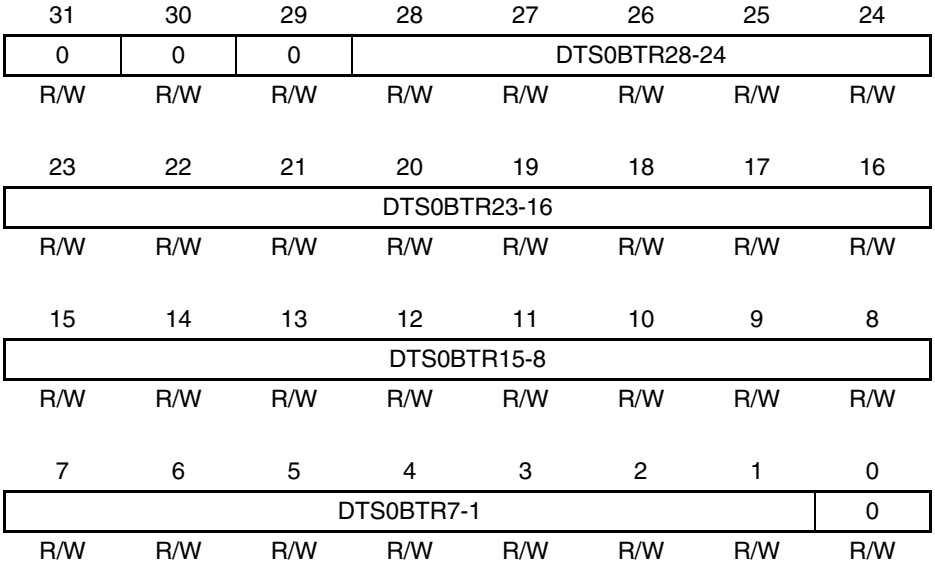


Table 6-37 DTS0BTR register contents

Bit position	Bit name	Function
28 to 1	–	This is a register for setting the start address of the table that is used by DTS to detect the TI position. Set the Data RAM area. Bits 31 to 29 and 0 are fixed to “0”. DTS uses DTS0BVR and this register to calculate the TI position corresponding to channels.

6.10.5 DTS0BVR – DTS base vector register [register group A]

Access This register can be read or written in 32-bit units.

Address $FFFF7C10_H$

Initial Value 00000000_H

31	30	29	28	27	26	25	24
0	0	0	DTS0BVR28-24				
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
DTS0BVR23-16							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
DTS0BVR15-8							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
DTS0BVR7-2						0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-38 DTS0BVR register contents

Bit position	Bit name	Function
28 to 2	–	This is a register for setting the start address of the vector that is used by DTS to detect the TI position. Set the Data RAM area. Bits 31 to 29, 1, and 0 are fixed to “0”. When a DTS request is generated, a 16-bit vector address is read from the table address calculated as “DTS0BTR + channel No. x 2”. The read value is added to this register's value to determine the TI's address.

6.10.6 DTS0ACR – DTS active channel register [register group A]

Access This register is read-only, in 16-bit units.

Address $FFFF7C14_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	DTS0ACT
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	DTS0CH						
R	R	R	R	R	R	R	R

Table 6-39 DTS0ACR register contents

Bit position	Bit name	Function
8	DTS0ACT	DTS ACTive (DTS execution status) This bit indicates DTS's execution status. 0: DTS is not operating. 1: DTS is operating.
6 to 0	DTS0CH	CHannel (active channel) These bits indicate the active channel when the DTS0ACT bit is “1”. When DTS0ACT bit is “0”, the channel used for the previous execution is retained.

6.10.7 DTS0TST – DTS TI hold status register [register group A]

Access This register is read-only, in 8-bit units.

Address $FFFF7C18_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	DTS0TON3	DTS0TON2	DTS0TON1	DTS0TON0
R	R	R	R	R	R	R	R

Table 6-40 DTS0TST register contents

Bit position	Bit name	Function
3	DTS0TON3	DTS Ti-hold ON 3 (TI hold 3 ON) This bit checks whether TI is held in TI hold buffer 3. 0: TI is not held. 1: TI is held.
2	DTS0TON2	DTS Ti-hold ON 2 (TI hold 2 ON) This bit checks whether TI is held in TI hold buffer 2. 0: TI is not held. 1: TI is held.
1	DTS0TON1	DTS Ti-hold ON 1 (TI hold 1 ON) This bit checks whether TI is held in TI hold buffer 1. 0: TI is not held. 1: TI is held.
0	DTS0TON0	DTS Ti-hold ON 0 (TI hold 0 ON) This bit checks whether TI is held in TI hold buffer 0. 0: TI is not held. 1: TI is held.

6.10.8 DTS0HC – DTS TI hold channel number register [register group A]

Access This register is read-only, in 32-bit units.
However, when bits 30 to 24, 22 to 16, 14 to 8, and 6 to 0 in the DTS0HC register are used as the DTS0HC3, DTS0HC2, DTS0HC1, and DTS0HC0 registers respectively, these registers are read-only, in 8-bit units.

Address DTS0HC: $FFFF7C20_H$
DTS0HC0: $FFFF7C20_H$, DTS0HC1: $FFFF7C21_H$, DTS0HC2: $FFFF7C22_H$, DTS0HC3: $FFFF7C23_H$

Initial Value 00000000_H

- DTS0HC

31	30	29	28	27	26	25	24
0	DTS0HC30-24						
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	DTS0HC22-16						
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	DTS0HC14-8						
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	DTS0HC6-0						
R	R	R	R	R	R	R	R

- DTS0HC3

7	6	5	4	3	2	1	0
0	DTS0HC30-24						
R	R	R	R	R	R	R	R

- DTS0HC2

7	6	5	4	3	2	1	0
0	DTS0HC22-16						
R	R	R	R	R	R	R	R

- DTS0HC1

7	6	5	4	3	2	1	0
0	DTS0HC14-8						
R	R	R	R	R	R	R	R

- DTS0HC0

7	6	5	4	3	2	1	0
0	DTS0HC6-0						
R	R	R	R	R	R	R	R

Table 6-41 DTS0HC register contents

Bit position	Bit name	Function
30 to 24	DTS0HC30 to DTS0HC24	These bits store the channel number of the TI that is held in TI hold buffer 3.
22 to 16	DTS0HC22 to DTS0HC16	These bits store the channel number of the TI that is held in TI hold buffer 2.
14 to 8	DTS0HC14 to DTS0HC8	These bits store the channel number of the TI that is held in TI hold buffer 1.
6 to 0	DTS0HC6 to DTS0HC0	These bits store the channel number of the TI that is held in TI hold buffer 0.

6.10.9 DTS0SAR – DTS source address register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU.

Access —

Address —

Initial Value 00000000_H

31	30	29	28	27	26	25	24
IMM31	IMM30	IMM29	DTS0SAR28-24 (IMM28-24)				

23	22	21	20	19	18	17	16
DTS0SAR23-16 (IMM23-16)							

15	14	13	12	11	10	9	8
DTS0SAR15-8 (IMM15-8)							

7	6	5	4	3	2	1	0
DTS0SAR7-0 (IMM7-0)							

Table 6-42 DTS0SAR register contents

Bit position	Bit name	Function
31 to 29	IMM31 to IMM29	IMMediate (immediate bits) When using the immediate function, these bits and DTS0SAR bits are used in combination as a 32-bit register.
28 to 0	DTS0SAR28 to DTS0SAR0	DTS Source Address Register (source address register) These bits comprise the source address register. When using the immediate function, they function as IMM28 to IMM0.

Caution DTS transfer of misaligned data is not supported. The lower 2 bits of an address corresponding to the transfer data size are as follows (x indicates any bit).

Operation is not guaranteed if a setting other than the following is made.

Data size	DTS0SAR1	DTS0SAR0
8 bits	x	x
16 bits	x	0
32 bits	0	0

6.10.10 DTS0DAR – DTS destination address register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU.

Access —

Address —

Initial Value 00000000_H

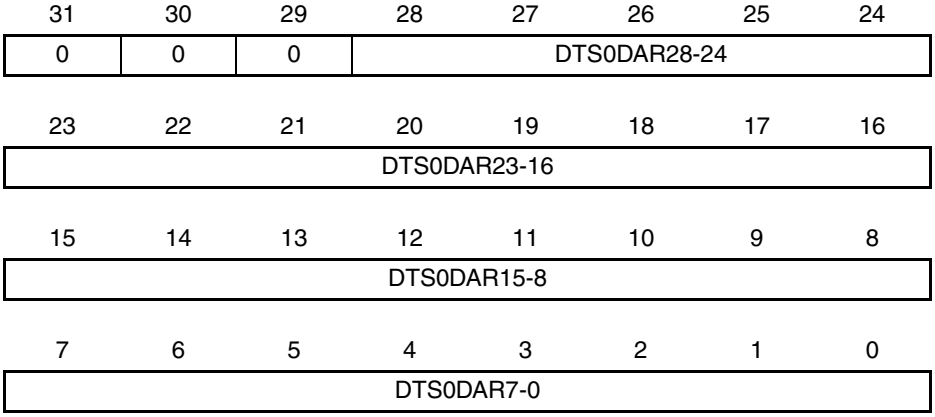


Table 6-43 DTS0DAR register contents

Bit position	Bit name	Function
28 to 0	DTS0DAR28 to DTS0DAR0	Destination Address Register (destination address register) These bits comprise the destination address register. Bits 31 to 29 are fixed to “0”.

Caution DTS transfer of misaligned data is not supported. The lower 2 bits of an address corresponding to the transfer data size are as follows (x indicates any bit).

Operation is not guaranteed if a setting other than the following is made.

Data size	DTS0DAR1	DTS0DAR0
8 bits	x	x
16 bits	x	0
32 bits	0	0

6.10.11 DTS0TCEA – DTS transfer counter or else address register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU.

Access —

Address —

Initial Value 0000_H

(1) Transfer count 255 mode

15	14	13	12	11	10	9	8
DTS0STR7-0							

7	6	5	4	3	2	1	0
DTS0STC7-0							

Table 6-44 DTS0TCEA register contents (255 mode)

Bit position	Bit name	Function
15 to 8	DTS0STR7 to DTS0STR0	Short Transfer Number Register (transfer count hold register) During transfer count 255 mode, this register holds the transfer count. If a DTS request is generated when the DTS0STC bit is “0”, the contents of this register are copied to the DTS0STC bit.
7 to 0	DTS0STC7 to DTS0STC0	Short Transfer Counter (short transfer count register) During transfer count 255 mode, this register counts the transfer count. The count is decremented once per DTS cycle. When 1 is set, one DTS is triggered, and when FFH is set, 255 DTS cycles are triggered. When the value of this register reaches “0”, an interrupt occurs (or the chain function can be selected). Continued execution of transfers can be selected. If a DTS request is generated when DTS0STC is “0”, the contents of the DTS0STR bits are copied to this register, and another decrementation count is performed.

(2) Transfer count 65535 mode

15	14	13	12	11	10	9	8
DTS0LTC15-8							

7	6	5	4	3	2	1	0
DTS0LTC7-0							

Table 6-45 DTS0TCEA register contents (65535 mode)

Bit position	Bit name	Function
15 to 0	DTS0LTC15 to DTS0LTC0	Long Transfer Counter (long transfer count register) During transfer count 65535 mode, this register counts the transfer count. The count is decremented once per DTS cycle. When 1 is set, one DTS is triggered, and when FFFFH is set, 65,535 DTS cycles are triggered. When the value of this register reaches “0”, an interrupt occurs (or the chain function can be selected).

(3) Extended chain mode

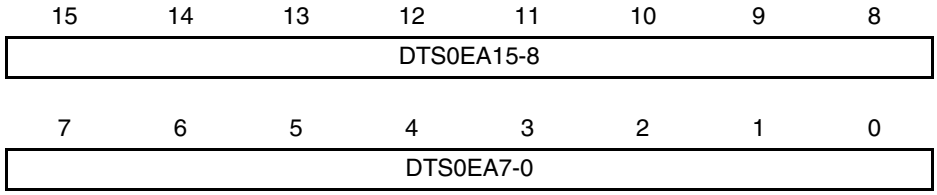


Table 6-46 DTS0TCEA register contents (extended chain mode)

Bit position	Bit name	Function
15 to 0	DTS0EA15 to DTS0EA0	Else Address (ELSE address) During extended chain mode, this register specifies the position of TI that is read when the condition is false (ELSE). To specify the TI position, set an offset using this register from the current position (TI-A address where the extended chain is specified). The value is specified as a signed 16-bit value from –32,768 to 32,760 (lower 3 bits are 3'b000).

6.10.12 DTS0SCS – DTS source address count size register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU. When DTS0IMM in DTS0CIR is “1”, the following operations are performed.

- When the data size indicated by DTS0DS1 and DTS0DS0 in DTS0CIR is 32 bits:

  DTS0SCS becomes “0”.

- When the data size indicated by DTS0DS1 and DTS0DS0 in DTS0CIR is 16 bits:

  DTS0SAR (IMM15 to IMM0) is counted (incremented/decremented) by DTS0SCS.

- When the data size indicated by DTS0DS1 and DTS0DS0 in DTS0CIR is 8 bits:

  DTS0SAR (IMM7 to IMM0) is counted (incremented/decremented) by DTS0SCS.

However, when in SG mode, this register's settings are ignored.

Access —

Address —

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	DTS0SCS2	DTS0SCS1	DTS0SCS0

Table 6-47 DTS0SCS register contents

Bit position	Bit name	Function
2 1 0	DTS0SCS2 DTS0SCS1 DTS0SCS0	Specifies the count size of the source address. 000: 0 (FIX) 001: +1 010: +2 011: +4 101: –1 110: –2 111: –4

6.10.13 DTS0DCS – DTS destination address count size register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU.

This register's settings are ignored when in SG mode.

Access —

Address —

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	DTS0DCS2	DTS0DCS1	DTS0DCS0

Table 6-48 DTS0DCS register contents

Bit position	Bit name	Function
2 1 0	DTS0DCS2 DTS0DCS1 DTS0DCS0	Specifies the count size of the destination address. 000: 0 (FIX) 001: +1 010: +2 011: +4 101: –1 110: –2 111: –4

6.10.14 DTS0CIR – DTS control information register [register group B]

This register is a DTS internal register. It is neither write-accessible nor read-accessible from the CPU.

Access —

Address —

Initial Value 0000_H

15	14	13	12	11	10	9	8
DTS0CAF	DTS0DPE1	DTS0SPE1	DTS0DISI	DTS0CM1	DTS0CM0	DTS0DS1	DTS0DS0

7	6	5	4	3	2	1	0
DTS0BEN	DTS0IMM	DTS0TTYP5	DTS0TTYP4	DTS0TTYP3	DTS0TTYP2	DTS0TTYP1	DTS0TTYP0

Table 6-49 DTS0CIR register contents (1/2)

Bit position	Bit name	Function
15	DTS0CAF	Condition Accumulation Flag This is the flag check and comp check result accumulation flag. When the results of a flag check or comp check (comparison) are false, DTS0CAF is set to “1” and this value is written back to TI-A. If this bit is “1” when TI-A is read, then “1” must be written back. Condition for setting (to “1”): • When compare result is false. • When flag check result is false. • When DTS0CAF read data is “1” (bit 15 is “1” at TI-A fetch). In other words, when “1” is set to bit 15 of TI-A, the value of DTS0CAF remains “1”. Condition for clearing (to “0”): • When the DTS0CAF bit read data is “0” (bit 15 is “0” at TI-A fetch).
14	DTS0DPE1	Destination select PE1 (destination PE1 select) This bit specifies the address range on the destination side. 0: Selects Data-Flash, external memory area, and peripheral I/O area. 1: Selects Code flash and Data RAM.
13	DTS0SPE1	Source select PE1 (source PE1 select) This bit specifies the address range on the source side. 0: Selects Data-Flash, external memory area, and peripheral I/O area. 1: Selects Code flash and Data RAM.
12	DTS0DISI	DISable Interruption (interrupt output disabled) This bit prohibits interrupt requests for DTSFSL. 0: Interrupt enabled. 1: Interrupt disabled.
11 10	DTS0CM1 DTS0CM0	Chain Mode (chain mode) 00: Does not perform a chain of transfers. [if (counter==0) then Interrupt else no operation] 01: Performs a chain of transfers when counter = 0. [if (counter==0) then chain & Interrupt else no operation] 10: Performs a chain of transfers if true. [if (true) then chain else Interrupt] 11: Always perform a chain of transfers.

Table 6-49 DTS0CIR register contents (2/2)

Bit position	Bit name	Function
9 8	DTS0DS1 DTS0DS0	Data Size (data size) These bits set the data size for transfer, compare, and flag check. 00: 8 bits 01: 16 bits 10: 32 bits 11: Setting prohibited
7	DTS0BEN	Block Enable This is the block enable signal. 0: Single mode 1: Block mode
6	DTS0IMM	IMMediate enable (immediate function) This bit enables/disables the immediate function. When the immediate function is enabled, DTS0SAR functions as the immediate register. 0: Immediate disabled 1: Immediate enabled
5 4 3 2 1 0	DTS0TTYP5 DTS0TTYP4 DTS0TTYP3 DTS0TTYP2 DTS0TTYP1 DTS0TTYP0	Transfer TYPe (transfer type select) These bits select the transfer type. When a transfer type is selected, a special function and transfer count are selected at the same time. See *6.11.5 "Transfer types"* for the list of transfer types.

6.10.15 DTS0ECSRA – DTS extension address count size/repeat address registers [register group B]

These registers are DTS internal registers. They are neither write-accessible nor read-accessible from the CPU.

Access —

Address —

Initial Value 00000000_H

(1) SG (Scatter & Gather) mode

31	30	29	28	27	26	25	24
DTS0EDCS15-8							

23	22	21	20	19	18	17	16
DTS0EDCS7-0							

15	14	13	12	11	10	9	8
DTS0ESCS15-8							

7	6	5	4	3	2	1	0
DTS0ESCS7-0							

Table 6-50 DTS0ECSRA register contents (SG mode)

Bit position	Bit name	Function
31 to 16	DTS0EDCS15 to DTS0EDCS0	These bits specify the count size for the destination address. Any signed 16-bit value from –32,768 to 32,767 can be specified. This register is valid only when SG mode is selected by the DTS's special function. When not in SG mode, the count size is determined by DTS0DCS.
15 to 0	DTS0ESCS15 to DTS0ESCS0	This bit specifies the count size for the source address. Any signed 16-bit value from –32,768 to 32,767 can be specified. This register is valid only when SG mode is selected by the DTS's special function. When not in SG mode, the count size is determined by DTS0SCS.

(2) SREP (Source Repeat) mode

31	30	29	28	27	26	25	24
DTS0SRA31-24							

23	22	21	20	19	18	17	16
DTS0SRA23-16							

15	14	13	12	11	10	9	8
DTS0SRA15-8							

7	6	5	4	3	2	1	0
DTS0SRA7-0							

Table 6-51 DTS0ECSRA register contents (SREP mode)

Bit position	Bit name	Function
31 to 0	DTS0SRA31 to DTS0SRA0	These bits comprise an area where the repeat address is held. The source repeat becomes valid only when SREP is selected by the DTS's special function.

(3) DREP (Destination Repeat) mode

31	30	29	28	27	26	25	24
DTS0DRA31-24							

23	22	21	20	19	18	17	16
DTS0DRA23-16							

15	14	13	12	11	10	9	8
DTS0DRA15-8							

7	6	5	4	3	2	1	0
DTS0DRA7-0							

Table 6-52 DTS0ECSRA register contents (DREP mode)

Bit position	Bit name	Function
31 to 0	DTS0DRA31 to DTS0DRA0	These bits comprise an area where the repeat address is held. The destination repeat becomes valid only when DREP is selected by the DTS's special function.

6.11 DTS Function Details

6.11.1 Transfer information (TI)

DTS internal registers (register group B) cannot be operated directly from an external source. The DTS reads transfer information from Data RAM, and then sets values to the internal registers. This transfer information is called TI.

TI is configured of either 96 bits (TI-A to TI-C) or 128 bits (TI-A to TI-D), and the setting in DTS0TTYP5 to DTS0TTYP0 determines which type is used.

(1) Configuration of TI

Each TI should be 32-bit aligned (the lower 2 bits of the address are 00).

The 128 bits from TI-A to TI-D should be allocated to Data RAM as one set.

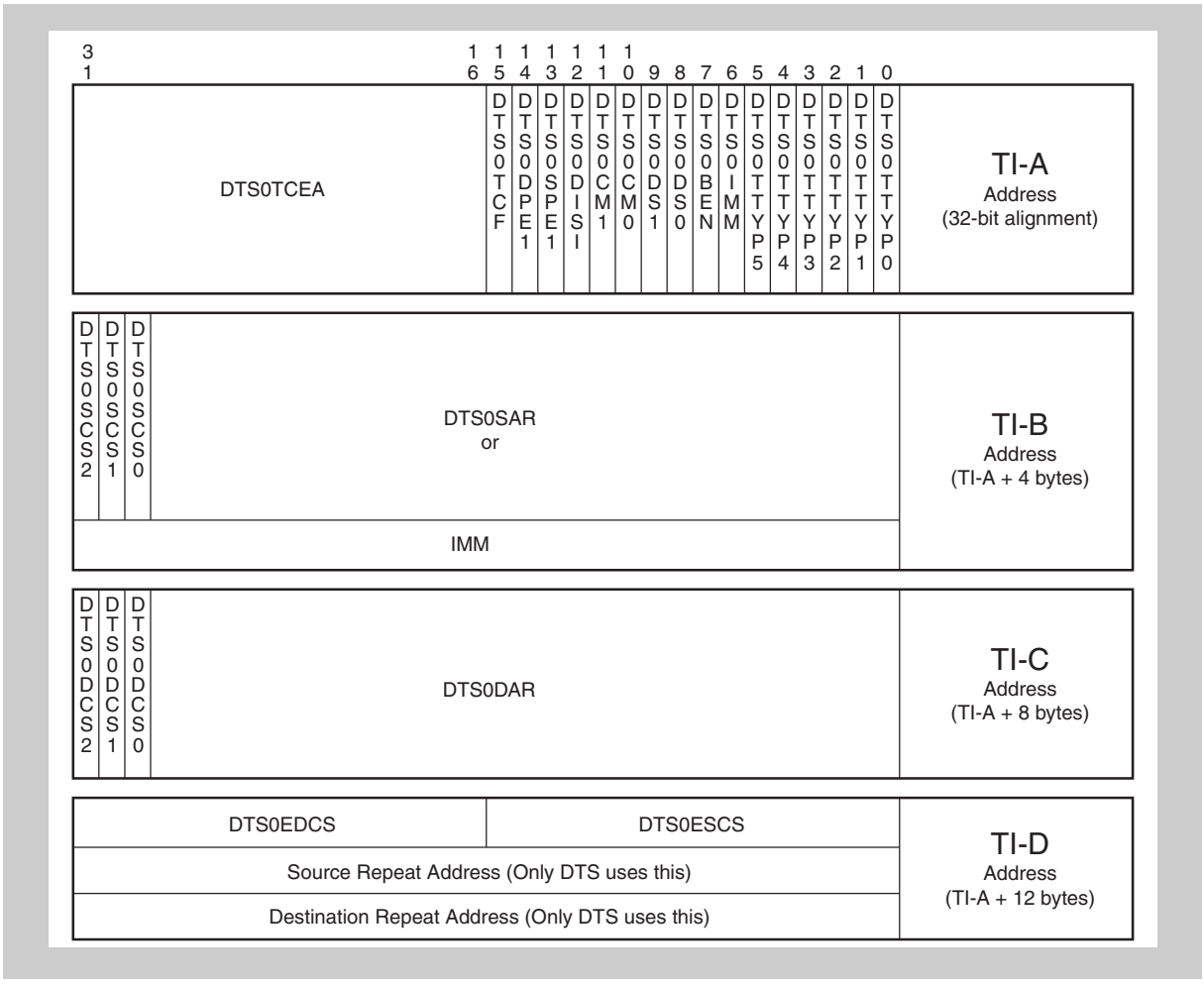


Figure 6-10 Configuration of TI

(2) Address where TI is allocated

The DTS uses DTS0BTR and DTS0BVR to determine the TI address corresponding to a channel.

TI can be freely allocated anywhere in the range from the address specified by DTS0BVR to the address to which FFFCH is added.

The method for determining DTS's TI address is as follows.

1. A 16-bit offset address is read from the table address [table address = DTS0BTR + channel number x 2] (the lower 2 bits of the offset address are ignored).
2. TI is read from the vector address [TI address = DTS0BVR + offset address]. In this case as well, the lower 2 bits are fixed to 00.

In the following figure, in which the request from CH3 is used as an example, the data for table address (1EC0_1000H + 4 x 2) is 0320H and the vector address is DTS0BVR + 0320H.

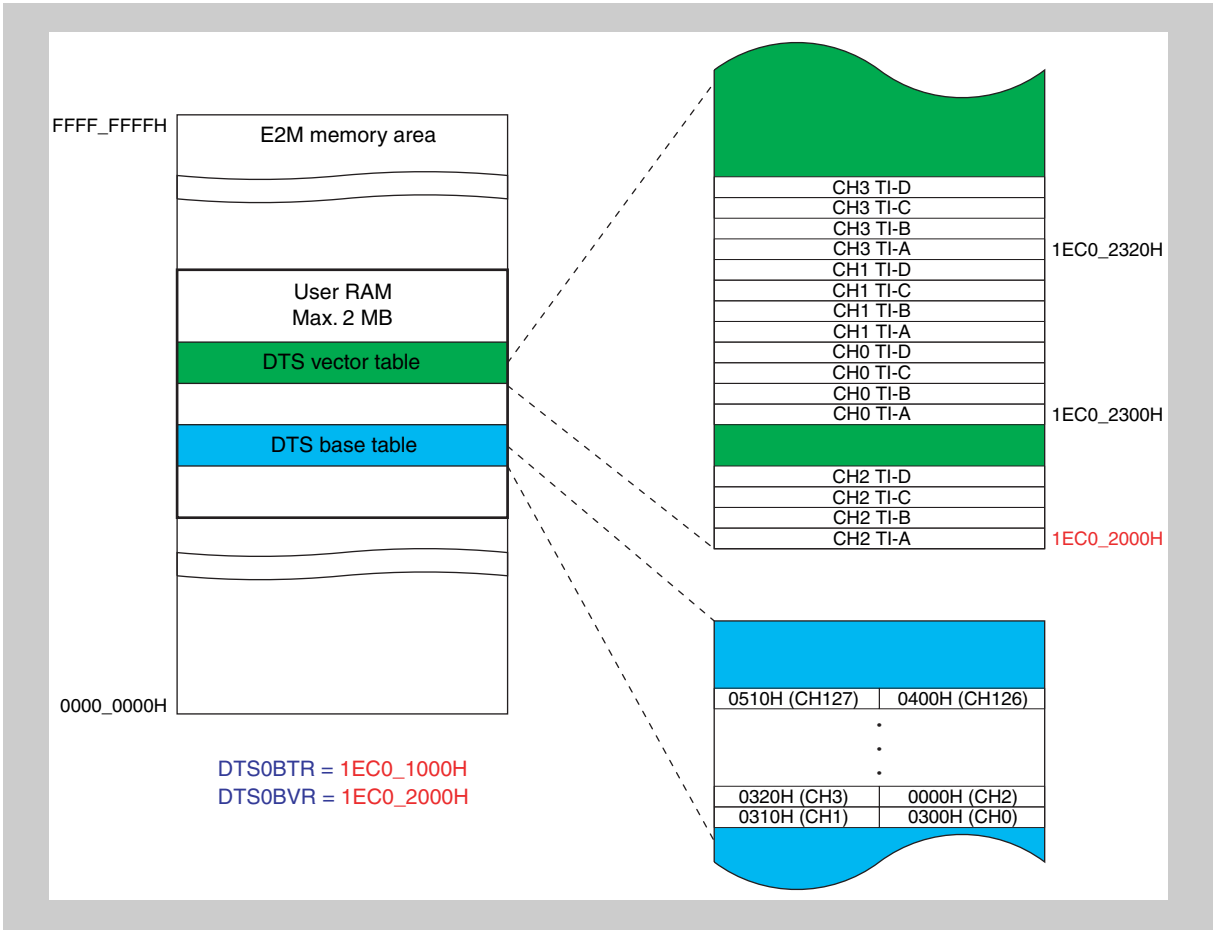


Figure 6-11 Example of address where TI is allocated

(3) Base table

The DTS references a base table to find the TI's position. This base table must be created by the user. The address set by DTS0BTR is the base table's start address, and 16 bits are allocated for each address, starting from channel 0. Data set to the table is offset from the DTS0BVR value (the offset range is 0 to 65,524 when using TI-A, TI-B, and TI-C and 0 to 65,520 when using TI-A, TI-B, TI-C, and TI-D).

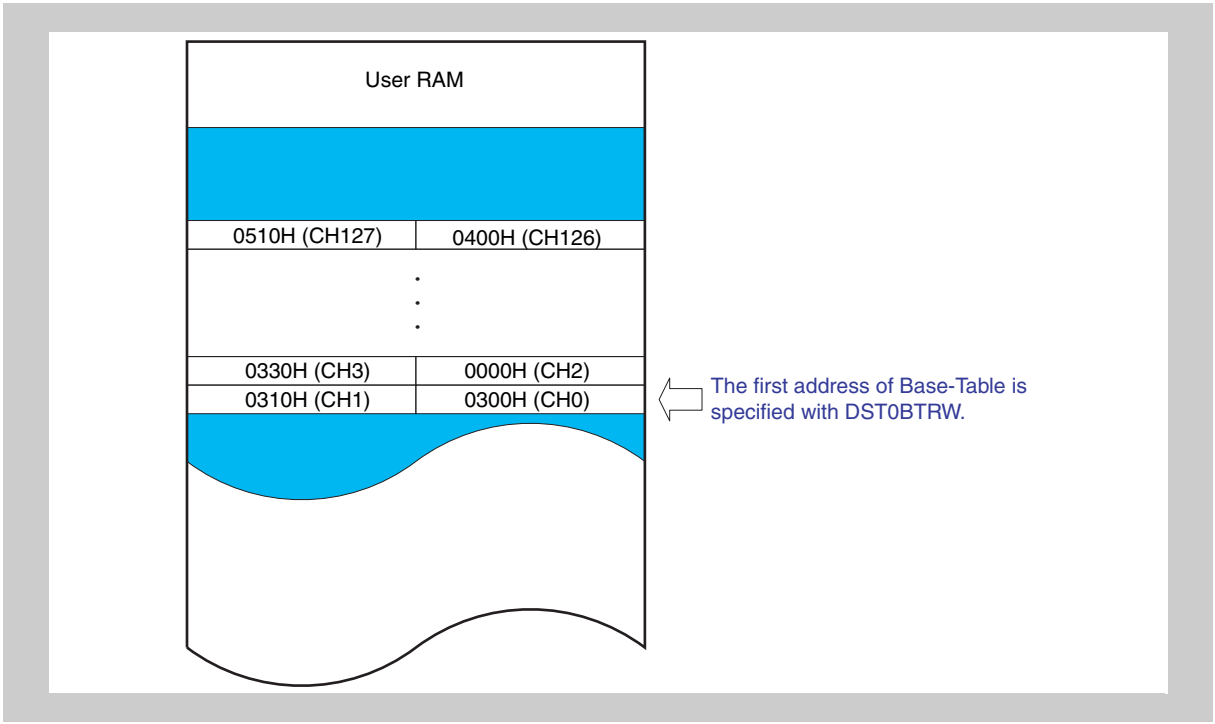


Figure 6-12 Description of base table

6.11.2 DTS transfer setting flow

Figure 6-13 "DTS transfer setting flow" shows the flow for setting DTS transfer by hardware when DTSEN is enabled after TI is set.

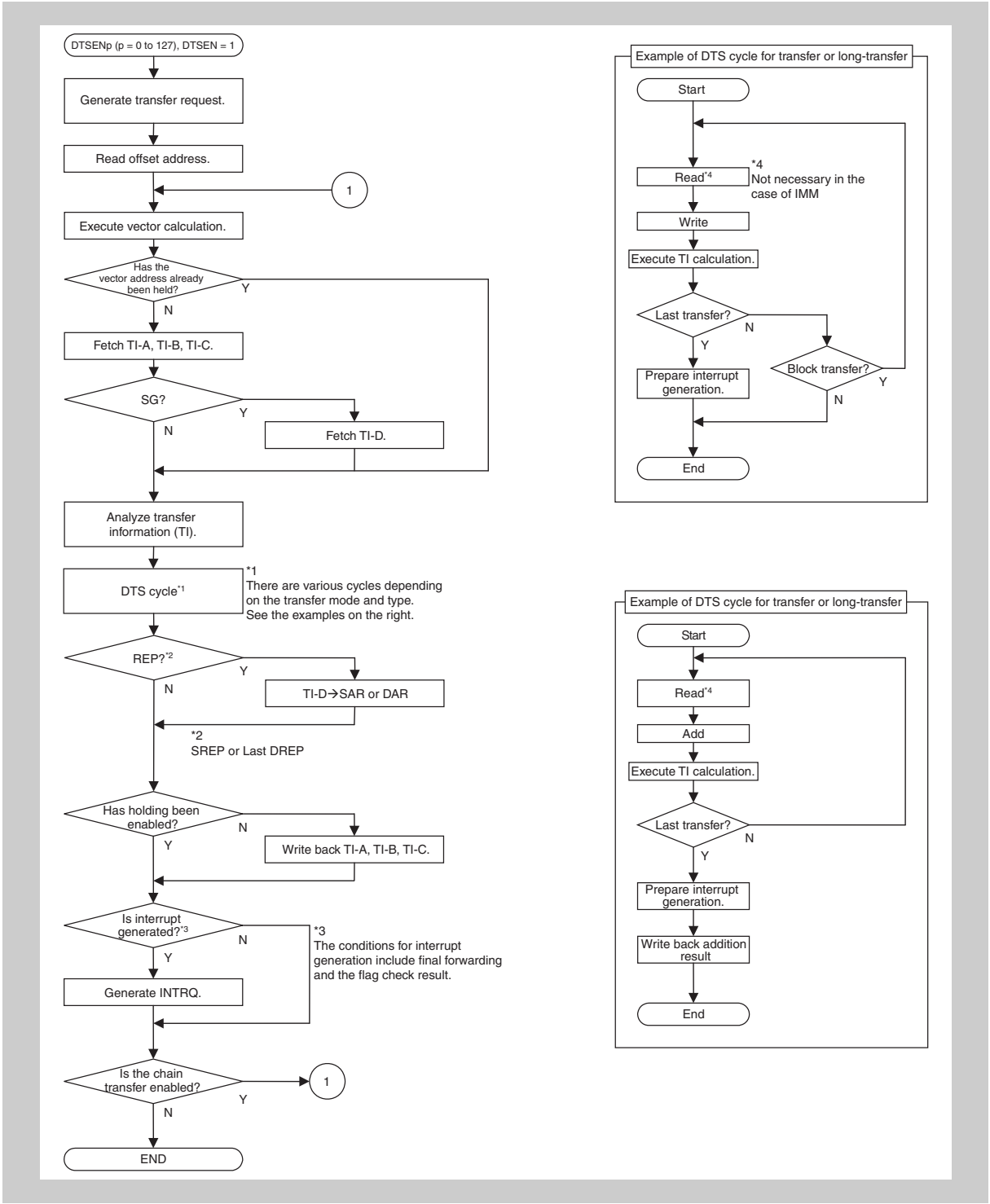


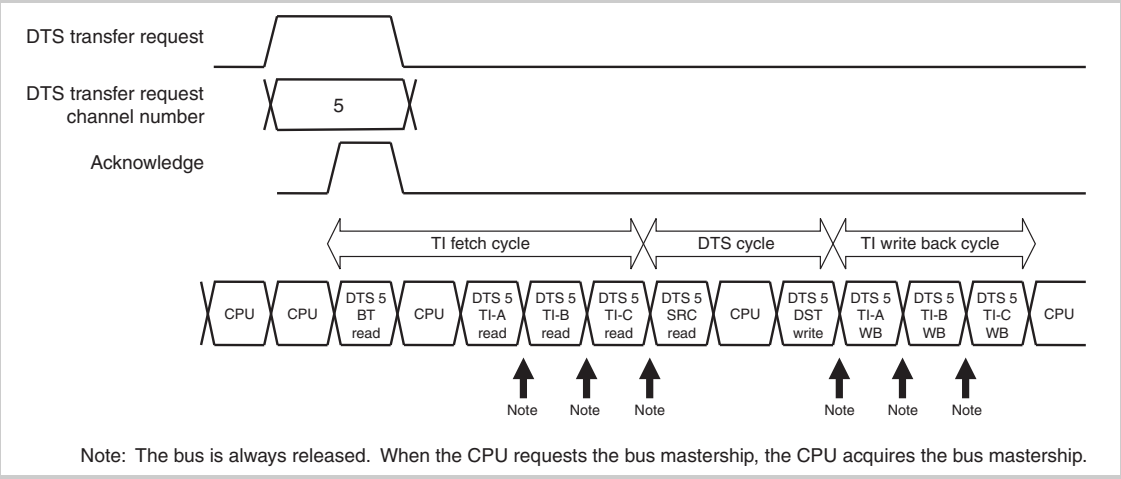
Figure 6-13 DTS transfer setting flow

6.11.3 Basic operations of DTS

The DTS fetches the information required for transfers (TI) from Data RAM, then executes transfers according to this TI, after which it updates the TI and writes it back to Data RAM.

(1) TI fetch and write back

For each DTS request, one round of "TI fetch cycle → DTS cycle → TI write back cycle" occurs. Since the bus is not locked, a CPU cycle may interrupt.



Note: The bus is always released. When the CPU requests the bus mastership, the CPU acquires the bus mastership.

Figure 6-14 Example of TI fetch and write back

6.11.4 Transfer modes

A single-transfer mode and a block transfer mode are supported as transfer modes.

(1) Single transfer mode

When a DTS request is generated in single transfer mode, one round of “TI fetch → DTS cycle → TI write back” is performed and then transfer request wait status is set. When the next DTS request is generated, these cycles are executed again starting from a TI fetch. These operations are repeated until the value of DTS0STC (or DTS0LTC when in 65535 mode) reaches “0”. Since the bus is not locked, a CPU cycle may interrupt.

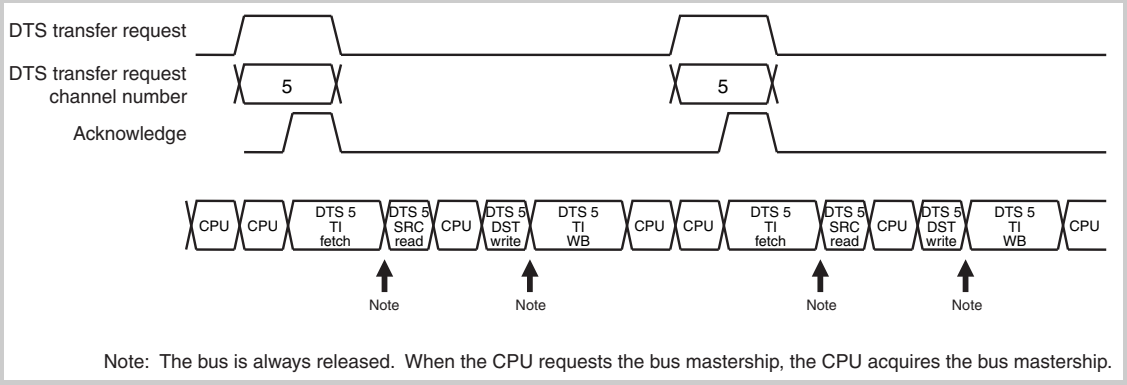


Figure 6-15 Example of single transfer

(2) Block transfer mode

When a DTS request is generated in block transfer mode, TI is fetched and then the DTS cycle is repeated until the value of DTS0STC (or DTS0LTC when in 65535 mode) reaches “0”, after which the TI is written back. Since the bus is not locked, a CPU cycle may interrupt.

Other DTS requests cannot be received during a block transfer (they are held pending in DTSFSL).

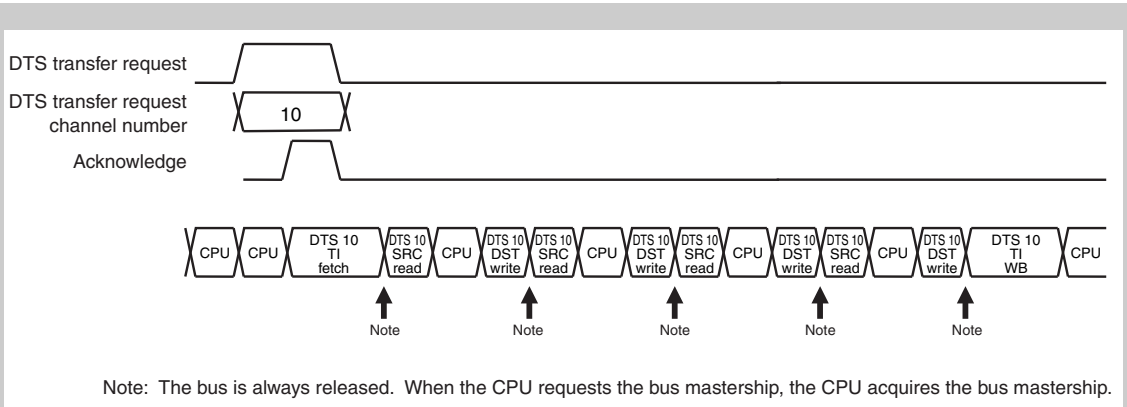


Figure 6-16 Example of block transfer

6.11.5 Transfer types

Transfer (2-cycle transfer), flag check, compare, and add are supported as transfer types. Although transfer types are specified by DTS0TTYP5 to DTS0TTYP0 in DTS0CIR, at the same time the transfer count and the special function[Note] are also set.

The DTS transfer types are listed below. The operation of each transfer type varies depending on the setting of the DTS0IMM bit in DTS0CIR.

Note For a description of the special function, see *6.11.6 "Special functions"*.

Table 6-53 List of DTS transfer types (1/2)

Settings of bits DTS0TTYP5 to DTS0TTYP0			MAX transfer count	Transfer type	Special function	TI-D
0	000	00	255	Transfer	None	Not used
		01			SREP	Used
		10			DREP	Used
		11			SG	Used
0	001	00	255	Add	None	Not used
		01			SREP	Used
		10			DREP	Used
		11			SG	Used
0	010	00	255	flag[0]	None	Not used
		01		flag[0]	SREP	Used
		10		flag[0]	DREP	Used
		11		flag[0]	(SG)	Used
0	011	00	255	flag[1]	None	Not used
		01		flag[1]	SREP	Used
		10		flag[1]	DREP	Used
		11		flag[1]	(SG)	Used
0	100	00	255	comp[=]	None	Not used
		01		comp[=]	SREP	Used
		10		comp[=]	DREP	Used
		11		comp[=]	(SG)	Used
0	101	00	255	comp[!=]	None	Not used
		01		comp[!=]	SREP	Used
		10		comp[!=]	DREP	Used
		11		comp[!=]	(SG)	Used
0	110	00	255	comp[<]	None	Not used
		01		comp[<]	SREP	Used
		10		comp[<]	DREP	Used
		11		comp[<]	(SG)	Used
0	111	00	255	comp[>]	None	Not used
		01		comp[>]	SREP	Used
		10		comp[>]	DREP	Used
		11		comp[>]	(SG)	Used

Table 6-53 List of DTS transfer types (2/2)

Settings of bits DTS0TTYP5 to DTS0TTYP0			MAX transfer count	Transfer type	Special function	TI-D
1	000	00	65536	Long transfer	None	Not used
		01			(SREP)	Used
		10			(DREP)	Used
		11			SG	Used
1	001	00	65536	Long add	None	Not used
		01			(SREP)	Used
		10			(DREP)	Used
		11			SG	Used
1	010		—	flag[0]	Extended chain	Not used
	011			flag[1]		
	100			comp[=]		
	101			comp[!=]		
	110			comp[<]		
	111			comp[>]		

(1) Transfer: TTYP5 to TTYP2 are 0000 or 1000

- When TTYP5 to TTYP2 are 0000 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “transfer”, the DTS cycle is used for data transfer. DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times. When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  The chain mode specification can be 00 (no chain), 01 (transfer count 0 chain), or 11 (always chain), and SREP (Source Repeat), DREP (Destination Repeat), or SG (Scatter & Gather) can be selected as the special function.

Operation Data read from address SAR is written to address DAR.

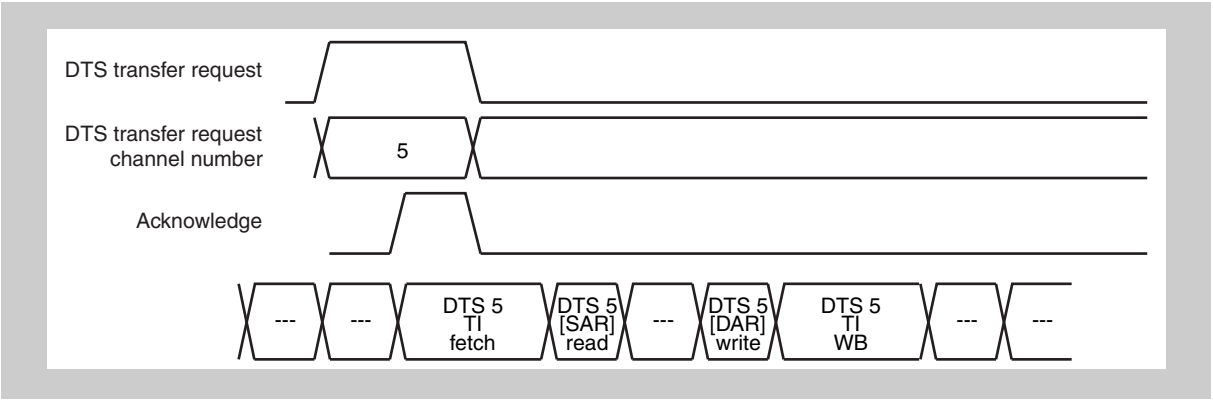


Figure 6-17 When TTYP5 to TYYP2 are 0000 and DTS0IMM in DTS0CIR is “0”

- When TTYP5 to TYYP2 are 0000 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM and the DTS0SAR value is written to address DTS0DAR.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times. When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  The chain mode specification can be 00, 01, or 11, and DREP or SG can be selected as the special function.

Operation The SAR data is written to address DAR without modification.

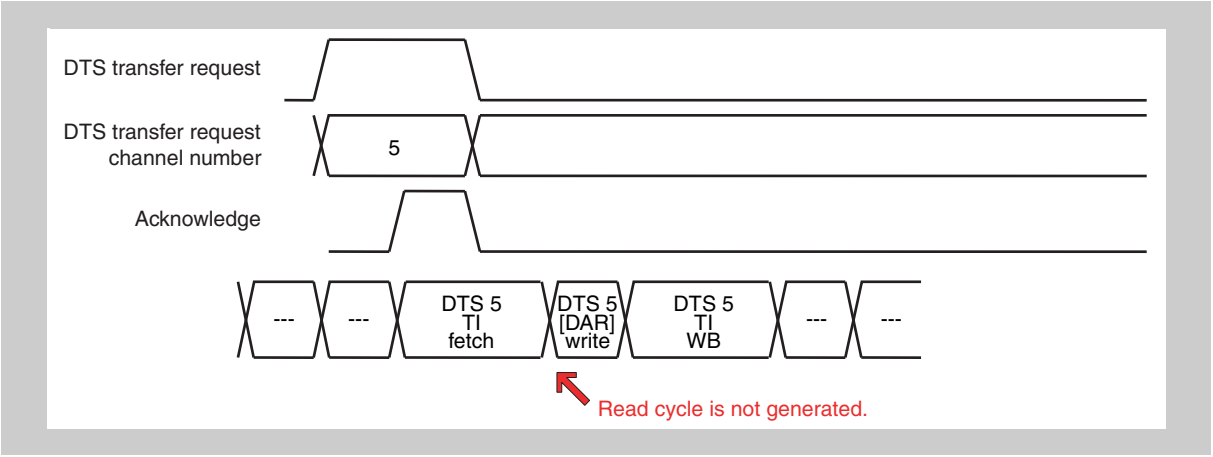


Figure 6-18 When TTYP5 to TYYP2 are 0000 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

- When TTYP5 to TTYP2 are 1000 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “long transfer”, the DTS cycle is used for data transfer. DTS0TCEA becomes DTS0LTC and a transfer count up to 65,535 times can be specified. Once the value of DTS0LTC reaches “0”, no more transfer requests can be accepted. Before executing a transfer, set DTS0LTC (TI-A's DTS0TCEA) again.

  The chain mode specification can be 00, 01, or 11, and SG can be selected as the special function.

Operation Data read from address SAR is written to address DAR.

The figure is the same as *Figure 6-17 “When TTYP5 to TYYP2 are 0000 and DTS0IMM in DTS0CIR is “0””*.

- When TTYP5 to TYYP2 are 1000 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM and the DTS0SAR value is written to address DTS0DAR.

  DTS0TCEA becomes DTS0LTC and a transfer count up to 65,535 times can be specified. When the value of DTS0LTC reaches “0”, no more transfer requests can be accepted. Before executing a transfer, set DTS0LTC (TI-A's DTS0TCEA) again.

  The chain mode specification can be 00, 01, or 11, and SG can be selected as the special function.

Operation The SAR data is written to address DAR without modification.

The figure is the same as *Figure 6-18 “When TTYP5 to TYYP2 are 0000 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)”*.

(2) Add: When TTYP5 to TTYP2 are 0001 or 1001

- When TTYP5 to TTYP2 are 0001 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “add”, the DTS cycle becomes an accumulation cycle.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  The chain mode specification can be 00 or 01, and SREP, DREP, or SG can be selected as the special function.

  DTS0BEN in DTS0CIR is ignored and block transfer mode is always set.

Operation Data is read from address SAR until DTS0STC reaches “0”.
The read data is added each time as unsigned data.
When DTS0STC reaches “0”, the addition result is written to address DAR.

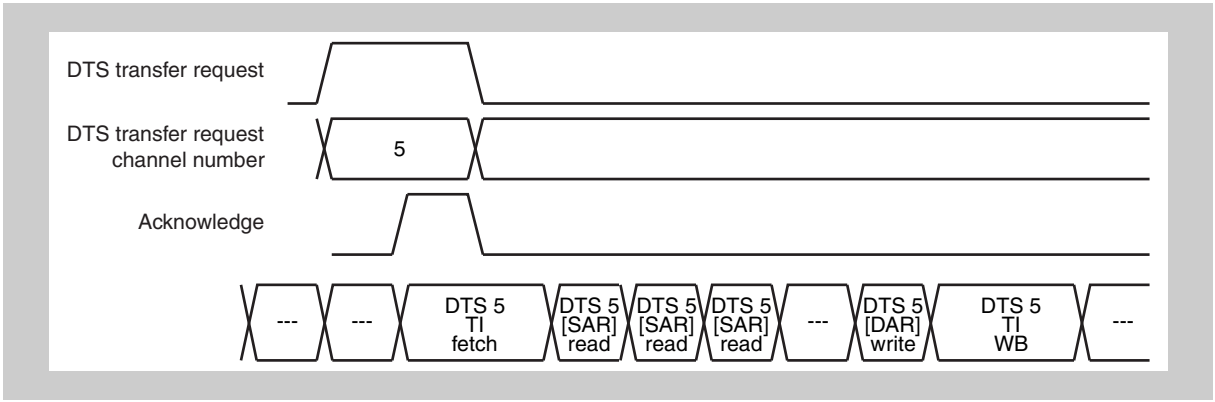


Figure 6-19 When TTYP5 to TTYP2 are 0001 and DTS0IMM in DTS0CIR is “0”

The cumulative value is stored to a 32-bit register regardless of the data size, and when the transfer count reaches “0”, data is written to DAR in 32-bit units. Any data exceeding 32 bits is truncated.

Accordingly, when the transfer size is 32 bits, if an overflow or borrow occurs the calculation result will not be correct. When the transfer size is 16 bits or 8 bits, an overflow or borrow will not occur if the maximum transfer count of 65,535 times is specified.

- When TTYP5 to TYYP2 are 0001 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  DTS0BEN in DTS0CIR is ignored and block transfer mode is always set. In other words, the DTS0SAR value is added the number of times specified by DTS0STC, and when the value of DTS0STC reaches “0”, the result is written to address DAR.

  The chain mode specification can be 00 or 01, and DREP or SG can be selected as the special function.

- When TTYP5 to TTYP2 are 1001 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “long add”, the DTS cycle becomes an accumulation cycle. DTS0TCEA becomes DTS0LTC and a transfer count up to 65,535 times can be specified. When the value of DTS0LTC reaches “0”, no more transfer requests can be accepted.

  Before executing a transfer, set DTS0LTC (TI-A's DTS0TCEA) again.

  The chain mode specification can be 00 or 01, and SG can be selected as the special function.

  DTS0BEN in DTS0CIR is ignored and block transfer mode is always set.

- When TTYP5 to TYYP2 are 1001 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM.

  DTS0TCEA becomes DTS0LTC and a transfer count up to 65,535 times can be specified.

  DTS0BEN in DTS0CIR is ignored and block transfer mode is always set. In other words, the DTS0SAR value is added the number of times specified by DTS0LTC, and when the value of DTS0LTC reaches “0”, the addition result is written to address DAR.

  The chain mode specification can be 00, or 01, and SG can be selected as the special function.

(3) flag[0]: When TTYP5 to TTYP2 are 0010 or 1010

- When TTYP5 to TTYP2 are 0010 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “flag [0]”, the DTS cycle becomes a flag check [0] cycle.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (SREP, DREP, or SG) and all chain modes can be selected.

  The function of flag check [0] is to check whether an arbitrary bit in the data read from address DAR is “0”.

  If at least one bit is “0”, the check result is “true”. The check result can be used as a chain condition or as interrupt output.

Operation First, the check bit specification data is read from address SAR, and then the data to be checked is read from address DAR to execute a flag check.

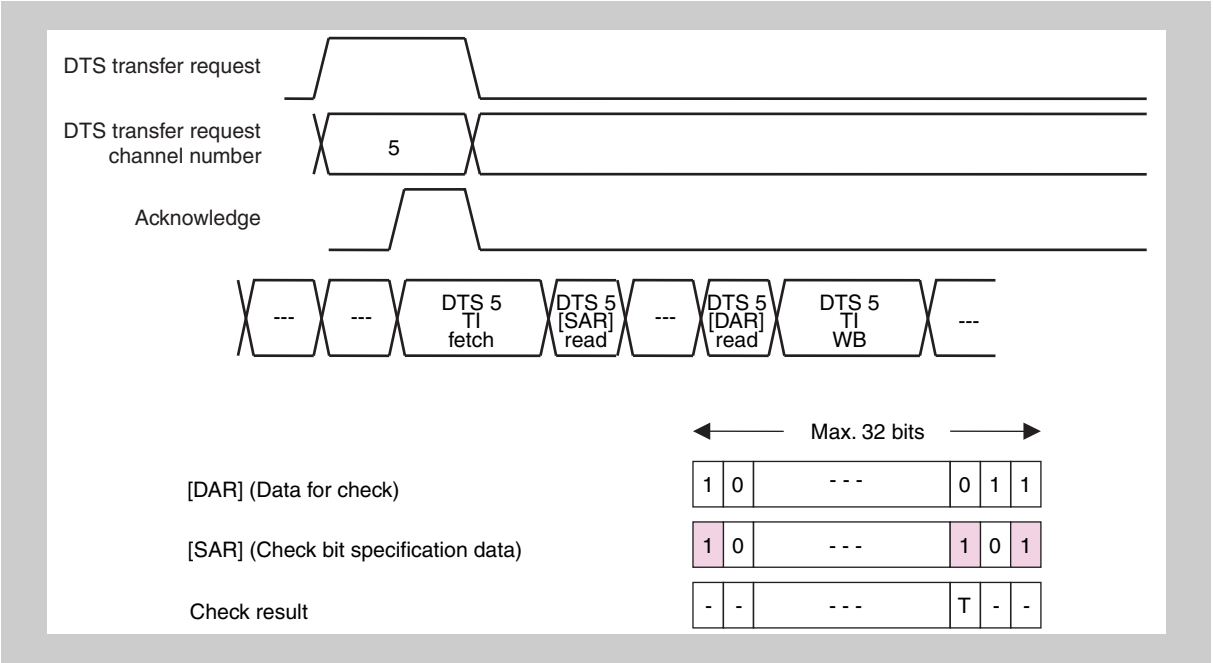


Figure 6-20 When TTYP5 to TTYP2 are 0010 and DTS0IMM in DTS0CIR is “0”

- When TTYP5 to TYYP2 are 0010 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the check bit.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (DREP or SG) and all chain modes can be selected.

  The function of flag check [0] is to check whether an arbitrary bit in the data read from address DAR is “0”.

  If at least one bit is “0”, the check result is “true”.

  The check result can be used as a chain condition or as interrupt output.

Operation The data to be checked is read from address DAR, and a flag check is executed using the SAR value as the check bit specification data.

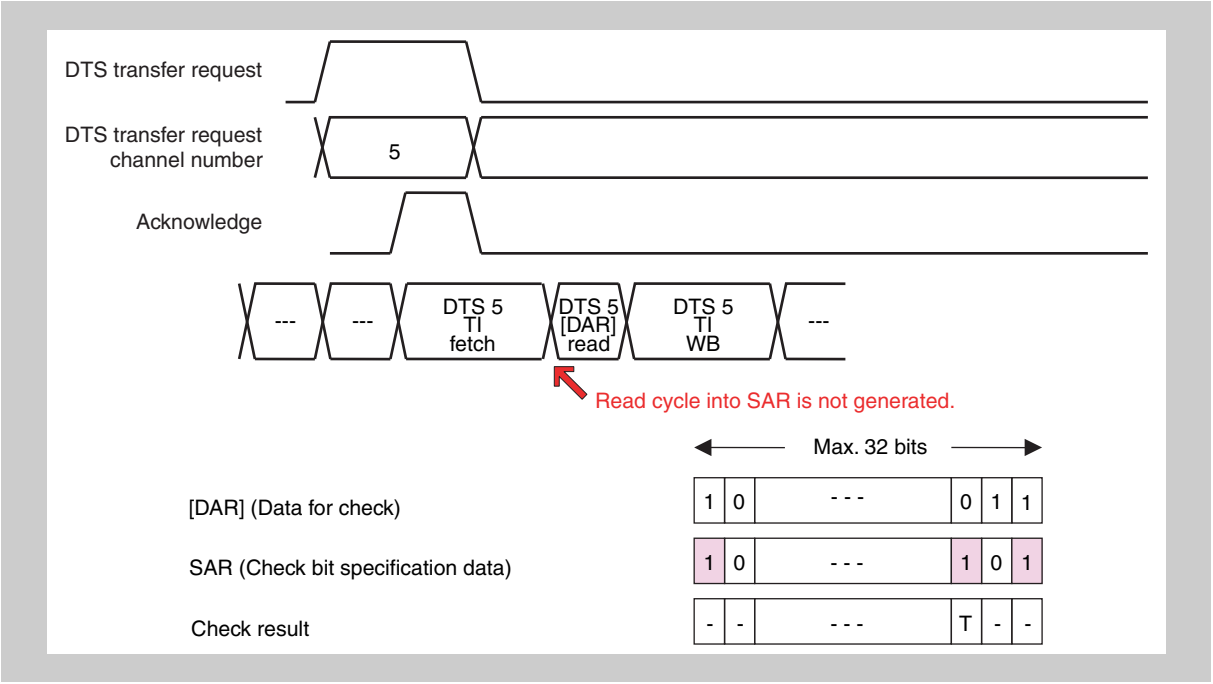


Figure 6-21 When TTYP5 to TYYP2 are 0010 and immediate function is enabled (DTS0IMM in DTS0CIR is "1")

- When TTYP5 to TTYP2 are 1010 and DTS0IMM in DTS0CIR is "0"

  When the transfer type is set to "extended flag[0]", the DTS cycle becomes a flag check [0] cycle.
  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is "false".
  Accordingly, the transfer count cannot be specified. When the check result is "true", TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is "false", TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.
  Neither the chain mode nor a special function can be selected.

Operation First, the check bit specification data is read from address SAR, and then the data to be checked is read from address DAR to execute a flag check.
A chain of transfers is always performed.

- When TTYP5 to TYYP2 are 1010 and immediate function is enabled (DTS0IMM in DTS0CIR is "1")

  When DTS0IMM is "1", DTS0SAR becomes IMM, and DTS0SAR becomes the check bit.
  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is "false".
  Accordingly, the transfer count cannot be specified. When the check result is "true", TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is "false", TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.
  Neither the chain mode nor a special function can be selected.

Operation The data to be checked is read from address DAR, and a flag check is executed using the SAR value as the check bit specification data.
A chain of transfers is always performed.

(4) flag[1]: When TTYP5 to TTYP2 are 0011 or 1011

- When TTYP5 to TTYP2 are 0011 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “flag [1]”, the DTS cycle becomes a flag check [1] cycle.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (SREP, DREP, or SG) and all chain modes can be selected. The function of flag check [1] is to check whether an arbitrary bit in the data read from address DAR is “1”.

  If at least one bit is “1”, the check result is “true”.

  The check result can be used as a chain condition or as interrupt output.

Operation First, the check bit specification data is read from address SAR, and then the data to be checked is read from address DAR to execute a flag check.

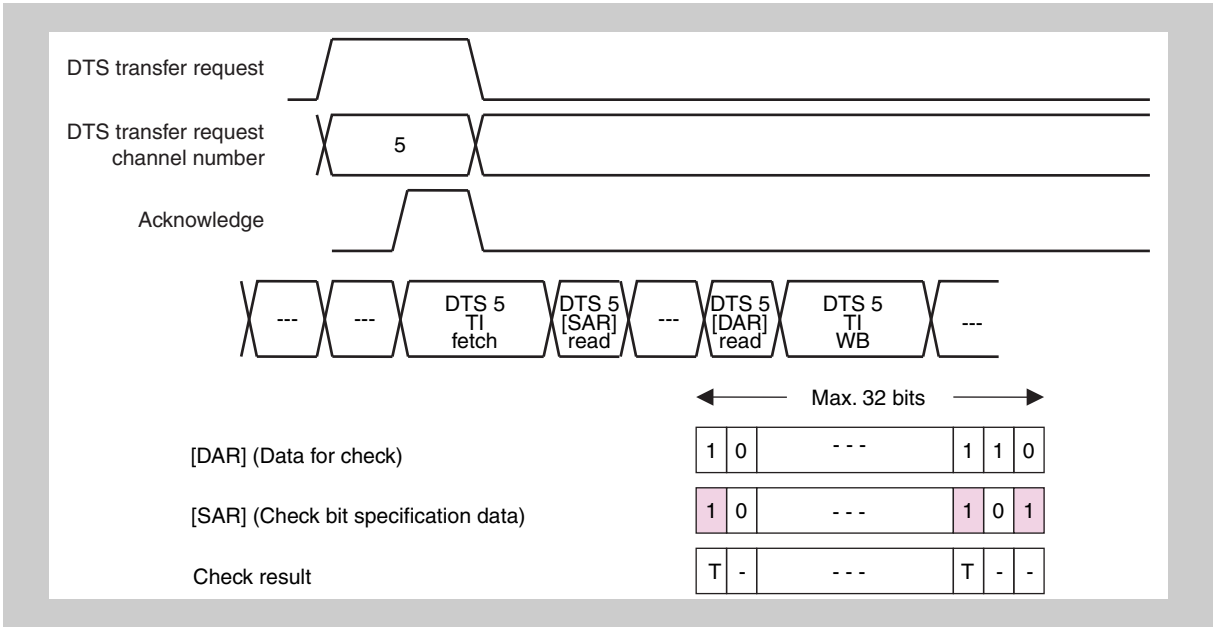


Figure 6-22 When TTYP5 to TTYP2 are 0011 and DTS0IMM in DTS0CIR is “0”

- When TTYP5 to TYYP2 are 0011 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the check bit.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (DREP or SG) and all chain modes can be selected.

The function of flag check [1] is to check whether an arbitrary bit in the data read from address DAR is “1”.

If at least one bit is “1”, the check result is “true”. The check result can be used as a chain condition or as interrupt output.

Operation The data to be checked is read from address DAR, and a flag check is executed using the SAR value as the check bit specification data.

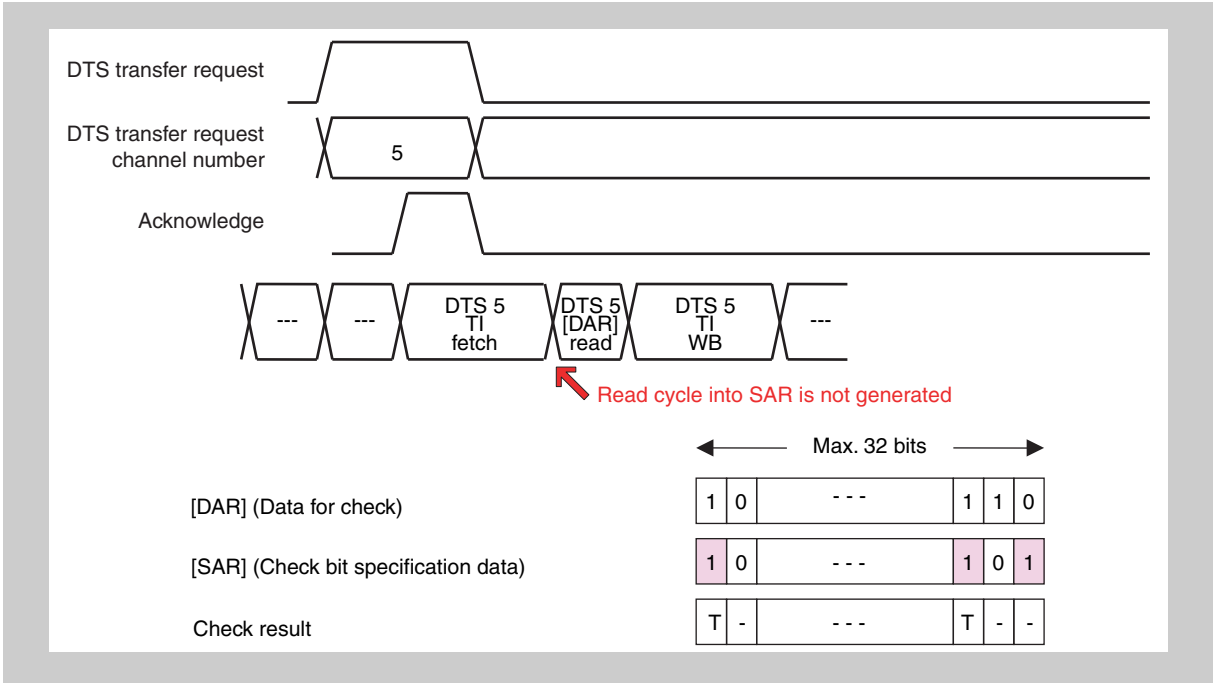


Figure 6-23 When TTYP5 to TYYP2 are 0011 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

- When TTYP5 to TTYP2 are 1011 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “extended flag[1]”, the DTS cycle becomes a flag check [1] cycle.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation First, the check bit specification data is read from address SAR, and then the data to be checked is read from address DAR to execute a flag check.
A chain of transfers is always performed.

- When TTYP5 to TYYP2 are 1011 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the check bit.

DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

Neither the chain mode nor a special function can be selected.

Operation The data to be checked is read from address DAR, and a flag check is executed using the SAR value as the check bit specification data.
A chain of transfers is always performed.

(5) comp[=]: When TTYP5 to TTYP2 are 0100 or 1100

- When TTYP5 to TTYP2 are 0100 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “comp[=]”, the DTS cycle becomes a comparison [=] cycle. DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (SREP, DREP, or SG) and all chain modes can be selected.

  With comp[=], data read from address SAR is compared with data read from address DAR, and if these data match, the compare result is “true”.

  The check result can be used as a chain condition or as interrupt output.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [SAR] == [DAR] is checked.

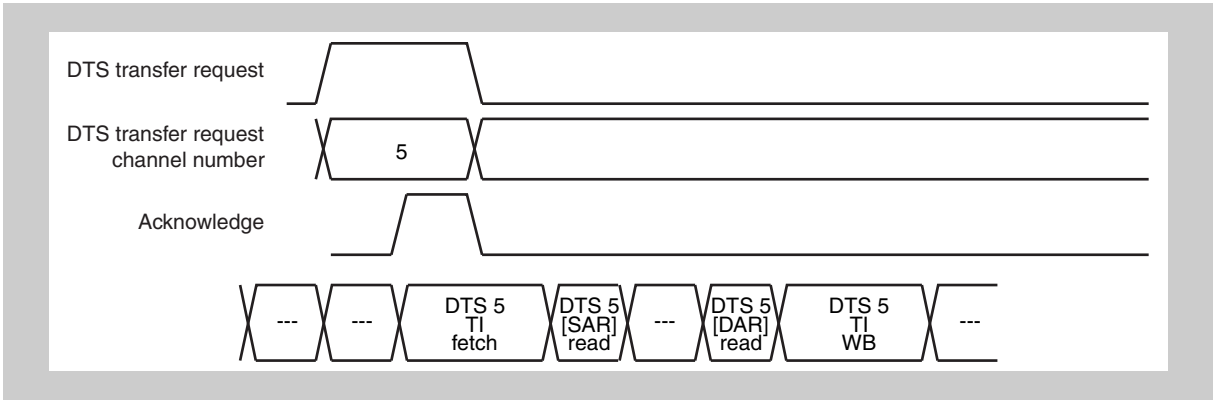


Figure 6-24 When TTYP5 to TTYP2 are 0100 and DTS0IMM in DTS0CIR is “0”

- When TTYP5 to TYYP2 are 0100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

A special function (DREP or SG) and all chain modes can be selected.
The check result can be used as a chain condition or as interrupt output.

Operation Data is read from address DAR, and whether SAR == [DAR] is checked.

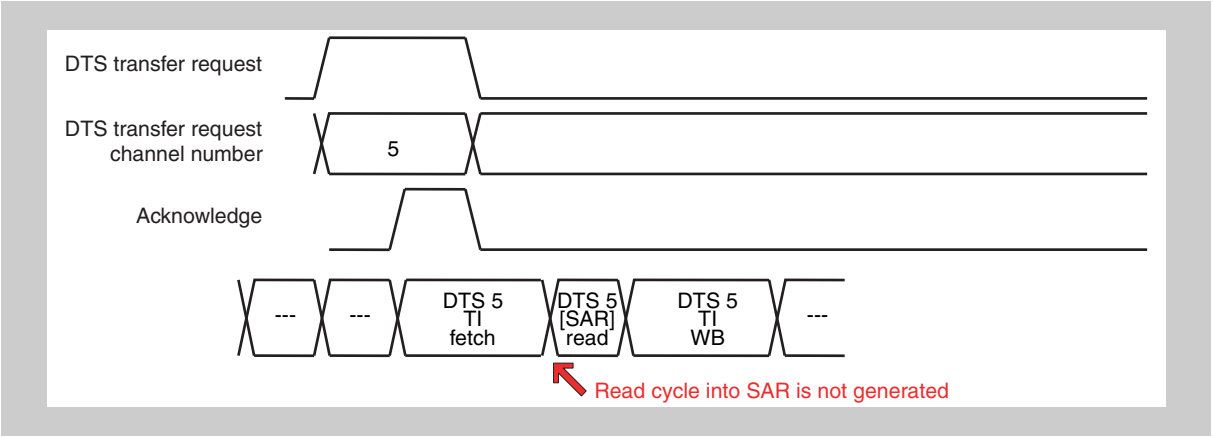


Figure 6-25 When TTYP5 to TYYP2 are 0100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

- When TTYP5 to TTYP2 are 1100 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “extended comp[=]”, the DTS cycle becomes a comparison [=] cycle.
  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.
  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.
  Neither the chain mode nor a special function can be selected.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [SAR] == [DAR] is checked.
A chain of transfers is always performed.

- When TTYP5 to TYYP2 are 1100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.
  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.
  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.
  Neither the chain mode nor a special function can be selected.

Operation The data to be compared is read from address DAR, and then compared with the SAR value.
A chain of transfers is always performed.

(6) comp[!=]: When TTYP5 to TTYP2 are 0101 or 1101

- When TTYP5 to TTYP2 are 0101 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “comp[!=]”, the DTS cycle becomes a comparison [!=] cycle.
  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.
  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.
  A special function (SREP, DREP, or SG) and all chain modes can be selected.
  With comp[!=], data read from address SAR is compared with data read from address DAR, and if these data do not match, the compare result is “true”.
  The check result can be used as a chain condition or as interrupt output.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [SAR] != [DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and DTS0IMM in DTS0CIR is “0”.

- When TTYP5 to TYYP2 are 0101 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.
  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.
  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.
  A special function (DREP or SG) and all chain modes can be selected.
  The check result can be used as a chain condition or as interrupt output.

Operation Data is read from address DAR, and whether SAR != [DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”).

- When TTYP5 to TTYP2 are 1101 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “extended comp[!=]”, the DTS cycle becomes a comparison [!=] cycle.
  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.
  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

Neither the chain mode nor a special function can be selected.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [SAR] != [DAR] is checked.
A chain of transfers is always performed.

- When TTYP5 to TYYP2 are 1101 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation The data to be compared is read from address DAR, and then compared with the SAR value.
A chain of transfers is always performed.

(7) comp[<]: When TTYP5 to TTYP2 are 0110 or 1110

- When TTYP5 to TTYP2 are 0110 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “comp[<]”, the DTS cycle becomes a comparison [<] cycle.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (SREP, DREP, or SG) and all chain modes can be selected.

  With comp[<], data read from address SAR is compared with data read from address DAR, and if the data read from address SAR is smaller, the compare result is “true”.

  The check result can be used as a chain condition or as interrupt output.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [(unsigned) SAR] < [(unsigned) DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and DTS0IMM in DTS0CIR is “0”.

- When TTYP5 to TTYP2 are 0110 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

A special function (DREP or SG) and all chain modes can be selected.

The check result can be used as a chain condition or as interrupt output.

Operation Data is read from address DAR, and whether (unsigned) SAR < [(unsigned) DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”).

- When TTYP5 to TTYP2 are 1110 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “extended comp[<]”, the DTS cycle becomes a comparison [<] cycle.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [(unsigned) SAR] < [(unsigned) DAR] is checked.

A chain of transfers is always performed.

- When TTYP5 to TTYP2 are 1110 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation The data to be compared is read from address DAR, and then compared with the SAR value.
A chain of transfers is always performed.

(8) comp[>]: When TTYP5 to TTYP2 are 0111 or 1111

- When TTYP5 to TTYP2 are 0111 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “comp[>]”, the DTS cycle becomes a comparison [>] cycle.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (SREP, DREP, or SG) and all chain modes can be selected.

  With comp[>], data read from address SAR is compared with data read from address DAR, and if the data read from address SAR is greater, the compare result is “true”.

  The check result can be used as a chain condition or as interrupt output.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [(unsigned) SAR] > [(unsigned) DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and DTS0IMM in DTS0CIR is “0”.

- When TTYP5 to TTYP2 are 0111 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

  DTS0TCEA is split into DTS0STR and DTS0STC, and DTS0STC can be used to specify a transfer count value up to 255 times.

  When the value of DTS0STC reaches “0”, the value of DTS0STR is copied to DTS0STC and another transfer can be performed.

  A special function (DREP or SG) and all chain modes can be selected.

  The check result can be used as a chain condition or as interrupt output.

Operation Data is read from address DAR, and whether (unsigned) SAR > [(unsigned) DAR] is checked.

The transfer cycle is the same as when TTYP5 to TTYP2 are 0100 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”).

- When TTYP5 to TTYP2 are 1111 and DTS0IMM in DTS0CIR is “0”

  When the transfer type is set to “extended comp[>]”, the DTS cycle becomes a comparison [>] cycle.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation First, data is read from address SAR, then data is read from address DAR. Next, whether [(unsigned) SAR] > [(unsigned) DAR] is checked.
A chain of transfers is always performed.

- When TTYP5 to TTYP2 are 1111 and immediate function is enabled (DTS0IMM in DTS0CIR is “1”)

  When DTS0IMM is “1”, DTS0SAR becomes IMM, and DTS0SAR becomes the data to be compared.

  DTS0TCEA becomes DTS0EA, functioning as a register that specifies an offset address used when the check result is “false”.

  Accordingly, the transfer count cannot be specified. When the check result is “true”, TI is fetched from the current TI fetch address added to 10H to perform a chain of transfers. When the check result is “false”, TI is fetched from the current TI fetch address added to the value of DTS0EA to perform a chain of transfers.

  Neither the chain mode nor a special function can be selected.

Operation The data to be compared is read from address DAR, and then compared with the SAR value.
A chain of transfers is always performed.

6.11.6 Special functions

SREP, DREP, and SG are supported as special functions.

The special function is selected using DTS0TTYP1 and DTS0TTYP0 in DTS0CIR.

(1) SREP (Source address REPeat)

When the value of DTS0STC reaches “0”, this function returns DTS0SAR to its initial value. The TI-D area is used.

Set a value equal to the DTS0SAR value in advance to the TI-D area.

DTS cycles are executed until DTS0STC reaches “0”, then when the last transfer occurs the TI-D area's value is read to DTS0SAR, and a write back is performed to achieve SREP. With SREP, when DTS0STC reaches “0”, a write back of TI-B is always performed (it is never skipped).

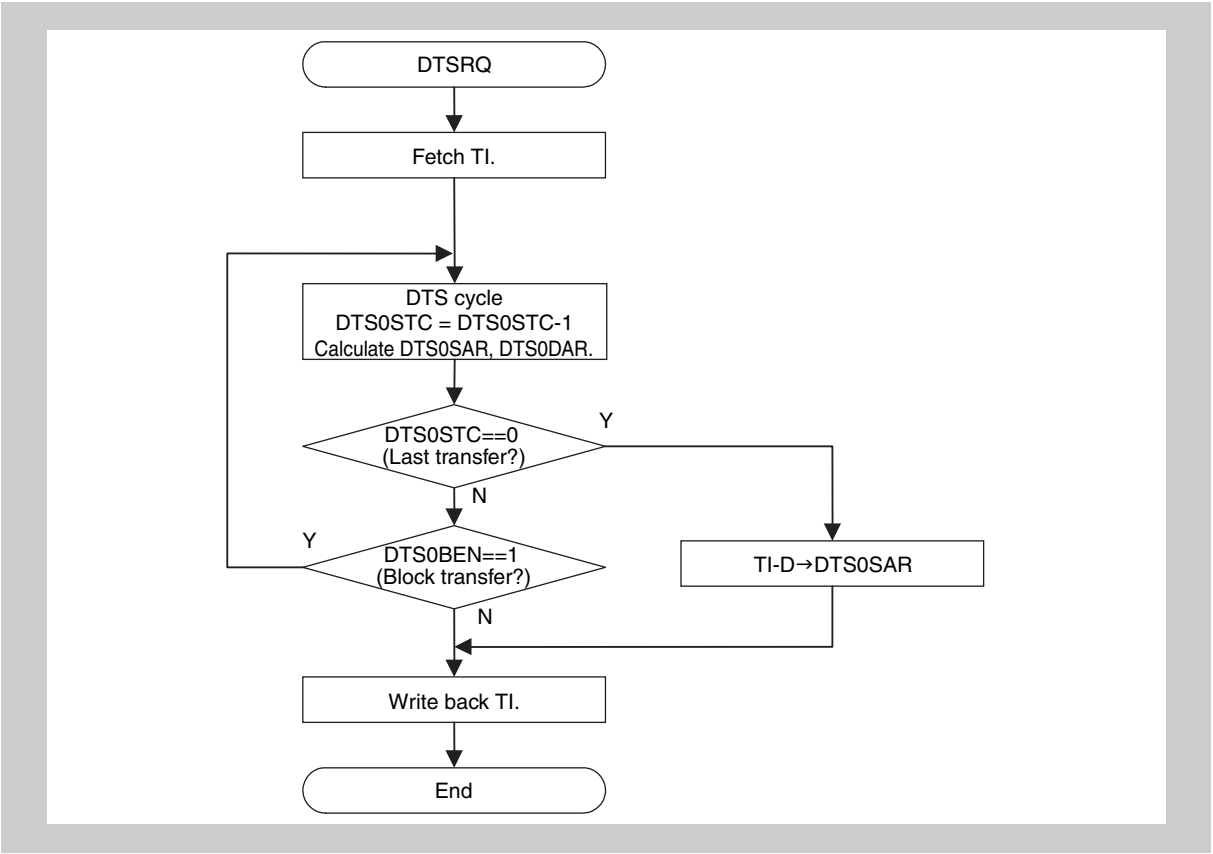


Figure 6-26 SREP processing flow

(2) DREP (Destination address REPeat)

When the value of DTS0STC reaches “0”, this function returns DTS0DAR to its initial value. The TI-D area is used.

The value is returned using the same method as for SREP, and the TI-D value is written back to DTS0DAR.

With DREP, when DTS0STC reaches “0”, a write back of TI-C is always performed (it is never skipped).

Note SREP and DREP cannot be set at the same time (since the TI-D area is used).

(3) SG (Scatter & Gather)

In this mode, the count sizes of the source address and destination address are specified as signed 16-bit values. The TI-D area is used. When in modes other than SG, the DTS's address count size can only be specified using DTS0SCS and DTS0DCS, as a value in the range from –4 to +8, but in SG mode DTS0ESCS and DTS0EDCS are used, which enables the count size to become much larger (data in the TI-D area is read to DTS internal registers DTS0ESCS and DTS0EDCS).

6.11.7 Transfer count

The transfer count is MAX255 times or MAX65535 times. In either case, when the specified number of transfers is completed, an interrupt request is output to DTSFSL.

(1) MAX255 times

Although the maximum transfer count is 255 times, the transfer count can be recovered when the transfer count (DTS0STC) reaches “0”, so in fact unlimited transfers are supported.

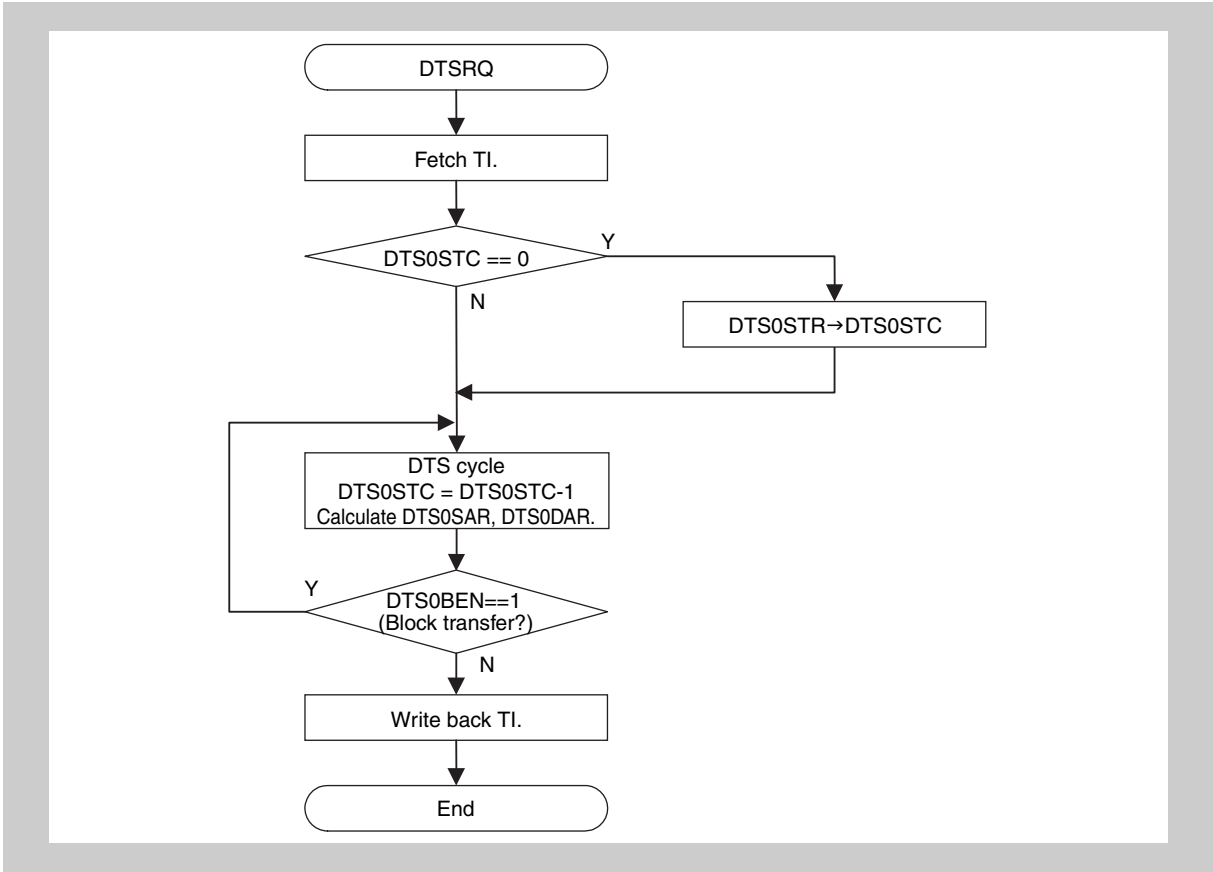


Figure 6-27 Transfer count processing flow

(2) MAX65535 times

When DTS0TCEA is used as a 16-bit counter, a transfer count of up to 65,535 times can be specified.

6.11.8 Chain function

DTS includes a chain function that performs multiple data transfers in succession.

The condition for the chain function can be set by using DTS0CM1 and DTS0CM0 in DTS0CIR. Also, DTS0TTYP4 to DTS0TTYP0 in DTS0CIR can be used to select an extended chain function.

When extended chain function (special flag[0], special flag[1], or special comp[=, !=, <, >]) has been selected, the settings in DTS0CM1 and DTS0CM0 are ignored.

After TI fetch → DTS cycle → TI write back is completed, the chain function performs a TI fetch, achieving a chain of transfers.
The fetch address for this TI fetch operation is aways the address

TI-D + 4 = TI-A + 16,

regardless of the presence or absence of TI-D in the previous TI.

Table 6-54 List of chain functions

Transfer mode	Special mode	Settings of DTS0CM1 and DTS0CM0 in DTS0CIR	Operation
transfer	—	00 (No chain)	Does not perform chain of transfers.
		01 (Last transfer)	Performs chain of transfers when DTS0STC (or DTS0LTC when 65535) is 0.
		10 (Chain if true)	Does not perform chain of transfers.
		11 (Always chain)	Performs chain of transfers following DTS cycle. Same as “01” during block transfer.
add	—	00 (No chain)	Does not perform chain of transfers.
		01 (Last transfer)	Performs chain of transfers when DTS0STC (or DTS0LTC when 65535) is 0.
		10 (Chain if true)	Does not perform chain of transfers.
		11 (Always chain)	Performs chain of transfers following DTS cycle. Same as “01” during block transfer.
flag	None SREP DREP	00 (No chain)	Does not perform chain of transfers.
		01 (Last transfer)	Performs chain of transfers when DTS0STC is 0.
		10 (Chain if true)	Performs chain of transfers only when condition is true. An interrupt is generated when it is false.
		11 (Always chain)	Performs chain of transfers following DTS cycle. Same as “01” during block transfer.
	Extended chain	Ignored	Always performs chain of transfers. When condition is true, TI is fetched from current TI fetch address added to 10H to perform chain of transfers (same manner as normal chain transfer). When condition is false, TI is fetched from current TI fetch address added to ±DTS0TCEA to perform chain of transfers.
comp	None SREP DREP	00 (No chain)	Does not perform chain of transfers.
		01 (Last transfer)	Performs chain of transfers when DTS0STC is 0.
		10 (Chain if true)	Performs chain of transfers only when condition is true. An interrupt is generated when it is false.
		11 (Always chain)	Performs chain of transfers following DTS cycle. Same as “01” during block transfer.
	Extended chain	Ignored	Always performs chain of transfers. When condition is true, TI is fetched from current TI fetch address added to 10H to perform chain of transfers (same manner as normal chain transfer). When condition is false, TI is fetched from current TI fetch address added to ±DTS0TCEA to perform chain of transfers.

6.11.9 TI hold function

The DTSHENm (m = 0 to 3) register of DTSFSL can be used to enable the TI hold function (ON).

(1) Single transfer when TI hold is ON

When the TI hold function is set to ON in single transfer mode, TI fetch → DTS cycle is executed for one DTS request, then transfer request wait status is set (TI is set to TI hold status). After this, if a DTS request occurs for another channel, TI fetch → DTS cycle → TI write back is executed for that channel. If a DTS request occurs for the same channel as the TI hold, processing goes as far as reading the BT (base table), but since the TI has already been set to TI hold status, a TI fetch is not executed when the DTS cycle occurs.

This operation is repeated until the value of DTS0STC (or DTS0LTC when in 65535 mode) reaches “0”.

When DTS0STC (or DTS0LTC) reaches “0”, TI hold is set to OFF and the TI is written back.

Since the bus is not locked, a CPU cycle may interrupt.

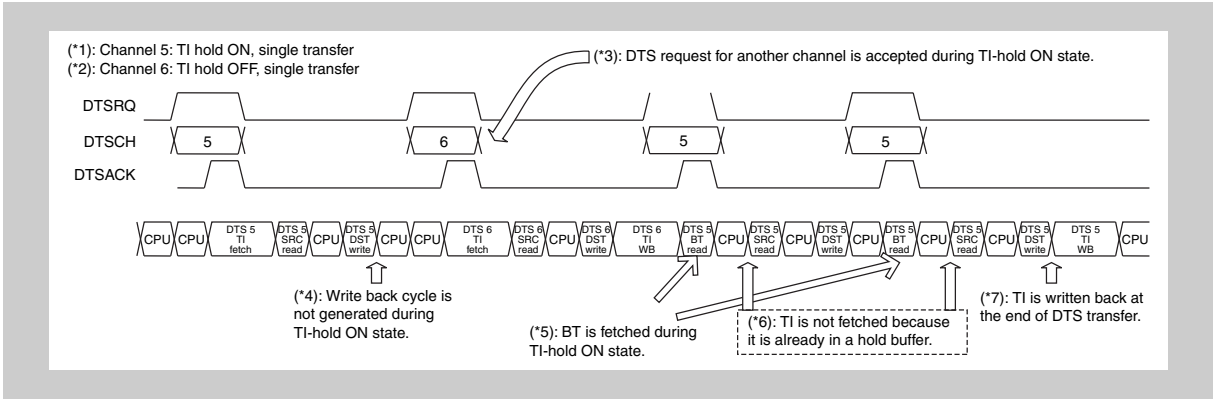


Figure 6-28 TI hold function

(2) Number of TI hold buffers

Four sets of TI hold buffers are provided. The hold instruction for these hold buffers is specified using the DTSHENm (m = 3 to 0) register of DTSFSL.

6.11.10 Interrupt output function

DTS is able to output interrupts to the CPU via DTSFSL. DTS outputs an interrupt source signal to DTSFSL, and DTSFSL then decodes this signal and transmits it to the CPU as an interrupt signal.

The conditions for enabling the interrupt source signal (DTSINTRQ) involves the transfer count, flag check result, and compare result.

With regard to the transfer count, when only one transfer remains (the last transfer), DTSINTRQ activates upon completion of this transfer.

With regard to the flag check and compare results , if either of their judgment results is false, DTSINTRQ activates upon completion of the transfer.

The interrupt source signal (DTSINTRQ) can be masked using the DTS0DISI bit in DTS0CIR.

DTSINTRQ does not become active when the DTS0DISI bit in DTS0CIR is “1”.

6.11.11 Interrupt and chain of transfers during block transfer

In the case of a block transfer, judgments concerning interrupts and chain of transfers are performed only once when the number of transfers specified by DTS0TCEA has been completed.

With regard to interrupts, an interrupt is always output when the specified number of transfers has been completed (the counter has reached “0”). (Note that interrupt masking is possible using DTS0DISI in DTS0CIR.)

With regard to a chain of transfers, when DTS0CM1 and DTS0CM0 in DTS0CIR are “01” (chain when counter is 0) or “11” (always chain), a chain of transfers is always performed. When these bits are “10” (chain if true), a chain of transfers is performed only when all the flag check and compare results are true (and a chain of transfers is not performed if any result is false).

DTS0TCF in DTS0CIR becomes “1” if any of the flag check and compare results is false.

6.11.12 TI write back skip function

DTS skips any TI write back that is not necessary. The conditions for skipping write back are as follows.

Table 6-55 Write back skip conditions

	DTS0CIR				Transfer count	SCS	SAM
	DS	IMM	Type	Special function			
TI-B write back skip conditions	—	—	—	SG	—	—	0
	—	0	Not add	SREP	Some remain	0	—
	—	0	—	DREP/ None	—	0	—
	32 bits	1	Not add	SREP	Some remain	—	—
	8/16 bits	1	Not add	SREP	Some remain	0	—
	32 bits	1	—	DREP/ None	—	—	—
	8/16 bits	1	—	DREP/ None	—	0	—

	DTS0CIR		Transfer count	SCS	SAM
	Type	Special function			
TI-B write back skip conditions	—	SG	—	—	0
	—	SREP	—	0	—
	Not add	DREP	Some remain	0	—
	—	None	—	0	—

6.11.13 DTS channel priority control

The priority function can set up to four priority levels using DTSFSL.

6.11.14 Valid DTS transfer request conditions

When the DTSEN bit in DTSEN000 to DTSEN127 of DTSFSL is set to “0”, a transfer request to the DTS is not generated for the corresponding channel, but sampling of DTS transfer factors [255:128] is performed. When sampled, if the DTSEN bit is set to “1”, a DTS transfer request is generated.

When the DTSEN bit is “1”, if INTIN occurs when the DTS is transferring data or is stopped either by an ELBERR response or by a user, the INITIN is held pending in DTSFSL until the DTS is able to accept it.

Table 6-56 Valid DTS transfer request conditions

DTS	DTS	DTSFSL	Status of DTS	Transfer request by DTS transfer factors [255:128]
DTS0TSR.bit1	DTS0TSR.bit0	DTSEN		
x	x	0	—	Invalid (held)
0	0	1	Acceptable	Valid (if next request occurs during a transfer, it is held)
x	1	1	Not acceptable	Held in DTSFSL
1	x	1	Not acceptable	Held in DTSFSL

6.11.15 Aborting/resuming DTS transfer

Table 6-57 DTS transfer abort timing

Transfer mode	Transfer count	Aborted after:
Single transfer with TI hold OFF	Last transfer	TI write back
	Some remain	
Block transfer with TI hold OFF	Last transfer	TI write back
	Some remain	DTS cycle
Single transfer with TI hold ON	Last transfer	TI write back
	Some remain	DTS cycle
Block transfer with TI hold ON	Last transfer	TI write back
	Some remain	DTS cycle

(1) Aborting or resuming DTS transfer for all channels through hardware (NMI)

Not supported

(2) Aborting or resuming DTS transfer for all channels through software

By setting the DTS transfer abort request trigger bit (DTS0UST) in the DTS transfer request control register (DTS0TRC), the next DTS transfer and those that follow can be aborted. If this occurs during a DTS cycle, the next DTS transfer is aborted according to the condition shown in *Table 6-57 "DTS transfer abort timing"*.

To resume the aborted DTS transfer, set the DTS0UCL bit in the DTS transfer request control register (DTS0TRC), and then clear the DTS0STPU bit. If a DTS transfer is requested at that point, the DTS transfer is resumed.

(Additional information will be provided.)

(3) Aborting or resuming DTS transfer by using DTS transfer enable bit (DTSEN)

This function is performed by the DTSFSL side.

6.11.16 Error response support

(1) Aborting DTS transfer by error response

When DMAT receives an error response from the DMA data bus, the DTS cycle end signal (CYCEND) and the DMA data bus transfer error detection signal (ELBERR) are asserted at the same time. When ELBERR is detected at high level, DTS sets the DTS0STPE bit in the DTS transfer status register (DTS0TSR) to "1" to abort the DTS transfer immediately. Subsequent DTS transfer requests are not accepted. Meanwhile, the DTS transfer error interrupt signal (DTSERR) is asserted for one clock cycle. This abort occurs as soon as an error response is received, regardless of the transfer mode and TI hold ON/OFF status.

The user can learn in which channel the error has occurred, by reading the DTS active channel register, after confirming that DTS0STPE has been set.

Notes

1. If an error response is received on a channel with TI hold ON, TI write back is not performed.
2. It is not possible to determine the cycle (TI fetch cycle, etc.) where the error response occurred.

(2) Canceling transfer abort caused by error response

To cancel a DTS transfer abort caused by an error response, check the DTS active channel register (DTS0ACR), use the DTS0TIT and DTS0ICH bits in the DTS initialization control register (DTS0ICR) to clear the TI for the channel in which the error has occurred, and then set the DTS0ECL bit in the DTS transfer request control register (DTS0TRC) to "1" to clear DTS0STPE to "0".

6.11.17 Stand-by support

Table 6-58 Stand-by timing

Transfer mode	Transfer count	Aborted after:
Single transfer with TI hold OFF	Last transfer	TI write back
	Some remain	
Block transfer with TI hold OFF	Last transfer	TI write back
	Some remain	DTS cycle
Single transfer with TI hold ON	Last transfer	TI write back
	Some remain	DTS cycle
Block transfer with TI hold ON	Last transfer	TI write back
	Some remain	DTS cycle

When the stand-by request signal (STBRQ) is sampled at high level at the rising edge of the system clock (hpclk), it is judged as a stand-by request and bit 0 (DTS0STPU) in DTS0TSR is set to “1” to abort DTS transfers. Next, the stand-by acknowledge signal (STBAK) is asserted and completion of stand-by setup is reported. If this occurs during a DTS cycle, the DTS transfer is aborted according to the condition shown in *Table 6-58 “Stand-by timing”*.

Note that, even during stand-by mode, only the subsequent DTS transfers are not performed, and there are no special operations (such as stopping the clock). To cancel stand-by mode, negate STBRQ.

When the stand-by request signal (STBRQ) is sampled at low level at the rising edge of hpclk, it is judged as a stand-by cancel request, so STBAK is negated and stand-by mode is canceled.

To enable DTS transfers after stand-by is canceled, clear DTS0STPU to “0”.

6.12 DTSFSL Function

The DTS factor selector (DTSFSL) function selects DTS trigger factors from among interrupt signals. Four priority levels can be set for DTS trigger factors. Factors that are not used with DTS can be input to the interrupt controller via DTS transfer-factor-through outputs (IRQ[127:0]).

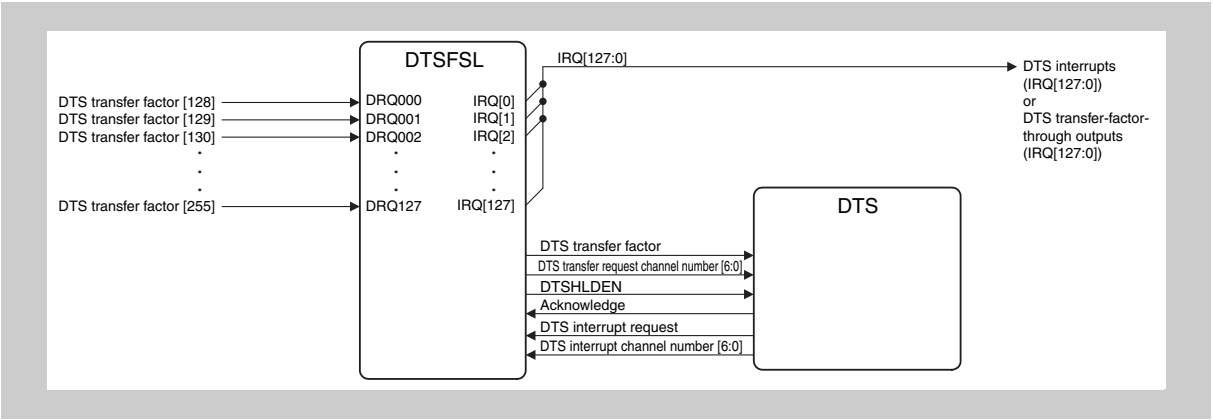


Figure 6-29 Connections between DTSFSL and DTS

6.12.1 Features

A maximum of up to 128 factors can be assigned as DTS transfer factors.

Any of four priority levels can be set for each of these factors.

Based on an interrupt request source from the DTS, an interrupt is output (shared output) to the CPU via a DTS interrupt (IRQ[127:0]) corresponding to the particular channel.

S/W DTS request A DTS request can also be set by S/W by setting DTSENn.DRQSET = 1.

Note The types of interrupt request sources from the DTS are “all transfer counts ended”, “flag check result is false”, and “compare result is false”.

Factors that are not used with DTS are input to the interrupt controller via DTS transfer-factor-through outputs (IRQ[127:0]) (shared output).

6.13 DTSFSL Control Registers

6.13.1 DTSENn – DTS transfer enable register

Access This register can be read or written in 8-bit units.

Address FFFF 7D00$_H$ + 2 x n

Initial Value 00$_H$

7	6	5	4	3	2	1	0
DRQ	DRQSET	DRQCLR	0	0	DTSEN	DTSPR1	DTSPR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-59 DTSFSL register contents

Bit position	Bit name	Function
7	DRQ	This bit indicates whether a transfer request to the DTS is being held pending. This is a read-only bit, and writing to this bit is ignored. 0: No DTS transfer requests 1: DTS transfer requests exists (wait status) Condition for setting (to “1”): When the rising edge of DRQn is detected Condition for clearing (to “0”): When “1” is written to the DRQCLR bit, or when the DTS accepts a transfer request
6	DRQSET	Writing “1” to this bit sets the DRQ bit to “1”. Writing “0” to this bit is ignored.
5	DRQCLR	Writing “1” to this bit clears the DRQ bit to “0”. Writing “0” to this bit is ignored.
2	DTSEN	DTS transfer request enable bit for DRQn 0: Does not use DRQn as DTS transfer request (DTS request disabled) 1: Uses DRQn as DTS transfer request (DTS request enabled)
1 0	DTSPR1 DTSPR0	These bits specify the priority order. “00” specifies the highest priority level and “11” specifies the lowest priority level.

6.13.2 DTSHENm – DTS hold enable register (m = 0 to 3)

Access This register can be read or written in 16-bit units.

Address DTSHEN0: FFFF7E00$_H$, DTSHEN1: FFFF7E02$_H$, DTSHEN2: FFFF7E04$_H$, DTSHEN3: FFFF7E06$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
HEN	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	dtsch6-0						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 6-60 DTSHENm register contents

Bit position	Bit name	Function
15	HEN	This bit enables/disables operation of the DTS hold enable register. 1: Enables operation of hold enable register. 0: Disables operation of hold enable register.
6 to 0	dtsch6 to dtsch0	These bits select the target channel for hold. Settings to these bits are valid only when HEN is “1”. The set values are listed in the table below.

Table 6-61 List of channels subject to hold selected by setting dtsch6 to dtsch0

dtsch6 to dtsch0	Hold channel	dtsch6 to dtsch0	Hold channel	dtsch6 to dtsch0	Hold channel	dtsch6 to dtsch0	Hold channel
000_0000	CH000	010_0000	CH032	100_0000	CH064	110_0000	CH096
000_0001	CH001	010_0001	CH033	100_0001	CH065	110_0001	CH097
000_0010	CH002	010_0010	CH034	100_0010	CH066	110_0010	CH098
000_0011	CH003	010_0011	CH035	100_0011	CH067	110_0011	CH099
000_0100	CH004	010_0100	CH036	100_0100	CH068	110_0100	CH100
000_0101	CH005	010_0101	CH037	100_0101	CH069	110_0101	CH101
000_0110	CH006	010_0110	CH038	100_0110	CH070	110_0110	CH102
000_0111	CH007	010_0111	CH039	100_0111	CH071	110_0111	CH103
000_1000	CH008	010_1000	CH040	100_1000	CH072	110_1000	CH104
000_1001	CH009	010_1001	CH041	100_1001	CH073	110_1001	CH105
000_1010	CH010	010_1010	CH042	100_1010	CH074	110_1010	CH106
000_1011	CH011	010_1011	CH043	100_1011	CH075	110_1011	CH107
000_1100	CH012	010_1100	CH044	100_1100	CH076	110_1100	CH108
000_1101	CH013	010_1101	CH045	100_1101	CH077	110_1101	CH109
000_1110	CH014	010_1110	CH046	100_1110	CH078	110_1110	CH110
000_1111	CH015	010_1111	CH047	100_1111	CH079	110_1111	CH111
001_0000	CH016	011_0000	CH048	101_0000	CH080	111_0000	CH112
001_0001	CH017	011_0001	CH049	101_0001	CH081	111_0001	CH113
001_0010	CH018	011_0010	CH050	101_0010	CH082	111_0010	CH114
001_0011	CH019	011_0011	CH051	101_0011	CH083	111_0011	CH115
001_0100	CH020	011_0100	CH052	101_0100	CH084	111_0100	CH116
001_0101	CH021	011_0101	CH053	101_0101	CH085	111_0101	CH117
001_0110	CH022	011_0110	CH054	101_0110	CH086	111_0110	CH118
001_0111	CH023	011_0111	CH055	101_0111	CH087	111_0111	CH119
001_1000	CH024	011_1000	CH056	101_1000	CH088	111_1000	CH120
001_1001	CH025	011_1001	CH057	101_1001	CH089	111_1001	CH121
001_1010	CH026	011_1010	CH058	101_1010	CH090	111_1010	CH122
001_1011	CH027	011_1011	CH059	101_1011	CH091	111_1011	CH123
001_1100	CH028	011_1100	CH060	101_1100	CH092	111_1100	CH124
001_1101	CH029	011_1101	CH061	101_1101	CH093	111_1101	CH125
001_1110	CH030	011_1110	CH062	101_1110	CH094	111_1110	CH126
001_1111	CH031	011_1111	CH063	101_1111	CH095	111_1111	CH127

6.14 DTSFSL Function Details

6.14.1 Interrupt output function

DTSFSL includes a function that outputs interrupts to the CPU. These interrupts to the CPU are output from IRQ000 to IRQ127. The interrupt sources in this case are DTSINTRQ from DTS.

When an interrupt is requested by the DTS interrupt request (DTSINTRQ), it is assigned to one of IRQ0 to IRQ127 according to the DTS interrupt channel number (DTSINTCH[6:0]).

When the DTS interrupt channel number (DTSINTCH[6:0]) is 00H, the interrupt request is assigned to IRQ0, and when the number is 7FH, it is assigned to IRQ127.

Note Requests for an interrupt by the DTS interrupt request (DTSINTRQ) or by the DTS interrupt channel number (DTSINTCH[6:0]) are automatically processed by the DTS, so no external settings are needed.

For example, when DTSEN in DTSEN002 is set to “1” to generate a DTS request from DRQ002, the DTS performs a series of processing, and at the last transfer, the DTS automatically sets the DTS interrupt request (DTSINTRQ) to “1” and the DTS interrupt channel number (DTSINTCH) to 02H so that the request is transmitted to DTSFSL. This is decoded within DTSFSL, and an interrupt request is output to the CPU from IRQ002.

Interrupt requests from the DTS cannot be masked by DTSFSL, so mask processing should be performed on the DTS side.

Chapter 7 Flash Memory

The following V850E2/Fx4-H devices are equipped with internal flash memory as follows:

Series name	Product name	Code flash	Data flash
FK4-H-2M	μPD70F3561	2 MB	64 KB
FL4-H-2M	μPD70F3564		

The code flash memory is attached to the dedicated instruction fetch bus of the V850E2 CPU core. It is used for non-volatile storage of program code and constant data.

The data flash memory is accessible via the memory interface bus. It holds nonvolatile user's data, which are subject to be altered during normal program operation.

The flash memory can be written in different ways:

- mounted on the target board by connecting a dedicated flash programmer to the target system (Serial-Programming)
- by the microcontroller's application software (Self-Programming)

Additionally the flash memory is equipped with a configuration area to hold various configuration settings. Via these options start-up configurations can be set for e.g. the Watchdog Timer. The flash configuration options can be written by use of an external flash programmer and in Self-Programming mode. They are not accessible via the normal CPU address space.

7.1 Code Flash Memory Overview

7.1.1 Code flash memory features

- All-blocks or multiple blocks batch erase or single block erase
- Erase/write with single power supply
- Two programming modes:
  - Serial-Programming with flash programmer using dedicated serial interfaces
  - Flash memory programming by Self-Programming

7.1.2 Code flash memory map

The microcontroller's internal code flash memory area is divided into blocks of 4 KB blocks and can be programmed/erased in block units.

Following tables list the block structures and address assignments for all V850E2/Fx4-H devices with code flash memory.

Table 7-1 Code flash memory configuration for V850E2/Fx4-H devices

Block 511 (4 KB)	**001F FFFF$_H$** **001F 0FFF$_H$**	**Address**
...	**...**	
Block 1 (4 KB)	**0000 1FFF$_H$** **0000 1000$_H$**	
Block 0 (4 KB)	**0000 0FFF$_H$** **0000 0000$_H$**	
2 MB	**Code flash size**	
8/16/32/64/128/256 KB	**Boot swap cluster sizes**	
• FK4-H-2M: µPD70F3561 • FL4-H-2M: µPD70F3564	**Products**	

7.1.3 Data flash memory map

The data flash memory is organized in blocks of 2 KB size.

Following tables list the block structures and address assignments for all V850E2/Fx4-H devices with data flash memory.

Table 7-2 V850E2/Fx4-H data flash memory

Product	Data flash size	Number of 2 KB blocks	Address range
FK4-H-2M: µPD70F3561	64 KB	32	0200 0000$_H$ - 0200 FFFF$_H$
FL4-H-2M: µPD70F3564			

7.2 Code Flash Memory functional Outline

Serial-Programming The internal flash memory of the microcontroller can be written by using the erase and write functions of a flash programmer, also while the microcontroller has already been mounted on the target system.

Self-Programming The Self-Programming facility allows rewriting of the flash memory by the user program. It is ideal for program updates after production and shipment, since no additional programming equipment is required. During Self-Programming some software services as well as interrupt serving can still be in operation, e.g to sustain communication with other devices.

While the Self-Programming mode can be initiated from the normal operation mode the external flash programmer mode is entered immediately after release of a system reset.
Refer to *"Flash memory programming control"* in the section *"Flash Programming with Flash Programmer"* in this chapter for details on how to enter normal operation or Serial-Programming mode.

Configuration area The flash memory contains a configuration area, used to store the settings of security and protection functions, initial settings for some modules.

Boot swap A boot swap function makes safe re-programming of the flash memory possible and is used to maintain an operable software version, even if re-programming fails for any reason, e.g. in a power fail situation.
For further information concerning boot swapping refer to *"Secure Self-Programming (boot cluster swapping)"* below in this chapter.

Protection A set of protection flags can be specified during flash memory programming to prohibit access to the flash memory in different ways, such as read-out, write and erase protections for the external programmer interface. By these means the flash memory can be protected against read-out and rewrite of the flash memory content by unauthorized persons.
For further information concerning data protection refer to the chapter *"Code Protection and Security"*.

Table 7-3 Flash memory write methods

Environment	Interface	Outline	Operation Mode
Serial-Programming	Dedicated serial interface	Flash memory programming is done by an external flash programmer. The device is mounted on the target system. The communication between the device and the flash progammer is using a dedicated serial interface. For details refer to the section *"Flash Programming with Flash Programmer"* in this chapter.	Flash memory programming mode
Self-Programming	Self-Programming library	Flash memory can be rewritten by executing a user program that has been written to the flash memory in advance by means of Serial-Programming. The Self-Programming library provides all necessary functions to be called by the application software. For details refer to the section *"Code Flash Self-Programming"* in this chapter.	Normal operation mode

Table 7-4 "Basic functions for flash memory modifications" on page 450 summarizes the functions used to modify flash memory content.

Table 7-4 Basic functions for flash memory modifications

Function	Functional outline	Support (√: supported, ×: not supported)	
		Serial-Programming	Self-Programming
Block erasure	The contents of specified memory blocks are erased.	√	√
Chip erasure[a]	The contents of the entire flash memory area is erased all at once. The configuration area - except the boot cluster protection flag - is also erased. **Caution** The chip erase function erases also the data flash memory.	√	x
Write	Writing to specified addresses, and a verify check to see if write level is secured are performed.	√	√
Erase/write	A 32 KB block is erased while another 32 KB block is written.	√	√
Verify	Data read from the flash memory is compared with data transferred from the flash programmer.	√	x [b]
Checksum	Microcontroller internally calculated CRC checksum over the entire flash memory content is compared with the checksum calculated by the serial programmer	√	x
Blank check	The erasure status of the entire memory is checked.	√	√
Protection settings	Following functions can be prohibited: • chip erase • block erase • write • read • rewriting of the boot cluster • flash shield	√	√ [c]

a) Chip erasure is not possible if the boot block protection is enabled (by setting the boot block cluster protection flag) or chip erasure is disabled in general (by setting the chip erase protection flag).

b) Can be carried out by the user's program.

c) Except protection against rewriting of the boot cluster all other protections have no effect in Self-Programming mode.
Protection settings can be activated in Self-Programming mode. Already activate protection settings can not be deactivated.

The following table lists the available flash memory protection functions.

For details refer to the chapter *"Code Protection and Security"*.

Table 7-5 Protection functions

Function	Functional outline	Applicable (√: applies, ×: doesn't apply)	
		Serial-Programming	Self-Programming
Chip erase command prohibit	Erasure of the entire flash (including the configuration area[a] and the data flash) or single blocks impossible.	√	×
Block erase command prohibit	Erasure of single blocks impossible.	√	×
Program command prohibit	Erasure and rewrite of single blocks impossible.	√	×
Read command prohibit	Read-out of any flash content impossible.	√	×
Rewriting boot area prohibit	Erasure (by block or chip erase) or writing of the boot cluster impossible.	√	√
Flash shield	Write/erase protection outside a defined window.	√	√

a) The boot cluster protection flag is not erased.

7.2.1 Code flash memory erasure and rewrite

Erasure According to its block structure the flash memory can be erased in two different modes.

- All-blocks batch erasure (chip erase, only in Serial -Programming mode)
  All blocks are erased all together.
- Block erasure

  Each 4 KB flash memory block can be erased separately.
  In Self-Programming mode any number of contiguous flash memory blocks can be erased all together.

Rewrite In Self- and Serial -Programming mode it is possible to rewrite the flash memory in smaller units than one block. Once a complete block has been erased it can be rewritten in units of 16 byte. Each unit can be rewritten only once after erasure of the complete block.

Erase/write Erase/write mode allows to erase a 32 KB block of flash memory , while another 32 KB block is written.

7.3 Data Flash Memory

The V850E2/Fx4-H Series products contain data flash in addition to the code flash.

7.3.1 Data flash memory features

The data flash has the following features:

- data flash memory in 2 KB blocks
- write access in 32-bit steps
- erase in 2 KB blocks
- write, erase operations to the data flash while application code can be executed from code flash

7.3.2 Data flash reading and writing

The data flash can be read and written by using an external flash programming tool or the data flash access library.

Programming during normal operation is achieved by using the data flash access software library. For details refer to the User's Manual "Data Flash Access Library FDL - T05" (U20279EExVxUM00).

Note The chip erase command of an external programmer erases also the data flash.

7.4 Flash Programming with Flash Programmer

A dedicated flash programmer can be used for external writing of the flash memory in Serial-Programming mode.

Serial-Programming

During Serial-Programming the target microcontroller remains mounted on its board. The board is equipped with a connector, that connects the flash programmer to the target microcontroller.

The microcontroller must be fully functional and operated in Serial-Programming mode, in particular

- All external power supplies must be active.
- The external resonator must be connected to the X1/X2 pins.

All other necessary microcontroller configurations are conducted by the on-chip firmware, that is processed in Serial-Programming mode.

Caution

Connecting the flash programmer to the on-board microcontroller may yield conflicts with other sginals. Pay attention to the hints given in section *"Potential conflicts with on-board signal connections" on page 458*.

7.4.1 Programming environment

The recommended environment to write data to the flash memory of the microcontroller is shown below.

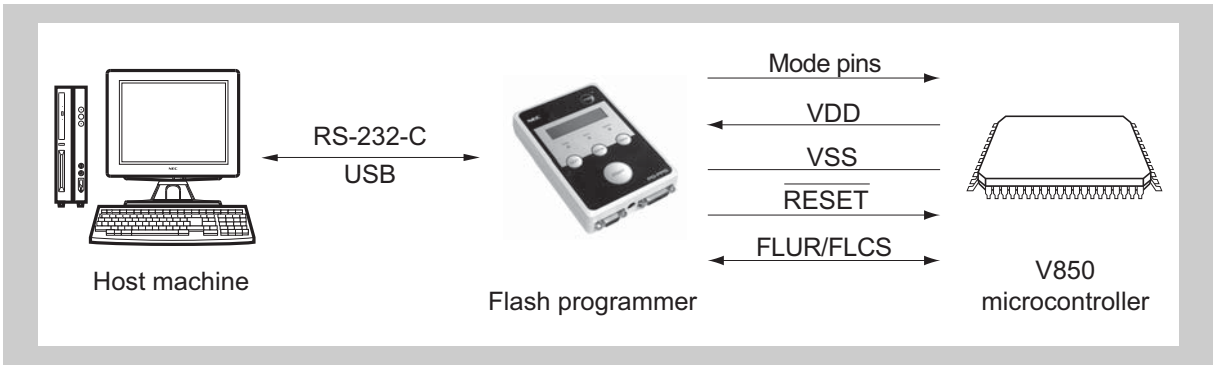


Figure 7-1 Environment to write program to flash memory

A host machine is required for configure the flash programmer.
The used flash programmer may also feature a stand-alone mode, so a host machine may not be necessary.

Following dedicated microcontroller serial interfaces can be used as the interface between the flash programmer and the microcontroller:

- single-wire asynchronous serial interface FLUR0
- clocked serial interface FLCS0

Mode pins

The mode pins are used to set the microcontroller into flash programming mode.
Refer to the section *"Operation modes"* in the chapter *"CPU System Functions"* for further information.

Note In normal operation mode, i.e. not in flash programming mode, the serial interfaces FLUR0 and FLCS0 are not available. The used ports in flash programming mode are specified in the following sections.
These are automatically configured for communicating with the flash programmer in flash programming mode.

7.4.2 Communication modes

(1) Asynchronous flash programming interface FLUR0

The single-wire asynchronous Serial-Programming interface FLUR0 uses following port for connecting to the flash programmer:

- JP0_0: reception/transmission data FLUR0RTX

The external flash programmer offers various choices of available baudrates.

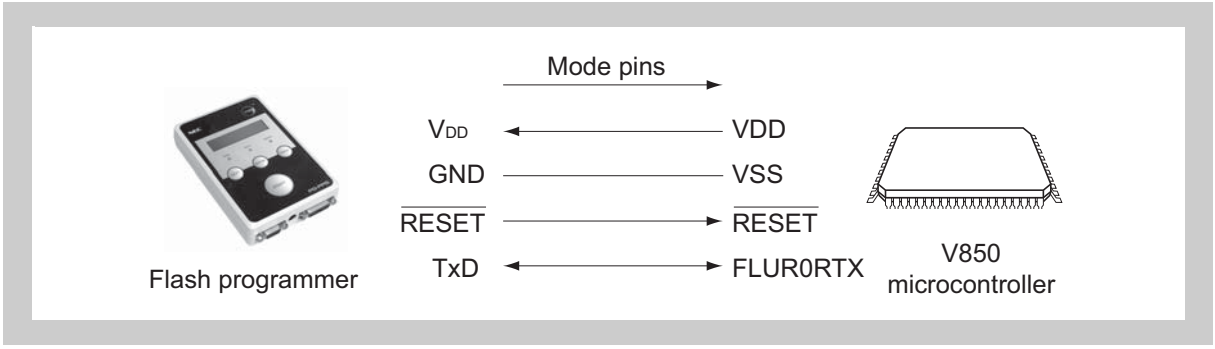


Figure 7-2 Communication with flash programmer via FLUR0

(2) Synchronous flash programming interface FLCS0

The synchronous Serial-Programming interface FLCS0 uses following ports for connecting to the flash programmer:

- JP0_0: serial data input FLCS0SI
- JP0_1: serial data output FLCS0SO
- JP0_2: serial data clock input FLCS0SCI

The external flash programmer offers various choices of available clock rates.

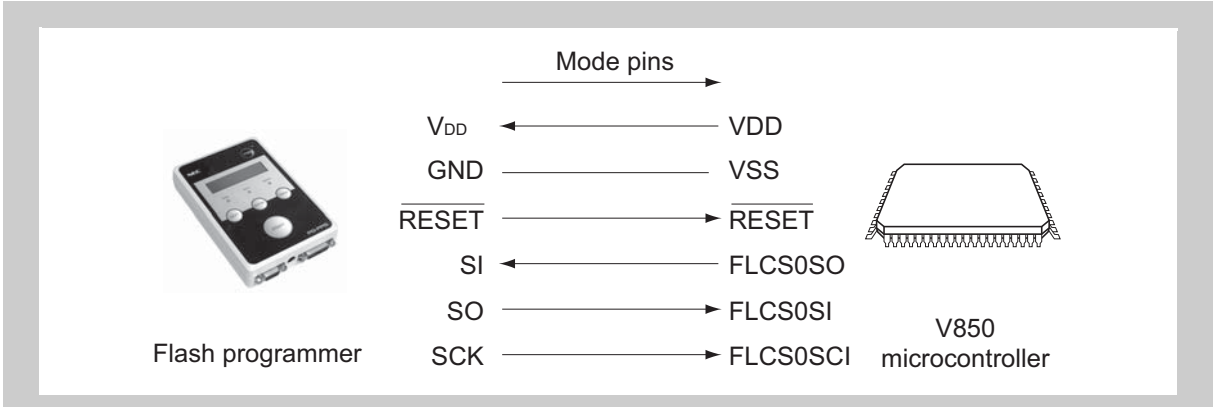


Figure 7-3 Communication with flash programmer via FLCS0

The flash programmer outputs the serial data clock SCK and the microcontroller operates as a slave.

7.4.3 Pin connection with flash programmer PG-FP5

A connector must be mounted on the target system to connect the flash programmer for Serial-Programming. In addition, functions to switch between the normal operation mode and flash memory programming mode and to control the microcontroller's $\overline{\text{RESET}}$ pin must be provided on the board.

Mode pins

The mode pins are used to set the microcontroller into flash programming mode.
Refer to the section *"Operation modes"* in the chapter *"CPU System Functions"* for further information.

When the microcontroller flash memory programming mode is set, all the pins not used for flash memory programming are in the same status as immediately after reset.

If the PG-FP5 is used as the flash programmer, connect the PG-FP5 target interface connector to the microcontroller as follows:

Table 7-6 Connection of microcontroller flash programmer PG-FP5

Flash programmer FG-FP5 connection pin			Microcontroller signal (port) name			
Signal name	I/O	Function	FLUR0		FLCS0	
			Signal	Port	Signal	Port
SO/TxD	O	• FLUR0: receive/transmit data • FLCS0: transmit data	FLUR0RTX	JP0_0	FLCS0SI	JP0_0
SI/RxD	I	Receive data	leave open		FLCS0SO	JP0_1
SCK	O	Transfer clock	leave open		FLCS0SCI	JP0_2
CLK	O	Clock to microcontroller	leave open		leave open	
			leave open		leave open	
$\overline{\text{RESET}}$	O	Reset signal	$\overline{\text{RESET}}$	–	$\overline{\text{RESET}}$	–
FLMD0	I	Mode selection	FLMD0	–	FLMD0	–
FLMD1	I	Mode selection	FLMD1[a]	P0_1	FLMD1[a]	P0_1
H/S	I	Handshake signal	leave open		leave open	
V_{DD}	I	Microcontroller supply voltage monitoring	Power supply of JP0 port group buffers[b]		Power supply of JP0 port group buffers[b]	
V_{DD2}	–	Supply voltage	leave open		leave open	
V_{PP}	–	Flash programming voltage	leave open		leave open	
GND	–	Ground	VSS		VSS	
VDE	–	Reserved	leave open		leave open	
RFU-1	–	Reserved	leave open		leave open	

a) If FLMD1 is fixed to low level on the target board, the programmer's FLMD1 signal can be left unconnected.
b) Refer to the chapter *"Power Supply"* to obtain the correct microcontroller's power supply pins for the JP0 port group.

Refer also to the section *"Operation Modes"* in chapter *"CPU System Functions"* for more details about flash programming mode setting.

For details concerning the PF-FP5 programmer, refer to the PG-FP5 User's Manual, document number R20UT0008EJxxxx (xxxx denotes the current version number).

7.4.4 Flash memory programming control

The procedure to program the flash memory is illustrated below.

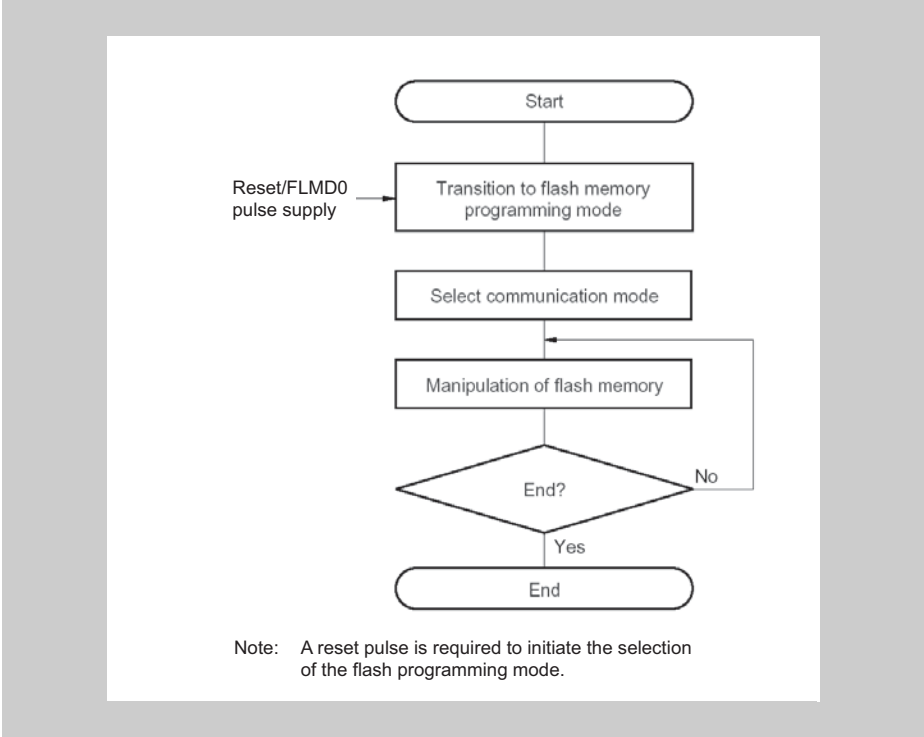


Figure 7-4 Flash memory programming procedure

(1) Operation mode control

To rewrite the contents of the flash memory by using the flash programmer, set the microcontroller in the flash memory programming mode.

To enter the Serial-Programming mode, i.e. on-board programming by an external flash programmer, the FLMD0 pin has to be supplied with VDD and FLMD1 with VSS level at $\overline{\text{RESET}}$ release.

In the normal operation mode, VSS is input to the FLMD0 pin. A pull-down resistor at the FLMD0 pin ensures normal operation mode if no flash programmer is connected.

Note Refer to the sections *"Operation modes"* and " Mode pins and JP0 connections" in the *"CPU System Functions"* chapter for details about operation modes settings.

An example of connection of the FLMD0 and FLMD1 pins is shown below. FLMD1 can be connected to ground via a resistor. Alternatively the FLMD1 pin may also be connected directly to the FLMD1 signal of the flash programmer.

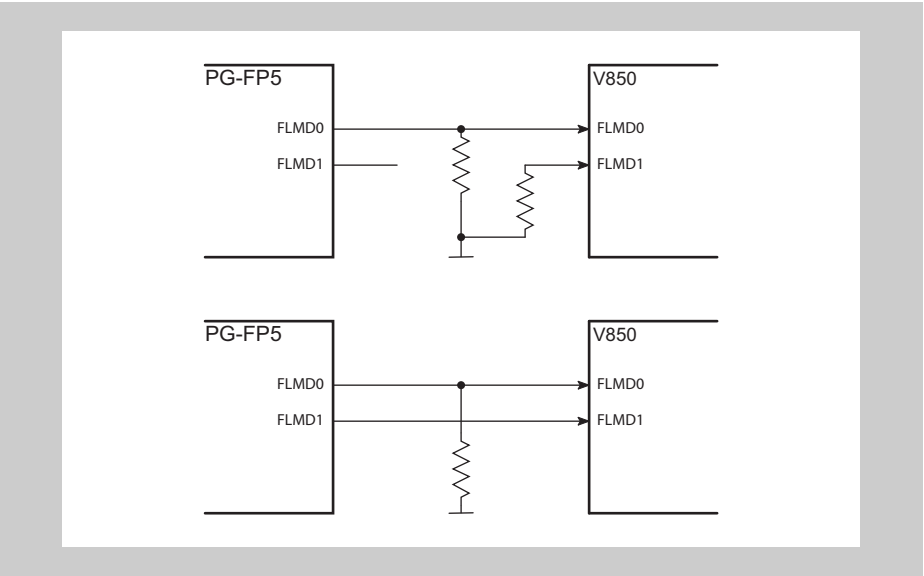


Figure 7-5 Example of connection to flash programmer PG-FP5

Once started in normal operation mode (FLMD0 = 0), FLMD0 pin is used for enabling Self-Programming. Refer also to *7.5 "Code Flash Self-Programming" on page 463* .

(2) Potential conflicts with on-board signal connections

Serial I/O signals

If other devices are connected to the serial interface pins in use for flash memory programming in Serial-Programming mode take care that the concerned signals do not conflict with the signals of the flash programmer and the microcontroller. Output pins of the other devices must be isolated or set in high impedance state. Ensure that the other devices do not malfunction because of flash programmer signals.

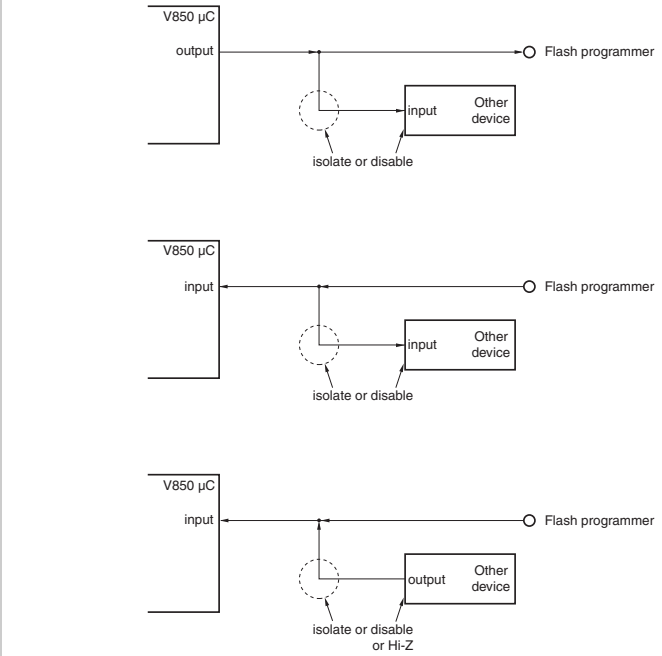


Figure 7-6 Potential conflicts with serial interfaces signals

$\overline{\text{RESET}}$

Pay attention in particular if the flash programmer's $\overline{\text{RESET}}$ signal is connected also to an on-board reset generation circuit. The reset output of the reset generator may ruin the flash programming process and may need to be isolated or disabled.

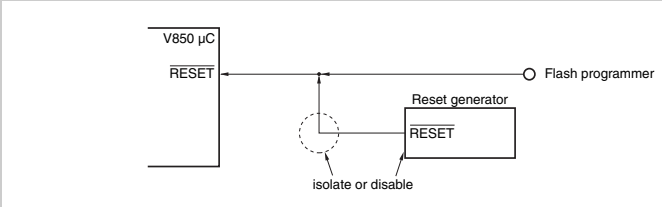


Figure 7-7 Potential conflict with $\overline{\text{RESET}}$

Ports The V850 port pins adopts following status during Serial-Programming:

Ports used for programming are configured as FLUR0 respectively FLCS0 pins.

All other pins remain in their default state after reset release.

In case the default state after reset of the pins not used for programming is in port or high -impedance output port, pay attention to other devices connected to these pins. If these devices require defined levels at the pins, the ports may have to be connected to VDD or VSS via a resistors.

Oscillators Connect all oscillator pins in the same way as in the normal operation mode.

$\overline{\text{DCUTRST}}$ During flash memory programming, input a low level to $\overline{\text{DCUTRST}}$ (port JP0_4) or leave it open. Do not input a high level.

Power supply Supply the same power to all power supply pins, including reference voltages, power regulator pins, etc., as in the normal operation mode.

(3) Selection of the communication mode

The communication interface is chosen by applying a specified number of pulses to the FLMD0 pin after reset release. Note that this is handled by the flash programmer.

Figure 7-8 “Selection of communication mode” on page 460 gives an example how the FLCS0 is established for the communication between the flash programmer and the microcontroller.

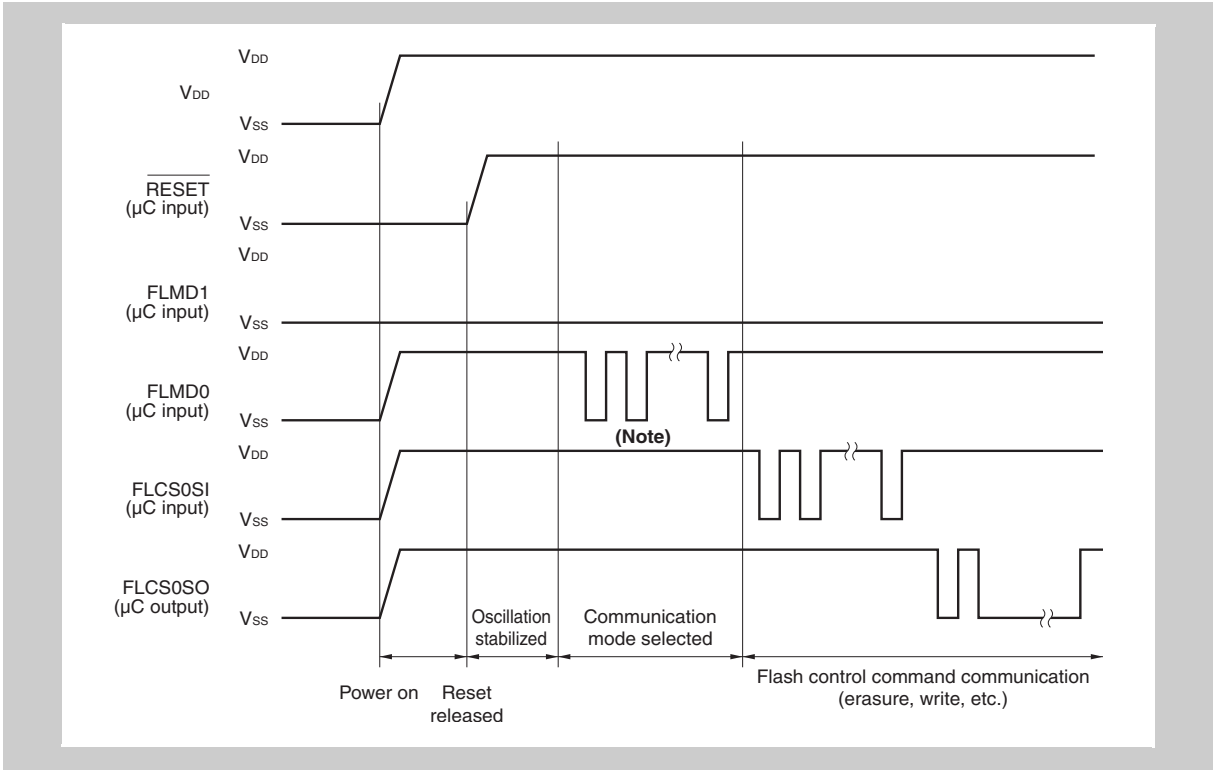


Figure 7-8 Selection of communication mode

Note The number of clocks to be inserted differs depending on the chosen communication mode. For details, refer to *Table 7-7 “FLMD0 pulses for communication mode setting” on page 460* .

Table 7-7 FLMD0 pulses for communication mode setting

FLMD0 pulses	Communication mode	Remarks
0	FLUR0	Communication rate: 9600 bps (after reset), LSB first
8	FLCS0	Microcontroller performs slave operation, MSB first
Other	–	Setting prohibited

When FLUR0 has been selected after reception of the FLMD0 pulses with 9600 bps, the flash programmer changes the baudrate according to the user’s choice via the flash programmer’s user interface.

(4) Communication commands

The flash programmer sends commands to the microcontroller. Depending on the commands, the microcontroller returns status information or the requested data.

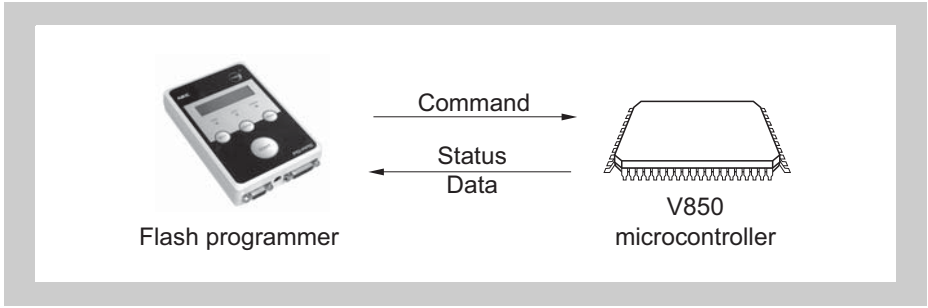


Figure 7-9 Communication commands exchange

The following table lists the flash memory control commands of the microcontroller. All these commands are issued by the flash programmer, and the microcontroller performs the corresponding processing.

Table 7-8 Flash memory control commands (1/2)

Classification	Command name	Support		Function
		FLCS0	FLUR0	
Blank check	Block blank check	√	√	Checks erasure status of entire memory.
Erase	Chip erase	√	√	Erases all flash memories contents, including code flash, data flash and configuration area.
	Block erase	√	√	Erases the memory contents of specified blocks. Note that the configuration area remains untouched.
Write	Write	√	√	Writes data by specifying write address and number of bytes to be written, and executes verify check.
Erase/write	Erase and write	√	√	Erases a specified number of flash blocks of the code flash or the data flash. Note that the configuration area remains untouched.
	Chip erase and write	√	√	Erases and write the entire flash memory, including code flash, data flash and configuration area.
Read	Read	√	√	Reads data by specifying write address and number of bytes to be read.
Verify	Verify	√	√	Compares input data with all memory contents.
ID	Set ID code	√	√	Set the On-chip Debug ID to the registers OCDIDL, OCDIDM, OCDIDH
	Get ID code	√	√	Reads the On-chip Debug ID from the registers OCDIDL, OCDIDM, OCDIDH

Table 7-8 Flash memory control commands (2/2)

Classification	Command name	Support		Function
		FLCS0	FLUR0	
CRC check	CRC check	√	√	Calculates a checksum over a specified number of flash blocks of the code flash or the data flash. Note that this checksum does not cover the configuration area.
	Chip CRC check	√	√	Calculates a checksum over the entire flash memory, including code flash, data flash and configuration area.
Flash mask option	Set flash mask option	√	√	Set the flash mask options to register OPBT0
	Get flash mask option	√	√	Reads the flash mask options from register OPBT0
Protection	Protection setting	√	√	Sets protection against chip erasure, block erasure, and writing.
	Get protection settings	√	√	Reads the protection settings.
System setting and control	Reset	√	√	Escapes from each status.
	Oscillation frequency setting	√	√	Sets oscillation frequency.
	Baudrate setting	–	√	Sets baudrate when UART is used.
	Silicon signature	√	√	Reads silicon signature information.

7.5 Code Flash Self-Programming

This V850 microcontroller supports a flash macro service that allows the user program to rewrite the internal flash memory by itself.

By using this flash macro service and a Self-Programming library (SPL), provided by Renesas, the user's program is able to rewrite the flash memory with data, transferred in advance to the internal RAM or the external memory.

Thus the user program can be upgraded and constant data can be rewritten in the field.

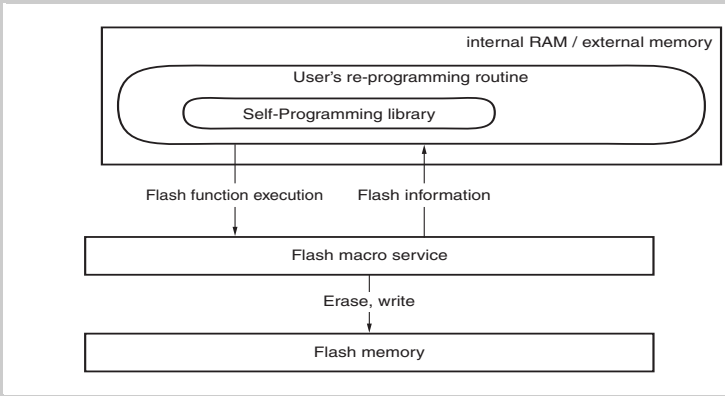


Figure 7-10 Concept of Self-Programming

During Self-Programming access to the flash memory is not possible. Thus program execution is only possible by instruction fetching from internal RAM or external memory.

Consequently the instructions of user re-programming software routines, which shall remain in operation during the Self-Programming procedure, must be copied from the flash memory to the internal RAM or external memory prior to activating the Self-Programming. Since interrupt processing by using the interrupt vectors in the flash memory is also impossible during Self-Programming, interrupt acknowledges need to be re-routed to interrupt vectors in the internal RAM (refer to *"Interrupt handling during flash Self-Programming" on page 470*).

It is recommended to refer to the User's Manual "Flash Self-Programming Library FSL - T05" for comprehensive information concerning flash Self-Programming. This document explains also the functions of the Self-Programming library.

7.5.1 Self-Programming enable

The Self-Programming function can be started out of the normal user mode of the microcontroller.

Self-Programming must be in particular enabled in order to avoid unintended re-programming of the flash. Two ways to enable Self-Programming are provided:

- by setting the external FLMD0 pin to high level
  This requires some external components or wiring, e.g. connecting an output port to FLMD0.
- by setting the internal register bit FLMDCNT.FLMDPUP
  This way does not need any special external components or wiring.

The following register is used to enable Self-Programming internally by software.

Note The FLMD0 pin must be connected to ground via a 82 kΩ resistor. Refer to the section *"Mode pins and JP0 connections"* in the *"CPU System Functions"* chapter and the Electrical Target Specification for further details.

(1) FLMDCNT – FLMD control register

This register controls an internal pull-up respectively pull-down resistor, connected to the FLMD0 pin, and thus enables respectively disables Self-Programming.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register FLMDPCMD.
Refer to the section *"Write protected Registers"* in chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 8-bit units.

Address FF43 8000$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FLMDPUP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 7-9 FLMDCNT register contents

Bit position	Bit name	Function
0	FLMDPUP	FLMD0 pull-up/pull-down control 0: pull-down resistor active at FLMD0 (Self-Programming mode disabled) 1: pull-up resistor active at FLMD0 (Self-Programming mode enabled)

7.5.2 Self-Programming library functions

Code flash memory Self-Programming by the user's program is supported by the Self-Programming library.

This library provides a set of C function calls to carry out basic functions like

- blank-check/erase/rewrite/verify of the flash
- boot cluster swapping, including definition of boot clusters
- setting of protection flags
- obtain various information concerning the code flash memory

Detailed information how to use the library functions is given in the User's Manual "Flash Self-Programming Library FSL - T05".

7.5.3 Self-Programming internal RAM occupancy

During Self-Programming the upper 4 KB of the internal RAM (address FEDF $F000_H$ - FEDF $FFFF_H$) are occupied by the Self-Programming Library. Thus the content of this 4 KB RAM is altered during Self-Programming and may have to be recovered by the user's program.

Note Additional RAM may be necessary for intermediate storage of user data and code to be copied from flash memory to RAM during Self-Programming.

7.5.4 Secure Self-Programming (boot cluster swapping)

The V850 flash microcontrollers support a mechanism to swap a cluster of code flash memory blocks, starting from address 0000 0000_H, with another cluster of the same size, located immediately above the first one.

Boot swap cluster

A group of boot blocks to be swapped. The cluster of blocks starting at address 0000 0000_H is named active boot swap cluster, since it contains the entry point of the user's program at the default reset vector 0000 0000_H.

Boot swap flag

Which of the two clusters is the active boot cluster is controlled by the boot swap flag, that can be defined during flash programming via the Self-Programming library.
The boot swap flag is stored in the flash memory configuration area.

Figure 7-11 "Boot swap cluster swapping function" on page 466 shows an example of the boot block swapping function with a cluster size of 4 flash memory blocks. After inverting the boot_flag - it becomes not(boot_flag) - blocks 4 to 7 become the active boot cluster. Thus after next reset release the user's program starts from the new boot swap cluster.

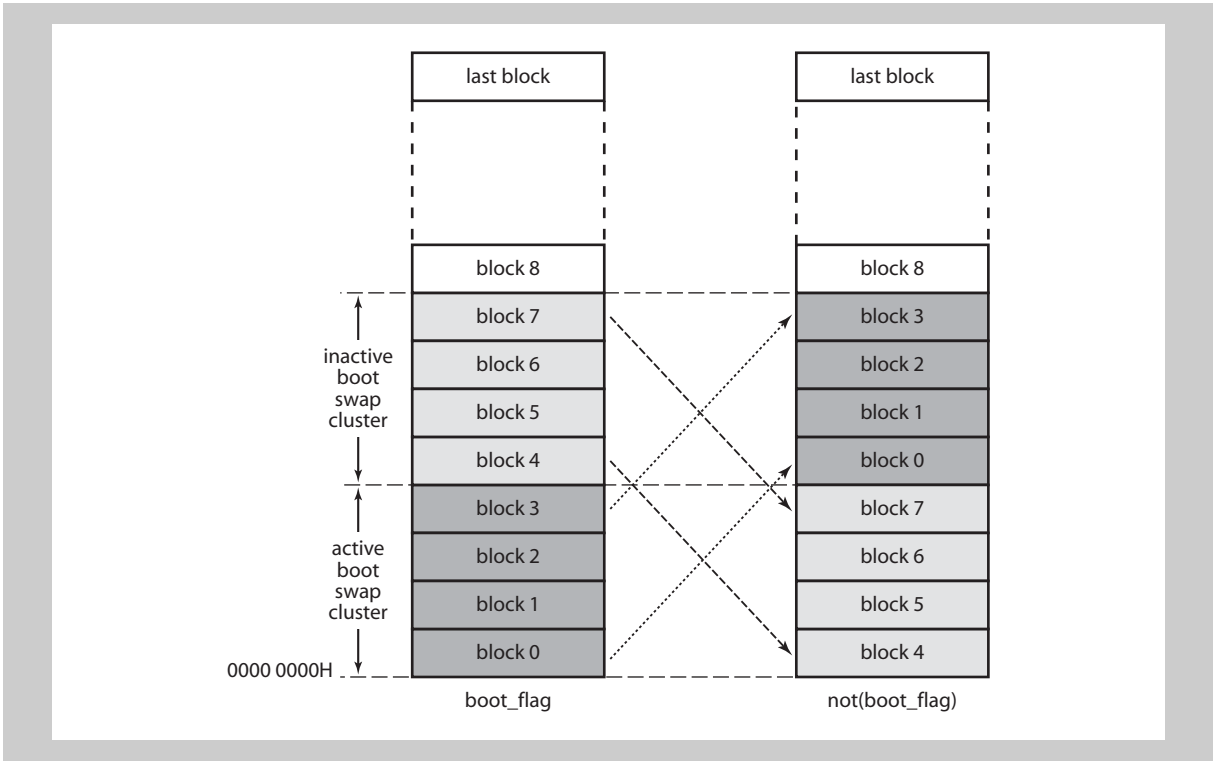


Figure 7-11 Boot swap cluster swapping function

Secure Self-Programming

The boot cluster swapping function enables secure Self-Programming. In case the boot code shall be rewritten, the new code can be written to the inactive boot cluster, while the boot_flag remains in its previous state.
If rewriting of the boot cluster has been completed successfully, the boot_flag can be inverted, making the new boot code active.
If rewriting of the new boot code fails for any reason, e.g. power fail or unintended reset, the old boot code still remains active and rewriting can be started again.

Boot cluster

The boot code size itself may be smaller than the boot swap cluster size.

The number of flash memory blocks, which are part of the boot code, are named boot cluster. The number of boot blocks, which are member of the cluster, can be defined during Self-Programming via the Self-Programming library.

The boot cluster size determines the boot swap cluster size. This is automatically evaluated from the number of boot blocks, defined during Self-Programming.

Table 7-10 “Relation between boot block and boot swap cluster” on page 468 shows the relation between the number of boot blocks, the boot cluster size and the boot swap cluster.

Number of boot blocks

The number of boot blocks has to be defined by the user during self-programming. It determines the blocks, which are subject to the boot cluster protection, that allows to protect the boot blocks from any erase or write process.

Boot block protection

To prohibit rewriting of the boot blocks, the boot cluster protection flag can be set during flash memory programming. When this flag is set, the blocks of the active boot cluster can neither be erased nor written. Boot cluster swapping is impossible as well.
Note that only the blocks of the active boot cluster are protected. In the example according to *Figure 7-12 “Boot cluster swapping function” on page 469* , for instance, blocks 0 and 1 would be prohibited, while blocks 2 and 3 could still be erased and written.

Caution

Once the boot cluster protection has been activated, it can never be deactivated again.

For further information concerning flash memory protection flags refer to the chapter *“Code Protection and Security”*.

Table 7-10 Relation between boot block and boot swap cluster

Number of boot blocks	Boot cluster size	Boot swap	Boot cluster protection	
		Active boot swap cluster ↔ inactive boot swap cluster	Size	Address
00_H	4 KB	$0000\ 0000_H$ - $0000\ 0FFF_H$ ↔ $0000\ 1000_H$ - $0000\ 1FFF_H$	4 KB	$0000\ 0000_H$ - $0000\ 0FFF_H$
01_H	8 KB	$0000\ 0000_H$ - $0000\ 1FFF_H$ ↔ $0000\ 2000_H$ - $0000\ 3FFF_H$	8 KB	$0000\ 000_H$ - $0000\ 1FFF_H$
02_H	16 KB	$0000\ 0000_H$ - $0000\ 3FFF_H$ ↔ $0000\ 4000_H$ - $0000\ 7FFF_H$	12 KB	$0000\ 0000_H$ - $0000\ 2FFF_H$
03_H			16 KB	$0000\ 0000_H$ - $0000\ 3FFF_H$
04_H	32 KB	$0000\ 0000_H$ - $0000\ 7FFF_H$ ↔ $0000\ 8000_H$ - $0000\ FFFF_H$	20 KB	$0000\ 0000_H$ - $0000\ 4FFF_H$
...			...	...
07_H			32 KB	$0000\ 0000_H$ - $0000\ 7FFF_H$
08_H	64 KB	$0000\ 0000_H$ - $0000\ FFFF_H$ ↔ $0001\ 0000_H$ - $0001\ FFFF_H$	36 KB	$0000\ 0000_H$ - $0000\ 8FFF_H$
...			...	...
$0F_H$			64 KB	$0000\ 0000_H$ - $0000\ FFFF_H$
10_H	128 KB	$0000\ 0000_H$ - $0001\ FFFF_H$ ↔ $0002\ 0000_H$ - $0003\ FFFF_H$	68 KB	$0000\ 0000_H$ - $0001\ 0FFF_H$
...			...	...
$1F_H$			128 KB	$0000\ 0000_H$ - $0001\ FFFF_H$
20_H	256 KB	$0000\ 0000_H$ - $0003\ FFFF_H$ ↔ $0004\ 0000_H$ - $0007\ FFFF_H$	132 KB	$0000\ 0000_H$ - $0002\ 0FFF_H$
...			...	...
FF_H			512 KB	$0000\ 0000_H$ - $0003\ FFFF_H$

Maximum boot swap cluster The maximum boot cluster size is 256 KB. Thus code flash above 512 KB can not be subject to boot cluster swapping.

Figure 7-12 “Boot cluster swapping function” on page 469 illustrates an example with following settings:

- number of boot blocks: 2 (boot cluster contains 2 blocks), thus the active boot cluster comprises
  - if boot_flag: blocks 0 and 1
  - if not(boot_flag): blocks 4 and 5
- active boot swap clusters comprises
  - if boot_flag: blocks 0 to 3
  - if not(boot_flag): blocks 4 to 7

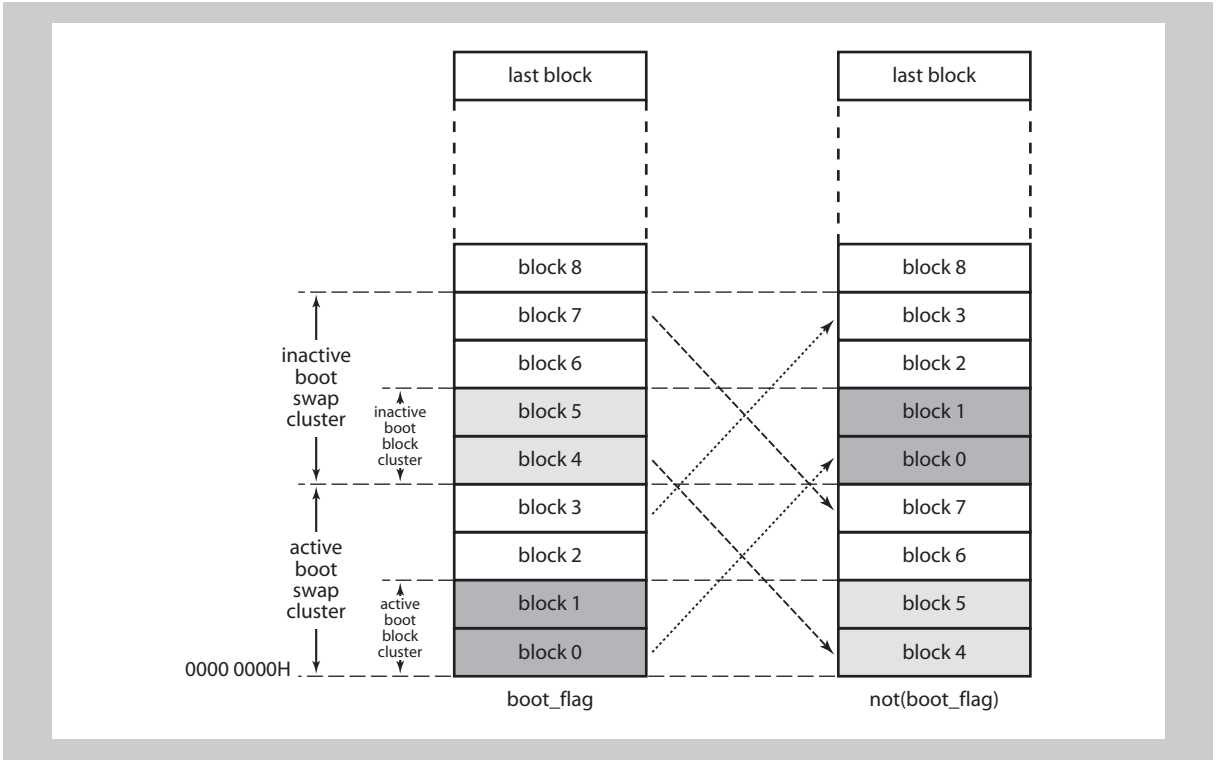


Figure 7-12 Boot cluster swapping function

7.5.5 Interrupt handling during flash Self-Programming

This microcontroller provides functions to maintain interrupt servicing during the Self-Programming procedure.

Since neither the interrupt vector table nor the interrupt handler routines, which are normally located in the flash memory, are accessible while Self-Programming is active, interrupt acknowledges have to be re-routed to non-flash memory, i.e. to the internal RAM.

Therefore two prerequisites are necessary to enable interrupt servicing during Self-Programming:

- The concerned interrupt handler routine needs to be copied to non-flash memory, e.g. to the internal RAM. The user has to initiate this copy process.
- The concerned interrupt acknowledge has to be re-routed to that handler. Re-routing to the handler is done by use of the CPU registers SW_CFG/ SW_BASE respectively EH_CFG/EH_BASE. Refer to the document “V850E2M 32-bit Microprocessor Core Architecture User’s Manual” for further details about these CPU registers.

Re-routing of the interrupt vectors offer two options:

- All interrupt can be mapped to the single interrupt vector of interrupt channel 0.
- The base address of the interrupt vector table can be mapped to a different address. In this case the offsets of the interrupt channels are added to the new base address and the correct interrupt vector is obtained upon interrupt acknowledgement.

It is recommended to refer to the User’s Manual “Flash Self-Programming Library FSL - T05”.

7.6 Flash Mask Options

The flash memory contains an extra area, called flash mask options, that holds user specified data for various purposes.

These flash mask options become effective upon a release of an external $\overline{\text{RESET}}$, a Power-On-Clear reset POCRES or a debugger reset DBRES, thus determining initial settings of various modules.

Caution If the flash memory is programmed during a debug session with the on-chip debugger and any flash mask options have been changed, a target reset command has to be issued in order to make the new option settings effective.

All flash mask options can be read in all operation modes.

Modification of flash mask options depend partly on the operation mode, some can not be modified at all.
The following table summarizes all flash mask options and about the possibility to modify them in the different operation modes.

Table 7-11 Flash mask options and setting

Function	Flash mask option	Modification possible in mode			
		Normal	Serial flash programming	Flash self-programming	Debug
JTAG port group JP0 control	OPBT0.OPBT0[31]	no	yes	yes	yes
VAC enable/ disable of WDTA1	OPBT0.OPBT0[26]	no	yes	yes	yes
Automatic or S/W start of WDTA1	OPBT0.OPBT0[24]	no	yes	yes	yes
Enable/disable of WDTA1	OPBT0.OPBT0[23]	no	yes	yes	yes
VAC enable/ disable of WDTA0	OPBT0.OPBT0[22]	no	yes	yes	yes
Automatic or S/W start of WDTA0	OPBT0.OPBT0[20]	no	yes	yes	yes
Enable/disable of WDTA0	OPBT0.OPBT0[19]	no	yes	yes	yes
Initial value of WDTAn count clock	OPBT0.OPBT0[18:16]	no	yes	yes	yes
Power Sequencer timer value	OPBT0.OPBT0[15:3]	no	yes	yes	yes
Power Sequencer power up mode control	OPBT0.OPBT0[0]	no	yes	yes	yes

7.6.1 OPBT0 - Flash mask option register 0

Access In normal operation mode this register can be read in 32-bit units. Writing to this register is only possible in flash programming and self-programming mode.

Address FF47 000C$_H$

Initial Value User defined

31	30	...	0
OPBT0 [31]	OPBT0 [30]	...	OPBT0 [1]	OPBT0 [0]
R	R	...	R	R

Table 7-12 OPBT0 register contents (1/2)

Bit position	Bit name	Connected to		Function
		Module	Signal	
31	OPBT0[31]	JTAG port group JP0	OPJTAG	Controls the functions of the JTAG port group JP0: 0: JP0 used for general purpose/alternative functions ports 1: JP0 used as JTAG ports
30 to 27	OPBT0[30:25]	–	–	Reserved, set to “0”
26	OPBT0[26]	WDTA1	OPWDVAC	Enables/disables the Variable Activation Code function (VAC) of WDTA1 0: VAC is disabled 1: VAC is enabled
25	OPBT0[25]	–	–	Reserved, set to “0”
24	OPBT0[24]	WDTA1	OPWDRUN	Specifies the start mode of WDTA1: 0: Software trigger start mode 1: Automatic start mode
23	OPBT0[23]		OPWDEN	Enables/disables WDTA1: 0: WDTA1 is disabled 1: WDTA1 is enabled
22	OPBT0[22]	WDTA0	OPWDVAC	Enables/disables the Variable Activation Code function (VAC) of WDTA0 0: VAC is disabled 1: VAC is enabled
21	OPBT0[21]	–	–	Reserved, set to “0”
20	OPBT0[20]	WDTA0	OPWDRUN	Specifies the start mode of WDTA0: 0: Software trigger start mode 1: Automatic start mode
19	OPBT0[19]		OPWDEN	Enables/disables WDTA0: 0: WDTA0 is disabled 1: WDTA0 is enabled
18 to 16	OPBT0[18:16]	WDTA0 WDTA1	OPWDOVF [2-0]	Specifies the reset value of the count clock WDTA0 and WDTA1 control bits WDTAnMD.WDTAnOVF[2:0].

Table 7-12 OPBT0 register contents (2/2)

Bit position	Bit name	Connected to		Function
		Module	Signal	
15 to 3	OPBT0[15:3]	Power Sequencer	TDON[12:0]	Power Sequencer timer value[a]
2 to 1	OPBT0[2:1]	–	–	Reserved, set to “0”
0	OPBT0[0]	Power Sequencer	PWGDEN	Power Sequencer power up mode control[a] **Caution** This bit must always be set to 1.

a) For details concerning the Power Sequencer and the power-up and -down procedures refer to the section *“Power-up and down procedures”* in the chapter *“Power supply”*.

Chapter 8 Data CRC Function A (DCRA)

This chapter contains a generic description of the Data CRC Function A (DCRA).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

8.1 V850E2/Fx4-H DCRA Features

Instances

This microcontroller has the following number of instances of the Data CRC Function A.

Table 8-1 Instances of DCRA

Data CRC Function A	
Instance	1
Name	DCRA0

Instances index n

Throughout this chapter, the individual instances of a Data CRC Function A is identified by the index "n" (n = 0), for example, DCRAnCTL for the DCRAn control register.

Register addresses

All DCRAn register addresses are given as address offsets to the individual base address <DCRAn_base>.
The base address <DCRAn_base> of each DCRAn is listed in the following table:

Table 8-2 Register base addresses <DCRAn_base>

DCRAn instance	<DCRAn_base> address
DCRA0	FF81 $F000_H$

Clock supply All Data CRC Function A provide one clock input:

Table 8-3 DCRAn clock supply

DCRAn instance	DCRAn clock	Connected to
DCRA0	PCLK	Clock Generator CKSCLK_101

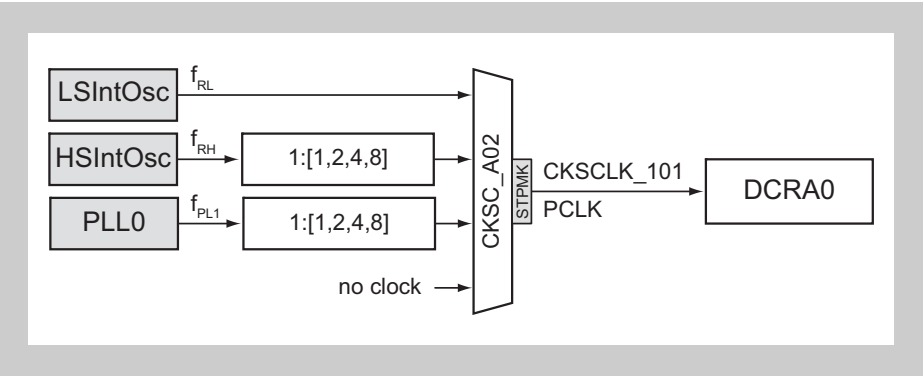


Figure 8-1 DCRA clock supply

DCRA H/W reset The Data CRC Function A and its registers are initialized by the following reset signal:

Table 8-4 DCRAn reset signal

DCRAn	Reset signal
DCRA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

8.2 Functional Overview

Features summary

The Data CRC Function A can be used to verify or generate CRC protected data streams of arbitrary length and different bit widths.

- 32-bit Ethernet CRC ($04C11DB7_H$)

  $(X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X^1+1)$

- 16-bit CCITT CRC (1021_H)

  $(X^{16}+X^{12}+X^5+1)$

- CRC generation to an arbitrary data block length
- After initialization of the CRC input register every write access to the CRC input register generates a new CRC according to the chosen polynomial and the result is stored in the CRC data register.

The following picture shows the block diagram of the Data CRC Function A.

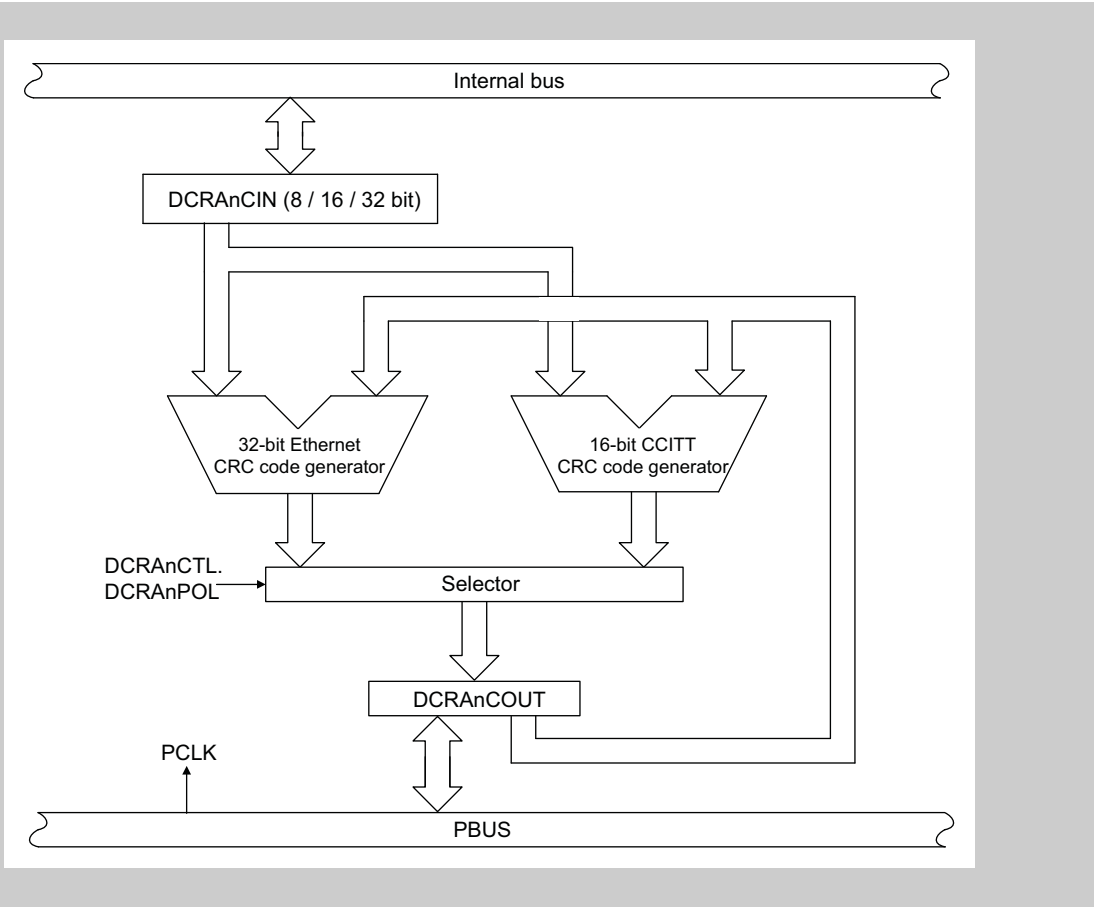


Figure 8-2 Block diagram of Data CRC Function A

8.3 Functional Description

The Data CRC Function A generates a CRC (cyclic redundancy check) of an arbitrary data block length. The data is forwarded to the Data CRC Function in 8-, 16- or 32-bit units. The CRC polynomial can either be selected for 32-bit Ethernet or 16-bit CCITT, the initial starting value must be set at the DCRAnCOUT register before the first write access to the CRC input register (DCRAnCIN) is performed.

The flow chart below shows the CRC generating procedure.

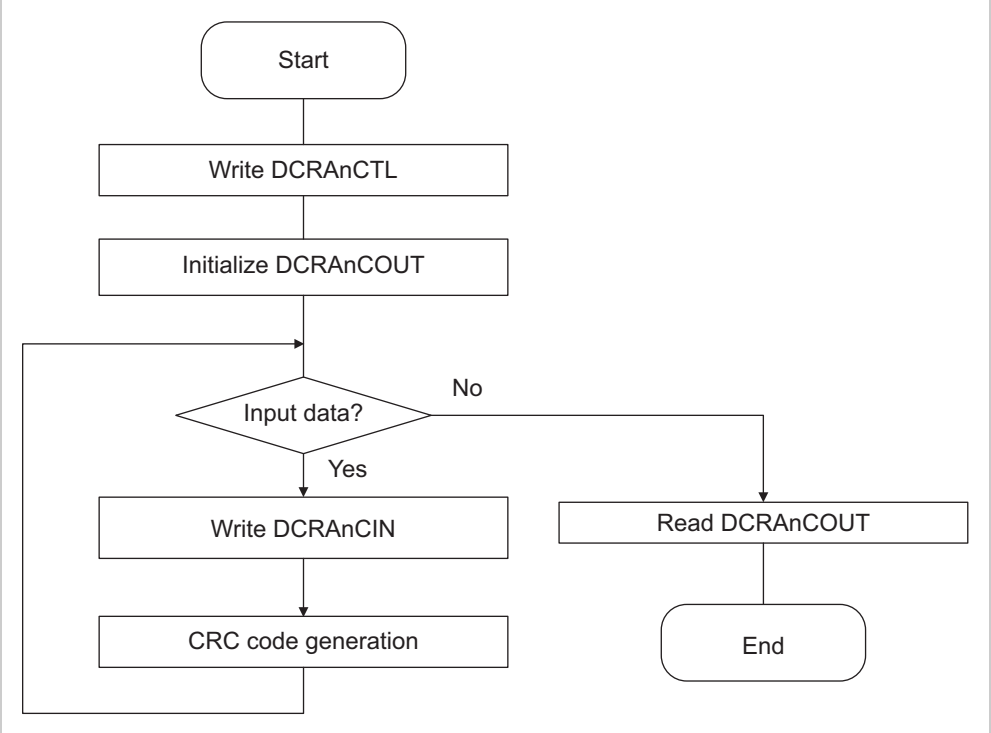


Figure 8-3 Data CRC Function A flow diagram

Notes

1. Before writing the first data to DCRAnCIN the CRC output register DCRAnCOUT must be initialized with the start value.
2. Re-initialization of DCRAnCOUT is also necessary when the polynomial is changed by changing DCRAnCTL.DCRAnPOL.

8.4 Registers

This section contains a description of all registers of the DCRA.

8.4.1 DCRA registers overview

The DCRA is controlled and operated by the following registers:

Table 8-5 DCRA registers overview

Register name	Shortcut	Address
DATA-DCRA input register	DCRAnCIN	<DCRAn_base>
DATA-DCRA data register	DCRAnCOUT	<DCRAn_base> + 4_H
DCRA control register	DCRAnCTL	<DCRAn_base> + 20_H

<DCRAn_base> The base addresses <DCRAn_base> of the Data CRC Function A are defined in the first section of this chapter under the key word “Register addresses”.

8.4.2 DCRA registers details

(1) DCRAnCIN - CRC input register

This register holds the input data for the CRC calculation. The effective bit width used for CRC calculation must be set by DCRAnCTL.DCRAnISZ[1:0].

When data is written to this register, the CRC code is generated.

The CRC calculation is immediately started after the DCRAnCIN register is written. The DCRAnCOUT register must be initialized, with the initial starting value, before the first data of the data block is written to DCRAnCIN register.

Byte order The byte order in DCRAnCIN depends on the selected CRC generating function:

- 32-bit Ethernet CRC polynomial generation (DCRAnCTL.DCRAnPOL = 0)

  The byte order is LSB (Least Significant Byte) first, means LSB at bit position 7...0 of the DCRAnCIN register.

- 16-bit CCITT CRC polynomial generation (DCRAnCTL.DCRAnPOL = 1)

  The byte order is MSB (Most Significant Byte) first, means MSB at bit position 7...0 of the DCRAnCIN register.

Access This register can be read/written in 32-bit units.

Address <DCRAn_base>

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCRAnCIN[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCRAnCIN[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 8-6 DCRAnCIN register contents

Bit position	Bit name	Function
31 to 0	DCRAnCIN[31:0]	Input data for CRC calculation. The valid bits are: • For 32 bit effective bit width: DCRAnCIN[31:0] • For 16 bit effective bit width: DCRAnCIN[15:0] • For 8 bit effective bit width: DCRAnCIN[7:0]

(2) DCRAnCOUT - CRC data register

This register stores the result of the CRC code generated by the 32-bit Ethernet or 16-bit CCITT polynomial.

Access This register can be read/written in 32-bit units.

Address <DCRAn_base> + 4_H

Initial Value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DCRAnCOUT[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DCRAnCOUT[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 8-7 DCRAnCOUT register contents

Bit position	Bit name	Function
31 to 0	DCRAnCOUT[31:0]	Result of the CRC code generation. When the 16-bit CCITT polynomial is enabled, the bits 15 to 0 show the CRC result. The bits 31 to 16 are undefined.

Caution This register must be initialized with the start value before the first data of the data block is written to DCRAnCIN register.

(3) DCRAnCTL - CRC control register

This register controls the CRC generation process.

Access This register can be read/written in 8-bit units.

Address <DCRAn_base> + 20_H

Initial Value 00_H. This register is initialized by any reset.

7	6	5	4	3	2	1	0
0	0	0	0	0	DCRAnISZ[1:0]		DCRAnPOL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 8-8 DCRAnCTL register contents

Bit position	Bit name	Function
2 to 1	DCRAnISZ[1:0]	Specifies the CRC input bit width: 00: 32 bit (DCRAnCIN[31:0]) 01: 16 bit (DCRAnCIN[15:0]) 10: 8 bit (DCRAnCIN[7:0]) 11: Setting prohibited
0	DCRAnPOL	Specifies the CRC generating function: 0: 32-bit Ethernet CRC polynomial generation. The byte order of the DCRAnCIN register is LSB (Least Significant Byte) first, means LSB at bit position 7...0 of the DCRAnCIN register. 1: 16-bit CCITT CRC polynomial generation. The byte order of the DCRAnCIN register is MSB (Most Significant Byte) first, means MSB at bit position 7...0 of the DCRAnCIN register.

Note After changing the CRC generating function (DCRAnCTL.DCRAnPOL) the DCRAnCOUT register must be initialized.

Caution The CRC bit width (DCRAnCTL.DCRAnISZn) must be set according to the data block bit width. Switching the CRC bit width is not allowed during processing of a data block (a data block consists of N bytes, half words or words). After the final CRC result is read from DCRAnCOUT register, the bit width can be changed and the DCRAnCOUT register must be initialized with the initial value afterwards.

Chapter 9 Clock Controller

This chapter describes the Clock Controller functions of the V850E2/Fx4-H microcontrollers.

The clock signals, their control registers, etc. follow a defined naming convention, that reflects their membership to a certain power domain and clock domain.

Power domain index m

The index m is used to define the power area:

- m = 0: denotes the Isolated-Area-0 (Iso0)
- m = 1: denotes the Isolated-Area-1 (Iso1)

m = A: denotes the Always-On-Area (AWO)

Clock domain index n

The index n is used to define the clock domain.

- n_0 denotes the clock domain indices of Isolated-Area-0 clock domains
- n_1 denotes the clock domain indices of Isolated-Area-1 clock domains
- n_A denotes the clock domain indices of Always-On-Area clock domains

Power domain	Clock domains
Isolated-Area-0	n_0 = 00, 05, 06, 07, 11, 12, 16
Isolated-Area-1	n_1 = 01, 02, 03, 04, 05, 06, 07, 08, 09, 11, 12, 13, 14, 15, 22, 28
Always-On-Area	n_A = 02 to 07, 09

Examples:
The clock selector registers CKCS_0n select the clocks for the Isolated-Area-0, which are named CKSCLK_0n.
The clock signal CKSCLK_A06 is the clock, supplied to the clock domain 06 (n = 06) on Always-On-Area (m = A). This clock is selected via the clock selector register CKSC_A06.

PLL index k

Throughout this chapter, the individual instances of the PLL circuits are identified by the index “k” (k = 0, 1, 2), for example PLL0 for the PLL circuit 0.

9.1 Clock Controller Overview

Features summary

The Clock Controller has the following functions:

- four oscillators:
  - Low Speed Internal Oscillator with a nominal frequency of 240 kHz
  - High Speed Internal Oscillator with a nominal frequency of 8 MHz
  - Sub Oscillator 32.768 kHz
  - Main Oscillator 4 MHz to 20 MHz
- three PLL circuits:
  - two Spread Spectrum PLL circuits (PLL0, PLL2) with adjustable frequency dithering parameters
  - one PLL (PLL1) with fixed output frequency
- separate clock selector for each clock domain with individual stand-by control
- three Clock Monitors (CLMA0, CLMA2, CLMA3)
  - CLMA0: supervision of MainOsc
  - CLMA2: supervision of High Speed IntOsc
  - CLMA3: supervision of PLL0
- selectable and adjustable frequency output (FOUT)

Note For the specification of the clock generators frequencies, their tolerance and other parameters, refer to the document Electrical Target Specification .

Clock sources overview

The following table summarizes all clock sources and their typical respectivelly maximum frequencies:

Table 9-1 V850E2/Fx4-H clock sources overview

Source	Input	Frequency
MainOsc	external crystal	nom. 4 MHz to 20 MHz
SubOsc	external crystal	nom. 32.768 kHz
PLL0/SSCG0	MainOsc	max. 160 MHz[a]
PLL1	MainOsc	max. 120 MHz
PLL2/SSCG2	MainOsc	max. 120 MHz[a]
Low Speed IntOsc	–	nom. 240 kHz
High Speed IntOsc	–	nom. 8 MHz

a) + 5 % in SSCG mode

Clock Monitors outputs

The Clock Monitors are generating following output signals to indicate abnormal clock frequencies of the supervised clock:

- $\overline{\text{CLMAnRES}}$: the reset ouputs can generate a reset
- INTCLMAn: the interrupt can generate
  - an interrupt, if connected to the Interrupt Controller
  - a wake-up from stand-by mode, if connected to the Stand-by Controller and if enabled as wake-up factor (refer to the chapter *"Stand-by Controller"* for details).

Table 9-2 CLMAn output signals

CLMA	$\overline{\text{CLMAnRES}}$ connected to	INTCLMAn connected to
CLMA0	Reset Controller	Interrupt Controller Stand-by Controller
CLMA2	Reset Controller	Stand-by Controller
CLMA3	Reset Controller	Stand-by Controller

The following figure shows the main components of the Clock Controller.

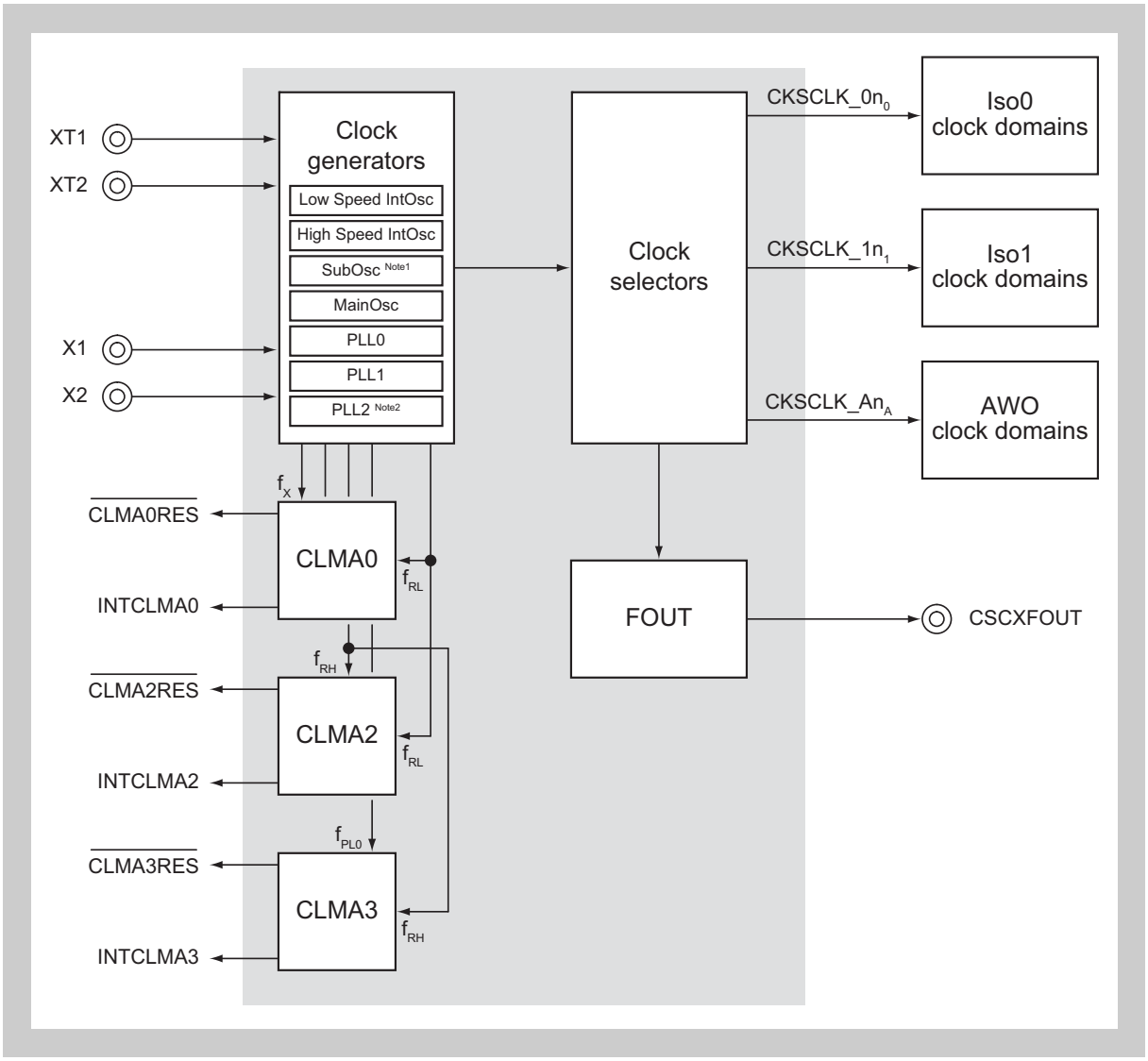


Figure 9-1 Clock Controller overview

9.2 General Description of Clock Generation and Control

The Clock Controller generates a set of clock signals for each of the three power domains

- CKSCLK_0n for clocks on the Isolated-Area-0 (Iso0)
- CKSCLK_1n for clocks on the Isolated-Area-1 (Iso1)
- CKSCLK_An for clocks on the Always-On-Area (AWO).

Each clock signal CKSCLK_mn supplies the clock for the clock domain “mn”.

The following diagram outlines the basic structure of the Clock Controller.

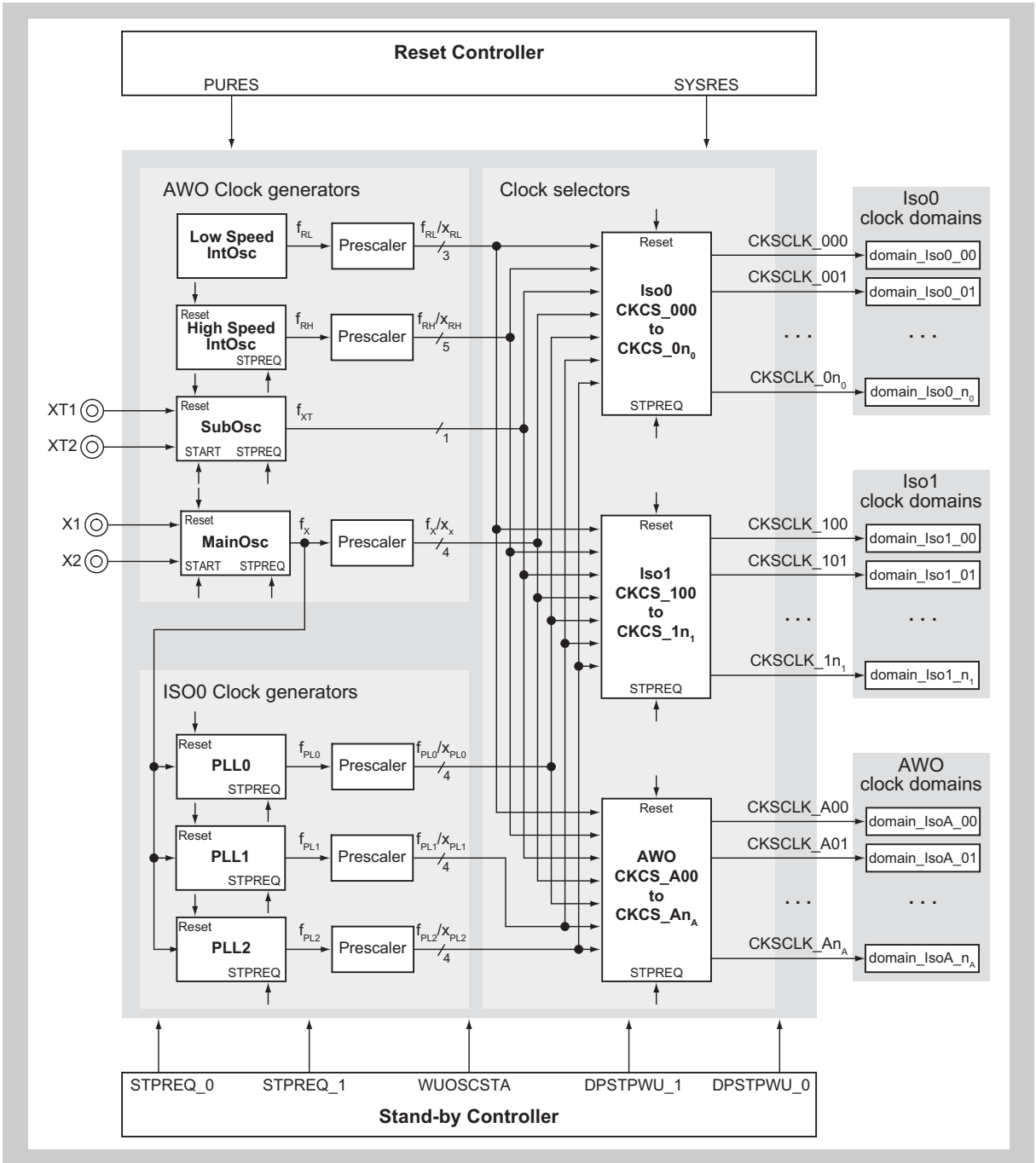


Figure 9-2 Clock Controller structure

The Clock Controller provides

- a set of different clock generators, that generate clocks in different frequency ranges
- a set of clock selectors, that allow to select a clock, or fractions thereof, from the clock generators to supply all clock domains on the three power domains.

The clock generators as well as the selectors are controlled by Reset and Stand-by Controller signals:

- Reset Controller signals
  - PURES: power-up reset signal (Power-On-Clear POCRES or Debugger DBRES reset), which is asserted during power-up of the microcontroller
  - SYSRES: system reset signal, which is asserted by any internal and external reset source, including PURES

  For detailed information about the Reset Controller signals, refer to the chapter *12 “Reset Controller” on page 575* .
- Stand-by Controller signals
  - STPREQ_0: Isolated Area-0 stop request signal
    STPREQ_0 is asserted, when the microcontroller enters an Isolated-Area-0 stand-by mode and is used to stop the clock generators and the Isolated-Area-0 and Always-On-Area domain clocks via the clock selectors CKSC_0n_0 and CKSC_An$_A$.
  - STPREQ_1: Isolated Area-1 stop request signal
    STPREQ_1 is asserted, when the microcontroller enters an Isolated-Area-1 stand-by mode and is used to stop Isolated-Area-1 domain clocks via the clock selectors CKSC_1n_1.
  - DPSTPWU_0: Isolated Area-0 DEEPSTOP wake-up signal
    DPSTPWU_0 is asserted, when the microcontroller wakes up from an Isolated-Area-0 DEEPSTOP mode and is used to reset the Isolated-Area-0 clock selectors.
  - DPSTPWU_1: Isolated Area-0 DEEPSTOP wake-up signal
    DPSTPWU_1 is asserted, when the microcontroller wakes up from an Isolated-Area-1 DEEPSTOP mode and is used to reset the Isolated-Area-1 clock selectors.
  - WUOSCSTA: Wake-up oscillator start signal
    WUOSCSTA can optionally be asserted, when the MainOsc and SubOsc shall start operation upon wake-up from a stand-by mode. The generation of WUOSCSTA is controlled by a register of the Stand-by Controller.

For detailed information about the Stand-by Controller signals, refer to the chapter *“Stand-by Controller”*.

Common to all clock generators and clock domains of a certain power domain is, that they can be stopped all together in STOP stand-by mode, that requests the clocks of this domain to be stopped. The Stand-by Controller issues the signals STPREQ_0 respectively STPREQ_1 stop requests to the generators and selectors. However all clock generators and most of the domain clock selectors allow to mask the stop request in order to continue clock output to the concerned clock domain also in stand-by mode.

Since the Isolated-Area-0 and -1 clock selectors reside on the respective power domain, and thus their power supply are switched off in a DEEPSTOP mode, the clock selectors are reset upon DEEPSTOP wake-up. So they have to be completely re-initialized afterwards.

The following figure shows the reset and stop request signals wiring of the clock generators and selectors.

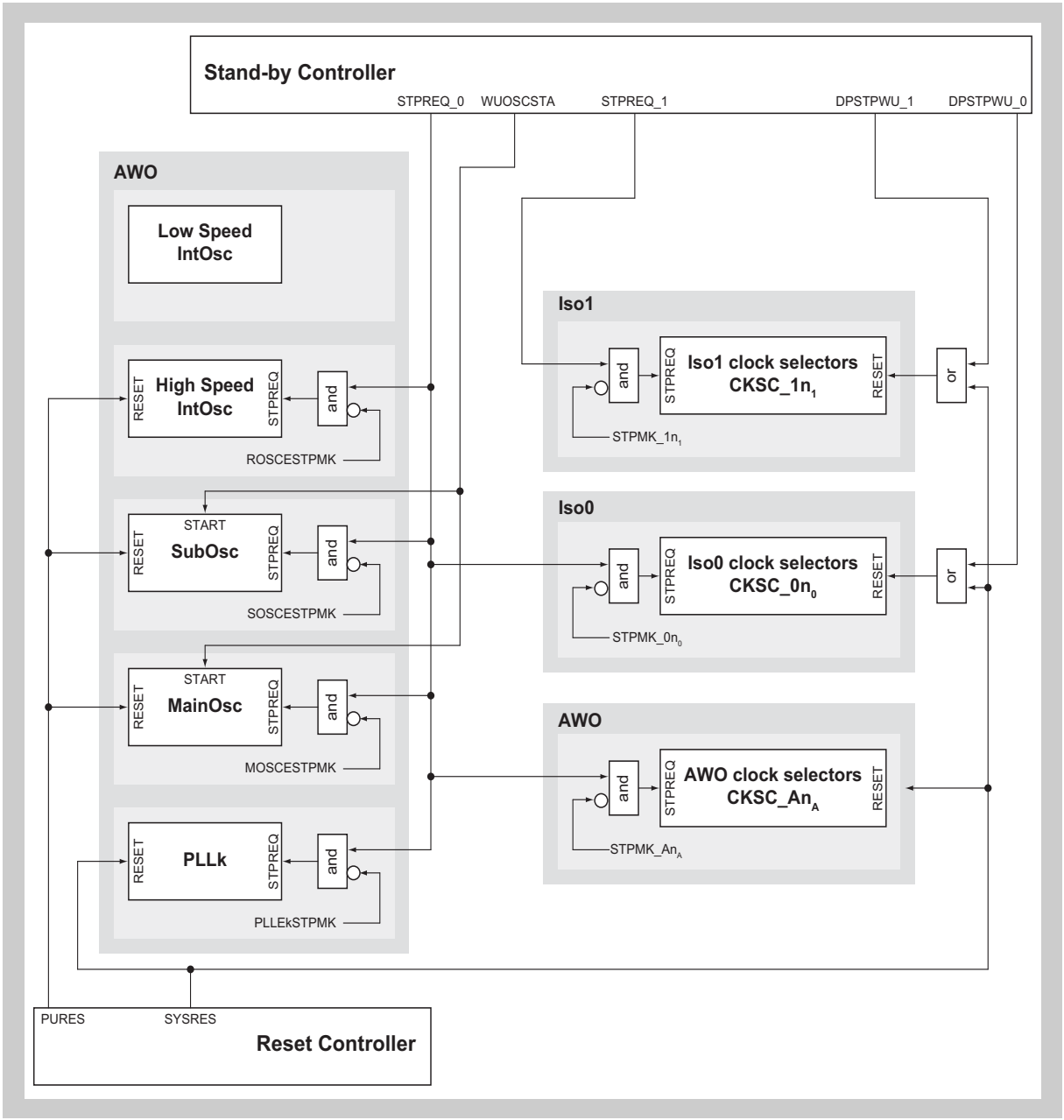


Figure 9-3 Clock Controller stand-by mode and reset signals

9.2.1 Clock generators

Seven clock generators are provided:

- Low Speed Internal Oscillator (Low Speed IntOsc)
  This oscillator starts operation after power up and can not be stopped, hence it is always operating. It generates the clock f_{RL} with a nominal frequency of 240 kHz.
  It does not require any external components.
- High Speed Internal Oscillator (High Speed IntOsc)
  The High Speed IntOsc generates a clock with the nominal frequency of 8 MHz without any external components.
- Sub Oscillator (SubOsc)
  The SubOsc requires an external resonator (connected to XT1, XT2) to generate the sub clock f_{TX} with a nominal frequency of 32768 Hz. This clock is mainly used for real-time clock applications.
- Main Oscillator (MainOsc)
  The MainOsc output clock f_X is the main system root clock, as f_X is input to the PLLs. The MainOsc operates with an external resonator (X1, X2).
- PLL0 to PLL2
  The PLL circuits generate all high speed operation clocks f_{PL0}, f_{PL1}, f_{PL2} for normal operation of the microcontroller.

The Low Speed IntOsc, High Speed IntOsc, SubOsc and MainOsc oscillators are located on the Always-On-Area, while the PLLk circuits reside on the Isolated-Area-0. Thus in DEEPSTOP stand-by mode the PLLk circuits are switched off.

It is optionally possible to individually disable the generators (except the Low Speed IntOsc) in STOP stand-by mode.

The output clocks of the clock generators, except of the SubOsc f_{TX}, are input to prescalers, which provide different fractions of these clocks.
Note that the prescalers for all clocks have different division factors.

(1) Automatic start of the MainOsc and SubOsc after stand-by wake-up

The MainOsc and SubOsc can be automatically started upon stand-by mode wake-up, independent of their status before stand-by.
Refer to the *"Wake-up"* section in the chapter *10 "Stand-by Controller (STBC)" on page 515* for details about this feature.

(2) Clock generators reset

The MainOsc, SubOsc and High Speed IntOsc clock generators are only reset by the power-up reset PURES (Power-On-Clear reset POCRES or Debugger reset DBRES).
Thus any internal reset neither stops these clock generators nor changes their configuration. Consequently the microcontroller start-up time after release from an internal reset is minimum, because the clock generator's stabilization times have not to be regarded.

All PLLs are reset by all internal and external reset sources via SYSRES.

(3) STOP stand-by mode request

The Stand-by Controller signal STPREQ_0 is asserted upon entering Isolated-Area-0 STOP mode and is applied to the stop requests STPREQ of the MainOsc, SubOsc, High Speed IntOsc and all PLLs.

STPREQ can be individually masked by the respective clock generator control register:

- MainOsc: MOSCE.MOSCESTPMK = 0:
  STPREQ is not masked, thus f_X is stopped, if STPREQ_0 is asserted
- SubOsc: SOSCE.SOSCESTPMK = 0:
  STPREQ is not masked, thus f_{XT} is stopped, if STPREQ_0 is asserted
- High Speed IntOsc: ROSCE.ROSCESTPMK = 0:
  STPREQ is not masked, thus f_{RH} is stopped, if STPREQ_0 is asserted
- PLLk: PLLEk.PLLEkSTPMK = 0:
  STPREQ is not masked, thus f_{PLk} is stopped, if STPREQ_0 is asserted

If the stop request is not masked, i.e. the clock generator is stopped during stand-by mode, it automatically restarts operation after wake-up from stand-by mode, provided it was operating before stand-by mode entry.
If the clock generator was stopped before stand-by mode, it remains stopped. However the MainOsc and SubOsc can be started after wake-up, even if they were not operating before stand-by. Refer to *1 “Automatic start of the MainOsc and SubOsc after stand-by wake-up” on page 489* .

If the stop request is masked, the clock generator’s status is not changed during stand-by mode.

(4) DEEPSTOP stand-by mode

A DEEPSTOP stand-by mode asserts also the STPREQ_0 signal and impacts the clock generators in the same way like STOP stand-by mode.

Since the power supplies of the Isolated-Area-0 and Isolated-Area-1 are switched off in DEEPSTOP stand-by mode, all PLL clocks f_{PLk} stop operation.
After release from DEEPSTOP mode the PLLk control registers take on their reset state and the PLL circuits have to be completely re-initialized.

Notes

1. Refer to *9.3 “Clock Generators” on page 493* for a detailed description of the clock generators.
2. For the specification of the clock generators frequencies, their tolerance and other parameters, refer to the document Electrical Target Specification.

9.2.2 Clock selectors

The clocks, generated by the clock generators, are input to the clock selectors CKSC_mn. A separate clock selector register CKSC_mn is provided for each clock domain.
Note that not all available clocks from the clock generators are input to each clock selector.

The three sets of clock selector registers are dedicated to the clock domains on one of the three power domains:

- n_0 registers CKSC_000 to CKSC_0n_0 for clock domains on the Isolated-Area-0.
- n_1 registers CKSC_100 to CKSC_1n_1 for clock domains on the Isolated-Area-1
- n_A registers CKSC_A00 to CKSC_An_A for clock domains on the Always-On-Area

By use of a clock selector register CKSC_mn one of its input clocks is selected as the clock CKSCLK_mn.

The clock selectors CKSCLK_mn, selecting the clock domain clocks for power domain m, are also located on the same power domain.

(1) Clock selectors reset

The clock selectors are reset by the SYSRES signal, which is asserted by any microcontroller internal or external reset. Hence after any reset the supply of all clock domains are set to their default configuration.

Additionally the clock selectors located on the Isolated-Area-0 and Isolated-Area-1 power domain are reset if their domain is woken up from DEEPSTOP mode.

(2) STOP stand-by mode request

The stop request for a certain power domain is issued by the Stand-by Controller by asserting any of the stop request signals STPREQ_0 (for Isolated-Area-0 and Always-On-Area) respectively STPREQ_1 (for Isolated-Area-1) upon entering STOP mode.

The stop request, supplied to all clock selectors of a power domain, can be individually masked by the clock selector register CKSC_mn:

- CKSC_mn.STPMK_mn = 0:
  Stop request to clock selector CKSC_mn is not masked, thus CLSCLK_mn is stopped, if STPREQ_0 respectively STPREQ_1 is asserted.
- CKSC_mn.STPMK_mn = 1:
  Stop request to clock selector CKSC_mn is masked, thus CLSCLK_mn remains in operation, even if STPREQ_0 respectively STPREQ_1 is asserted.

Note Not all clock selectors CKSC_mn provide the stop mask function. These clock domains remain in operation during STOP mode. Refer to the section *"Clock Selection"* below in this chapter for details about each clock selector.

(3) DEEPSTOP stand-by mode

In a DEEPSTOP stand-by mode the power supplies of the Isolated-Area-0 respectively Isolated-Area-1 are switched off.
Since the clock selectors CKSC_mn reside on the power domain, they are serving, they are switched off as well in DEEPSTOP mode.
Upon wake-up they receive the reset signal DPSTPWU_0 respectively DPSTPWU_1, and are in their default - i.e. reset - state after DEEPSTOP wake-up. Consequently the concerned clock selectors have to be completely re-initialized.

In particular this means:

- After wake-up from DEEPSTOP mode, i.e. power supply of Isolated-Area-0 and Isolated-Area-1 are switched off, the clock selectors of Isolated-Area-0 CKSC_0n_0 and Isolated-Area-1 CKSC_1n_1 have to be re-initialized.
- After wake-up from RUN mode (ISO1 DEEPSTOP), i.e. power supply of Isolated-Area-1 is switched off, the clock selectors of Isolated-Area-1 CKSC_1n_1 have to be re-initialized.

Notes

1. If the Isolated-Area-0 is set into DEEPSTOP mode, all clock generators and the clock selectors CKSC_An$_A$ on the Always-On-Area receive the stop request signal STPREQ_0.
2. Refer to the section *"Clock Selection"* below for a detailed description of the clock selectors.

9.3 Clock Generators

9.3.1 Main Oscillator (MainOsc) clock generator

The Main Oscillator generates the clock f_X, which is supplied to the clock domain clock selectors CKSC_mn. f_X is also used as the PLL input clock PLLCLKIN.

The diagram below shows the basic structure and signals of the MainOsc clock generator.

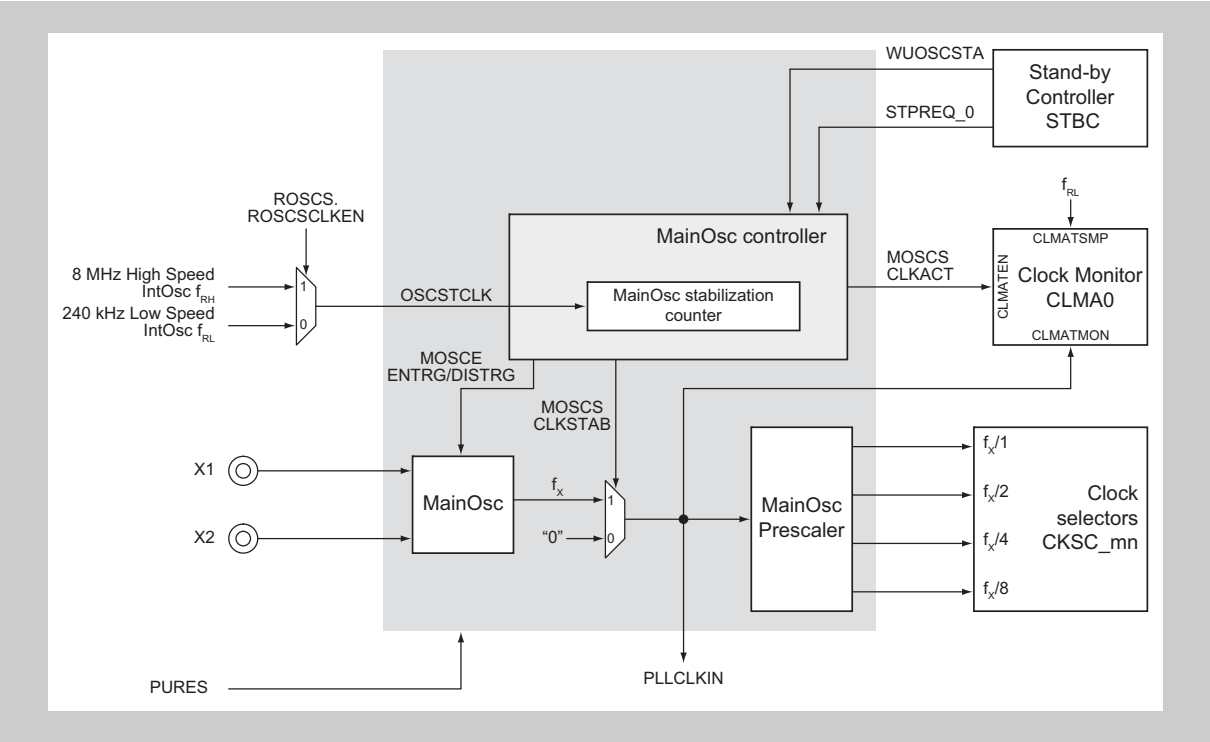


Figure 9-4 Main Oscillator clock generator

After release of the power-up reset PURES the MainOsc is disabled.

Note During assertion of the external $\overline{\text{RESET}}$ the MainOsc is always operating with its maximum amplification gain.
Refer to the descrition of the external $\overline{\text{RESET}}$ in the *"Reset Controller"* chapter for details.
All other resets except PURES and $\overline{\text{RESET}}$ do not affect the MainOsc.

MainOsc enable/disable

The MainOsc can be enabled and disabled and its status can be checked:

- enable trigger: MOSCE.MOSCEENTRG = 1,
  enabled status: MOSCS.MOSCSCLKEN = 1
- disable trigger: MOSCE.MOSCEDISTRG = 1
  disabled status: MOSCS.MOSCSCLKEN = 0

MainOsc stabilization

The MainOsc stabilization counter starts counting down the stabilization time. During count down the MOSCSCLKSTAB signal disables the f_X output to the MainOsc prescaler.

If the stabilization counter has reached the stabilization count value, as defined in MOSCST.MOST[3:0], f_X is judged as stable and change of the MOSCSCLKSTAB signal inputs f_X to the prescaler. This status is indicated by the bit MOSCS.MOSCSCLKSTAB = 1.
All prescaler outputs are available for the clock selectors CKSC_mn. This status is indicated by MOSCS.MOSCSCLKACT = 1.

The stabilization counter clock OSCSTCLK is selected from two sources:

- OSCSTCLK = 8 MHz High Speed IntOsc clock f_{RH}, if this oscillator is operating (ROSCS.ROSCSCLKEN = 1)
- OSCSTCLK = 240 kHz Low Speed IntOsc clock f_{RL}, if the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0)

The stabilization counter clock source is selected automatically by the High Speed IntOsc operation status.

MOSCST.MOST[3:0] determines the number of OSCSTCLK periods as the MainOsc stabilization time, which can be specified in the range of 2^2 to 2^{17} OSCSTCLK periods.

Short stabilization time mode

If the MainOsc has been stopped during stand-by mode (refer to the *"STOP stand-by mode request"* below) and is restarted after wake-up, the stabilization time - and so the wake-up time - can be shortened by selecting the short stabilization time mode (MOSCC.MOSCCSHTSTBY = 1). This sets the amplification gain of the MainOsc circuit to maximum.
Note that selection of the amplication gain by MOSCC.MOSCCSHTSTBY has only effect during the MainOsc stabilization. If the MainOsc is stable the amplification gain is determined by MOSCCC.MOSCCAMPSEL[1:0].

Note

Make sure to use the f_X clock only if MOSCS.MOSCSCLKACT = 1.

MainOsc amplification gain

By use of the control bits MOSCC.MOSCCAMPSEL[1:0] the oscillator's amplifier gain can be adjusted to cope with different external resonators and their external circuitry.
The amplification gain can be forced to its maximum, thus disregarding MOSCC.MOSCCAMPSEL[1:0], by setting MOSCC.MOSCCSHTSTBY = 1. This may result in a shorter stabilization time.
Note that the amplification gain is set to maximum by default.
For detailed information about the electrical conditions of the different amplification gain settings refer to the Electrical Target Specification.

Stand-by mode requests

By use of the STPREQ_0 signal the Stand-by Controller indicates stand-by status of the Isolated-Area-0 and the Always-On-Area. The stop request mask bit MOSCE.MOSCESTPMK controls whether the MainOsc is stopped during stand-by or continues operation.
If the MainOsc is stopped during stand-by (MOSCE.MOSCESTPMK = 0), it is automatically re-started upon wake-up from stand-by mode, provided it was operating before stand-by mode entry.
If the MainOsc was stopped before stand-by mode, it remains stopped.

Note The MainOsc can also be activated after stand-by wake-up, independent of its status before stand-by mode, via the wake-up oscillator start signal WUOSCSTA. Generation of WUOSCSTA is controlled by the oscillator wake-up mask register OSCWUFMSK. Refer to section *"Wake-up"* in the chapter *"Stand-by Controller"* for details.

Clock Monitor control The MainOsc activity flag MOSCS.MOSCSCLKACT is output to the Clock Monitor CLMA0 to control its operation. In case the MainOsc is inactive, supervision of its output clock f_X by CLMA0 is also deactivated.

The table below summarizes the different conditions for the Clock Monitor control.

Table 9-3 Clock Monitor 0 status control

MainOsc enable status	MainOsc stand-by control		MainOsc stand-by status	CLMA0 status
MOSCSCLKEN	MOSCESTPMK	STPREQ_0	MOSCSCLKACT	
0	X	X	0	stopped
1	0	0	1	active
		1	0	stopped
	1	X	1	active

MainOsc enable/ disable trigger The MainOsc can be enabled and disabled by the enable and disable trigger control bits:

- enable trigger MOSCE.MOSCEENTRG = 1 starts the MainOsc
  Note that setting the enable trigger is only effective if the MainOsc is inactive, i.e. if MOSCS.MOSCSCLKACT = 0.
- disable trigger MOSCE.MOSCEDISTRG = 1 stops the MainOsc
  Note that setting the disable trigger is only effective if the MainOsc is active, i.e. if MOSCS.MOSCSCLKACT = 1.

9.3.2 Sub Oscillator (SubOsc) clock generator

The Sub Oscillator generates the sub clock f_{TX}, which is supplied to the clock domain clock selectors CKSC_mn. f_{TX} has usually a nominal frequency of 32.768 kHz and is mainly used as for the Real-time Clock.

The diagram below shows the basic structure and signals of the SubOsc clock generator.

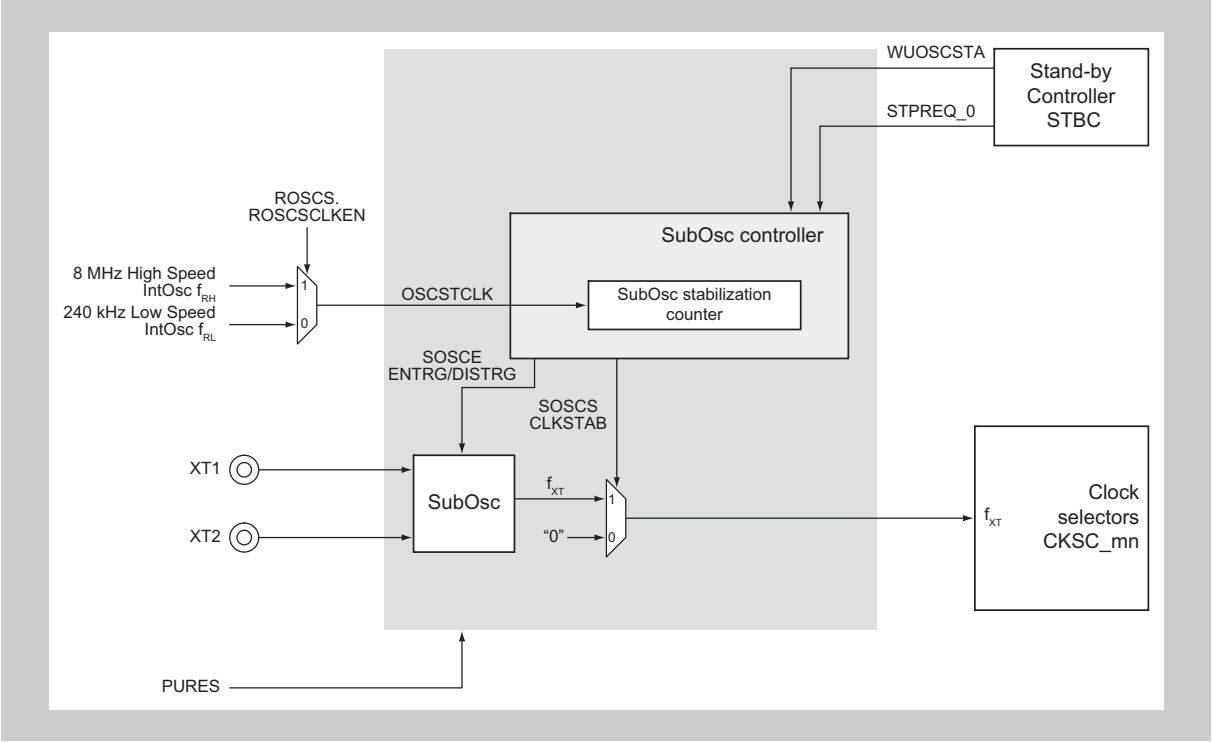


Figure 9-5 Sub Oscillator clock generator

After release of the power-up reset PURES the SubOsc is disabled.

Note All other resets except PURES do not affect the SubOsc.

SubOsc enable/disable

The SubOsc can be enabled and disabled and its status can be checked:

- enable trigger: SOSCE.SOSCEENTRG = 1,
  enabled status: SOSCS.SOSCSCLKEN = 1

disable trigger: SOSCE.SOSCEDISTRG = 1
disabled status: SOSCS.SOSCSCLKEN = 0.

SubOsc stabilization

The SubOsc stabilization counter starts counting down the stabilization time.

As long as the SubOsc is not stable, the SOSCS.SOSCSCLKSTAB signal disables the f_{XT} output to the clock selectors CKSC_mn.
If the SubOsc stabilization counter has reached the value, defined by SOSCST.SOST[2:0], f_{XT} is judged as stable and change of SOSCS.SOSCSCLKSTAB inputs f_{XT} to the clock selectors CKSC_mn.
The f_{XT} clock stable status is indicated by the bit SOSCS.SOSCSCLKSTAB = 1 and its activation by SOSCS.SOSCSCLKACT = 1.

The stabilization counter clock OSCSTCLK is selected from two sources:

- OSCSTCLK = 8 MHz High Speed IntOsc clock f_{RH}, if this oscillator is operating (ROSCS.ROSCSCLKEN = 1)
- OSCSTCLK = 240 kHz Low Speed IntOsc clock f_{RL}, if the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0)

The stabilization counter clock source is selected automatically by the High Speed IntOsc operation status.

SOSCST.SOST[2:0] determines the number of OSCSTCLK periods as the SubOsc stabilization time, which can be specified in the range of 2^{19} to 2^{26} OSCSTCLK periods.

Note Make sure to use the f_{XT} clock only if SOSCS.SOSCSCLKACT = 1.

SubOsc input frequencies

The SubOsc input frequency is typically 32.768 kHz.

Stand-by mode requests

By use of the STPREQ_0 signal the Stand-by Controller indicates stand-by status of the Isolated-Area-0 and the Always-On-Area. The stop request mask bit SOSCE.SOSCESTPMK controls whether the SubOsc is stopped during stand-by or continues operation.
If the SubOsc is stopped during stand-by (SOSCE.SOSCESTPMK = 0), it is automatically re-started upon wake-up from stand-by mode, provided it was operating before stand-by mode entry.
If the SubOsc was stopped before stand-by mode, it remains stopped.

Note The SubOsc can also be activated after stand-by wake-up, independent of its status before stand-by mode, via the wake-up oscillator start signal WUOSCSTA. Generation of WUOSCSTA is controlled by the oscillator wake-up mask register OSCWUFMSK.
Refer to section *"Wake-up"* in the chapter *"Stand-by Controller"* for details.

SubOsc enable/ disable trigger

The SubOsc can be enabled and disabled by the enable and disable trigger control bits:

- enable trigger SOSCE.SOSCEENTRG = 1 starts the SubOsc
  Note that setting the enable trigger is only effective if the SubOsc is inactive, i.e. if SOSCS.SOSCSCLKACT = 0.
- disable trigger SOSCE.SOSCEDISTRG = 1 stops the SubOsc
  Note that setting the disable trigger is only effective if the SubOsc is active, i.e. if SOSCS.SOSCSCLKACT = 1.

9.3.3 Low Speed Internal Oscillator (Low Speed IntOsc) clock generator

The Low Speed Internal Oscillator generates the clock f_{RL}, which is supplied to the clock domain clock selectors CKSC_mn. f_{RL} has a nominal frequency of 240 kHz.

The diagram below shows the basic structure and signals of the Low Speed IntOsc clock generator.

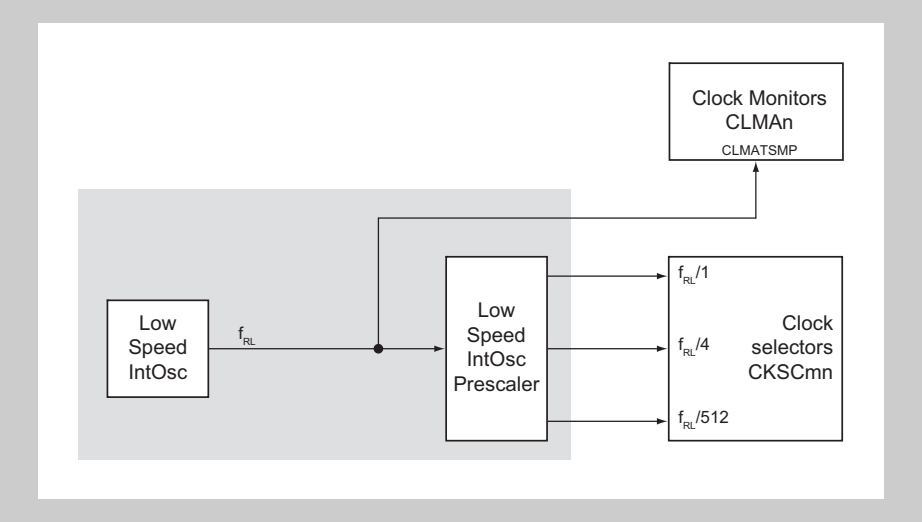


Figure 9-6 Low Speed Internal Oscillator clock generator

After power-up the Low Speed IntOsc starts operation. It can not be stopped.

The Low Speed IntOsc clock f_{RL} is used as the sampling clock for Clock Monitors CMLA0 and CLMA2.

9.3.4 High Speed Internal Oscillator (High Speed IntOsc) clock generator

The High Speed Internal Oscillator generates the clock f_{RH}, which is supplied to the clock domain clock selectors CKSC_mn. f_{RH} has a nominal frequency of 8 MHz.

The diagram below shows the basic structure and signals of the High Speed IntOsc clock generator.

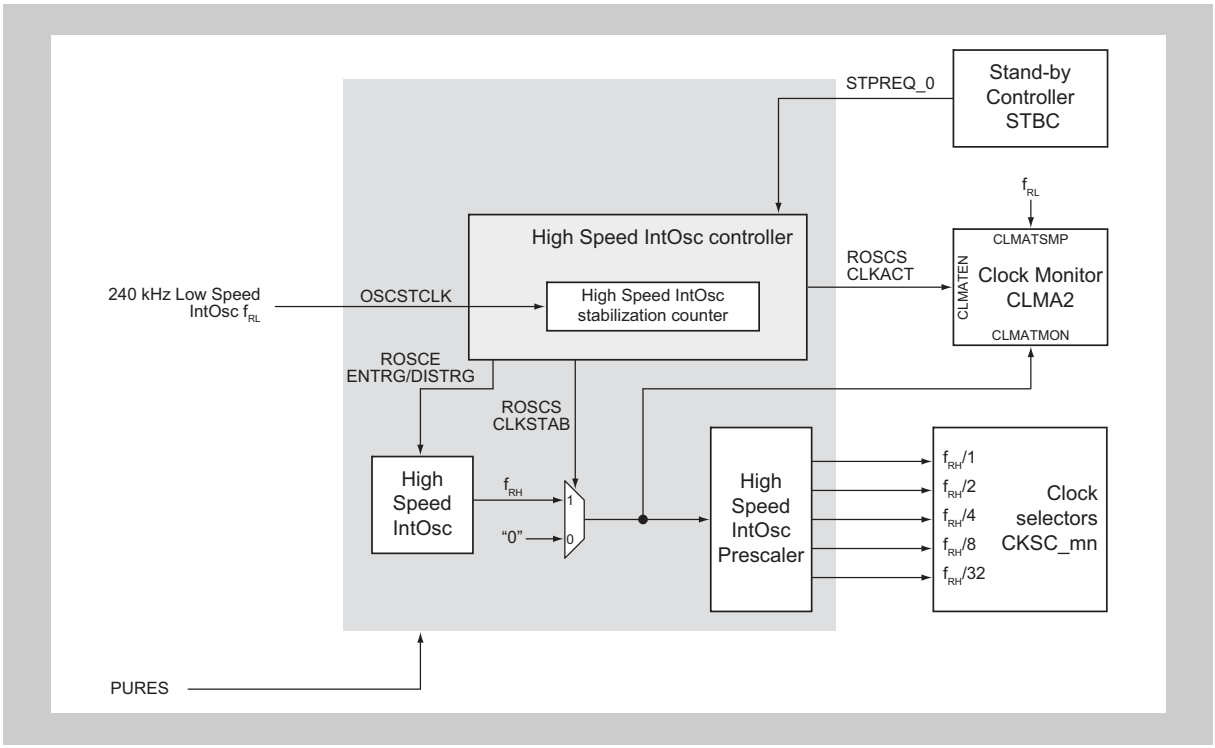


Figure 9-7 High Speed Internal Oscillator clock generator

After PURES release the High Speed IntOsc starts operation.

Note All other resets except PURES do not affect the High Speed IntOsc.

High Speed IntOsc enable/disable

The High Speed IntOsc can be enabled and disabled and its status can be checked:

- enable trigger: ROSCE.ROSCEENTRG = 1,
  enabled status: ROSCS.ROSCSCLKEN = 1
- disable trigger: ROSCE.ROSCEDISTRG = 1
  disabled status: ROSCS.ROSCSCLKEN = 0

Caution

Pay attention to clock domains, which are using the High Speed IntOsc clock as source via the clock selectors CKSC_mn, when disabling f_{RH} by ROSCE.ROSCEDISTRG = 1.
In particular all resets set all clock selectors CKSCLK_mn to their default selection, but only the PURES enables the High Speed IntOsc.
As a consequence all PBUS clocks PCLK will not operate after a reset - except PURES -, since the High Speed IntOsc is selected as their clock source, but the High Speed IntOsc is not enabled.

High Speed IntOsc stabilization

The High Speed IntOsc stabilization counter starts counting down the stabilization time.

As long as the High Speed IntOsc is not stable, the CLKSEL signal disables the f_{RH} output to the MainOsc prescaler.
If the High Speed IntOsc stabilization counter has counted four periods of its input clock OSCSTCLK, f_{RH} is judged as stable and change of CLKSEL inputs f_{RH} to the prescaler, thus all prescaler outputs are available for the clock selectors CKSCmn.
The f_{RH} clock stable status is indicated by the bit ROSCS.ROSCSCLKSTAB = 1 and its activation by ROSCS.ROSCSCLKACT = 1.

The stabilization counter clock OSCSTCLK is the 240 kHz Low Speed IntOsc clock f_{RL}.

Note

Make sure to use the f_{RH} clock only if ROSCS.ROSCSCLKACT = 1.

Stand-by mode requests

By use of the STPREQ_0 signal the Stand-by Controller indicates stand-by status of the Isolated-Area-0 and the Always-On-Area. The stop request mask bit ROSCE.ROSCESTPMK controls whether the High Speed IntOsc is stopped during stand-by or continues operation.
If the High Speed IntOsc is stopped during stand-by (ROSCE.ROSCESTPMK = 0), it is automatically re-started upon wake-up from stand-by mode, provided it was operating before stand-by mode entry.
If the High Speed IntOsc was stopped before stand-by mode, it remains stopped.

Clock Monitor control

The High Speed IntOsc activity flag ROSCS.ROSCSCLKACT is output to the Clock Monitor CLMA2 to control its operation. In case the High Speed IntOsc is inactive, supervision of its output clock f_{RH} by CLMA2 is also deactivated.

The table below summarizes the different conditions for the Clock Monitor control.

Table 9-4 Clock Monitor 2 status control

High Speed IntOsc enable status	High Speed IntOsc stand-by control		High Speed IntOsc stand-by status	CLMA2 status
ROSCSCLKEN	ROSCESTPMK	STPREQ_0	ROSCSCLKACT	
0	X	X	0	stopped
1	0	0	1	active
		1	0	stopped
	1	X	1	active

The High Speed IntOsc clock f_{RH} is used as the sampling clock for Clock Monitor CMLA3.

High Speed IntOsc enable/disable trigger

The High Speed IntOsc can be enabled and disabled by the enable and disable trigger control bits:

- enable trigger ROSCE.ROSCEENTRG = 1 starts the High Speed IntOsc Note that setting the enable trigger is only effective if the High Speed IntOsc is inactive, i.e. if ROSCS.ROSCSCLKACT = 0.
- disable trigger ROSCE.ROSCEDISTRG = 1 stops the High Speed IntOsc Note that setting the disable trigger is only effective if the High Speed IntOsc is active, i.e. if ROSCS.ROSCSCLKACT = 1.

9.3.5 Phase-Locked Loop (PLL) clock generators

The Main Oscillator clock f_X is input to the Phase-Locked Loops clock generators PLLk. The PLLk output clocks f_{PLk} are a multiple of f_X and serve as the main operation clocks for the microcontroller.

The diagram below shows the basic structure and signals of the PLLk clock generator.

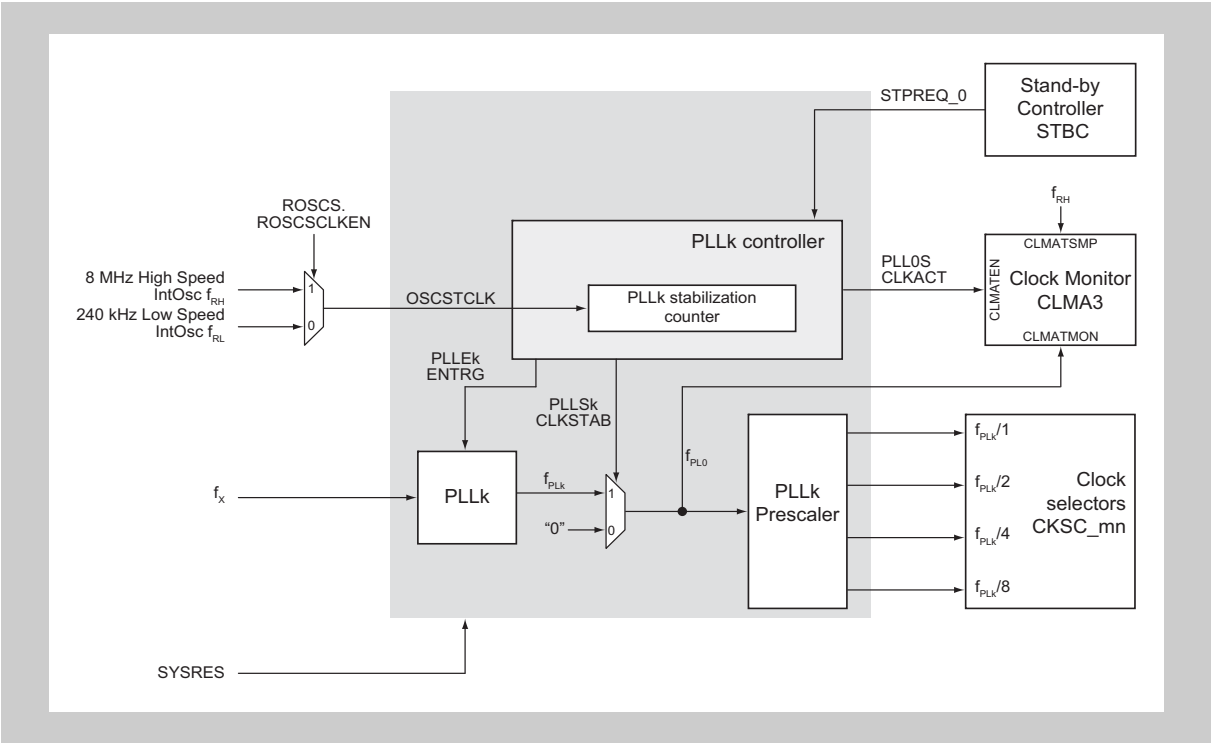


Figure 9-8 PLLk clock generator

After power-up of the microcontroller the PLLk are disabled.

Note All resets (SYSRES) stop the PLLk.

PLLk enable/disable

The PLLk can be enabled and disabled and its status can be checked:

- enable trigger: PLLEk.PLLEkENTRG = 1,
  enabled status: PLLSk.PLLSkCLKEN = 1 = 1
- disable trigger: PLLEk.PLLEkDISTRG = 1
  disabled status: PLLSk.PLLSkCLKEN = 1 = 0

PLLk stabilization

The PLLk stabilization counter starts counting down the stabilization time.

As long as the PLLk is not stable, the CLKSEL signal disables the f_{PLk} output to the PLLk prescaler.
If the PLLk stabilization counter has reached the value, defined by PLLSTk.PLLSTk[2:0], f_{PLk} is judged as stable and change of CLKSEL inputs f_X to the prescaler, thus all prescaler outputs are available for the clock selectors CKSC_mn.
The f_{PLk} clock stable status is indicated by the bit PLLSk.PLLSkCLKSTABk = 1 and its activation by PLLSk.PLLSkCLKACT = 1.

The stabilization counter clock OSCSTCLK is selected from two sources:

- OSCSTCLK = 8 MHz High Speed IntOsc clock f_{RH}, if this oscillator is operating (ROSCS.ROSCSCLKEN = 1)
- OSCSTCLK = 240 kHz Low Speed IntOsc clock f_{RL}, if the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0)

The stabilization counter clock source is selected automatically by the High Speed IntOsc operation status.

PLLSTk.PLLSTk[2:0] determines the number of OSCSTCLK periods as the PLLk stabilization time, which can be specified in the range of 2^7 to 2^{14} OSCSTCLK periods.

Note Make sure to use the f_{PLk} clock only if PLLSk.PLLSkCLKACT = 1.

Stand-by mode requests

By use of the STPREQ_0 signal the Stand-by Controller indicates stand-by status of the Isolated-Area-0 and the Always-On-Area. The stop request mask bit PLLEk.PLLEkSTPMK controls whether the PLLk is stopped during stand-by or continues operation.
If the PLLk is stopped during stand-by (PLLEk.PLLEkSTPMK = 0), it is automatically re-started upon wake-up from stand-by mode, provided it was operating before stand-by mode entry.
If the PLLk was stopped before stand-by mode, it remains stopped.

Clock Monitor control

The PLL0 activity signal CLKACT is output to the Clock Monitor CLMA3 to control its operation. In case the PLL0 is inactive, supervision of its output clock f_{PL0} by CLMA3 is also deactivated.
The table below summarizes the different conditions for the Clock Monitor control.

Note The output clocks the other PLLs, except PLL0, are not supervised by a Clock Monitor.

Table 9-5 Clock Monitor 3 status control

PLL0 enable status	PLL0 stand-by control		PLL0 stand-by status	CLMA3 status
PLLS0CLKEN	PLLE0STPMK	STPREQ_0	PLLS0CLKACT	
0	X	X	0	stopped
1	0	0	1	active
		1	0	stopped
	1	X	1	active

PLLk enable/disable trigger

The PLLk can be enabled and disabled by the enable and disable trigger control bits:

- enable trigger PLLEk.PLLEkENTRG = 1 starts the PLLk
  Note that setting the enable trigger is only effective if the PLLk is inactive, i.e. if PLLSk.PLLSkCLKACT = 0.
- disable trigger PLLEk.PLLEkDISTRG = 1 stops the PLLk
  Note that setting the disable trigger is only effective if the PLLk is active, i.e. if PLLSk.PLLSkCLKACT = 1.

(1) PLL parameters

The PLL is configured by a set of parameters, derived from the control register PLLCk.

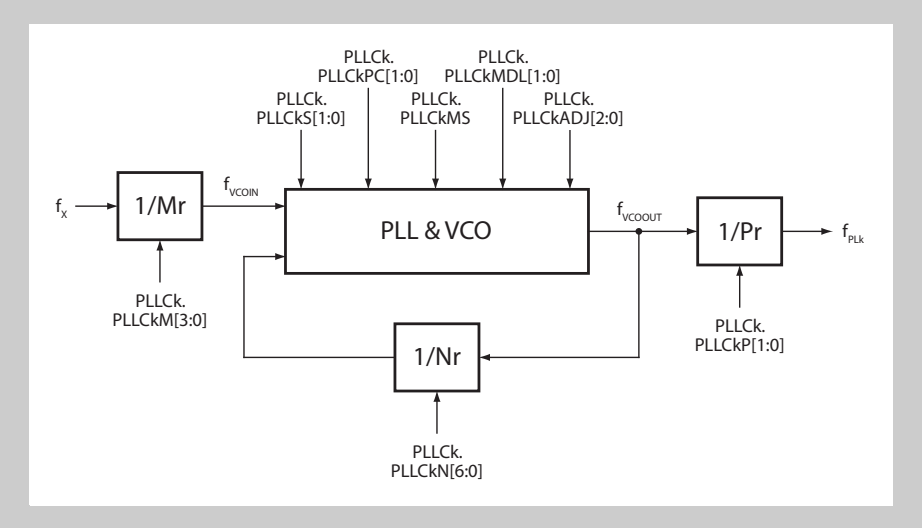


Figure 9-9 PLL circuit

PLL mode

The PLL mode is selected by

- PLLCk.PLLCkMS = 0: SSCG (Spread Spectrum Clock Generator) mode with dithered output frequency
- PLLCk.PLLCkMS = 1: PLL mode with fixed output frequency

Note The SSCG mode is only available for PLL0 and PLL2.

f_{PLk}

The frequency f_{PLk} is calculated as follows:

$$f_{PLk} = f_x \bullet \frac{Nr}{Mr \bullet Pr}$$

The values Nr, Mr and Pr are derived from PLLCk register bits:

- Nr = PLLCk.PLLCkN[6:0] + 1
- Mr depends on the PLL mode:

  - SSCG mode (PLLCk.PLLCkMS = 0): Mr = PLLCk.PLLCkM[3:0] + 1
  - PLL mode (PLLCk.PLLCkMS = 1): Mr = 1 (PLLCk.PLLCkM[3:0] = 0_H)
- Pr is determined by PLLCk.PLLCkP[1:0] according to the following table:

PLLCk.PLLCkP[1:0]	Pr	PLL output frequency f_{PL0} range
00_B	–	setting prohibited
01_B	1	100 MHz to 200 MHz
10_B	2	50 MHz to 100 MHz
11_B	4	25 MHz to 50 MHz

Frequency dithering

If frequency dithering is enabled (PLLCk.PLLCkMS = 0), additional parameters must be set in PLLCk:

- PLLCk.PLLCkPC[1:0] determines dithering mode
  - PC[1:0] = $0X_B$: fixed output frequency
  - PC[1:0] = 11_B: center spread mode, i.e. the nominal frequency is the center output frequency
- PLLCk.PLLCkMDL[1:0] determines the modulation frequency range, i.e. the time the output frequency changes from its maximum to its minimum and vice versa
- PLLCk.PLLCkADJ[2:0] determines the frequency dither range, i.e. the maximum and minimum frequency

Concerning the allowed settings of the PLL parameters refer to the description of the PLLCk register.

Note The SSCG mode is only available for PLL0 and PLL2.

PLLk input frequencies

In SSCG mode (PLLCk.PLLCkMS = 0) the VCO input frequency f_{VCOIN} must lie in the range of 1 MHz to 2 MHz. Thus the division factor Mr must be set correctly so, that

$$f_{VCOIN} = f_X/Mr = 1 \text{ MHz to } 2 \text{ MHz}.$$

PLLCk.PLLCkS[1:0] must be set in accordance with the frequency f_{VCOIN}.

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

9.4 Clock Selection

This section specifies all clock selection options for all clock domain on the three power areas of the V850E2/Fx4-H products.

The names of clock selector control and status registers are using two indices to identify the power domain and the clock domain:

(1) Clock selectors indices

m = 0: Isolated-Area-0 The clock selectors CKSC_0n control the CKSCLK_0n clocks of any clock domain within Isolated-Area-0.

m = 1: Isolated-Area-1 The clock selectors CKSC_1n control the CKSCLK_1n clocks of any clock domain within Isolated-Area-1.

m = A: Always-On-Area The clock selectors CKSC_An control the CKSCLK_An clocks of any clock domain within the Always-On-Area.

For each clock selector register a separate table is provided that informs about

- the power and the clock domain
- the clock selector register name, its address and initial value
- the domain clock name
- the clock selection options, their ID and - if applicable - clock limitations
- the availability of the clock stop mask bit CKSC_mn.STPMK_mn, that allows to determine the domain clock operation during STOP stand-by mode.

(2) Clock ID and default clock selection

Clock ID Each input clock to a clock selector CKSC_mn is identified by a unique ID. This ID has to be written to the clock selector register CKSC_mn to select the clock.

Note The ID must be written to the bits CKSC_mn.CKSCID_mn[30:0], which are located in the CKSC_mn register bits 31 to 1. Thus to select the ID clock

CKSC_mn = 2 x ID

must be written.

For details refer to the description of the CKSC_mn register in the section *"Clock Controller Registers"* of this chapter.

Default clock In the clock selection tables of the following section the default clock selection is emphasized in bold.

(3) Clock STOP mode

In STOP stand-by mode the clock output of the clock selector may be optionally stopped by setting the clock selection register bit CKSC_mn.STPMK_mn = 0. If

- STPMK_mn: available

  Clock CKSCLK_mn operation in STOP mode can be selected:

  – if CKSC_mn.STPMK_mn = 0: CKSCLK_mn is stopped in STOP mode

  – if CKSC_mn.STPMK_mn = 1: CKSCLK_mn continues in STOP mode

- STPMK_mn: not available, no STOP

  CKSC_mn.STPMK_mn is not available (STPMK_mn is fixed to 0).
  The clock CKSCLK_mn can not be stopped in STOP mode.

(4) Clock switching

Caution When changing the clock selection for a clock domain from one clock to another, make sure that both clocks are operating.

Legal ID When changing the source of a domain clock CKSCLK_mn by writing to the respective clock selector register CKSC_mn proceed as follows:

1. Write the ID of the new clock source:
   – CKSC_mn.CKSCID_mn = new_ID
2. Wait the time, as specified below under the key word *"Clock switching timing"*, before checking the status of the clock selector ID change in the next step.
3. Check that the domain clock CKSCLK_mn has changed to the new clock source, indicated by
   – CSCSTAT_mn.CLKSELID_mn[30:0] = new_ID
   – CSCSTAT_mn.CLKACT_mn = 1 (new clock active)

Caution Make sure that no accesses (by the CPU, DMA, etc.) to modules, supplied by the domain clock to be changed, are performed before activation of the new_ID is confirmed by CSCSTAT_mn.CLKACT_mn = 1.

Illegal ID If an illegal clock source ID, i.e. an ID not permitted for a certain clock domain m_n, is written to CKSC_mn.CKSCID_mn, the clock selector behaves as follows:

- clock source ID 0000_H (no clock selected) is a *legal* selection
  - illegal clock ID is not accepted, instead clock ID 0000_H is selected, thus the domain clock CKSCLK_mn is stopped
  - clock selector control register shows the illegal_ID:
    CKSC_mn.CKSCID = illegal_ID
  - clock selector status register shows ID 0000_H as inactive:
    CSCSTAT_mn.CLKSELID_mn[30:0] = 0000_H
    CSCSTAT_mn.CLKACT_mn = 0

- clock source ID 0000_H (no clock selected) is *not a legal* selection
  - illegal clock ID is not accepted and the old clock ID remains valid, thus the domain clock CKSCLK_mn does not change
  - clock selector control register shows the old ID: CKSC_mn.CKSCID = old_ID
  - clock selector status register shows the old ID as active: CSCSTAT_mn.CLKSELID_mn[30:0] = old_ID CSCSTAT_mn.CLKACT_mn = 1

Clock switching timing

The time between writing an ID to a clock selector control register CKSC_mn (for mn ≠ 000) until the status can be evaluated via the content of its status register CSCSTAT_mn is calculated as follows:
Conditions: new_ID ≠ 0000_H, old_ID ≠ 0000_H, new_ID ≠ old_ID

- if the High Speed IntOsc is active (ROSCS.ROSCSCLKEN = 1): $13 / f_{RH} + 3 / f_{old\_ID} + 3 / f_{new\_ID}$
- if the High Speed IntOsc is inactive (ROSCS.ROSCSCLKEN = 0): $13 / f_{RL} + 3 / f_{old\_ID} + 3 / f_{new\_ID}$

with

f_{RH}	=	frequency of the High Speed IntOsc (nom. 8 MHz)
f_{RL}	=	frequency of the Low Speed IntOsc (nom. 240 kHz)
f_{old_ID}	=	frequency of the old_ID clock source
f_{new_ID}	=	frequency of the new_ID clock source

ID = 0000_H

If the old_ID = 0000_H or new_ID = 0000_H, i.e. no clock selected, the respective summand ($3/f_{new\_ID}$) or ($3/f_{new\_ID}$), respectively, becomes 0 ($f \rightarrow \infty$ in above formulas).

new_ID = old_ID

If new_ID = old_ID the clock switching time becomes 0.

Clock switching time for CKSCLK_000 (CPUCLK) domain clock

If the CPU clock CPUCLK (domain clock CKSCLK_000) is changed, the CPU operation stalls during the change of the clock.
The CPU stall time can be calculated as follows:
Conditions: new_ID ≠ old_ID

- if the High Speed IntOsc is active (ROSCS.ROSCSCLKEN = 1): $4 / f_{RH} + 3 / f_{new\_ID}$
- if the High Speed IntOsc is inactive (ROSCS.ROSCSCLKEN = 0): $4 / f_{RL} + 3 / f_{new\_ID}$

9.4.1 Clock domains of Always-On-Area

(1) Clock domain AWO_2

Clock selector control register: CKSC_A02		**Power domain:**	**Always–On–Area**	
Address:	**FF42 2020$_H$**	**Clock domain:**	**AWO_2**	
Initial value:	**0000 000E$_H$**	**STPMK_A02:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 80 MHz	CKSCLK_A02	RTCA0: PCLK WDTA0: PCLK VCPC0: PCLK CLMA0: PCLK CLMA2: PCLK KR0: PCLK FOUT: PCLK AWO port control and filters: PCLK[a]
0007$_H$	**High Speed IntOsc [8 MHz] / 1**			
0008$_H$	High Speed IntOsc [8 MHz] / 2			
0009$_H$	High Speed IntOsc [8 MHz] / 4			
000A$_H$	High Speed IntOsc [8 MHz] / 8			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

a) Domain clock CKSCLK_A02 supplies the port control and filter modules of the Always-On-Area, i.e. for port groups P0, JP0.

(2) Clock domain AWO_3

Clock selector control register: CKSC_A03		**Power domain:**	**Always–On–Area**	
Address:	**FF42 2030$_H$**	**Clock domain:**	**AWO_3**	
Initial value:	**0000 000E$_H$**	**STPMK_A03:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 80 MHz	CKSCLK_A03	TAUJ0: PCLK
0007$_H$	**High Speed IntOsc [8 MHz] / 1**			
000C$_H$	MainOsc / 1			
0012$_H$	SubOsc [32 kHz]			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(3) Clock domain AWO_4

Clock selector control register: CKSC_A04		Power domain:	Always–On–Area	
Address:	FF42 2040$_H$	Clock domain:	AWO_4	
Initial value:	0000 000E$_H$	STPMK_A04:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 80 MHz	CKSCLK_A04	TAUJ1: PCLK
0007$_H$	**High Speed IntOsc [8 MHz] / 1**			
000C$_H$	MainOsc / 1 / 1			
0012$_H$	SubOsc [32 kHz]			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(4) Clock domain AWO_5

Clock selector control register: CKSC_A05		Power domain:	Always–On–Area	
Address:	FF42 2050$_H$	Clock domain:	AWO_5	
Initial value:	0000 000E$_H$	STPMK_A05:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 40 MHz	CKSCLK_A05	BURAM: PCLK
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
others than above	Setting prohibited			

(5) Clock domain AWO_6

Clock selector control register: CKSC_A06		Power domain:	Always–On–Area	
Address:	**FF42 2060$_H$**	**Clock domain:**	**AWO_6**	
Initial value:	**0000 0000$_H$**	**STPMK_A06:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 120 MHz	CKSCLK_A06	FOUT: FOUTDIVCLK
0007$_H$	High Speed IntOsc [8 MHz] / 1			
000C$_H$	MainOsc/ 1			
0012$_H$	SubOsc [32 kHz]			
0017$_H$	PLL0 / 4			
001F$_H$	PLL1 / 4			
002C$_H$	CKSCLK_000: CPU system clock			
002D$_H$	CKSCLK_A09: RTCA0 clock RTCATCKI			
002E$_H$	CKSCLK_A07: WDTA0 clock WDTACKI			
002F$_H$	CKSCLK_007: WDTA1 clock WDTACKI			
0030$_H$	CKSCLK_113: FCN0, FCN1, FCN2 clock PCLK			
0000$_H$	**No clock selected, domain clock is stopped**			
others than above	Setting prohibited			

(6) Clock domain AWO_7

Clock selector control register: CKSC_A07		Power domain:	Always–On–Area	
Address:	**FF42 2070$_H$**	**Clock domain:**	**AWO_7**	
Initial value:	**0000 0006$_H$**	**STPMK_A07:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	–	CKSCLK_A07	WDTA0: WDTACKI
0003$_H$	**Low Speed IntOsc [240 kHz] / 4**			
0005$_H$	Low Speed IntOsc [240 kHz] / 512			
others than above	Setting prohibited			

(7) Clock domain AWO_9

Clock selector control register: CKSC_A09		Power domain:	Always–On–Area	
Address:	FF42 2090$_H$	Clock domain:	AWO_9	
Initial value:	0000 0000$_H$	STPMK_A09:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0009$_H$	High Speed IntOsc [8 MHz] / 4	32 kHz up to 4.194304 MHz	CKSCLK_A09	RTCA0: RTCATCKI
000A$_H$	High Speed IntOsc [8 MHz] / 8			
000C$_H$	MainOsc / 1			
000D$_H$	MainOsc / 2			
000E$_H$	MainOsc / 4			
000F$_H$	MainOsc / 8			
0012$_H$	SubOsc [32 kHz]			
0000$_H$	**No clock selected, domain clock is stopped**			
others than above	Setting prohibited			

9.4.2 Clock domains of Isolated-Area-0

(1) Clock domain ISO0_0

Clock selector control register: CKSC_000		Power domain:	Isolated-Area-0	
Address:	FF42 6000$_H$	Clock domain:	ISO0_0	
Initial value:	0000 0074$_H$	STPMK_000:	available, must remain 0[a]	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0008$_H$	High Speed IntOsc [8 MHz] / 2	≤ 160 MHz	CKSCLK_000	CPU Subsystem: CPUCLK
0009$_H$	High Speed IntOsc [8 MHz] / 4			
000A$_H$	High Speed IntOsc [8 MHz] / 8			
000B$_H$	High Speed IntOsc [8 MHz] / 32			
000C$_H$	MainOsc / 1			
0014$_H$	PLL0 / 1			
0015$_H$	PLL0 / 2			
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
003A$_H$	**High Speed IntOsc [8 MHz] (Low Speed IntOsc [240 kHz])[b]**			
others than above	Setting prohibited			

a) CPUCLK must always be stopped in Isolated-Area-0 stand-by mode. Thus STPMK_000 must remain in its initial state 0.

b) If the High Speed IntOsc is disabled, the Low Speed IntOsc is automatically selected.

HCLK The domain clock CKSCLK_000 divided by 2 supplies also the HCLK of the following modules:

- ETHA0: HCLK
- RAM0: HCLK
- RAM1: HCLK

(2) Clock domain ISO0_5

Clock selector control register: CKSC_005		Power domain:	Isolated-Area-0	
Address:	**FF42 6050$_H$**	**Clock domain:**	**ISO0_5**	
Initial value:	**0000 00E$_H$**	**STPMK_005:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 80 MHz	CKSCLK_005	WDTA1: PCLK CLMA3: PCLK FOUTDIV: PCLK Iso0 port control and filters: PCLK[a]
0007$_H$	**High Speed IntOsc [8 MHz] / 1**			
0008$_H$	High Speed IntOsc [8 MHz] / 2			
0009$_H$	High Speed IntOsc [8 MHz] / 4			
000A$_H$	High Speed IntOsc [8 MHz] / 8			
0014$_H$	PLL0 / 1			
0015$_H$	PLL0 / 2			
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

a) Domain clock CKSCLK_005 supplies the port control and filter modules of the Isolated-Area-0, i.e. for port groups P1 to P4, P10, P11.

(3) Clock domain ISO0_6

Clock selector control register: CKSC_006		Power domain:	Isolated-Area-0	
Address:	**FF42 6060$_H$**	**Clock domain:**	**ISO0_6**	
Initial value:	**0000 000E$_H$**	**STPMK_006:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_006	TAUA0: PCLK TAUB1: PCLK TAPA0: PCLK ENCA0: PCLK DLYA: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(4) Clock domain ISO0_7

Clock selector control register: CKSC_007		**Power domain:**	**Isolated-Area-0**	
Address:	**FF42 6070$_H$**	**Clock domain:**	**ISO0_7**	
Initial value:	**0000 0006$_H$**	**STPMK_007:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	–	CKSCLK_007	WDTA1: WDTATCKI
0003$_H$	**Low Speed IntOsc [240 kHz] / 4**			
others than above	Setting prohibited			

(5) Clock domain ISO0_11

Clock selector control register: CKSC_011		Power domain:	Isolated-Area-0	
Address:	FF42 60B0$_H$	Clock domain:	ISO0_11	
Initial value:	0000 000E$_H$	STPMK_011:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_011	URTE10: PCLK LMA10: PCLK URTE11: PCLK LMA11: PCLK CNTA2: PCLK CSIG4: PCLK
000C$_H$	MainOsc			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(6) Clock domain ISO0_12

Clock selector control register: CKSC_012		Power domain:	Isolated-Area-0	
Address:	FF42 60C0$_H$	Clock domain:	ISO0_12	
Initial value:	0000 000E$_H$	STPMK_012:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_012	ADCA0: PCLK PMCA0: PCLK
000C$_H$	MainOsc			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(7) Clock domain ISO0_16

Clock selector control register: CKSC_016		Power domain:	Isolated-Area-0	
Address:	FF42 6100$_H$	Clock domain:	ISO0_16	
Initial value:	0000 000E$_H$	STPMK_016:	not available, no STOP	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_016	Iso0 port filters: DNFATCKI
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
others than above	Setting prohibited			

9.4.3 Clock domains of Isolated-Area-1

Caution If the Isolated-Area-1 is in DEEPSTOP mode, the clock selector control registers CKSC_1n and clock selector status registers CSCSTAT_1n are not accessible.
Any CPU attempt to access these registers will lead to a microcontroller deadlock.

(1) Clock domain ISO1_1

Clock selector control register: CKSC_101		**Power domain:**	**Isolated-Area-1**	
Address:	**FF42 A010$_H$**	**Clock domain:**	**ISO1_1**	
Initial value:	**0000 000E$_H$**	**STPMK_101:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0001$_H$	Low Speed IntOsc [240 kHz] / 1	≤ 80 MHz	CKSCLK_101	DCRA0: PCLK RNGA0: PCLK Iso1 port control and filters: PCLK[a]
0007$_H$	**High Speed IntOsc [8 MHz] / 1**			
0008$_H$	High Speed IntOsc [8 MHz] / 2			
0009$_H$	High Speed IntOsc [8 MHz] / 4			
000A$_H$	High Speed IntOsc [8 MHz] / 8			
0014$_H$	PLL0 / 1			
0015$_H$	PLL0 / 2			
0017$_H$	PLL0 / 4			
001A$_H$	PLL0 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

a) Domain clock CKSCLK_101 supplies the port control and filter modules of the Isolated-Area-1, i.e. for port groups P12, P13, P21, P24 to P29.

(2) Clock domain ISO1_2

Clock selector control register: CKSC_102		Power domain:	Isolated-Area-1	
Address:	$FF42\ A020_H$	Clock domain:	ISO1_2	
Initial value:	$0000\ 000E_H$	STPMK_102:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_H	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_102	FLX0: eray_bclk
0014_H	PLL0 / 1			
0015_H	PLL0 / 2			
0017_H	PLL0 / 4			
$001C_H$	PLL1 / 1			
$001D_H$	PLL1 / 2			
$001F_H$	PLL1 / 4			
0024_H	PLL2 / 1			
0025_H	PLL2 / 2			
0027_H	PLL2 / 4			
0000_H	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(3) Clock domain ISO1_3

Clock selector control register: CKSC_103		Power domain:	Isolated-Area-1	
Address:	$FF42\ A030_H$	Clock domain:	ISO1_3	
Initial value:	$0000\ 000E_H$	STPMK_103:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_H	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_103	FLX0: eray_sclk
0014_H	PLL0 / 1			
0015_H	PLL0 / 2			
0017_H	PLL0 / 4			
$001C_H$	PLL1 / 1			
$001D_H$	PLL1 / 2			
$001F_H$	PLL1 / 4			
0024_H	PLL2 / 1			
0025_H	PLL2 / 2			
0027_H	PLL2 / 4			
0000_H	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(4) Clock domain ISO1_5

Clock selector control register: CKSC_105		Power domain:	Isolated-Area-1	
Address:	$FF42\ A050_H$	Clock domain:	ISO1_5	
Initial value:	$0000\ 000E_H$	STPMK_105:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_H	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_105	TAUC4: PCLK
$000C_H$	MainOsc / 1			
$001C_H$	PLL1 / 1			
$001D_H$	PLL1 / 2			
$001F_H$	PLL1 / 4			
0022_H	PLL1 / 8			
0024_H	PLL2 / 1			
0025_H	PLL2 / 2			
0027_H	PLL2 / 4			
$002A_H$	PLL2 / 8			
0000_H	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(5) Clock domain ISO1_6

Clock selector control register: CKSC_106		Power domain:	Isolated-Area-1	
Address:	$FF42\ A060_H$	Clock domain:	ISO1_6	
Initial value:	$0000\ 000E_H$	STPMK_106:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_H	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_106	TAUC3: PCLK
$000C_H$	MainOsc / 1			
$001C_H$	PLL1 / 1			
$001D_H$	PLL1 / 2			
$001F_H$	PLL1 / 4			
0022_H	PLL1 / 8			
0024_H	PLL2 / 1			
0025_H	PLL2 / 2			
0027_H	PLL2 / 4			
$002A_H$	PLL2 / 8			
0000_H	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(6) Clock domain ISO1_7

Clock selector control register: CKSC_107		Power domain:	Isolated-Area-1	
Address:	$FF42\ A087_{0H}$	Clock domain:	ISO1_7	
Initial value:	$0000\ 000E_{H}$	STPMK_107:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_{H}	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_107	CSIG2: PCLK
$000C_{H}$	MainOsc / 1			
$001C_{H}$	PLL1 / 1			
$001D_{H}$	PLL1 / 2			
$001F_{H}$	PLL1 / 4			
0022_{H}	PLL1 / 8			
0024_{H}	PLL2 / 1			
0025_{H}	PLL2 / 2			
0027_{H}	PLL2 / 4			
$002A_{H}$	PLL2 / 8			
0000_{H}	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(7) Clock domain ISO1_8

Clock selector control register: CKSC_108		Power domain:	Isolated-Area-1	
Address:	$FF42\ A080_{H}$	Clock domain:	ISO1_8	
Initial value:	$0000\ 000E_{H}$	STPMK_108:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007_{H}	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_108	CSIG0: PCLK IICB0: PCLK
$000C_{H}$	MainOsc / 1			
$001C_{H}$	PLL1 / 1			
$001D_{H}$	PLL1 / 2			
$001F_{H}$	PLL1 / 4			
0022_{H}	PLL1 / 8			
0024_{H}	PLL2 / 1			
0025_{H}	PLL2 / 2			
0027_{H}	PLL2 / 4			
$002A_{H}$	PLL2 / 8			
0000_{H}	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(8) Clock domain ISO1_9

Clock selector control register: CKSC_109		Power domain:	Isolated-Area-1	
Address:	FF42 A090$_H$	Clock domain:	ISO1_9	
Initial value:	0000 000E$_H$	STPMK_109:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_109	CSIH0: PCLK CSIH1: PCLK CSIH2: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(9) Clock domain ISO1_11

Clock selector control register: CKSC_111		Power domain:	Isolated-Area-1	
Address:	FF42 A0B0$_H$	Clock domain:	ISO1_11	
Initial value:	0000 000E$_H$	STPMK_111:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_111	TAUB2: PCLK TAUC5: PCLK TAUC6: PCLK TAUC7: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(10) Clock domain ISO1_12

Clock selector control register: CKSC_112		Power domain:	Isolated-Area-1	
Address:	FF42 A0C0$_H$	Clock domain:	ISO1_12	
Initial value:	0000 000E$_H$	STPMK_112:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_112	URTE0: PCLK LMA0: PCLK URTE1: PCLK LMA1: PCLK CNTA0: PCLK OSTM0: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(11) Clock domain ISO1_13

Clock selector control register: CKSC_113		Power domain:	Isolated-Area-1	
Address:	FF42 A0D0$_H$	Clock domain:	ISO1_13	
Initial value:	0000 000E$_H$	STPMK_113:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_113	FCN0: PCLK FCN1: PCLK FCN2: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(12) Clock domain ISO1_14

Clock selector control register: CKSC_114		Power domain:	Isolated-Area-1	
Address:	**FF42 A0E0$_H$**	**Clock domain:**	**ISO1_14**	
Initial value:	**0000 000E$_H$**	**STPMK_114:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_114	URTE2: PCLK LMA2: PCLK URTE3: PCLK LMA3: PCLK URTE4: PCLK LMA4: PCLK URTE5: PCLK LMA5: PCLK URTE6: PCLK LMA6: PCLK URTE7: PCLK LMA7: PCLK URTE8: PCLK LMA8: PCLK URTE9: PCLK LMA9: PCLK CNTA1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(13) Clock domain ISO1_15

Clock selector control register: CKSC_115		Power domain:	Isolated-Area-1	
Address:	**FF42 A0F0$_H$**	**Clock domain:**	**ISO1_15**	
Initial value:	**0000 000E$_H$**	**STPMK_115:**	**available**	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_115	FCN3: PCLK FCN4: PCLK DCN0: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(14) Clock domain ISO1_22

Clock selector control register: CKSC_122		Power domain:	Isolated-Area-1	
Address:	FF42 A160$_H$	Clock domain:	ISO1_22	
Initial value:	0000 000E$_H$	STPMK_122:	available	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_122	ADCA1: PCLK
000C$_H$	MainOsc / 1			
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
0024$_H$	PLL2 / 1			
0025$_H$	PLL2 / 2			
0027$_H$	PLL2 / 4			
002A$_H$	PLL2 / 8			
0000$_H$	No clock selected, domain clock is stopped			
others than above	Setting prohibited			

(15) CClock domain ISO1_28

Clock selector control register: CKSC_128		Power domain:	Isolated-Area-1	
Address:	FF42 A1C0$_H$	Clock domain:	ISO1_28	
Initial value:	0000 000E$_H$	STPMK_128:	not available, no STOP	
Clock source ID	**Clock source**	**Clock limitation**	**Domain clock**	**Module: clock**
0007$_H$	**High Speed IntOsc [8 MHz] / 1**	≤ 80 MHz	CKSCLK_128	Iso1 port filters: DNFATCKI
001C$_H$	PLL1 / 1			
001D$_H$	PLL1 / 2			
001F$_H$	PLL1 / 4			
0022$_H$	PLL1 / 8			
others than above	Setting prohibited			

9.5 Clock Domain Figures

The following figures show the PBUS structures with regards to the PBUS modules in the different clock domains.

Note CPU (domain clock CKSCLK_000) accesses to modules, located in a different clock domain are passed through bus synchronizers, as shown in the following figures.
These synchronizers induce an access latency of several clock cycles.
Refer to the section *"CPU Access Bus Structures and Latencies"* in the chapter *"CPU System Function"* for details.

(1) Clock domains AWO_2 to AWO_5

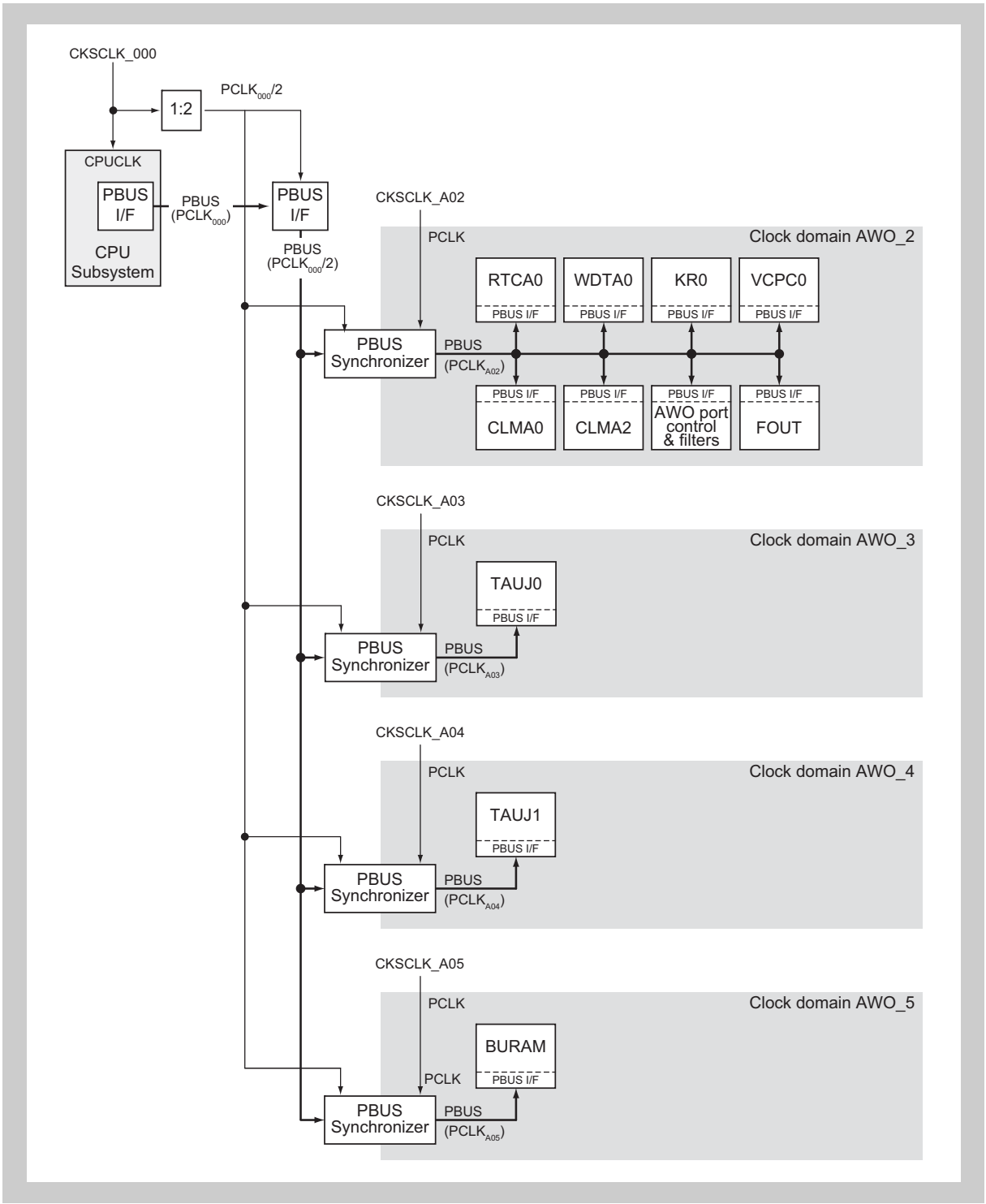


Figure 9-10 Clock domains AWO_2 to AWO_5

(2) Clock domains AWO_6, AWO_7, AWO_9

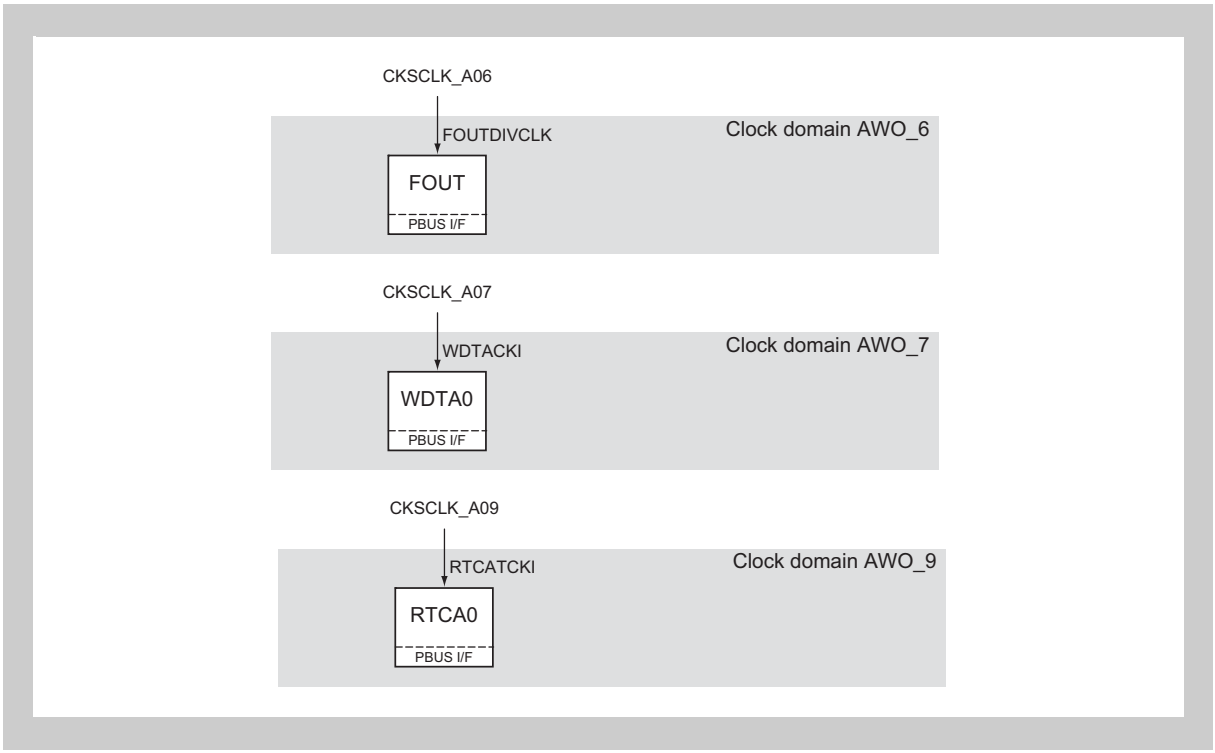


Figure 9-11 Clock domains AWO_6, AWO_7, AWO_9

(3) Clock domains ISO0_0, _5, ISO0_6, ISO0_11, ISO0_12

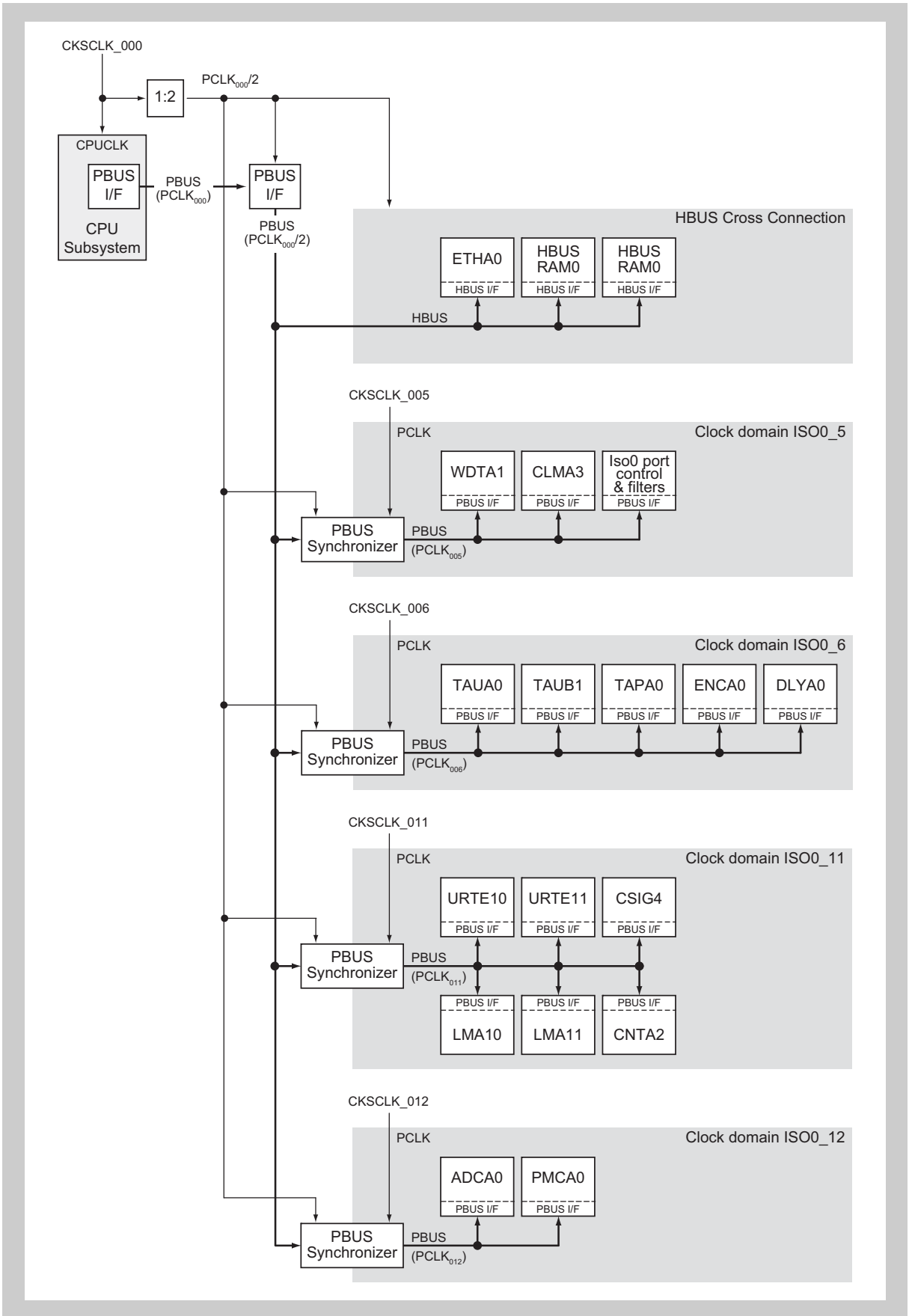


Figure 9-12 Clock domains ISO0_0, _5, ISO0_6, ISO0_11, ISO0_12

(4) Clock domains ISO0_7, ISO0_16

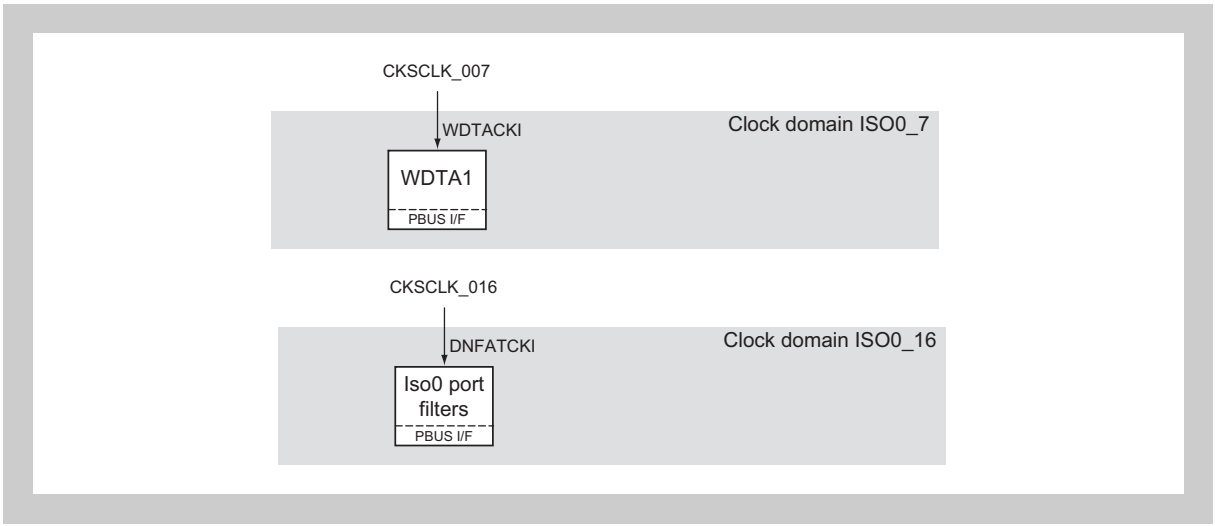


Figure 9-13 Clock domains ISO0_7, ISO0_16

(5) Clock domains ISO1_1, ISO1_2, ISO1_5 to ISO1_7

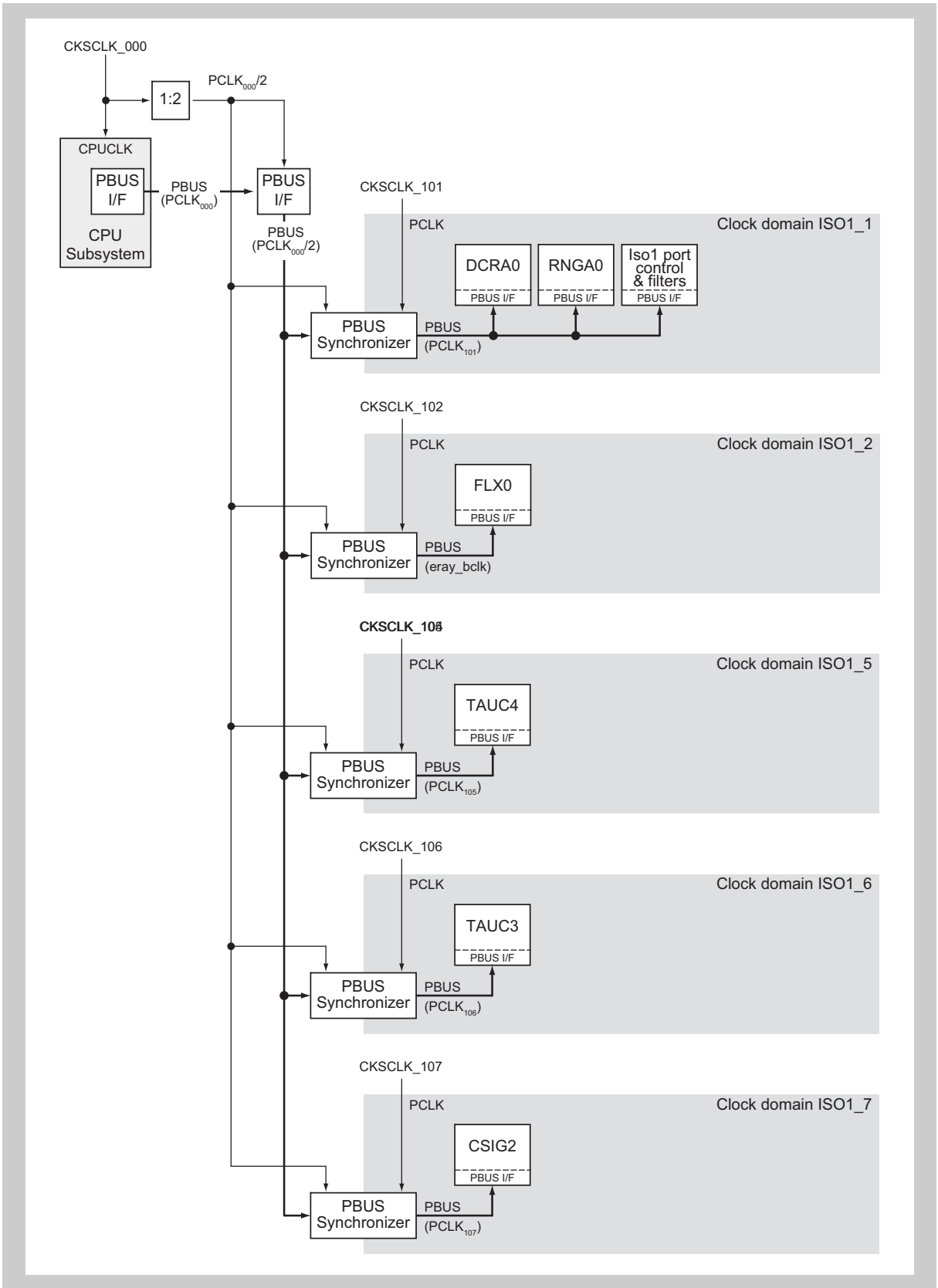


Figure 9-14 Clock domains ISO1_1, ISO1_2, ISO1_5 to ISO1_7

(6) Clock domains ISO1_8, ISO1_9, ISO1_11 to ISO1_13

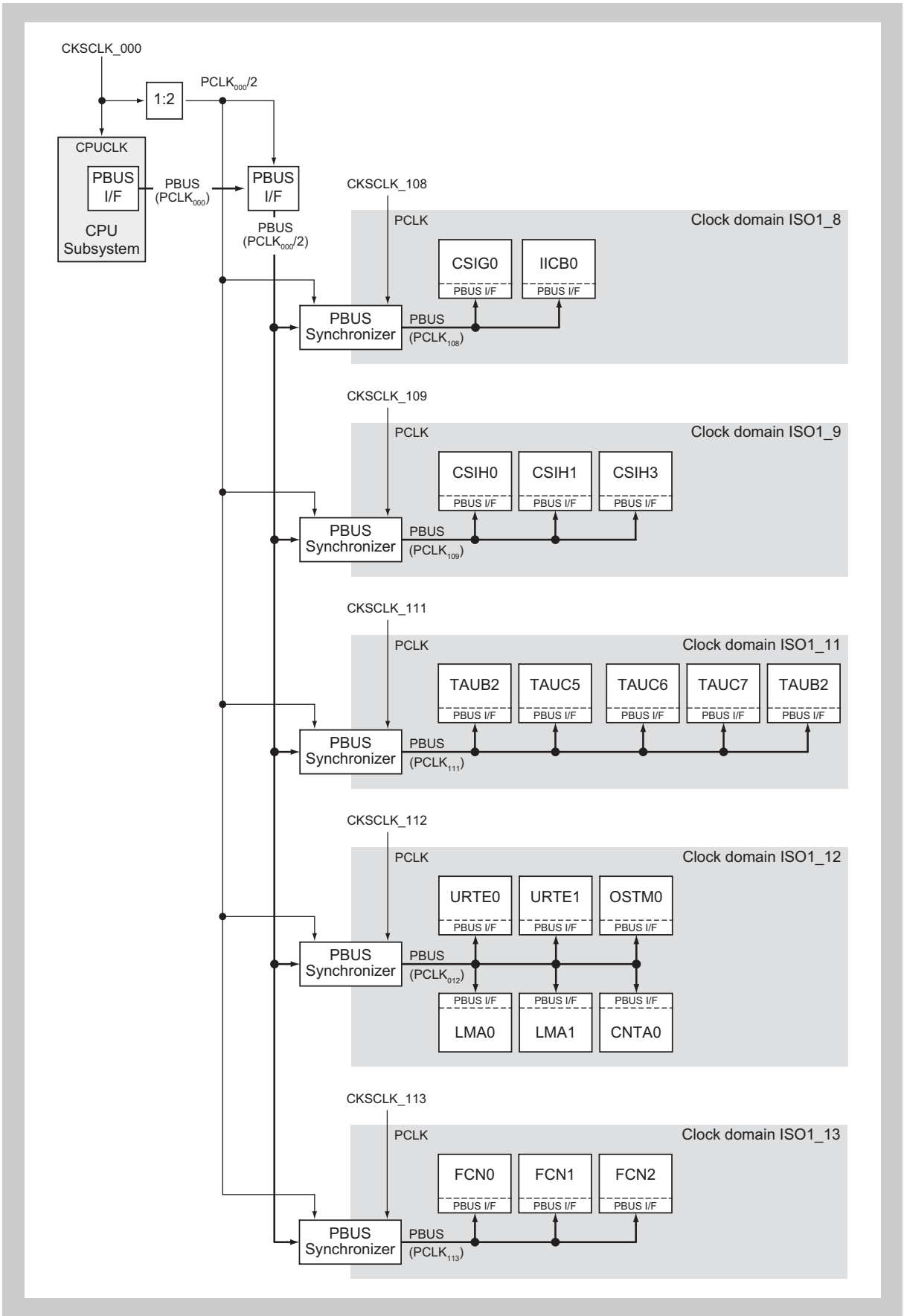


Figure 9-15 Clock domains ISO1_8, ISO1_9, ISO1_11 to ISO1_13

(7) Clock domains ISO1_14, ISO1_15, ISO1_22

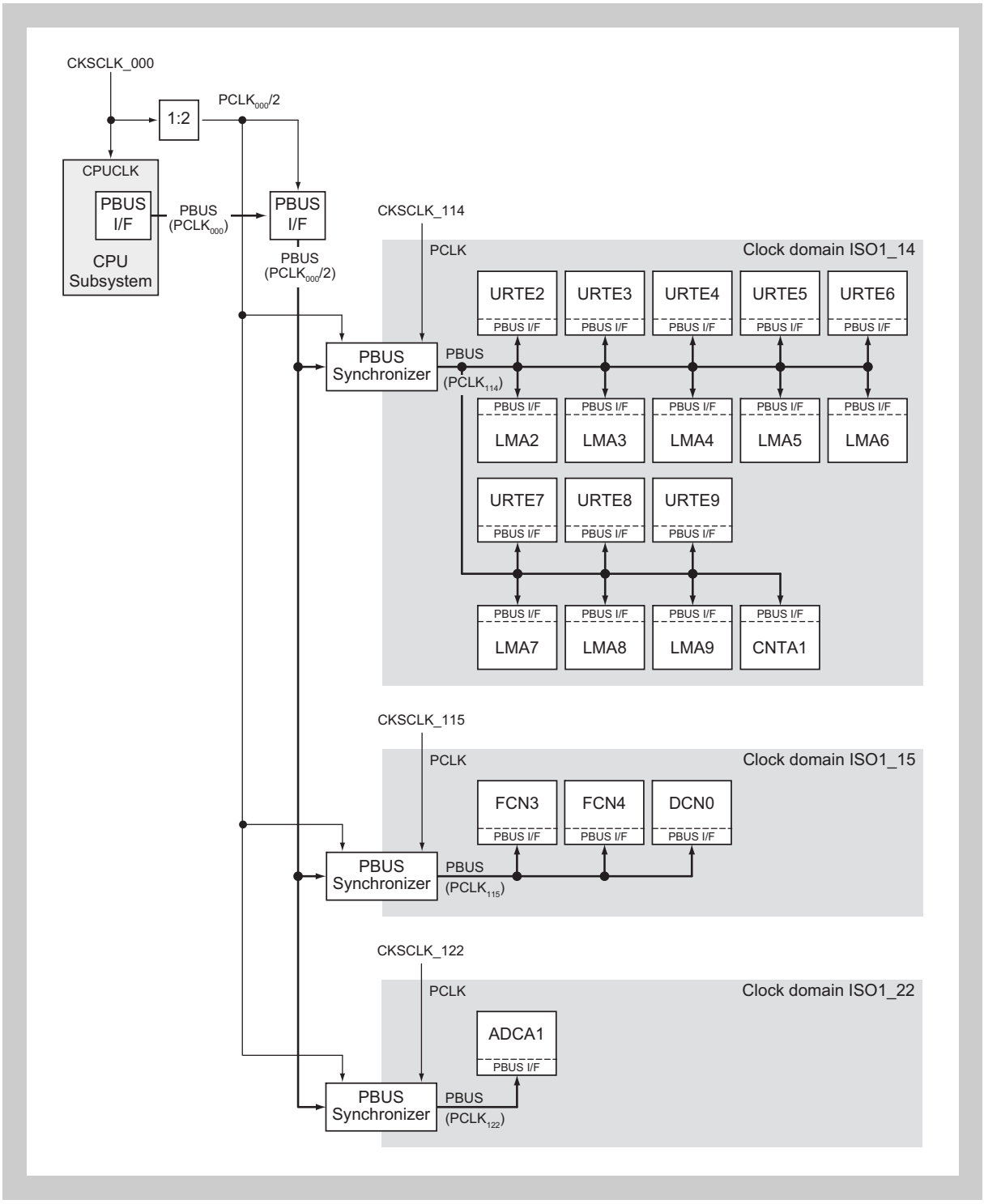


Figure 9-16 Clock domains ISO1_14, ISO1_15, ISO1_22

(8) Clock domains ISO1_3, _28

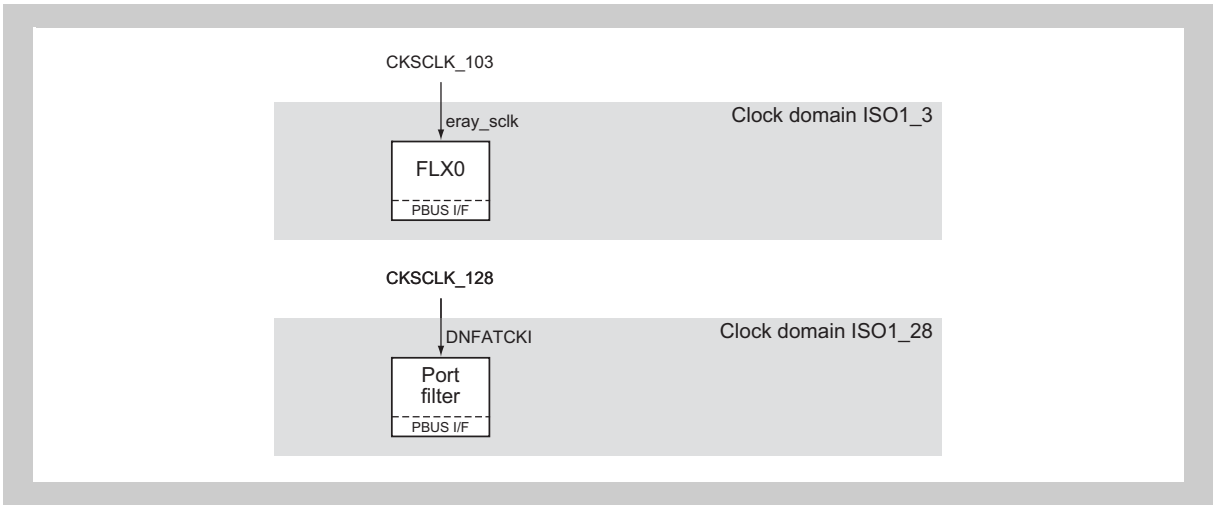


Figure 9-17 Clock domains ISO1_3, _28

9.6 Frequency Output Function (FOUT)

The Frequency Output function allows to output the clock FOUTDIVCLK = CKSCLK_A06 and its inversion as the external signals CSCXFOUT and $\overline{\text{CSCXFOUT}}$.

Further the frequency of CKSCLK_A06 can be reduced by a Clock Divider before it is output.

The figure below outlines the Frequency Output function.

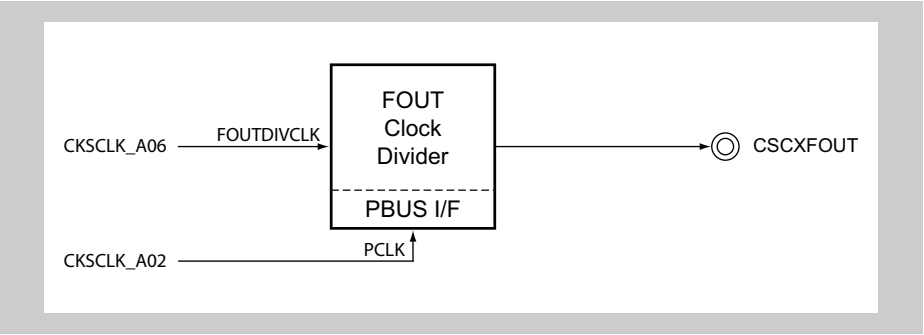


Figure 9-18 Frequency Output function

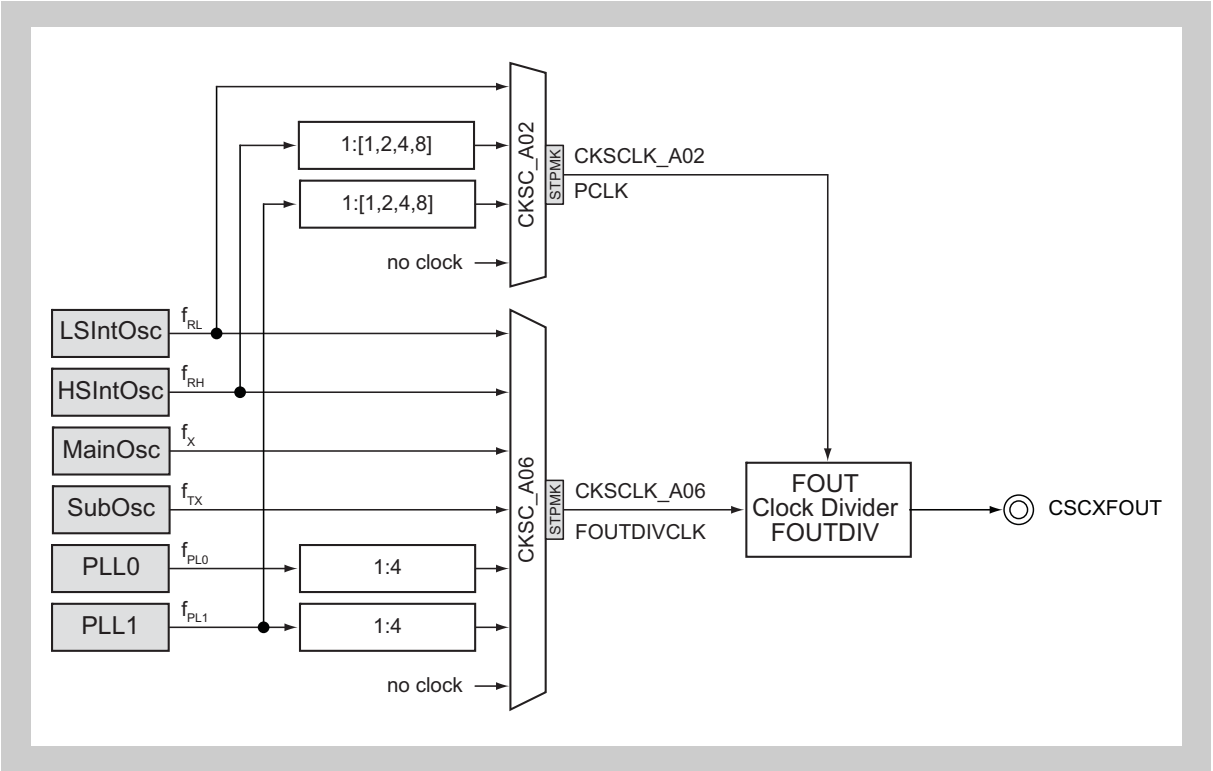


Figure 9-19 FOUT clock supply

Caution CSCXFOUT may output high frequency signals. Evaluate carefully whether the CSCXFOUT signal frequency and its harmonics causes interferences to the application system.

9.6.1 FOUT Clock Divider (FOUTDIV)

The FOUT Clock Divider divides its input clock FOUTDIVCLK by 1 to 511 and thus provides a variable CSCXFOUT clock.

The divisor value N is defined by FOUTDIV.FOUTDIV[8:0], where

$N = FOUTDIV.FOUTDIV[8:0]$ (for $FOUTDIV.FOUTDIV[8:0] > 000_H$)

The frequency $f_{CSCXFOUT}$ of the output clock is calculated by

$$f_{CSCXFOUT} = f_{FOUTDIVCLK} / N$$

(1) Duty cycle

The duty cycle of the clock divider depends on the divisor N, as shown in below figure:

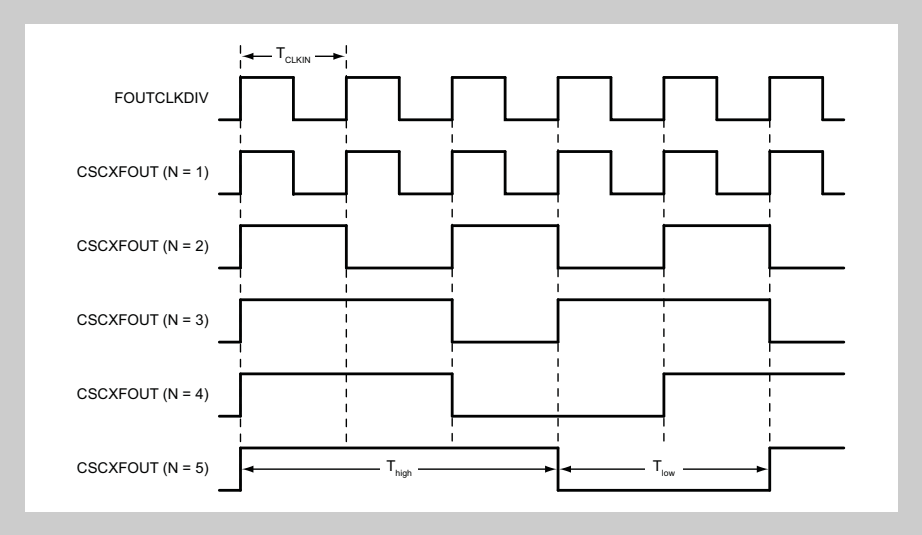


Figure 9-20 Clock divider output timing

The high and low time and the duty cycle can be calculated as follows:

Table 9-6 Divider output timing

N	Divider output		
	Low time T_{low}	High time T_{high}	Duty cycle
0	Clock output disabled		
1	$T_{CLKIN} / 2$	$T_{CLKIN} / 2$	50 %
Even	$[N / 2] \times T_{CLKIN}$	$N / 2 \times T_{CLKIN}$	50 %
Odd	$[(N - 1) / 2] \times T_{CLKIN}$	$[(N + 1) / 2] \times T_{CLKIN}$	$[(N + 1) / 2N] \times 100$

(2) Clock change delay

If the clock divisor N is changed in the FOUTDIV register, the new output clock is stable after a certain delay time:

$$\text{Delay time} = 20 * T_{PCLK} + 20 * T_{FOUTDIVCLK} + T_{CSCXFOUT}$$

with
T_{PCLK} = period of the PCLK (= CKSCLK_005) clock
$T_{FOUTDIVCLK}$ = period of the FOUTDIVCLK (= CKSCLK_A06) clock
$T_{CSCXFOUT}$ = period of the CSCXFOUT clock *before* divisor N change

(3) Clock output disable

If FOUTDIV.FOUTDIV[8:0] = 000_H, the clock output of the Clock Divider is disabled.

(4) FOUTDIV – FOUT clock divisor register

This register defines the clock divisor.

Access This register can be read/written in 16-bit units.

Address $FF81\ B000_H$

Initial Value $1FF_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FOUTDIV[8:0]								
R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-7 FOUTDIV register contents

Bit position	Bit name	Function
8 to 0	FOUT DIV[8:0]	Clock divisor N 000_H: clock output of clock divider is disabled (no CSCXFOUT output) 001_H: N = 1 002_H: N = 2 ... $1FE_H$: N = 510 $1FF_H$: N = 511 If the clock divider N is changed, the new output clock becomes effective after a certain delay time. Refer to *"Clock change delay"* above for the calculation of this delay.

9.7 Clock Monitor A (CLMA)

This section contains a generic description of the Clock Monitor A (CLMA).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

9.7.1 V850E2/Fx4-H CLMA features

Instances

This microcontroller has the following number of instances of the Clock Monitor A.

Table 9-8 Instances of CLMA

Clock Monitor A	
Instances	3
Names	CLMA0, CLMA2, CLMA3

Instances index n

Throughout this chapter, the individual instances of a Clock Monitor A are identified by the index “n” (n = 0, 2, 3), for example, CLMAnCTL0 for the control register 0 of CLMAn.

Register addresses

All CLMAn register addresses are given as address offsets from the individual base address <CLMAn_base>.
The base address <CLMAn_base> of each CLMAn is listed in the following table:

Table 9-9 Register base addresses <CLMAn_base>

CLMAn instance	<CLMAn_base> address
CLMA0	FF80 2000_H
CLMA2	FF80 4000_H
CLMA3	FF80 5000_H

Clock supply The monitored and the sampling clocks of all Clock Monitors A are listed in the following table:

Table 9-10 CLMAn clock supply

CLMAn clock	CLMAn clock	Connected to
CLMA0:		
CLMATSMP	CLMA0 sampling clock	Low speed IntOsc
CLMATMON	CLMA0 monitored clock	MainOsc
PCLK	PBUS clock	CKSCLK_A02
CLMA2:		
CLMATSMP	CLMA2 sampling clock	Low speed IntOsc
CLMATMON	CLMA2 monitored clock	High speed IntOsc
PCLK	PBUS clock	CKSCLK_A02
CLMA3:		
CLMATSMP	CLMA3 sampling clock	High speed IntOsc
CLMATMON	CLMA3 monitored clock	PLL0
PCLK	PBUS clock	CKSCLK_005

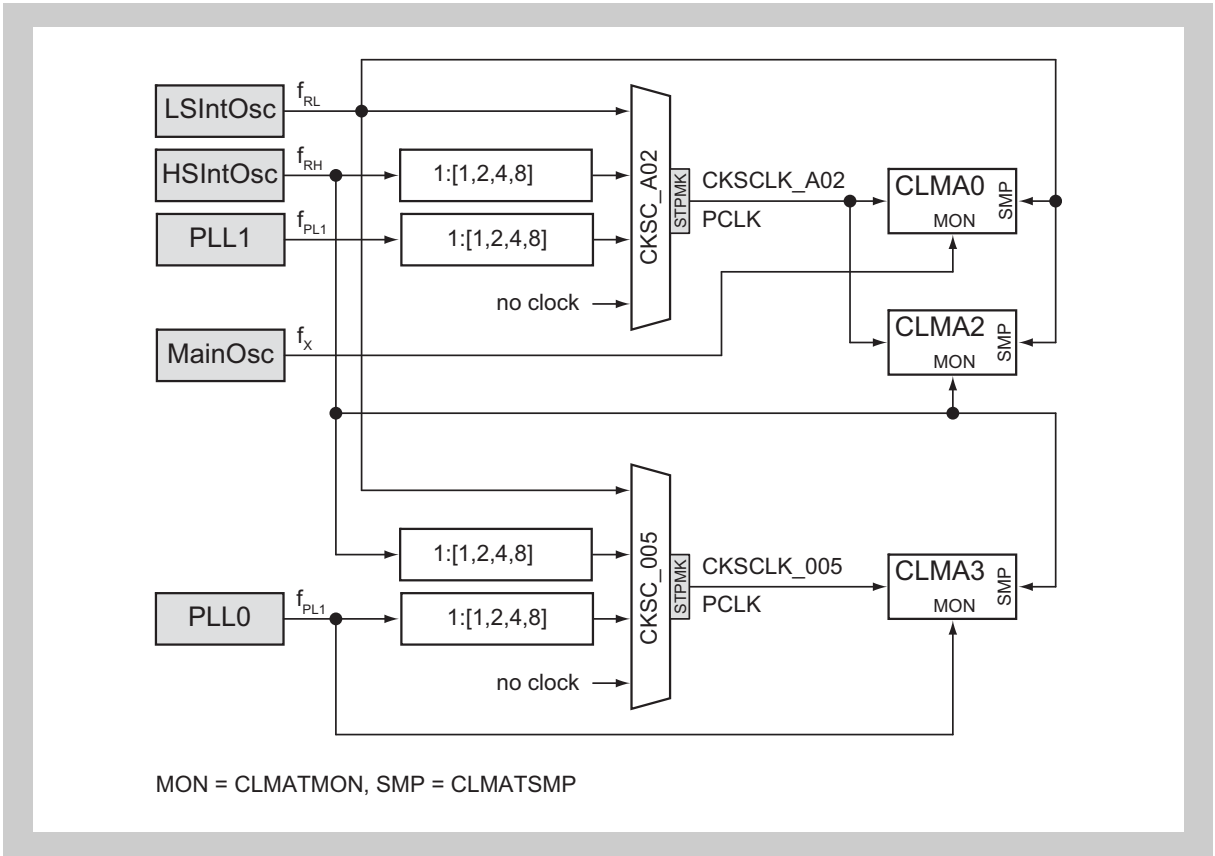


Figure 9-21 CLMA clock supply

Interrupts and reset outputs

The interrupts and reset outputs of the CLMAn are listed in the table below.

Table 9-11 CLMA interrupts and reset outputs

CLMAn signals	Function	Connected to
CLMA0:		
$\overline{\text{CLMARES}}$	Error reset	Reset Controller $\overline{\text{CLMA0RES}}$
CLMATI	Error interrupt	not connected
CLMATERR	Error signal	Motor control signals Hi-Z control[a]
CLMA2:		
$\overline{\text{CLMARES}}$	Error reset	Reset Controller $\overline{\text{CLMA2RES}}$
CLMATI	Error interrupt	not connected
CLMATERR	Error signal	Motor control signals Hi-Z control[a]
CLMA3:		
$\overline{\text{CLMARES}}$	Error reset	Reset Controller $\overline{\text{CLMA3RES}}$
CLMATI	Error interrupt	not connected
CLMATERR	Error signal	Motor control signals Hi-Z control[a]

a) Refer to *20 “Motor Control” on page 1576* for details.

CLMA H/W reset

The Clock Monitors A and their registers are initialized by the following reset signal:

Table 9-12 CLMAn reset signal

CLMAn instance	Reset signal
CLMA0, CLMA2	• Reset Controller SYSRES
CLMA3	• Reset Controller SYSRES • Stand-by Controller DPSTPWU_0 (wake-up from DEEPSTOP mode)

9.7.2 CLMA enable and start-up options

(1) CLMA enable and suspend/resume

Enable

A Clock Monitor's operation must be generally enabled by CLMAnCTL0.CLMAnCLME = 1.
General enable after reset is also controlled by the start-up option CLMATC0INI, refer to the section *"CLMA start-up options"* below.

Suspend/resume

If generally enabled, a Clock Monitor automatically

- suspends clock supervision, if the clock to be monitored is inactive
- resumes clock supervision, if the clock to be monitored is active.

For that purpose the clock generator, whose output is to be supervised, indicates its active - and stable - status by an active signal, that suspends respectively resumes the associated Clock Monitor's operation via the Clock Monitor's enable signal CLMATEN.

In case the monitored clock is stopped in STOP stand-by mode, the respective Clock Monitor suspends its operation and resumes it if the monitored clock is active and stable again.

Table 9-13 Clock Monitors start/stop signals

CLMAn instance	CLMAn enable signal	Clock generator active signal	Comment
CLMA0	CLMATEN	MainOsc MOSCSCLKACT	CLMA0 operation suspends/ resumes, if MainOsc clock f_X is inactive/active.
CLMA2	CLMATEN	High Speed IntOsc ROSCSCLKACT	CLMA2 operation suspends/ resumes, if High Speed IntOsc clock f_{RH} is inactive/ active.
CLMA3	CLMATEN	PLL0 PLLS0CLKACT	CLMA3 operation suspends/ resumes, if PLL0 clock f_{PL0} is inactive/active.

(2) CLMA start-up options

The start-up options determine the start-up configuration of the CLMA after reset release. A description of the start-up options is given in the following table.

Table 9-14 CLMA start-up options

Start-up option	Function	Description	Connected to
CLMATC0INI	Defines whether CLMAn is automatically enabled or stays disabled after a reset	Specifies the initial value of CLMAnCTL0.CLMAnCLME: 0: CLMA is disabled 1: CLMA is enabled	0
CLMATC1INI	Specifies the error indication method	Specifies the initial value of CLMAnCTL1.CLMAnOSEL: 0: Reset request $\overline{\text{CLMATRES}}$ on error 1: Interrupt request CLMATI on error	0
CLMATCSEL	Defines whether the default lower and upper thresholds are supplied by CLMATCLINI[11:0]/ CLMATCHINI[11:0]	Specifies how the threshold registers CLMAnCMPL/CLMAnCMPH are set after reset release: 0: set to default reset values 1: set by CLMATCLINI[11:0] and CLMATCHINI[11:0]	0
CLMATCHINI [11:0]	Sets the initial value of the threshold register CLMAnCMPH	Only effective if CLMATCSEL = 1	000_H
CLMATCLINI [11:0]	Sets the initial value of the threshold register CLMAnCMPL	Only effective if CLMATCSEL = 1	000_H

9.7.3 Functional Overview

The Clock Monitor CLMAn indicates an abnormal frequency of the monitored clock.

Features summary It has the following features:

- continuous monitoring of the frequency of an input clock CLMATMON by using a sampling clock CLMATSMP
- indication of abnormal clock frequencies by the following means:
  - output of a reset signal, or
  - output of an error signal in combination with the generation of an interrupt request
- configuration after reset is based on start-up options

Note Once enabled, the CLMAn can neither be configured nor stopped by software. Only a reset can stop the CLMAn.

The following figure shows the main components of the Clock Monitor.

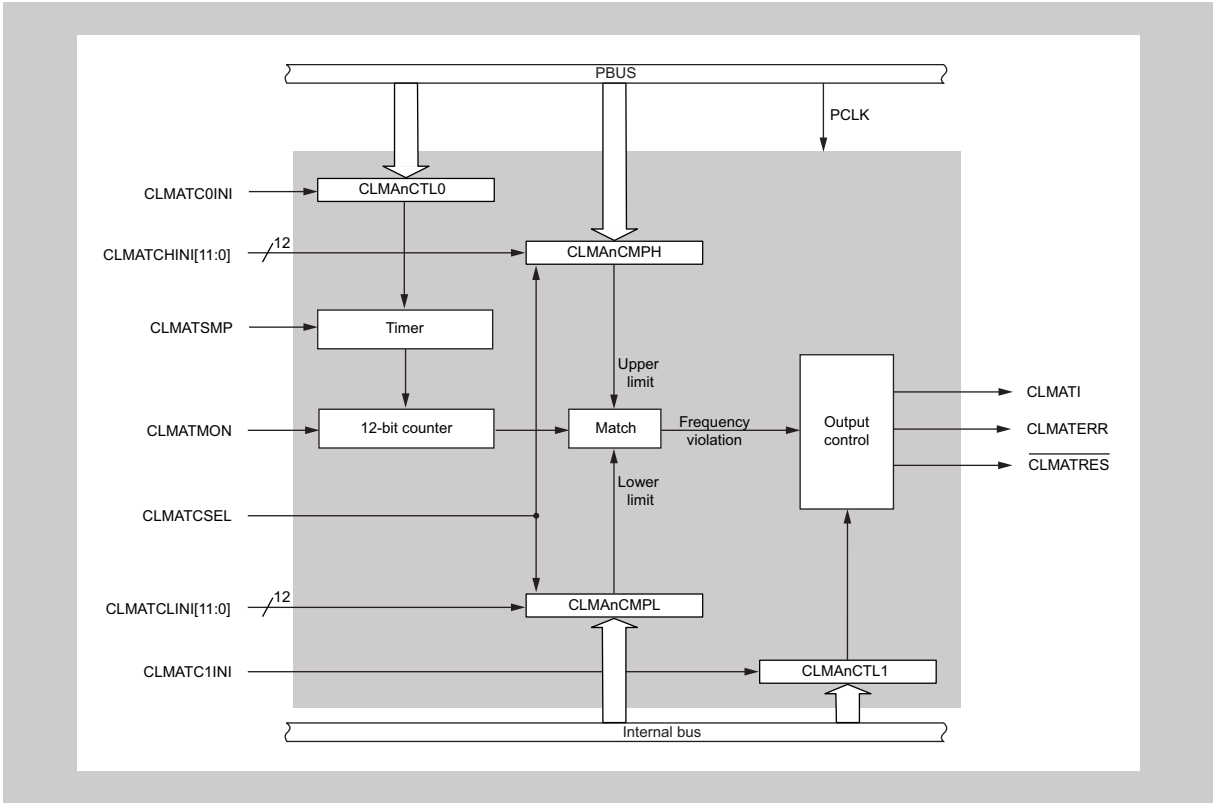


Figure 9-22 Block diagram of the Clock Monitor A

9.7.4 Functional Description

The Clock Monitor CLMAn is used to ensure that the frequency of a clock (CLMATMON) stays between certain limits.

(1) Detection of abnormal clock frequencies

Method

1. CLMAn counts the rising edges of the monitored clock CLMATMON within 16 cycles of the sampling clock CLMATSMP and then compares the counter with the configured thresholds:
   – CLMAnCMPL.CLMAnCMPL[11:0] defines the lower threshold.
   – CLMAnCMPH.CLMAnCMPH[11:0] defines the upper threshold.
2. When CLMATMON stops or its frequency is too low, the counter falls below CLMAnCMPL.CLMAnCMPL[11:0].
3. When the frequency of CLMATMON is too high, the counter exceeds CLMAnCMPH.CLMAnCMPH[11:0].

In both cases, CLMAn indicates an abnormal clock frequency as described in *2 “Indication of abnormal clock frequency” on page 547* .

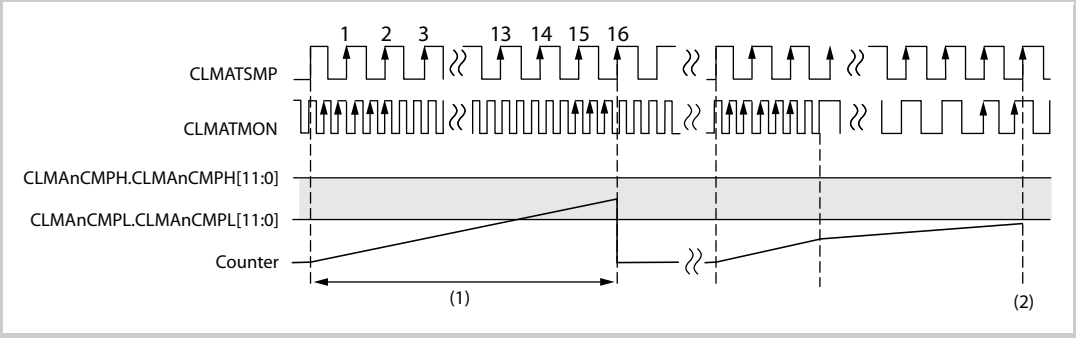


Figure 9-23 Example: $f_{CLMATMON}$ is too low

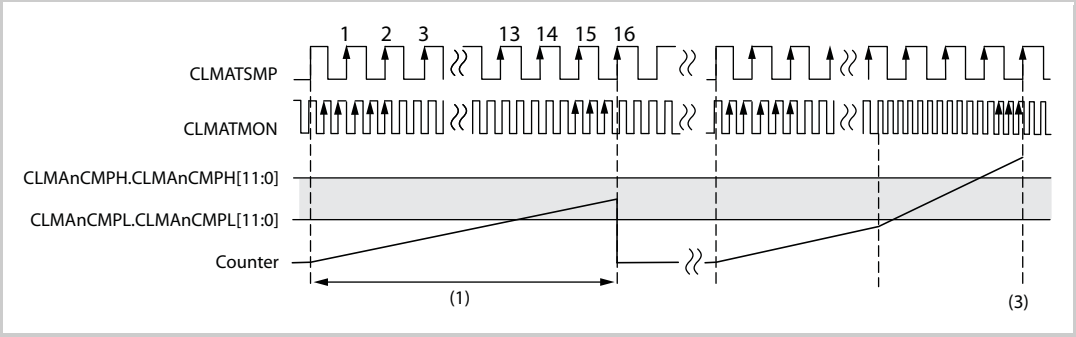


Figure 9-24 Example: $f_{CLMATMON}$ is too high

Note When $f_{CLMATMON}$ changes within the sampling interval, the counter might be within the valid range although $f_{CLMATMON}$ became too high/low.

The abnormal $f_{CLMATMON}$ is detected one sampling interval later.

(a) Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0]

The compare registers CLMAnCMPL and CLMAnCMPH are configured with the minimum and maximum number of clock cycles of CLMATMON that are assumed to be valid within 16 cycles of the sampling clock CLMATSMP. The expected number of clock cycles is denoted by N.

$$\frac{16}{f_{CLMATSMP}} = \frac{N}{f_{CLMATMON}}$$

$$N = \frac{f_{CLMATMON}}{f_{CLMATSMP}} \times 16$$

Considering the allowed frequency deviations of CLMATMON and CLMATSMP, the threshold values can be calculated by the following formulas:

$$\text{Lower threshold} = N_{min}$$

$$= \text{round down } \left(\frac{f_{CLMATMON(min)}}{f_{CLMATSMP(max)}} \times 16 - 1\right)$$

$$\text{Upper threshold} = N_{max}$$

$$= \text{round up } \left(\frac{f_{CLMATMON(max)}}{f_{CLMATSMP(min)}} \times 16 + 1\right)$$

Example For $f_{CLMATSMP}$ = 240 kHz (± 8%) and $f_{CLMATMON}$ = 16 MHz (± 5%) the recommended threshold values are the following:

$$N_{min} = [(15.2\ MHz / 259.2\ kHz) \times 16] - 1$$

$$= 937.27$$

$$CLMAnCMPL = 937 = 03A9_H$$

$$N_{max} = [(16.8\ MHz / 220.8\ kHz) \times 16] + 1$$

$$= 1218.38$$

$$CLMAnCMPH = 1219 = 04C3_H$$

Minimum thresholds The following restrictions must be taken into account:

- $CLMAnCMPL \geq 0001_H$
- $CLMAnCMPH \geq CLMAnCMPL + 0003_H$

(b) Definition of the initial value input of the threshold registers

CLMATCSEL = 1 When CLMATCSEL is active, the initial values of the threshold registers are set by the following input signals:

- CLMAnCMPL[11:0] is set by CLMATCLINI[11:0].
- CLMAnCMPH[11:0] is set by CLMATCHINI[11:0].

CLMATCSEL = 0 If CLMATCSEL is inactive, the initial values of the threshold registers are set in such a way, that the maximum frequency deviation of the monitored clock is allowed:

- CLMAnCMPL[11:0] = 0001_H
- CLMAnCMPH[11:0] = $03FF_H$

Note Refer to the above section *"CLMA enable and start-up options"* about the setting of the CLMATCSEL signal.

(2) Indication of abnormal clock frequency

In case an abnormal frequency is detected, following indications are generated:

Table 9-15 Abnormal frequency indications

CLMAnCTL1. CLMAnOSEL	Monitored clock frequency $f_{CLMATMON}$ is	
	too low	too high
1	Interrupt: assertion of CLMATI Error: assertion of CLMATERR	Reset: assertion of $\overline{\text{CLMATRES}}$
0	Reset: assertion of $\overline{\text{CLMATRES}}$	

Note that if a too high frequency is detected the reset CLMATRES is always generated, while in case of too low frequency detection the control bit CLMAnCTL1.CLMAnOSEL determines the generated indicator.

Note The initial value of CLMAnCTL1.CLMAnOSEL is defined by the input signal CLMATC1INI. Refer to the section *"CLMA enable and start-up options"* above.

(3) Enabling and disabling CLMAn

Enabling CLMAn CLMAn is enabled by CLMAnCTL0.CLMAnCLME = 1.

The initial value of CLMAnCTL0 is defined by the input signal CLMATC0INI, refer to the section *"CLMA enable and start-up options"* above.
This allows to define whether CLMAn is automatically enabled after a reset or stays disabled.

Table 9-16 Initial value of CLMAnCTL0

CLMATC0INI	Initial value of CLMAnCTL0	Function
Low	00_H	Disable CLMAn
High	01_H	Enable CLMAn

Refer to the section *"CLMA enable and start-up options"* above for information about CLMATC0INI.

Disabling CLMAn CLMAn can only be disabled by a reset, not by writing to CLMAnCTL0.

Note Once enabled, the CLMAn can neither be configured nor stopped by software, except by a reset.

(4) Suspend/resume control

The operation is suspended respectively resumed under control of the CLMATEN signal.
Refer to the section *"CLMA enable and start-up options"* above.

9.7.5 Clock Monitor registers

This section contains a description of all registers of the Clock Monitor.

(1) Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Following Clock Monitor registers feature this special write protection:

- CLMAn control register 0 CLMAnCTL0

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

(2) Clock Monitor registers overview

The Clock Monitor is controlled and operated by the following registers.

Table 9-17 Clock Monitor registers overview

Register name	Shortcut	Address
CLMAn control register 0	CLMAnCTL0	<CLMAn_base> + 00_H
CLMAn control register 1	CLMAnCTL1	<CLMAn_base> + 04_H
CLMAn comparison register L	CLMAnCMPL	<CLMAn_base> + 08_H
CLMAn comparison register H	CLMAnCMPH	<CLMAn_base> + $0C_H$
CLMAn emulation register 0	CLMAnEMU0	<CLMAn_base> +18_H

<CLMAn_base> The base addresses <CLMAn_base> of the CLMAn are defined in the first part of section *"Clock Monitor A (CLMA)"* in this chapter under the key word *"Register addresses"*.

(3) CLMAnCTL0 – CLMAn control register 0

This register is used to enable the clock monitor CLMAn.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register CLMAnPCMD.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 8-bit units.

Address <CLMAn_base> + 00_H

Initial Value Depends on start-up option CLMATC0INI. Refer to the section *"CLMA enable and start-up options"* above.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CLMAn CLME
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-18 CLMAnCTL0 register contents

Bit position	Bit name	Function
0	CLMAnCLME	Enables/disables the clock monitor: 0: Disable CLMAn 1: Enable CLMAn This bit can only be cleared by a reset if not automatically enabled by CLMATC0INI.

(4) CLMAnCTL1 – CLMAn control register 1

This register specifies the abnormal frequency detection indicator generation output when the frequency of the monitored clock CLMATMON is too low.

Access This register can be read/written in 8-bit units.
It can only be written when CLMAn is disabled (CLMAnCTL0.CLMAnCLME = 0).

Address <CLMAn_base> + 04_H

Initial Value Depends on start-up option CLMATC1INI. Refer to the section *"CLMA enable and start-up options"* above.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CLMAn OSEL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-19 CLMAnCTL1 register contents

Bit position	Bit name	Function
0	CLMAnOSEL	Specifies the indicator signals that are generated when the frequency of CLMATMON is too low: 0: reset $\overline{\text{CLMATRES}}$ 1: interrupt CLMATI and error CLMATERR

(5) CLMAnCMPL – CLMAn comparison register L

This register specifies the *lower* frequency limit.

Refer to *a "Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0]" on page 546* for details.

Access This register can be read/written in 16-bit units.
It can only be written, when CLMAn is disabled (CLMAnCTL0.CLMAnCLME = 0).

Address <CLMAn_base> + 08_H

Initial Value 0001_H
When CLMATCSEL is active, the initial value is set by the input signals CLMATCLINI[11:0]. Refer to the section *"CLMA enable and start-up options"* above.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CLMAnCMPL[11:0]											
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-20 CLMAnCMPL register contents

Bit position	Bit name	Function
11 to 0	CLMAnCMPL[11:0]	Specifies the lower threshold • The recommended value is $(f_{CLMATMON(min)} \times 16) / f_{CLMATSMP(max)}$ • The minimum value is 0001_H.

(6) CLMAnCMPH – CLMAn comparison register H

This register specifies the *upper* frequency limit.

Refer to *a "Calculation of thresholds CLMAnCMPL.CLMAnCMPL[11:0] and CLMAnCMPH.CLMAnCMPH[11:0]" on page 546* for details.

Access This register can be read/written in 16-bit units.
It can only be written, when CLMAn is disabled (CLMAnCTL0.CLMAnCLME = 0).

Address <CLMAn_base> + $0C_H$

Initial Value $03FF_H$
When CLMATCSEL is active, the initial value is set by the input signals CLMATCHINI[11:0]. Refer to the section *"CLMA enable and start-up options"* above.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CLMAnCMPH[11:0]											
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-21 CLMAnCMPH register contents

Bit position	Bit name	Function
11 to 0	CLMAnCMPH[11:0]	Specifies the upper threshold • The recommended value is ($f_{CLMATMON(max)}$ x 16) / $f_{CLMATSMP(min)}$ • The minimum value is CLMAnCMPL + 0003_H.

(7) CLMAnEMU0 – CLMAn emulation register 0

This register provides bits to emulate a frequency deviation error while the microcontroller is set in break mode during debugging.

Access This register can be read/written in 8-bit units.

Address <CLMAn_base> + 18_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	CLMAn SLFST	CLMAn SLSLW
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-22 CLMAnEMU0 register contents

Bit position	Bit name	Function
0	CLMAnSLFST	Specifies whether $f_{CLMATMON}$ is assumed to be too high during emulation: 0: CLMATMON is within the normal frequency range 1: CLMATMON is too fast
1	CLMAnSLSLW	Specifies whether $f_{CLMATMON}$ is assumed to be too low during emulation: 0: CLMATMON is within the normal frequency range 1: CLMATMON is too slow

Caution It is prohibited to emulate a too low and too high CLMATMON at the same time. Thus CLMAnEMU0 must not be set to 03_H.

9.8 Clock Controller Registers

This section contains a description of all registers of the Clock Controller.

9.8.1 Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Following Clock Controller registers feature this special write protection:

- MainOsc enable register MOSCE
- SubOsc enable register SOSCE
- High Speed IntOsc enable register ROSCE
- PLL enable registers PLLEk
- Clock selector control registers CKSC_mn

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

9.8.2 Clock Controller registers overview

The Clock Controller is controlled and operated by the following registers:

Table 9-23 Clock Controller registers overview (1/2)

Register name	Shortcut	Address
Clock generators registers:		
MainOsc enable register	MOSCE	FF42 1010$_H$
MainOsc status register	MOSCS	FF42 1014$_H$
MainOsc control register	MOSCC	FF42 1018$_H$
MainOsc stabilization time register	MOSCST	FF42 101C$_H$
SubOsc enable register	SOSCE	FF42 1020$_H$
SubOsc status register	SOSCS	FF42 1024$_H$
SubOsc stabilization time register	SOSCST	FF42 102C$_H$
High Speed IntOsc enable register	ROSCE	FF42 1000$_H$
High Speed IntOsc status register	ROSCS	FF42 1004$_H$
PLL0 enable register	PLLE0	FF42 5000$_H$
PLL0 status register	PLLS0	FF42 5004$_H$
PLL0 control register	PLLC0	FF42 5008$_H$
PLL0 stabilization time register	PLLST0	FF42 500C$_H$
PLL1 enable register	PLLE1	FF42 5010$_H$
PLL1 status register	PLLS1	FF42 5014$_H$
PLL1 control register	PLLC1	FF42 5018$_H$
PLL1 stabilization time register	PLLST1	FF42 501C$_H$
PLL2 enable register	PLLE2	FF42 5020$_H$

Table 9-23 Clock Controller registers overview (2/2)

Register name	Shortcut	Address
PLL2 status register	PLLS2	FF42 5024_H
PLL2 control register	PLLC2	FF42 5028_H
PLL2 stabilization time register	PLLST2	FF42 $502C_H$
Always-On-Area clock selector registers:		
Clock selector control registers for Always_On_Area	CKSC_An	FF42 2000_H + n x 16
Clock selector status registers for Always_On_Area	CSCSTAT_An	FF42 2004_H + n x 16
Isolated-Area-0 clock selector registers:		
Clock selector control registers for Isolated-Area-0	CKSC_0n	FF42 6000_H + n x 16
Clock selector status registers for Isolated-Area-0	CSCSTAT_0n	FF42 6004_H + n x 16
Isolated-Area-1 clock selector registers:		
Clock selector control registers for Isolated-Area-1	CKSC_1n	FF42 $A000_H$ + n x 16
Clock selector status registers for Isolated-Area-1	CSCSTAT_1n	FF42 $A004_H$ + n x 16

9.8.3 Clock generators registers

(1) MOSCE - MainOsc enable register

This register is used to start and stop the MainOsc and to specify its operation during stand-by modes.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 1010_H

Initial Value 0000 0004_H. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	MOSCESTPMK	MOSCEDISTRG	MOSCEENTRG
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-24 MOSCE register contents

Bit position	Bit name	Function
2	MOSCE STPMK	MainOsc stop request mask 0: stop request not masked 1: stop request masked If the MainOsc stop request is masked (MOSCESTPMK = 1) a stop request does not change the status of the MainOSC. If MOSCESTPMSK = 0, the MainOsc is stopped in stand-by mode. Upon wake-up from stand-by mode, the MainOsc returns to its status as before stand-by mode, i.e. continues operation when it was operating or remains stopped, when it was stopped before. **Note:** The MainOsc can also be activated after stand-by wake-up, independent of its status before stand-by mode. This features is controlled by the oscillator wake-up mask register OSCWUFMSK. Refer to section *"Wake-up"* in the chapter *"Stand-by Controller"* for details.
1	MOSCE DISTRG	MainOsc disable trigger 0: no function 1: stops MainOsc Stopping the MainOsc by MOSCEDISTRG = 1 is only possible, if the MainOsc is active, i.e. MOSCS.MOSCSCLKACT = 1. Reading of this bit returns always 0.
0	MOSCE ENTRG	MainOsc enable trigger 0: no function 1: starts MainOsc Starting the MainOsc by MOSCEENTRG = 1 is only possible, if the MainOsc is inactive, i.e. MOSCS.MOSCSCLKACT = 0. Reading of this bit returns always 0.

(2) MOSCS - MainOsc status register

This register provides various status information about the MainOsc status.

Access This register can be read in 32-bit units.

Address FF42 1014_H

Initial Value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	MOSCS CLK EN	MOSCS CLK ACT	MOSCS CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-25 MOSCS register contents

Bit position	Bit name	Function
2	MOSCS CLKEN	MainOsc enable status 0: MainOsc is disabled 1: MainOsc is enabled
1	MOSCS CLKACT	MainOsc activation status 0: MainOsc is inactive 1: MainOsc is active
0	MOSCS CLKSTAB	MainOsc stabilization status 0: MainOsc is unstable 1: MainOsc is stable

(3) MOSCC - MainOsc control register

This register is used to specify several options of the MainOsc start-up behaviour.

This register can only be written, if the MainOsc is disabled (MOSCS.MOSCSCLKEN = 0).

Access This register can be read/written in 32-bit units.

Address FF42 1018_H

Initial Value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	MOSCC SHT STBY	MOSCC AMPSEL[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-26 MOSCC register contents

Bit position	Bit name	Function
2	MOSCC SHTSTBY	Short stabilization time mode 0: normal stabilization time mode: MOSCCAMPSEL[1:0] determines the MainOsc amplification gain 1: short stabilization time mode: MainOsc amplification gain is maximum **Note:** MOSCCSHTSTBY = 1 is only effective during the MainOsc stabilization time. After the MainOsc has become active (MOSCS.MOSCSCLKACT = 1), normal stabilization time is used and thus MOSCCAMPSEL[1:0] determines the MainOsc amplification gain.
1 to 0	MOSCC AMPSEL[1:0]	MainOsc amplification gain selection **MOSCC AMPSEL[1:0]** / **Amplification gain[a]** 00_B high 01_B medium 10_B low 11_B very low MOSCCAMPSEL[1:0] determines the MainOsc amplification gain in normal stabilization time mode (MOSCCSHTSTBY = 0). In short stabilization time mode (MOSCCSHTSTBY = 1), MOSCCAMPSEL[1:0] have no effect.

a) For detailed information about the electrical conditions of the different amplification gain settings refer to the Electrical Target Specification.

Caution During assertion of the external $\overline{\text{RESET}}$ the MainOsc is always operating with its maximum amplification gain.
Refer to the descrition of the external $\overline{\text{RESET}}$ in the *"Reset Controller"* chapter for details.

(4) MOSCST - MainOsc stabilization time register

This register determines the MainOsc stabilization time.

This register can only be written, if the MainOsc is disabled (MOSCS.MOSCSCLKEN = 0).

Access This register can be read/written in 32-bit units.

Address FF42 101C$_H$

Initial Value 0000 0000$_H$. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	MOST[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-27 MOSCST register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>MOST[3:0]</td><td>MainOsc stabilization time setting
Per default the MainOsc stabilization counter is operating with the High Speed IntOsc.
If the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0), the stabilization counter clock is automatically changed to the Low Speed IntOsc.
<table>
<tr><th rowspan="2">MOST[3:0]</th><th colspan="2">MainOsc stabilization time</th></tr>
<tr><th>High Speed IntOsc enabled (ROSCS.ROSCSCLKEN = 1)</th><th>High Speed IntOsc disabled (ROSCS.ROSCSCLKEN = 0)</th></tr>
<tr><td>0000_B</td><td>2^2 / 8 MHz = 0.5 µs</td><td>2^2 / 240 kHz = 17.7 µs</td></tr>
<tr><td>0001_B</td><td>2^3 / 8 MHz = 1 µs</td><td>2^3 / 240 kHz = 33.3 µs</td></tr>
<tr><td>0010_B</td><td>2^4 / 8 MHz = 2 µs</td><td>2^4 / 240 kHz = 66.7 µs</td></tr>
<tr><td>0011_B</td><td>2^5 / 8 MHz = 4 µs</td><td>2^5 / 240 kHz = 133 µs</td></tr>
<tr><td>0100_B</td><td>2^6 / 8 MHz = 8 µs</td><td>2^6 / 240 kHz = 267 µs</td></tr>
<tr><td>0101_B</td><td>2^7 / 8 MHz = 16 µs</td><td>2^7 / 240 kHz = 533 µs</td></tr>
<tr><td>0110_B</td><td>2^8 / 8 MHz = 32 µs</td><td>2^8 / 240 kHz = 1.067 ms</td></tr>
<tr><td>0111_B</td><td>2^9 / 8 MHz = 64 µs</td><td>2^9 / 240 kHz = 2.133 ms</td></tr>
<tr><td>1000_B</td><td>2^{10} / 8 MHz = 128 µs</td><td>2^{10} / 240 kHz = 4.267 ms</td></tr>
<tr><td>1001_B</td><td>2^{11} / 8 MHz = 256 µs</td><td>2^{11} / 240 kHz = 8.533 ms</td></tr>
<tr><td>1010_B</td><td>2^{12} / 8 MHz = 512 µs</td><td>2^{12} / 240 kHz = 17.06 ms</td></tr>
<tr><td>1011_B</td><td>2^{13} / 8 MHz = 1.024 ms</td><td>2^{13} / 240 kHz = 34.13 ms</td></tr>
<tr><td>1100_B</td><td>2^{14} / 8 MHz = 2.048 ms</td><td>2^{14} / 240 kHz = 68.27 ms</td></tr>
<tr><td>1101_B</td><td>2^{15} / 8 MHz = 4.096 ms</td><td>2^{15} / 240 kHz = 136.5 ms</td></tr>
<tr><td>1110_B</td><td>2^{16} / 8 MHz = 8.192 ms</td><td>2^{16} / 240 kHz = 273.1 ms</td></tr>
<tr><td>1111_B</td><td>2^{17} / 8 MHz = 16.38 ms</td><td>2^{17} / 240 kHz = 546.1 ms</td></tr>
</table>
Notes:
• 8 MHz is the nominal frequency of the High Speed IntOsc.
• 240 kHz is the nominal frequency of the Low Speed IntOsc.</td></tr>
</table>

Note Refer to the Electrical Target Specification for information about the MainOsc stabilization time.

(5) SOSCE - SubOsc enable register

This register is used to start and stop the SubOsc and to specify its operation during stand-by modes.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 1020_H

Initial Value 0000 0004_H. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOSCESTPMK	SOSCEDISTRG	SOSCEENTRG
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-28 SOSCE register contents

Bit position	Bit name	Function
2	SOSCE STPMK	SubOsc stop request mask 0: stop request not masked 1: stop request masked If the SubOsc stop request is masked (SOSCESTPMK = 1) a stop request does not change the status of the SubOSC. If SOSCESTPMSK = 0, the SubOsc is stopped in stand-by mode. Upon wake-up from stand-by mode, the SubOsc returns to its status as before stand-by mode, i.e. continues operation when it was operating or remains stopped, when it was stopped before. **Note:** The SubOsc can also be activated after stand-by wake-up, independent of its status before stand-by mode. This features is controlled by the oscillator wake-up mask register OSCWUFMSK. Refer to section *"Wake-up"* in the chapter *"Stand-by Controller"* for details.
1	SOSCE DISTRG	SubOsc disable trigger 0: no function 1: stops SubOsc Stopping the SubOsc by SOSCEDISTRG = 1 is only possible, if the SubOsc is active, i.e. SOSCS.SOSCSCLKACT = 1. Reading of this bit returns always 0.
0	SOSCE ENTRG	SubOsc enable trigger 0: no function 1: starts SubOsc Starting the SubOsc by SOSCEENTRG = 1 is only possible, if the SubOsc is inactive, i.e. SOSCS.SOSCSCLKACT = 0. Reading of this bit returns always 0.

(6) SOSCS - SubOsc status register

This register provides various status information about the SubOsc status.

Access This register can be read in 32-bit units.

Address FF42 1024_H

Initial Value $0000\ 0000_H$. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOSCS CLK EN	SOSCS CLK ACT	SOSCS CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-29 SOSCS register contents

Bit position	Bit name	Function
2	SOSCS CLKEN	SubOsc enable status 0: SubOsc is disabled 1: SubOsc is enabled
1	SOSCS CLKACT	SubOsc activation status 0: SubOsc is inactive 1: SubOsc is active
0	SOSCS CLKSTAB	SubOsc stabilization status 0: SubOsc is unstable 1: SubOsc is stable

(7) SOSCST - SubOsc stabilization time register

This register determines the SubOsc stabilization time.

This register can only be written, if the SubOsc is disabled (SOSCS.SOSCSCLKEN = 0).

Access This register can be read/written in 32-bit units.

Address FF42 102C$_H$

Initial Value 0000 0000$_H$. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	SOST[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-30 SOSCST register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>2 to 0</td><td>SOST[2:0]</td><td>SubOsc stabilization time setting
Per default the SubOsc stabilization counter is operating with the High Speed IntOsc.
If the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0), the stabilization counter clock is automatically changed to the Low Speed IntOsc.
<table>
<tr><th rowspan="2">SOST[2:0]</th><th colspan="2">SubOsc stabilization time</th></tr>
<tr><th>High Speed IntOsc enabled (ROSCS.ROSCSCLKEN = 1)</th><th>High Speed IntOsc disabled (ROSCS.ROSCSCLKEN = 0)</th></tr>
<tr><td>000_B</td><td>2^{19} / 8 MHz = 65.54 ms</td><td>2^{19} / 240 kHz = 2.184 s</td></tr>
<tr><td>001_B</td><td>2^{20} / 8 MHz = 131 ms</td><td>2^{20} / 240 kHz = 4.369 s</td></tr>
<tr><td>0010_B</td><td>2^{21} / 8 MHz = 262 ms</td><td>2^{21} / 240 kHz = 8.738 s</td></tr>
<tr><td>011_B</td><td>2^{22} / 8 MHz = 524ms</td><td>2^{22} / 240 kHz = 17.4 s</td></tr>
<tr><td>100_B</td><td>2^{23} / 8 MHz = 1.048 s</td><td>2^{23} / 240 kHz = 34.9 s</td></tr>
<tr><td>101_B</td><td>2^{24} / 8 MHz = 2.096 s</td><td>2^{24} / 240 kHz = 69.9 s</td></tr>
<tr><td>110_B</td><td>2^{25} / 8 MHz = 4.192 s</td><td>2^{25} / 240 kHz = 139 s</td></tr>
<tr><td>111_B</td><td>2^{26} / 8 MHz = 8.384 s</td><td>2^{26} / 240 kHz = 279 s</td></tr>
</table>
Note:
• 8 MHz is the nominal frequency of the High Speed IntOsc.
• 240 kHz is the nominal frequency of the Low Speed IntOsc.</td></tr>
</table>

Note Refer to the Electrical Target Specification for information about the SubOsc stabilization time.

(8) ROSCE - High Speed IntOsc enable register

This register is used to start and stop the High Speed IntOsc and to specify its operation during stand-by modes.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 1000_H

Initial Value 0000 0004_H. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	ROSCESTPMK	ROSCEDISTRG	ROSCEENTRG
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-31 ROSCE register contents

Bit position	Bit name	Function
2	ROSCE STPMK	High Speed IntOsc stop request mask 0: stop request not masked 1: stop request masked If the High Speed IntOsc stop request is masked (ROSCESTPMK = 1) a stop request does not change the status of the High Speed IntOsc. If ROSCESTPMSK = 0, the High Speed IntOsc is stopped in stand-by mode. Upon wake-up from stand-by mode, the High Speed IntOsc returns to its status as before stand-by mode, i.e. continues operation when it was operating or remains stopped, when it was stopped before.
1	ROSCE DISTRG	High Speed IntOsc disable trigger 0: no function 1: stops High Speed IntOsc Stopping the High Speed IntOsc by ROSCEDISTRG = 1 is only possible, if the High Speed IntOsc is active, i.e. ROSCS.ROSCSCLKACT = 1. Reading of this bit returns always 0.
0	ROSCE ENTRG	High Speed IntOsc enable trigger 0: no function 1: starts High Speed IntOsc Starting the High Speed IntOsc by ROSCEENTRG = 1 is only possible, if the High Speed IntOsc is inactive, i.e. ROSCS.ROSCSCLKACT = 0. Reading of this bit returns always 0.

Cautions

1. Pay attention to clock domains, which are using the High Speed IntOsc clock as source via the clock selectors CKSC_mn, when disabling f_{RH} by ROSCE.ROSCEDISTRG = 1.
   In particular all resets set all clock selectors CKSCLK_mn to their default selection, but only the PURES enables the High Speed IntOsc.
   As a consequence all PBUS clocks PCLK will not operate after a reset - except PURES -, since the High Speed IntOsc is selected as their clock source, but the High Speed IntOsc is not enabled.
2. If the High Speed IntOsc is stopped and a reset occurs during a Back-up RAM access, the Back-up RAM may get corrupted. Thus do not disable the High Speed IntOsc in order to avoid Back-up RAM corruption.

(9) ROSCS - High Speed IntOsc status register

This register provides various status information about the High Speed IntOsc status.

Access This register can be read in 32-bit units.

Address FF42 1004_H

Initial Value 0000 0004_H. This register is initialized by the power-up reset PURES (Power-On-Clear or debugger reset).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	ROSCS CLK EN	ROSCS CLK ACT	ROSCS CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-32 ROSCS register contents

Bit position	Bit name	Function
2	ROSCS CLKEN	High Speed IntOsc enable status 0: High Speed IntOsc is disabled 1: High Speed IntOsc is enabled
1	ROSCS CLKACT	High Speed IntOsc activation status 0: High Speed IntOsc is inactive 1: High Speed IntOsc is active
0	ROSCS CLKSTAB	High Speed IntOsc stabilization status 0: High Speed IntOsc is unstable 1: High Speed IntOsc is stable

(10) PLLEk - PLLk enable register

This register is used to start and stop the PLLk and to specify its operation during stand-by modes.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address PLLE0: FF42 5000$_H$, PLLE1: FF42 5010$_H$, PLLE2: FF42 5020$_H$

Initial Value 0000 0004$_H$. This register is initialized by the system reset SYSRES.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PLLEk STPMK	PLLEk DISTRG	PLLEk ENTRG
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-33 PLLEk register contents

Bit position	Bit name	Function
2	PLLEk STPMK	PLLk stop request mask 0: stop request not masked 1: stop request masked If the PLLk stop request is masked (PLLEkSTPMK = 1) a stop request does not change the status of the PLLk. If PLLEkSTPMSK = 0, the PLLk is stopped in stand-by mode. Upon wake-up from stand-by mode, the PLLk returns to its status as before stand-by mode, i.e. continues operation when it was operating or remains stopped, when it was stopped before.
1	PLLEk DISTRG	PLLk disable trigger 0: no function 1: stops PLLk Stopping the PLLk by PLLEkDISTRG = 1 is only possible, if the PLLk is active, i.e. PLLSk.PLLSkCLKACT = 1. Reading of this bit returns always 0.
0	PLLEk ENTRG	PLLk enable trigger 0: no function 1: starts PLLk Starting the PLLk by PLLEkENTRG = 1 is only possible, if the PLLk is inactive, i.e. PLLSk.PLLSkCLKACT = 0. Reading of this bit returns always 0.

(11) PLLSk - PLLk status register

This register provides various status information about the PLLk status.

Access This register can be read in 32-bit units.

Address PLLS0: FF42 5004_H, PLLS1: FF42 5014_H, PLLS2: FF42 5024_H

Initial Value $0000\ 0000_H$. This register is initialized by the system reset SYSRES.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PLLSk CLK EN	PLLSk CLK ACT	PLLSk CLK STAB
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 9-34 PLLSk register contents

Bit position	Bit name	Function
2	PLLSk CLKEN	PLLk enable status 0: PLLk is disabled 1: PLLk is enabled
1	PLLSk CLKACT	PLLk activation status 0: PLLk is inactive 1: PLLk is active
0	PLLSk CLKSTAB	PLLk stabilization status 0: PLLk is unstable 1: PLLk is stable

(12) PLLCk - PLLk control register

This register is used to specify the PLLk output clock f_{PLk} frequency.

This register can only be written, if the PLLk is disabled (PLLSk.PLLSkCLKEN = 0).

Access This register can be read/written in 32-bit units.

Address PLLC0: FF42 5008$_H$, PLLC1: FF42 5018$_H$, PLLC2: FF42 5028$_H$

Initial Value 0000 0000$_H$. This register is initialized by the system reset SYSRES.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0[a]	PLLCkMS	PLLCk PC[1:0]		PLLCk ADJ[2:0]			PLLCk MDL[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) The default value 0 of bit 24 must not be changed.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PLLCkS[1:0]		PLLCkM[3:0]				PLLCkP[1:0]		0	PLLCkN[6:0]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-35 PLLCk register contents (1/3)

Bit position	Bit name	Function
23	PLLCk MS	PLL mode selection 0: SSCG mode (dithering mode) 1: PLL mode **Caution** SSCG mode is only available for PLL0 and PLL2. Thus PLL1 must be operated in PLL mode, i.e. set PLLC1.PLLC1MS = 1.
22 to 21	PLLCk PC[1:0]	Dithering mode selection **PLLCk PC[1:0]** / **Dithering mode** $0X_B$ / fixed frequency 10_B / setting prohibited 11_B / center spread dithering

Table 9-35 PLLCk register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>20 to 18</td><td>PLLCk
ADJ[2:0]</td><td>Frequency dithering range selection
<table>
<tr><th>PLLCk
ADJ[1:0]</th><th>Frequency dithering range</th></tr>
<tr><td>000_B</td><td rowspan="2">1 %</td></tr>
<tr><td>001_B</td></tr>
<tr><td>010_B</td><td>2 %</td></tr>
<tr><td>011_B</td><td>3 %</td></tr>
<tr><td>100_B</td><td>4 %</td></tr>
<tr><td>101_B</td><td>5 %</td></tr>
<tr><td>110_B</td><td>setting prohibited</td></tr>
<tr><td>111_B</td><td>setting prohibited</td></tr>
</table></td></tr>
<tr><td>17 to 16</td><td>PLLCk
MDL[1:0]</td><td>Dithering modulation frequency selection
<table>
<tr><th>PLLCk
MDL[1:0]</th><th>Modulation frequency</th></tr>
<tr><td>00_B</td><td>40 kHz</td></tr>
<tr><td>01_B</td><td>50 kHz</td></tr>
<tr><td>10_B</td><td>60 kHz</td></tr>
<tr><td>11_B</td><td>setting prohibited</td></tr>
</table></td></tr>
<tr><td>15 to 14</td><td>PLLCk
S[1:0]</td><td>VCO input frequency division selection in SSCG mode (PLLCkMS = 0)
<table>
<tr><th>PLLCk
S[1:0]</th><th>Input frequency range</th></tr>
<tr><td>00_B</td><td>1.0 MHz ≤ f_{VCOIN} < 1.2 MHz</td></tr>
<tr><td>01_B</td><td>1.2 MHz ≤ f_{VCOIN} < 1.4 MHz</td></tr>
<tr><td>10_B</td><td>1.4 MHz ≤ f_{VCOIN} < 1.7 MHz</td></tr>
<tr><td>11_B</td><td>1.7 MHz ≤ f_{VCOIN} ≤ 2.0 MHz</td></tr>
</table></td></tr>
<tr><td>13 to 10</td><td>PLLCk
M[3:0]</td><td>Divider value Mr
- in SSCG mode: Mr = PLLCkM[3:0] + 1
- in PLL mode PLLCkM[3:0] = 0: Mr = 1</td></tr>
<tr><td>9 to 8</td><td>PLLCk
P[1:0]</td><td>P divider selection
<table>
<tr><th>PLLCk
P[1:0]</th><th>Divider value Pr</th><th>PLL output frequency range</th></tr>
<tr><td>00_B</td><td>–</td><td>setting prohibited</td></tr>
<tr><td>01_B</td><td>1</td><td>100 MHz to 200 MHz</td></tr>
<tr><td>10_B</td><td>2</td><td>50 MHz to 100 MHz</td></tr>
<tr><td>11_B</td><td>4</td><td>25 MHz to 50 MHz</td></tr>
</table></td></tr>
</table>

Table 9-35 PLLCk register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>6 to 0</td><td>PLLCk
N[6:0]</td><td>Divider value Nr
<table>
<tr><th rowspan="2">PLLCk
N[6:0]</th><th colspan="2">Divider value Nr = PLLCKN[6:0] + 1</th></tr>
<tr><th>SSCG mode</th><th>PLL mode</th></tr>
<tr><td>$000\ 0100_B$</td><td>–</td><td>5</td></tr>
<tr><td>$000\ 0101_B$</td><td>–</td><td>6</td></tr>
<tr><td>...</td><td>–</td><td>...</td></tr>
<tr><td>$011\ 0001_B$</td><td>50</td><td>50</td></tr>
<tr><td>$011\ 0010_B$</td><td>51</td><td>–</td></tr>
<tr><td>...</td><td>...</td><td>–</td></tr>
<tr><td>$110\ 0010_B$</td><td>99</td><td>–</td></tr>
<tr><td>$110\ 0011_B$</td><td>100</td><td>–</td></tr>
</table></td></tr>
</table>

(13) PLLSTk - PLLk stabilization time register

This register determines the PLLk stabilization time.

This register can only be written, if the PLLk is disabled (PLLSk.PLLSkCLKEN = 0).

Access This register can be read/written in 32-bit units.

Address PLLST0: FF42 500C$_H$, PLLST1: FF42 501C$_H$, PLLST2: FF42 502C$_H$

Initial Value 0000 0000$_H$. This register is initialized by the system reset SYSRES.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	PLLSTk[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-36 PLLSTk register contents

Bit position	Bit name	Function
2 to 0	PLLSTk[2:0]	PLLk stabilization time setting Per default the PLLk stabilization counter is operating with the High Speed IntOsc. If the High Speed IntOsc is disabled (ROSCS.ROSCSCLKEN = 0), the stabilization counter clock is automatically changed to the Low Speed IntOsc.

PLLSTk[2:0]	PLLk stabilization time	
	High Speed IntOsc enabled (ROSCS.ROSCSCLKEN = 1)	High Speed IntOsc disabled (ROSCS.ROSCSCLKEN = 0)
000_B	2^7 / 8 MHz = 16 µs	2^7 / 240 kHz = 533 µs
001_B	2^8 / 8 MHz = 32 µs	2^8 / 240 kHz = 1.067 ms
010_B	2^9 / 8 MHz = 64 µs	2^9 / 240 kHz = 2.133 ms
011_B	2^{10} / 8 MHz = 128 µs	2^{10} / 240 kHz = 4.267 ms
100_B	2^{11} / 8 MHz = 256 µs	2^{11} / 240 kHz = 8.533 ms
101_B	2^{12} / 8 MHz = 512 µs	2^{12} / 240 kHz = 17.057 ms
110_B	2^{13} / 8 MHz = 1.024 ms	2^{13} / 240 kHz = 34.133 ms
111_B	2^{14} / 8 MHz = 2.048 ms	2^{14} / 240 kHz = 68.267 ms

Note:
- 8 MHz is the nominal frequency of the High Speed IntOsc.
- 240 kHz is the nominal frequency of the Low Speed IntOsc.

Note Refer to the Electrical Target Specification for information about the PLLk stabilization time.

9.8.4 Clock selector control register

(1) CKSC_mn – Clock selector control registers

These registers select the clock for all clock domains which allow to select different clocks. For details on the selectable values for each register and its initial value refer to the section *"Clock Selection"* of this chapter.

Caution Changing safely the clock selection for a clock domain requires particular attention.

Refer to *"Clock switching*" in the section *"Clock Selection"* of this chapter.

Protection Writing to these registers is protected by a special sequence of instructions by using a protection command register, that depends on the power area:

- m = 0: Iso0
  The access to writing CKSC_0n registers is protected by the protection register PRTCMD0.
- m = 1: Iso1
  The access to writing CKSC_1n registers is protected by the protection register PRTCMD1.
- m = A: AWO
  The access to writing CKSC_An registers is protected by the protection register PRTCMD2.

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Function"* for a detailed description how to write to write protected registers.

Caution If the Isolated-Area-1 is in DEEPSTOP mode, the clock selector control registers CKSC_1n and the protection registers PROTCMD1/PROTS1 are not accessible.
Any CPU attempt to access these registers will lead to a microcontroller deadlock.

Access These registers can be read/written in 32-bit units.

Address Isolated-Area-0 clock selectors CKSC_0n: FF42 6000_H + n x 16
Isolated-Area-1 clock selectors CKSC_1n: FF42 $A000_H$ + n x 16
Always-On-Area clock selectors CKSC_An: FF42 2000_H + n x 16

Initial Value Refer to the section *"Clock Selection"*.

These registers are initialized following resets:

- Isolated-Area-0 clock selectors CKSC_0n:
  - Reset Controller: SYSRES
  - Stand-by Controller:
    DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
- Isolated-Area-1 clock selectors CKSC_1n:
  - Reset Controller: SYSRES
  - Stand-by Controller:
    DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)
- Always-On-Area clock selectors CKSC_An:
  - Reset Controller: SYSRES

31	30	29	28	27	26	25	24
CKSCID_mn[30:23]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
CKSCID_mn[22:15]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
CKSCID_mn[14:7]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
CKSCID_mn[6:0]							STPMK_mn
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 9-37 CKSC_mn register contents

Bit position	Bit name	Function
31 to 1	CKSCID _mn[30:0]	Clock Source ID: These bits specify the clock CKSCLK_mn for clock domain mn.
0	STPMK_mn	Controls the clock output CKSCLK_mn during stand-by mode: 0: output clock CKSCLK_mn is stopped during stand-by mode 1: output clock CKSCLK_mn continues during stand-by mode

(2) CSCSTAT_mn – Clock selector status registers

These status registers return the ID of the clock source, that is currently selected by CKSC_mn and its activity status.

Caution If the Isolated-Area-1 is in DEEPSTOP mode, the clock selector status registers CSCSTAT_1n are not accessible.
Any CPU attempt to access these registers will lead to a microcontroller deadlock.

Access These registers can be read in 32-bit units.

Address Isolated-Area-0 clock selectors status CSCSTAT_0n: FF42 6004_H + n x 16
Isolated-Area-1 clock selectors status CSCSTAT_1n: FF42 $A004_H$ + n x 16
Always-On-Area clock selectors status CSCSTAT_An: FF42 2004_H + n x 16

Initial Value Refer to the section *"Clock Selection"*.

These registers are initialized following resets:

- Isolated-Area-0 clock selectors CKSC_0n:
  - Reset Controller: SYSRES
  - Stand-by Controller:
    DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
- Isolated-Area-1 clock selectors CKSC_1n:
  - Reset Controller: SYSRES
  - Stand-by Controller:
    DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)
- Always-On-Area clock selectors CKSC_An:
  - Reset Controller: SYSRES

31	30	29	28	27	26	25	24
CLKSELID_mn[30:23]							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
CLKSELID_mn[22:15]							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
CLKSELID_mn[14:7]							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
CLKSELID_mn[6:0]							CLKACT_mn
R	R	R	R	R	R	R	R

Table 9-38 CSCSTAT_mn register contents

Bit position	Bit name	Function
31 to 1	CLKSELID _mn[30:0]	Clock source ID of the current CKSC_mn clock selection
0	CLKACT_mn	CKSCLK_mn activity indicator 0: CKSCLK_mn is disabled 1: CKSCLK_mn is enabled

Chapter 10 Stand-by Controller (STBC)

10.1 V850E2/Fx4-H Stand-by Controller Features

STBC reset

The Stand-by Controller and its registers are initialized by the following reset signal:

Table 10-1 STBC reset signal

STBC	Reset signal
Stand-by Controller	• Reset Controller: SYSRES

Wake-up factors

The wake-up events for terminating a power save mode are controlled and monitored by the following Stand-by Controller registers:

- WUFLm, WUFMSKLm, WUFCLm
- WUFMm, WUFMSKMm, WUFCMm
- WUFHm, WUFMSKHm, WUFCHm

Index m

The index m = 0, 1 denotes the Isolated-Area-0 respectively Isolated-Area-1.

The assignment of the wake-up events to the wake-up control and status register bits is given in the table below.

- WUF[L,M,H]m[31:00] are bits of the wake-up factor registers WUF[L,M,H]m
- WUFMSK[L,M,H]m[31:00] are bits of the wake-up factor mask registers WUFMSK[L,M,H]m
- WUFC[L,M,H]m[31:00] are bits of the wake-up factor clear registers WUFC[L,M,H]m

For details about the wake-up control and status registers refer to the section “Wake-up factor controller registers details” later on in this chapter.

Table 10-2 Wake-up factors registers assignments (WUFLm/WUFMSKLm/WUFCLm with m = 0, 1) (1/2)

Assignment of WUFLm/WUFMSKLm/WUFCLm register bits						Wake-up source			
Always-On-Area and Isolated-Area-0 wake-up			Isolated-Area-1 wake-up			Wake up event	Module	Power area	Domain clock
WUFL000	WUFMSKL000	WUFCL000	WUFL100	WUFMSKL100	WUFCL100	NMI	Port	AWO	-
WUFL001	WUFMSKL001	WUFCL001	WUFL101	WUFMSKL101	WUFCL101	INTWDTA0	WDTA0	AWO	CKSCLK_A07
WUFL002	WUFMSKL002	WUFCL002	WUFL102	WUFMSKL102	WUFCL102	INTLVI	LVI	AWO	-
WUFL003	WUFMSKL003	WUFCL003	WUFL103	WUFMSKL103	WUFCL103	INTKR0	KR0	AWO	CKSCLK_A02
WUFL004	WUFMSKL004	WUFCL004	WUFL104	WUFMSKL104	WUFCL104	INTRTCA0AL	RTCA0	AWO	CKSCLK_A09
WUFL005	WUFMSKL005	WUFCL005	WUFL105	WUFMSKL105	WUFCL105	INTRTCA0R			
WUFL006	WUFMSKL006	WUFCL006	WUFL106	WUFMSKL106	WUFCL106	INTRTCA01S			

Table 10-2 Wake-up factors registers assignments (WUFLm/WUFMSKLm/WUFCLm with m = 0, 1) (2/2)

Assignment of WUFLm/WUFMSKLm/WUFCLm register bits						Wake-up source			
Always-On-Area and Isolated-Area-0 wake-up			Isolated-Area-1 wake-up			Wake up event	Module	Power area	Domain clock
WUFL007	WUFMSKL007	WUFCL007	WUFL107	WUFMSKL107	WUFCL107	INTP0	Port	AWO	-
WUFL008	WUFMSKL008	WUFCL008	WUFL108	WUFMSKL108	WUFCL108	INTP1	Port		-
WUFL009	WUFMSKL009	WUFCL009	WUFL109	WUFMSKL109	WUFCL109	INTP2	Port		-
WUFL010	WUFMSKL010	WUFCL010	WUFL110	WUFMSKL110	WUFCL110	INTP3	Port		-
WUFL011	WUFMSKL011	WUFCL011	WUFL111	WUFMSKL111	WUFCL111	INTP4	Port		-
WUFL012	WUFMSKL012	WUFCL012	WUFL112	WUFMSKL112	WUFCL112	INTP5	Port		-
WUFL013	WUFMSKL013	WUFCL013	WUFL113	WUFMSKL113	WUFCL113	INTP6	Port		-
WUFL014	WUFMSKL014	WUFCL014	WUFL114	WUFMSKL114	WUFCL114	INTP7	Port		-
WUFL015	WUFMSKL015	WUFCL015	WUFL115	WUFMSKL115	WUFCL115	INTP8	Port		-
WUFL016	WUFMSKL016	WUFCL016	WUFL116	WUFMSKL116	WUFCL116	INTP9	Port		-
WUFL017	WUFMSKL017	WUFCL017	WUFL117	WUFMSKL117	WUFCL117	INTP10	Port		-
WUFL018	WUFMSKL018	WUFCL018	WUFL118	WUFMSKL118	WUFCL118	INTP11	Port		-
WUFL019	WUFMSKL019	WUFCL019	WUFL119	WUFMSKL119	WUFCL119	INTP12	Port		-
WUFL020	WUFMSKL020	WUFCL020	WUFL120	WUFMSKL120	WUFCL120	INTP13	Port		-
WUFL021	WUFMSKL021	WUFCL021	WUFL121	WUFMSKL121	WUFCL121	INTP14	Port		-
WUFL022	WUFMSKL022	WUFCL022	WUFL122	WUFMSKL122	WUFCL122	INTP15	Port		-
WUFL023	WUFMSKL023	WUFCL023	WUFL123	WUFMSKL123	WUFCL123	FCN0RX	Port	AWO[a]	-
WUFL024	WUFMSKL024	WUFCL024	WUFL124	WUFMSKL124	WUFCL124	FCN1RX	Port		-
WUFL025	WUFMSKL025	WUFCL025	WUFL125	WUFMSKL125	WUFCL125	FCN2RX	Port		-
WUFL026	WUFMSKL026	WUFCL026	WUFL126	WUFMSKL126	WUFCL126	FCN3RX	Port		-
WUFL027	WUFMSKL027	WUFCL027	WUFL127	WUFMSKL127	WUFCL127	FCN4RX	Port		-
WUFL028	WUFMSKL028	WUFCL028	WUFL128	WUFMSKL128	WUFCL128	FCN5RX	Port		-

a) Though the CAN Controllers FCNn reside on the Isolated-Area-1, the CAN bus receive inputs FCNnRX can be selected as a wake-up source in any DEEPSTOP mode.

Table 10-3 Wake-up factors registers assignments (WUFMm/WUFMSKMm/WUFCMm with m = 0, 1) (1/2)

Assignment of WUFMm/WUFMSKMm/WUFCMm register bits						Wake-up source			
Always-On-Area and Isolated-Area-0 wake-up			Isolated-Area-1 wake-up			Wake up event	Module	Power area	Domain clock
WUFM000	WUFMSKM000	WUFCM000	WUFM100	WUFMSKM100	WUFCM100	INTVCPC0	VCPC0	AWO	CKSCLK_A02
WUFM001	WUFMSKM001	WUFCM001	WUFM101	WUFMSKM101	WUFCM101	INTVCPC1	VCPC1	AWO	CKSCLK_A02
WUFM002	WUFMSKM002	WUFCM002	WUFM102	WUFMSKM102	WUFCM102	INTTAUJ0I0	TAUJ0	AWO	CKSCLK_A03
WUFM003	WUFMSKM003	WUFCM003	WUFM103	WUFMSKM103	WUFCM103	INTTAUJ0I1			
WUFM004	WUFMSKM004	WUFCM004	WUFM104	WUFMSKM104	WUFCM104	INTTAUJ0I2			
WUFM005	WUFMSKM005	WUFCM005	WUFM105	WUFMSKM105	WUFCM105	INTTAUJ0I3			

Table 10-3 Wake-up factors registers assignments (WUFMm/WUFMSKMm/WUFCMm with m = 0, 1) (2/2)

Assignment of WUFMm/WUFMSKMm/WUFCMm register bits						Wake-up source			
Always-On-Area and Isolated-Area-0 wake-up			Isolated-Area-1 wake-up			Wake up event	Module	Power area	Domain clock
WUFM006	WUFMSKM006	WUFCM006	WUFM106	WUFMSKM106	WUFCM106	INTTAUJ1I0	TAUJ1	AWO	CKSCLK_A04
WUFM007	WUFMSKM007	WUFCM007	WUFM107	WUFMSKM107	WUFCM107	INTTAUJ1I1			
WUFM008	WUFMSKM008	WUFCM008	WUFM108	WUFMSKM108	WUFCM108	INTTAUJ1I2			
WUFM009	WUFMSKM009	WUFCM009	WUFM109	WUFMSKM109	WUFCM109	INTTAUJ1I3			
WUFM010	WUFMSKM010	WUFCM010	WUFM110	WUFMSKM110	WUFCM110	INTADCA0ERR	ADCA0	ISO0	CKSCLK_012
WUFM011	WUFMSKM011	WUFCM011	WUFM111	WUFMSKM111	WUFCM111	INTADCA0I0			
WUFM012	WUFMSKM012	WUFCM012	WUFM112	WUFMSKM112	WUFCM112	INTADCA0I1			
WUFM013	WUFMSKM013	WUFCM013	WUFM113	WUFMSKM113	WUFCM113	INTADCA0I2			
WUFM014	WUFMSKM014	WUFCM014	WUFM114	WUFMSKM114	WUFCM114	INTADCA0LLT			
WUFM015	WUFMSKM015	WUFCM015	WUFM115	WUFMSKM115	WUFCM115	INTWDTA1	WDTA1	ISO0	CKSCLK_007
WUFM017	WUFMSKM017	WUFCM017	WUFM117	WUFMSKM117	WUFCM117	INTLMA10IT	LMA10	ISO0	CKSCLK_011
WUFM018	WUFMSKM018	WUFCM018	WUFM118	WUFMSKM118	WUFCM118	INTLMA10IR			
WUFM019	WUFMSKM019	WUFCM019	WUFM119	WUFMSKM119	WUFCM119	INTLMA10IS			
WUFM020	WUFMSKM020	WUFCM020	WUFM120	WUFMSKM120	WUFCM120	INTLMA11IT	LMA11	ISO0	CKSCLK_011
WUFM021	WUFMSKM021	WUFCM021	WUFM121	WUFMSKM121	WUFCM121	INTLMA11IR			
WUFM022	WUFMSKM022	WUFCM022	WUFM122	WUFMSKM122	WUFCM122	INTLMA11IS			
WUFM023	WUFMSKM023	WUFCM023	WUFM123	WUFMSKM123	WUFCM123	INTCSIG4IC	CSIG4	ISO0	CKSCLK_011
WUFM024	WUFMSKM024	WUFCM024	WUFM124	WUFMSKM124	WUFCM124	INTCSIG4IR			
WUFM025	WUFMSKM025	WUFCM025	WUFM125	WUFMSKM125	WUFCM125	INTCSIG4IRE			
WUFM029	WUFMSKM029	WUFCM029	WUFM129	WUFMSKM129	WUFCM129	INTTAUA0I0	TAUA0	ISO0	CKSCLK_006
WUFM030	WUFMSKM030	WUFCM030	WUFM130	WUFMSKM130	WUFCM130	INTTAUA0I1			
WUFM031	WUFMSKM031	WUFCM031	WUFM131	WUFMSKM131	WUFCM131	INTTAUA0I2			

Table 10-4 Wake-up factors registers assignments (WUFHm/WUFMSKHm/WUFCHm with m = 0, 1)

Assignment of WUFHm/WUFMSKHm/WUFCHm register bits						Wake-up source			
Always-On-Area and Isolated-Area-0 wake-up			Isolated-Area-1 wake-up			Wake up event	Module	Power area	Domain clock
WUFH000	WUFMSKH000	WUFCH000	WUFH100	WUFMSKH100	WUFCH100	INTTAUA0I3	TAUA0	ISO0	ISO0_6
WUFH001	WUFMSKH001	WUFCH001	WUFH101	WUFMSKH101	WUFCH101	INTTAUA0I4			
WUFH002	WUFMSKH002	WUFCH002	WUFH102	WUFMSKH102	WUFCH102	INTTAUA0I5			
WUFH003	WUFMSKH003	WUFCH003	WUFH103	WUFMSKH103	WUFCH103	INTTAUA0I6			
WUFH004	WUFMSKH004	WUFCH004	WUFH104	WUFMSKH104	WUFCH104	INTTAUA0I7			
WUFH005	WUFMSKH005	WUFCH005	WUFH105	WUFMSKH105	WUFCH105	INTTAUA0I8			
WUFH006	WUFMSKH006	WUFCH006	WUFH106	WUFMSKH106	WUFCH106	INTTAUA0I9			
WUFH007	WUFMSKH007	WUFCH007	WUFH107	WUFMSKH107	WUFCH107	INTTAUA0I10			
WUFH008	WUFMSKH008	WUFCH008	WUFH108	WUFMSKH108	WUFCH108	INTTAUA0I11			
WUFH009	WUFMSKH009	WUFCH009	WUFH109	WUFMSKH109	WUFCH109	INTTAUA0I12			
WUFH010	WUFMSKH010	WUFCH010	WUFH110	WUFMSKH110	WUFCH110	INTTAUA0I13			
WUFH011	WUFMSKH011	WUFCH011	WUFH111	WUFMSKH111	WUFCH111	INTTAUA0I14			
WUFH012	WUFMSKH012	WUFCH012	WUFH112	WUFMSKH112	WUFCH112	INTTAUA0I15			
WUFH013	WUFMSKH013	WUFCH013	WUFH113	WUFMSKH113	WUFCH113	WDTA0NMI	WDTA0	AWO	CKSCLK_A07
WUFH014	WUFMSKH014	WUFCH014	WUFH114	WUFMSKH114	WUFCH114	WDTA1NMI	WDTA1	ISO0	CKSCLK_007
WUFH015	WUFMSKH015	WUFCH015	WUFH115	WUFMSKH115	WUFCH115	OCDSTPRQ	OCD	AWO	-

Wake-up by URTEnRX

Though the URTEnRx signals can not be directly used as wake-up factors, external interrupts INTPx can be used instead, since URTEnRX and INTPx are alternative port functions of the same pin.

10.2 Stand-by Controller functions

The V850E2/Fx4-H supports following operation respectively stand-by - or power save - modes:

- HALT mode
  HALT mode can be entered from normal run mode by performing the CPU instruction “HALT”. This stops the CPU operation, while all clocks continue to operate and all areas remain under power.
- STOP mode
  In STOP mode the clock supply of the entire CPU Subsystem, including the CPU, is stopped. Thus no program is executed in STOP mode.
  Additionally certain clock supplies of a power domain can be stopped.
  The selection of the clock to stop respectively continue in STOP mode, is done by the clock selector’s stop mask CKSC_mn.STPMK_mn.
- DEEPSTOP mode
  In order to reduce power consumption further, the power supply of the Isolated-Area-0 and Isolated-Area-1 (including the Graphics Subsystem) can be switched off. The power supply of the I/O buffer of the isolated area in DEEPSTOP can also be stopped.
- RUN
  All operation modes with the CPU system on Isolated-Area-0 in operation are called RUN modes.

Always-On-Area STOP mode

Always-On-Area STOP mode is no separate stand-by mode, but is activated together with

- STOP mode
- STOP mode (Isolated-Area-0 STOP)
- DEEPSTOP mode

and means, that Always-On-Area domain clocks CKSCLK_An can be stopped.

Note that the Always-On-Area’s power supply has to be active all the time.

10.2.1 Stand-by Controller signal connections

The Stand-by controller's main signal connections with other microcontroller modules is shown in the diagram below.

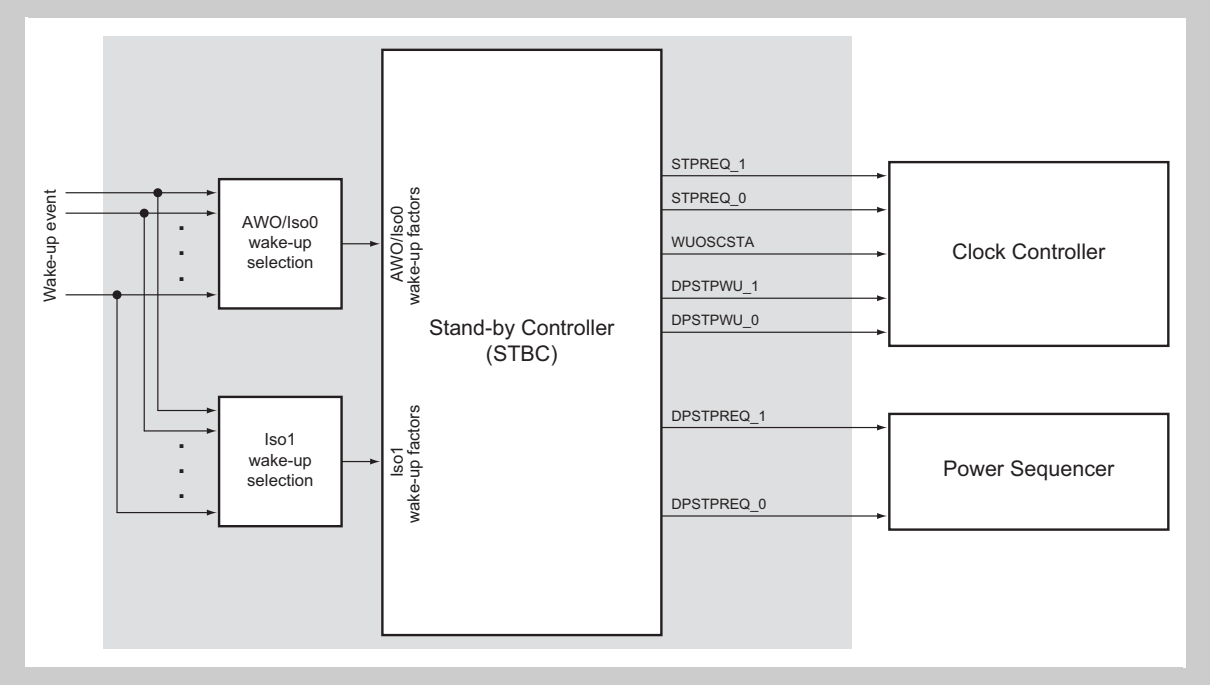


Figure 10-1 Stand-by Controller main signal connections

Wake-up events and factors

Wake-up events are used from various other modules. Each wake-up event can be separately enabled to wake-up

- Always-On-Area and Isolated-Area-0
- Isolated-Area-1

from stand-by mode, and thus becomes a wake-up factor.

Refer to the wake-up factor tables in the first section of this chapter under the key word *"Wake-up factors"*.

Power Sequencer signals

If isolated areas are set into DEEPSTOP mode, the Stand-by Controller generates following signals towards the Power Sequencer in order to switch off the power supply of the concerned isolated area:

- DPSTPREQ_0 is asserted, when the Isolated-Area-0 is set in DEEPSTOP mode.
- DPSTPREQ_1 is asserted, when the Isolated-Area-1 is set in DEEPSTOP mode.

Refer to the chapter *37 "Power Supply" on page 2602* for details concerning the Power Sequencer.

Clock Controller signals

Upon entering STOP mode or wake-up from DEEPSTOP the Stand-by Controller generates following signals towards the Clock Controller:

- STPREQ_0 is asserted upon entering Isolated-Area-0 STOP or DEEPSTOP mode and is used to stop clock generators and the Isolated-Area-0 and Always-On-Area domain clocks.

- STPREQ_1 is asserted upon entering Isolated-Area-1 STOP or DEEPSTOP mode and is used to stop Isolated-Area-1 domain clocks.
- DPSTPWU_0 is asserted upon wake-up from Isolated-Area-0 DEEPSTOP and is used to reset the Isolated-Area-0 clock selectors.
- DPSTPWU_1 is asserted upon wake-up from Isolated-Area-1 DEEPSTOP and is used to reset the Isolated-Area-1 clock selectors.
- WUOSCSTA can optionally be asserted, when the MainOsc and SubOsc shall start operation upon wake-up from a stand-by mode. The generation of WUOSCSTA is controlled by the OSCWUFMSK.

Refer to the *9 “Clock Controller” on page 420* chapter for details about the Clock Controller behaviour in stand-by.

10.2.2 Stand-by modes control

This section describes how the microcontroller is set into stand-by mode.

(1) HALT mode control

The HALT mode has no influence on any power or clock domain. Thus this mode is not subject to this chapter. Refer to the document *“V850E2M 32-bit Microprocessor Core Architecture”* for details concerning the HALT mode.

(2) STOP and DEEPSTOP control

STOP and DEEPSTOP control is accomplished by a set of stand-by control registers:

- PSC0 controls
  - STOP of Always-On-Area and Isolated-Area-0
  - DEEPSTOP of Isolated-Area-0
- PSC1 controls
  - STOP of Isolated-Area-1
  - DEEPSTOP of Isolated-Area-1

PSCm registers The power save control registers PSC0 and PSC1 hold bits to

- select a stand-by mode and initiate its entry (PSCmPOF, PSCmSTP)
- control the I/O buffers of the concerned power domain (PSCmIOHLDSET, PSCmIOHLDMSK, PSCmIOHLDCLR), for detail refer to *10.2.6 “I/O buffer control” on page 596* .
- control stand-by mode wake-up by software (PSCmISOWU)
- control the power-down and -up process for DEEPSTOP mode entry and wake-up (PSCmREGSTP), for detail refer to the chapter *37 “Power Supply” on page 2602* .

Note that not all PSCm registers provide all of the above control bits.

Stand-by mode entry

To enter a stand-by mode of the Isolated-Area-0 and Isolated-Area-1 power domain the stand-by mode trigger bit PSC0STP respectively PSC1STP has to be set to 1.

Whether STOP or DEEPSTOP mode is entered by PSC0STP = 1 respectively PSC1STP = 1, is determined by the power-off selection bit PSC0POF respectively PSC1POF:

- Isolated-Area-0:
  - PSC0STP = 1 and PSC0POF = 0: STOP mode
  - PSC0STP = 1 and PSC0POF = 1: DEEPSTOP mode
- Isolated-Area-1:
  - PSC1STP = 1 and PSC1POF = 0: STOP mode
  - PSC1STP = 1 and PSC1POF = 1: DEEPSTOP mode

The following table gives an overview about the stand-by control options:

Table 10-5 Stand-by modes control

Power domain	PSC0STP = 1		PSC1STP = 1	
	PSC0POF = 0	PSC0POF = 1	PSC1POF = 0	PSC1POF = 1
Always-On-Area	STOP	STOP	–	–
Isolated-Area-0	STOP	DEEPSTOP	–	–
Isolated-Area-1	–	–	STOP	DEEPSTOP

(3) Stand-by status

The status of each power domain can be evaluated by use of the power status registers PWSm. These registers hold bits, indicating the status of the respective power domain.

- operational or stand-by mode
  - PWSnPSS = 0: power domain n not in stand-by
  - PWSnPSS = 1: power domain n in stand-by respectively transition to stand-by ongoing
- power-off/-on, i.e. DEEPSTOP, mode
  - PWSnISO = 0: power domain n not in DEEPSTOP
  - PWSnISO = 1: power domain n in DEEPSTOP
- I/O buffer hold mode (PWSnIOHOLD)

The following table summarizes the stand-by status information.

Table 10-6 Stand-by status overview

Status bits		Mode
PWS1PSS	PWS1ISO	Isolated-Area-1
0	0	RUN
1	1	STOP
1	0	DEEPSTOP

Note Since in Isolated-Area-0 STOP and DEEPSTOP stand-by modes the CPU does not operate, these modes are not reflected in the stand-by status.

10.2.3 Stand-by modes overview

The clock and power supply options are summarized in the table below.

Table 10-7 Stand-by modes overview

Mode	Always-On	Isolated-Area-0		Isolated-Area-1	
	Clock	Power	Clock	Power	Clock
RUN	not stopped	on	not stopped	on	not stopped
	PSC0STP = 0			PSC1STP = 0	
RUN (Isolated-Area-1 STOP)	not stopped	on	not stopped	on	stopped
	PSC0STP = 0			PSC1STP = 1, PSC1POF = 0	
RUN (Isolated-Area-1 DEEPSTOP)	not stopped	on	not stopped	off	stopped
	PSC0STP = 0			PSC1STP = 1, PSC1POF = 1	
STOP	stopped	on	stopped	on	stopped
	PSC0STP = 1, PSC0POF = 0			PSC1STP = 1, PSC1POF = 0	
STOP (Isolated-Area-1 DEEPSTOP)	stopped	on	stopped	off	stopped
	PSC0STP = 1, PSC0POF = 0			PSC1STP = 1, PSC1POF = 1	
DEEPSTOP	stopped	off	stopped	off	stopped
	PSC0STP = 1, PSC0POF = 1			PSC1STP = 1, PSC1POF = 1	

Notes

1. The clock status in the above table has the following meaning:
   - “not stopped”:
     The clock selectors of the respective power domain do not receive a stop request. Thus its status are not changed.
   - “stopped”:
     The clock selectors of the respective power domain receive a stop request. Thus a certain clock domain clock is stopped, if the clock stop is unmasked by CKSC_mn.STPMK_mn = 0.

   Refer to *9.4 “Clock Selection” on page 444* for details concerning clock stop masking.
2. The power status in the above table has the following meaning:
   - on:
     The power supply of respective power domain is retained.

- off:
  The power supply of respective power domain is switched off.

Caution Setting the Isolated-Area-0 into DEEPSTOP mode (PSC0.POF = 1) while keeping the Isolated-Area-1 in operation is not permitted.

Stand-by and On-Chip Debug Setting the device in STOP or DEEPSTOP mode is not possible, if the microcontroller is operated under control of an On-Chip Debugger.

10.2.4 Clock generators in stand-by

Following table shows the status of the various clock generators in the different stand-by modes.

Table 10-8 Clock generators in power-save modes

Mode	Always-On-Area				Isolated-Area-0
	Low Speed IntOsc	High Speed IntOsc	MainOsc	SubOsc	PLLk
RUN	not stopped	not stopped	not stopped	not stopped	not stopped
RUN (Isolated-Area-1 STOP)		not stopped	not stopped	not stopped	not stopped
RUN (Isolated-Area-1 DEEPSTOP)		not stopped	not stopped	not stopped	not stopped
STOP		stopped	stopped	stopped	stopped
STOP (Isolated-Area-1 DEEPSTOP)		stopped	stopped	stopped	stopped
DEEPSTOP		stopped	stopped	stopped	off

Note The clock generator status in the above table has the following meaning:

- “not stopped”:
  The respective clock generator does not receive a stop request and its status is not changed. Thus it remains stopped when it was stopped or is operating in stand-by mode, when it was enabled.
- “stopped”:
  The respective clock generator receives a stop request. Thus it is stopped, if the clock stop is unmasked by associated stop request enable bit
  - MOSCE.MOSCESTPMK = 1 for the MainOsc
  - SOSCE.SOSCESTPMK = 1 for the SubOsc
  - ROSCE.ROSCESTPMK = 1 for the High Speed IntOsc
  - PLLEk.PLLEkSTPMK = 1 for the PLLk

  Refer to *8.13 “General Description of Clock Generation and Control” on page 1* for details concerning clock stop masking.
- “off”
  The power supply of the PLLs, which are located on the Isolated-Area-0, is switched off in DEEPSTOP mode.

(1) Module clocks during transition to stand-by mode

If any power domain (m = A, 0, 1) is set in any stand-by mode, the domain clocks CKSCLK_mn of all modules on this power domain must be

- *either* active, i.e. the clock source, selected by the clock ID CKSC_mn.CKSCID[30:0], must be active
- *or* stopped by setting CKSC_mn = 0.

Caution Do not set any power domain in stand-by mode, if any module on the power domain is connected to a non-active clock.

(2) Clock generators in stand-by mode

In case a clock generator shall be stopped in an Isolated-Area-0 stand-by mode - i.e. STOP, STOP (Isolated-Area-1 DEEPSTOP), STOP (Isolated-Area-0 STOP) and DEEPSTOP - and shall automatically resume operation after wake-up, restart of the clock generator may fail under certain conditions. Refer to the section *"Precaution: Clock generators and early wake-up"* below in this chapter for details.

10.2.5 Wake-up

(1) Wake-up factors

Following wake-up factors are provided to leave a stand-by mode:

Table 10-9 Wake-up events

Mode	Reset	NMI	INTLVI	INTPx[a]	Functional module on power domain[ab]	CAN reception FCNnRX[a]	S/W	OCD
RUN (Isolated-Area-1 STOP)	yes	yes	yes	yes	• all on AWO • all on Iso0	yes	yes	yes
RUN (Isolated-Area-1 DEEPSTOP)	yes	yes	yes	yes[c]	• all on AWO • all on Iso0	yes	yes	yes
STOP	yes	yes	yes	yes	• operating on AWO • operating on Iso0	yes	no	yes
STOP (Isolated-Area-1 DEEPSTOP)	yes	yes	yes	yes[c]	• all on AWO • operating on Iso0	yes	yes	yes
STOP (Isolated-Area-0 STOP)	yes	yes	yes	yes	• operating on AWO • operating on Iso0	yes	no	yes
DEEPSTOP	yes	yes	yes	yes[c]	• operating on AWO	yes	no	yes

a) The effective wake-up event must be available as a wake-up factor. Refer to the wake-up factor tables in the first section of this chapter under the key word *"Wake-up factors"*.

b) The functional module to issue a wake-up event must be supplied with its clock from the Clock Controller. If its clock domain is subject to the STOP mode, the stand-by mode request must be masked (CKSC_mn.STPMK_mn = 1), thus the clock for the functional module remains in operation during stand-by mode.

c) To use an external interrupt INTPx as a wake-up factor from DEEPSTOP mode, the INTPx analog filter must be set to edge detection mode (FCLAnINTm = 0). Refer to the section *"Port Filters Functional Description"* in *2 "Port Functions" on page 55* for details.

HALT mode wake-up

The wake-up events which terminate the HALT mode are described in the document "V850E2M 32-bit Microprocessor Core Architecture User's Manual".

External interrupts INTPx wake-up

All external interrupts INTPx can terminate all stand-by modes.
If an external interrupt INTPx shall be used to wake-up from a DEEPSTOP mode, the analog filter of INTPx must be configured for edge detection (FCLAnINTm = 0). Refer to the section *"Port Filters Functional Description"* in the *"Port Functions"* chapter for details.

CAN FCNnRX wake-up

A falling edge of a CAN reception signal FCNnRX can generate a wake-up from stand-by mode. For further information about CAN wake-up refer to the section "Stand-by Modes" in the chapter "CAN Controller (FCN)".
If FCNnRX wakes up from DEEPSTOP mode, the FCNnRX interrupt service routine is not processed. In this case the FCNnRX is only used as a wake-up factor, thus is not part of any data communication via the CAN interface.

Functional modules interrupt wake-up

Interrupts from a functional module can generate a wake-up, provided

- the functional module is not located on a power area that is in DEEPSTOP mode (power of this area is switched off) and
- the functional module is supplied with its operation clock (e.g. the function is located in a STOP mode area, but clock stop is masked by CKSC_mn.STPMK_m0 = 1) and
- the functional module event is enabled as a wake-up factor (refer to the wake-up factor tables in the first section of this chapter under the key word "Wake-up factors")

S/W wake-up

A S/W wake-up from RUN (Isolated-Area-1 STOP, Isolated-Area-1 DEEPSTOP) can be triggered by setting PSC1.PSC1ISOWU = 1.

On-Chip Debug wake-up

The On-Chip Debug unit (OCD) is generating a wake-up event while the microcontroller runs the application program in following cases:

- the debugger issues a stop request
- a breakpoint is hit

In either case any stand-by mode is terminated, provided the OCD debug event is enabled as a wake-up factor via the WUFMSKH register.

Caution

If the OCD wake-up event is disabled, it is not possible to wake-up the microcontroller from stand-by mode by a manual stop via the debugger. Thus it is recommended to enable the OCD wake-up for terminating all stand-by modes by setting

WUFMSKH0.WUFMSKH015 = WUFMSKH1.WUFMSKH115 = 0.

(2) Wake-up control

All wake-up events can be separately enabled to trigger a wake-up of Isolated-Area-0 and Isolated-Area-1 from stand-by mode by means of two sets of wake-up factors control registers:

- WUFMSKLm, WUFMSKMm, WUFMSKHm = WUFMSK[L,M,H]m
- WUFLm, WUFMm, WUFHm = WUF[L,M,H]m
- WUFCLm, WUFCMm, WUFCHm = WUFC[L,M,H]m,

where m = 0 denotes the registers for Isolated-Area-0 and m = 1 for Isolated-Area-1.

- Wake-up mask registers: WUFMSK[L,M,H]m
  Each bit of these registers is assigned to a certain wake-up event of Isolated-Area-m.
  Wake-up by this event is enabled if its mask bit WUFMSK[L,M,H]m.WUFMSK[L,M,H]m[31:0] is set to 0. Upon occurrence of the enabled wake-up event the Isolated-Area-m is woken up.
- Wake-up factor registers: WUF[L,M,H]m
  Upon occurrence of an unmasked wake-up event, the associated wake-up factor flag WUF[L,M,H]m.WUF[L,M,H]m[31:0] is set to 1.
  By use of this register the application program can identify the wake-up source.
- Wake-up factor clear registers: WUFC[L,M,H]m
  In order to reset a wake-up factor flag WUF[L,M,H]m[31:0] of a wake-up factor register, its assigned bit WUFC[L,M,H]m.WUFC[L,M,H]m[31:0] has to be set to 1.

Notes

1. The wake-up factor flags in the wake-up factors registers WUF[L,M,H]m indicate only the occurrence of a wake-up factor.
   Thus an asserted wake-up factor flag does not mean, that the transition from stand-by to operation mode of the concerned power domain is already accomplished.
   Refer to *3 “Stand-by status” on page 585* for information how to check the stand-by and operational status.
2. If an interrupt event can also be used as a wake-up event, and this wake-up event is not masked (WUFMSK[L,M,H]m[31:0] = 0), the occurrence of this interrupt event sets also its wake-up factor flag WUF[L,M,H]m[31:0], even if the device is not in stand-by mode.

(3) CPU wake-up interrupt factor processing

Depending on the stand-by mode, the CPU has to react on wake-up by an interrupt differently:

- Isolated-Area-0 not in DEEPSTOP mode
  If the microcontroller is woken up from any stand-by mode, where the Isolated-Area-0 was not in DEEPSTOP mode (i.e. the CPU was operating during stand-by), the wake-up interrupt event sets an interrupt request flag (ICn.RFn = 1), provided the interrupt event was not masked (ICn.MKn = 0). Thus normal interrupt acknowledge can be performed, after interrupt acknowledgement has been enabled by the “EI” instruction.
- Isolated-Area-0 in DEEPSTOP mode
  If the microcontroller is woken up from DEEPSTOP stand-by mode, (i.e. Isolated-Area-0 and thus also the CPU Subsystem was switched off during stand-by), the wake-up interrupt event is not recorded as an interrupt request, i.e. no interrupt request flag ICn.RFn = 1 is set.
  Thus the CPU has to read the wake-up factor registers WUF[L,M,H]m to evaluate the wake-up interrupt event and proceed accordingly.

(4) Oscillators wake-up

The MainOsc and SubOsc can be automatically started with the wake-up of an isolated area.
For that purpose the Stand-by Controller generates the wake-up oscillator start signal WUOSCSTA towards the Clock Controller, that starts the MainOsc and SubOsc.

Every wake-up factor, i.e. a wake-up event with its mask WUFMSK[L,M,H]mn cleared, can start the MainOsc and SubOsc.

Control bits in the OSCWUFMSK register allow to select the isolated area wake-up to start also the oscillators:

- OSCWUFMSK.OSCWUFMSK00 = 0
  enables the MainOsc and SubOsc to start operation with wake-up of the Isolated-Area-0.
- OSCWUFMSK.OSCWUFMSK01 = 0
  enables the MainOsc and SubOsc to start operation with wake-up of the Isolated-Area-1.

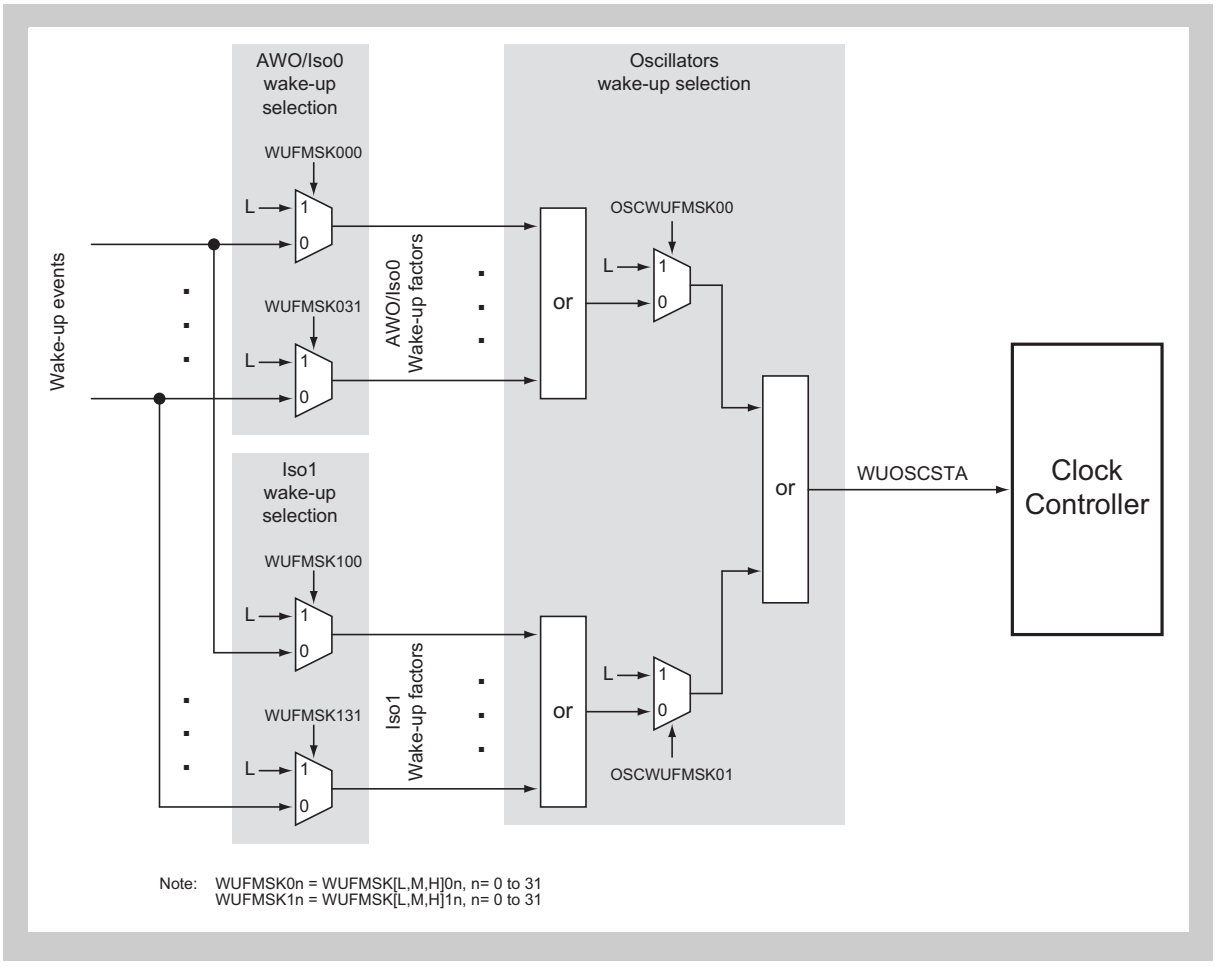


Figure 10-2 MainOsc and SubOsc start and wake-up

Note The oscillators can be stopped only in Isolated-Area-0 stand-by modes, i.e. if the STPREQ_0 is generated.

(a) MainOsc and/or SubOsc operating before Isolated-Area-0 stand-by

If the MainOsc and/or SubOsc were operating before Isolated-Area-0 stand-by, they are operating also after wake-up.

The behaviour of the MainOsc and SubOsc during stand-by mode is determined by the oscillator's stop masks MOSCE.MOSCESTPMK respectively SOSCE.SOSCESTPMK.

Note The oscillator's start refers to the start of their stabilization counter.

STOP mode In case of STOP mode the oscillators wake-up masks OSCWUFMSK00 and OSCWUFMSK01 have no influence.
If stopped during STOP mode, the oscillators are started at wake-up of Isolated-Area-0.

Table 10-10 Oscillator operating before STOP mode

OSCSTPMSK [a]	During STOP	OSCWUFMSK		Wake-up		After wake-up
		00	01	Wake-up factor to	Comment	
1	operating	X	X	Isolated-Area-0[b]	oscillators remain in operation	operating
0	stopped				oscillators are restarted at wake-up	

a) OSCSTPMSK stands for MOSCE.MOSCESTPMSK for the MainOsc and SOSCE.SOSCESTPMSK for the SubOsc.

b) A wake-up factor to Isolated-Area-1 does not restart the oscillators.

DEEPSTOP mode In case of DEEPSTOP mode the oscillators wake-up masks OSCWUFMSK00 and OSCWUFMSK01 have an influence on the time the oscillators are started, as shown in the figure below.

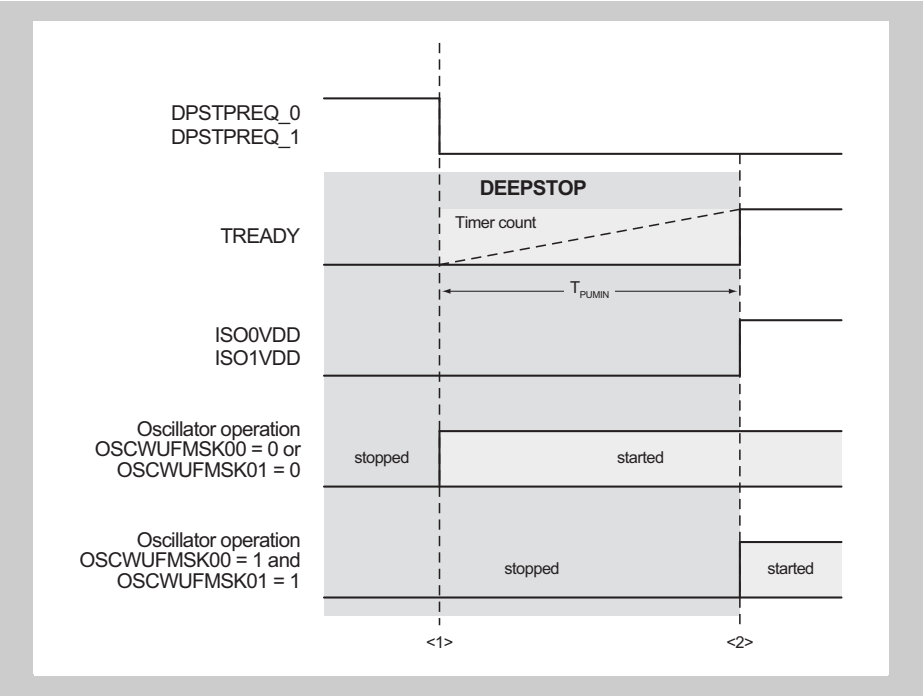


Figure 10-3 OSCWUFMSK effect on oscillators start time

<1> If any of the oscillator wake-up factor masks is not set (OSCWUFMSK00 or OSCWUFMSK01 is 0), the oscillators are started upon DEEPSTOP wake-up (DEEPSTOP request signals DPSTPREQ_0 and DPSTPREQ_1 are de-asserted).
Thus the oscillators are started before the Power Sequencer completes the power-up procedure of the isolated areas.

<2> If the oscillator wake-up factor masks are set (OSCWUFMSK00 and OSCWUFMSK01 is 1), the oscillators are started if the power supply of the isolated areas (ISO0VDD/ISO1VDD) are switched on and they are released from reset.
Thus the oscillators are started after the Power Sequencer completes the power-up procedure of the isolated areas.

Refer to *37 "Power Supply" on page 2602* for detailed information about the Power Sequencer and the wake-up procedures from DEEPSTOP mode.

Table 10-11 Oscillator operating before DEEPSTOP mode

OSCSTPMSK [a]	During DEEPSTOP	OSCWUFMSK		Wake-up		After wake-up
		00	01	Wake-up factor to	Comment	
1	operating	X	X	Isolated-Area-0	oscillators remain in operation	operating
0	stopped	1	1	Isolated-Area-0 Isolated-Area-1[b]	oscillators are restarted at isolated areas power on (<2>)	
		0	1		oscillators are restarted at wake-up (<1>)	
		1	0			
		0	0			

a) OSCSTPMSK stands for MOSCE.MOSCESTPMSK for the MainOsc and SOSCE.SOSCESTPMSK for the SubOsc.

b) Wake-up from DEEPSTOP mode affects always Isolated-Area-0 and Isolated-Area-1.

(b) MainOsc and/or SubOsc stopped before Isolated-Area-0 stand-by

If the MainOsc and/or SubOsc were stopped before Isolated-Area-0 stand-by, they remain stopped in stand-by in any case.
The oscillator's stop masks MOSCE.MOSCESTPMK respectively SOSCE.SOSCESTPMK have no impact.

Depending on the oscillators wake-up mask OSCWUFMSK00 and OSCWUFMSK01 and the isolated area, that is woken up, the oscillators may remain stopped or start automatically, as shown in the table below:

Table 10-12 Oscillator stopped before Isolated-Area-0 stand-by

OSCSTPMSK [a]	During Iso0 stand-by	OSCWUFMSK		Wake-up		After wake-up
		00	01	Wake-up factor to	Comment	
X	stopped	1	1	Isolated-Area-0	oscillators wake-up masked, no automatic start	stopped
				Isolated-Area-1	oscillators wake-up masked, no automatic start	stopped
		0	1	Isolated-Area-0	oscillators wake-up unmasked, automatic start at wake-up	operating
				Isolated-Area-1	oscillators wake-up masked, no automatic start	stopped
		1	0	Isolated-Area-0	oscillators wake-up masked, no automatic start	stopped
				Isolated-Area-1	oscillators wake-up unmasked, automatic start at wake-up	operating
		0	0	Isolated-Area-0	oscillators wake-up unmasked, automatic start at wake-up	operating
				Isolated-Area-1	oscillators wake-up unmasked, automatic start at wake-up	operating

a) OSCSTPMSK stands for MOSCE.MOSCESTPMSK for the MainOsc and SOSCE.SOSCESTPMSK for the SubOsc.

For details about the MainOsc and SubOsc control refer to the description of the MainOsc and SubOsc in *9 "Clock Controller" on page 420* .

10.2.6 I/O buffer control

This section describes the behaviour of the I/O buffers during various stand-by modes.

Note The buffers of the port groups P0 and JP0, which reside on the Always-On-Area, always remain in their active state, independent of any stand-by mode.

(1) I/O buffer hold state

If an I/O buffer is set into I/O buffer hold state, the state of the buffer is frozen. Thus its input and/or output remains in the state before entering I/O buffer hold state. No external or internal signal can change its state, until the I/O buffer hold state is terminated.

Besides I/O buffer hold states during stand-by modes, the application software can also set all buffers of the Isolated-Area-m into I/O buffer hold state via the power save control registers PSCm:

- PSCm.PSCmIOHLDSET = 1:
  Isolated-Area-m buffers enter I/O buffer hold state
- PSCm.PSCmIOHLDSET = 0:
  Isolated-Area-m buffers terminate I/O buffer hold state

Note The I/O buffers of the port groups P0 and JP0, which are located on the Always-On-Area, do not support the I/O hold function.

(2) I/O buffers during STOP mode

The I/O buffers of areas in STOP mode (clock has been stopped) remain in their configuration. Since the peripheral or port function, that is connected to the I/O buffer, also remains in the state before entering STOP mode, it still controls the I/O buffer.

(3) I/O buffers during DEEPSTOP mode

The I/O buffers of the isolated area in DEEPSTOP are changing into I/O buffer hold state by default, thus the buffer status is not changed.

After wake-up the user application has to re-configure the peripheral or port function, that generates the signals connected to a certain I/O buffer. Afterwards the I/O buffer hold state has to be terminated by setting PSCm.PSCmIOHLDCLR = 1.
After setting this I/O buffer hold clear function trigger the concerned I/O buffer operates as configured by the peripheral or port function.

The automatic change to I/O buffer hold state upon entering DEEPSTOP mode can be suppressed via the power save control registers PSCm:

- PSCm.PSCmIOHLDMSK = 0:
  Isolated-Area-m buffers *enter* I/O buffer hold state in DEEPSTOP
- PSCm.PSCmIOHLDMSK = 1:
  Isolated-Area-m buffers *do not enter* I/O buffer hold state in DEEPSTOP

10.2.7 Mode transitions

Note Following abbreviations are used in the figure above:

- AWO: Always-On-Area
- Iso0: Isolated-Area-0
- Iso1: Isolated-Area-1

(1) Stand-by mode transitions

The following diagram shows the possible transition between the various operation and stand-by modes.

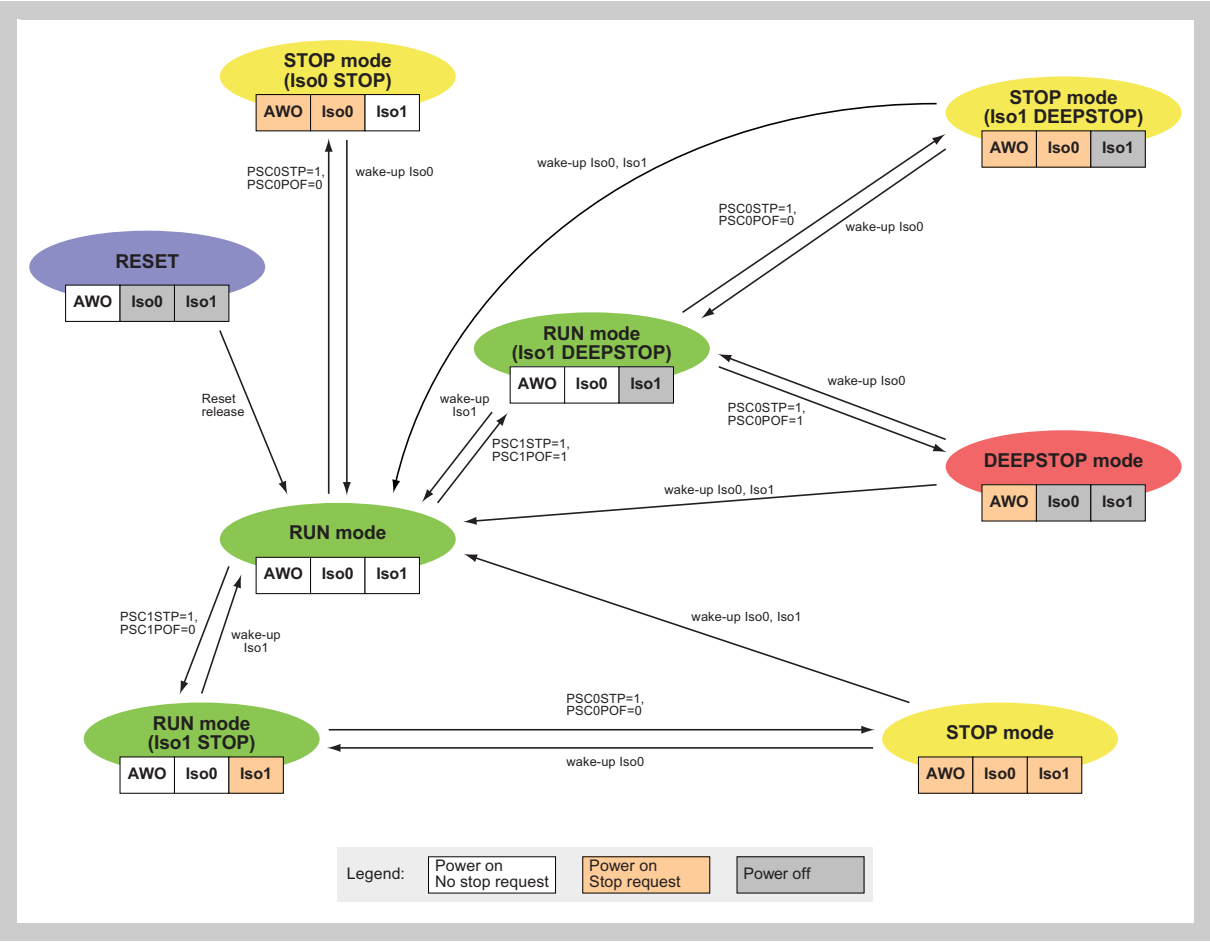


Figure 10-4 Stand-by mode transitions

10.3 Stand-by mode entry and exit example flows

In the following some recommended example flows are shown how to enter and exit stand-by modes.

Cautions

1. Make sure that all modules, which are subject to a stand-by mode are not in operation before starting stand-by mode preparations.
2. If you intend to stop any clock sources (High Speed IntOsc, MainOsc, SubOsc, PLLk) before entering any stand-by mode, make sure to meet one of the following conditions:
   - Make sure to switch clock domains using a clock source to be stopped to another clock source, that remains active.
   - Stop the respective clock domains by setting the CKSC_mn register to 0.

10.3.1 STOP mode

In STOP mode selectable domain clocks of the Always-On-Area (CKSCLK_An), Isolated-Area-0 (CKSCLK_0n) and Isolated-Area-1 (CKSCLK_1n) can be stopped.

(1) STOP preparation

Depending on the succession of the STOP mode preparation steps, early occurring wake-up events, to be used as valid wake-up factors, may be discarded or saved as wake-up factors.
Here only one option for preparing the STOP mode is described. Refer to the section *"Application hint: Handling of wake-up events during stand-by mode preparation"* below for other options.

DMA/interrupt stop

Stop all DMA Controller (DMAC) and interrupt activities.

Interrupts

Disable interrupt acknowledgement by the CPU instruction "DI".

Interrupt handling, in particular interrupt masking, has no influence on the wake-up behaviour. However if wake-up interrupts shall launch automatic interrupt acknowledgement after wake-up, proceed as follows:

Clear the interrupt flags (ICn.RFn = 0) and

- mask non-wake-up interrupts (ICn.MKn = 1)
- unmask wake-up interrupts (ICn.MKn = 0).

Prepare domain clocks

Select the clock domains to stop or continue operation during STOP. Set the clock stop masks:

- CKSC_mn.STPMK_mn = 1:
  domain clock CKSCLK_mn operates during STOP mode
- CKSC_mn.STPMK_mn = 0:
  domain clock CKSCLK_mn stops during STOP mode.

Prepare clock generators

- Choose clock generators to operate respectively stop during stand-by mode (by MOSCE.MOSCESTPMK, SOSCE.SOSCESTPMK, ROSCE.ROSCESTPMK, PLLEk.PLLEkSTPMK bits).
- Choose if the MainOsc and SubOsc shall be automatically started upon a wake-up from selected isolated areas via the OSCWUFMSK register.

Note Make sure that clock generators, used as the clock source for domain clocks CKSCLK_mn, which shall not stop in stand-by mode (CKSC_mn.STPMK_mn = 1), must also not be stopped in stand-by.

Caution If a clock generator shall be stopped during stand-by mode and shall automatically resume operation after wake-up, restart of the clock generator may fail under certain conditions. Refer to the section *"Precaution: Clock generators and early wake-up"* below in this chapter for details.

Wake-up factors

- Clear wake-up factor flags in the WUF[L,M,H]m registers (by WUFC[L,M,H]mn = 1 in WUFC[L,M,H]m registers) and
  - mask non-wake-up factors (WUFMSK[L,M,H]mn = 1 in WUFMSK[L,M,H]m registers)
  - unmask wake-up factors to allow them to issue a wake-up (WUFMSK[L,M,H]mn = 0 in WUFMSK[L,M,H]m registers).

Caution If any previously set wake-up factor flag remains set in WUF[L,M,H]m when starting transition to stand-by mode, an early wake-up will immediately occur and the microcontroller will not enter stand-by mode.

(2) STOP entry

Iso1 STOP entry

- Set PSC1.PSC1POF = 0 and PSC1.PSC1STP = 1 to trigger Isolated-Area-1 transition to STOP mode.

Caution The CPU must not access any register of an Isolated-Area-1 module during Isolated-Area-1 STOP mode.
Any attempt to access these registers will lead to a microcontroller deadlock.
This is valid even if the clock supply of these modules have not been stopped in STOP mode, due to a set clock stop mask CSCK_1n.STPMK_1n.

Iso0 STOP entry

- Set PSC0.PSC0POF = 0 and PSC0.PSC0STP = 1 to trigger Isolated-Area-0 transition to STOP mode.
- Since PWS0.PWS0PSS turns immediately to "1" after PSC0STP = 1, run an infinite loop by waiting for PWS0.PWS0PSS = 0.
- While the CPU remains in the infinite loop, transition to Isolated-Area-0 STOP starts.
- When transition to STOP mode is completed, all domain clocks are stopped, if their stop mask is not set (CKSC_mn.STPMK_mn = 0).
  The CPU is stopped in any case.

(3) Wake-up before STOP completion (early wake-up)

If a wake-up occurs before the transition to DEEPSTOP is completed, PWS0.PWS0PSS is set to 0 and the infinite loop is left, before the Isolated-Area-0 set in STOP mode. Wake-up service can start immediately.

Caution If a clock generator shall be stopped during stand-by mode and shall automatically resume operation after wake-up, restart of the clock generator may fail under certain conditions. Refer to the section *"Precaution: Clock generators and early wake-up"* below in this chapter for details.

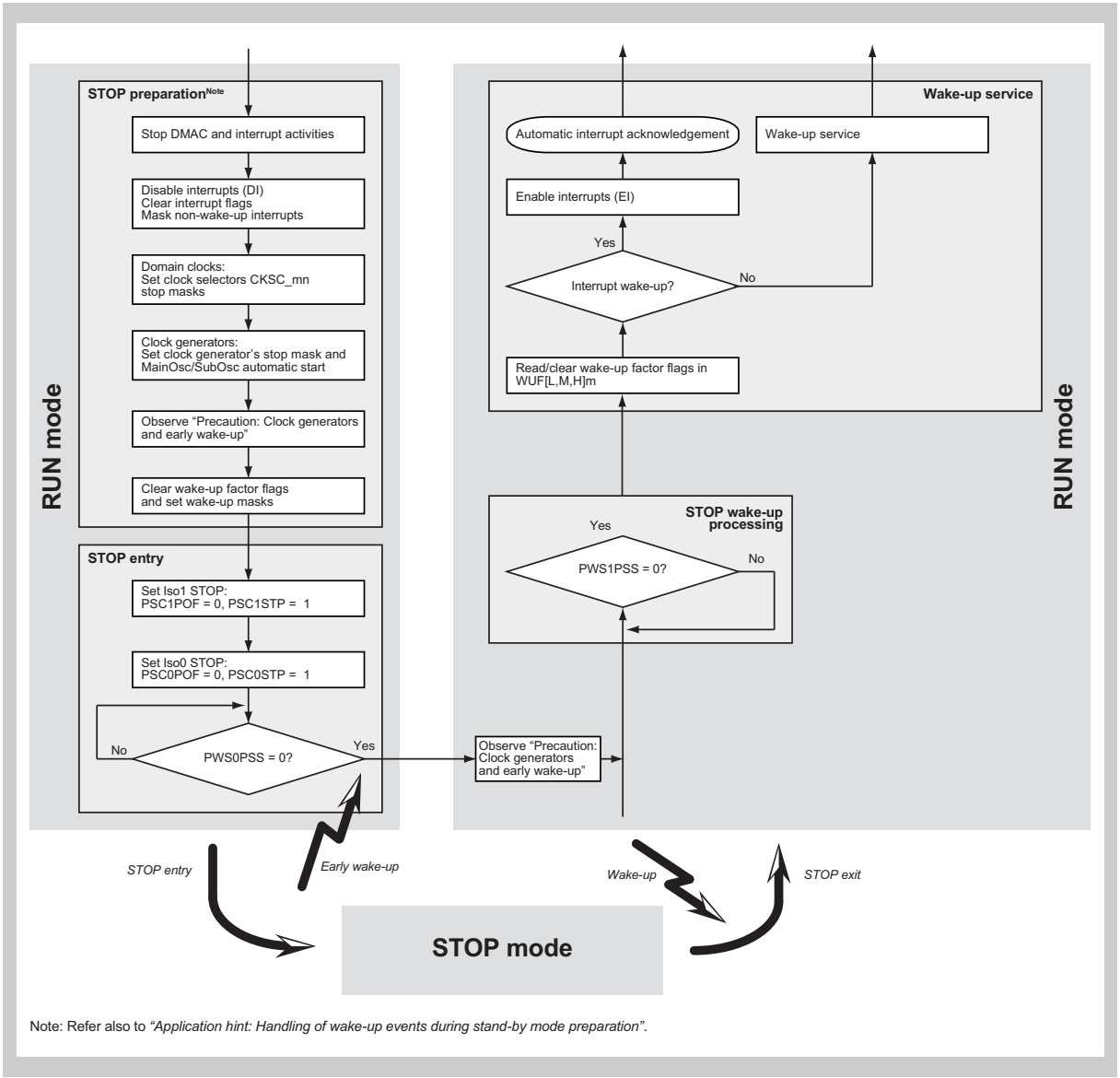
(4) STOP Wake-up processing

- Wait until Isolated-Area-1 is woken up, indicated by PWS1.PWS1PSS = 0.

(5) Wake-up service

- The wake-up factor is evaluated via the wake-up factor flags, which shall be cleared afterwards.
- If the wake-up factor was an interrupt, the interrupt request is stored in the Interrupt Controller and can be acknowledged after interrupt acknowledgement is enabled by the “EI” instruction, if it is unmasked.
- If the wake-up factor was not an interrupt, the wake-up service routine may be called.

Note Make sure to clear the oscillator OSCWUF wake-up flags via the OSCWUFC register.



Note: Refer also to *“Application hint: Handling of wake-up events during stand-by mode preparation”*.

Figure 10-5 Recommended flow of STOP mode

10.3.2 RUN mode (Isolated-Area-1 STOP)

In RUN mode (Isolated-Area-1 STOP) selectable domain clocks of the Isolated-Area-1 (CKSCLK_1n) can be stopped.

(1) Iso1 STOP preparation

Depending on the succession of the STOP mode preparation steps, early occurring wake-up events, to be used as valid wake-up factors, may be discarded or saved as wake-up factors.
Here only one option for preparing the STOP mode is described. Refer to the section *"Application hint: Handling of wake-up events during stand-by mode preparation"* below for other options.

DMA/interrupt stop

- Stop all DMA Controller (DMAC) and interrupt activities with regard to Isolated-Area-1 modules.

Interrupts

Interrupt handling, in particular interrupt masking, has no influence on the wake-up behaviour.
Since the CPU continues operation in RUN mode (Isolated-Area-1 STOP), all interrupts can be served during stand-by mode.
However if an interrupt shall not perform a wake-up, it must be masked by the wake-up factor mask registers WUFMSK[L,M,H]m.
If wake-up interrupts shall launch automatic interrupt acknowledgement and perform the wake-up processing, clear the wake-up interrupt flags (ICn.RFn = 0) and unmask wake-up interrupts (ICn.MKn = 0).

Prepare domain clocks

- Select the Isolated-Area-1 clock domains to stop or continue operation during STOP. Set the clock stop masks:
  - CKSC_mn.STPMK_1n = 1:
    domain clock CKSCLK_1n operates during STOP mode
  - CKSC_mn.STPMK_1n = 0:
    domain clock CKSCLK_1n stops during STOP mode.

Wake-up factors

- Clear wake-up factor flags in the WUF[L,M,H]m registers (by WUFC[L,M,H]mn = 1 in WUFC[L,M,H]m registers) and
  - mask non-wake-up factors (WUFMSK[L,M,H]mn = 1 in WUFMSK[L,M,H]m registers)
  - unmask wake-up factors to allow them to issue a wake-up (WUFMSK[L,M,H]mn = 0 in WUFMSK[L,M,H]m registers).

Caution

If any previously set wake-up factor flag remains set in WUF[L,M,H]m when starting transition to stand-by mode, an early wake-up will immediately occur and the microcontroller will not enter stand-by mode.

(2) Iso1 STOP entry

- Set PSC1.PSC1POF = 0 and PSC1.PSC1STP = 1 to trigger Isolated-Area-1 transition to STOP mode.

The CPU remains in operation.

Caution The CPU must not access any register of an Isolated-Area-1 module during Isolated-Area-1 STOP mode.
Any attempt to access these registers will lead to a microcontroller deadlock.
This is valid even if the clock supply of these modules have not been stopped in STOP mode, due to a set clock stop mask CSCK_1n.STPMK_1n.

(3) Iso1 STOP Wake-up processing

- Wait until Isolated-Area-1 is woken up, indicated by PWS1.PWS1PSS = 0.

(4) Wake-up service

- The wake-up factor is evaluated via the wake-up factor flags, which can be cleared afterwards.
- If the wake-up factor was an interrupt, the interrupt request is stored in the Interrupt Controller and can be acknowledged after interrupt acknowledgement is enabled by the “EI” instruction, if it is unmasked.
- If the wake-up factor was not an interrupt, the wake-up service routine is called.

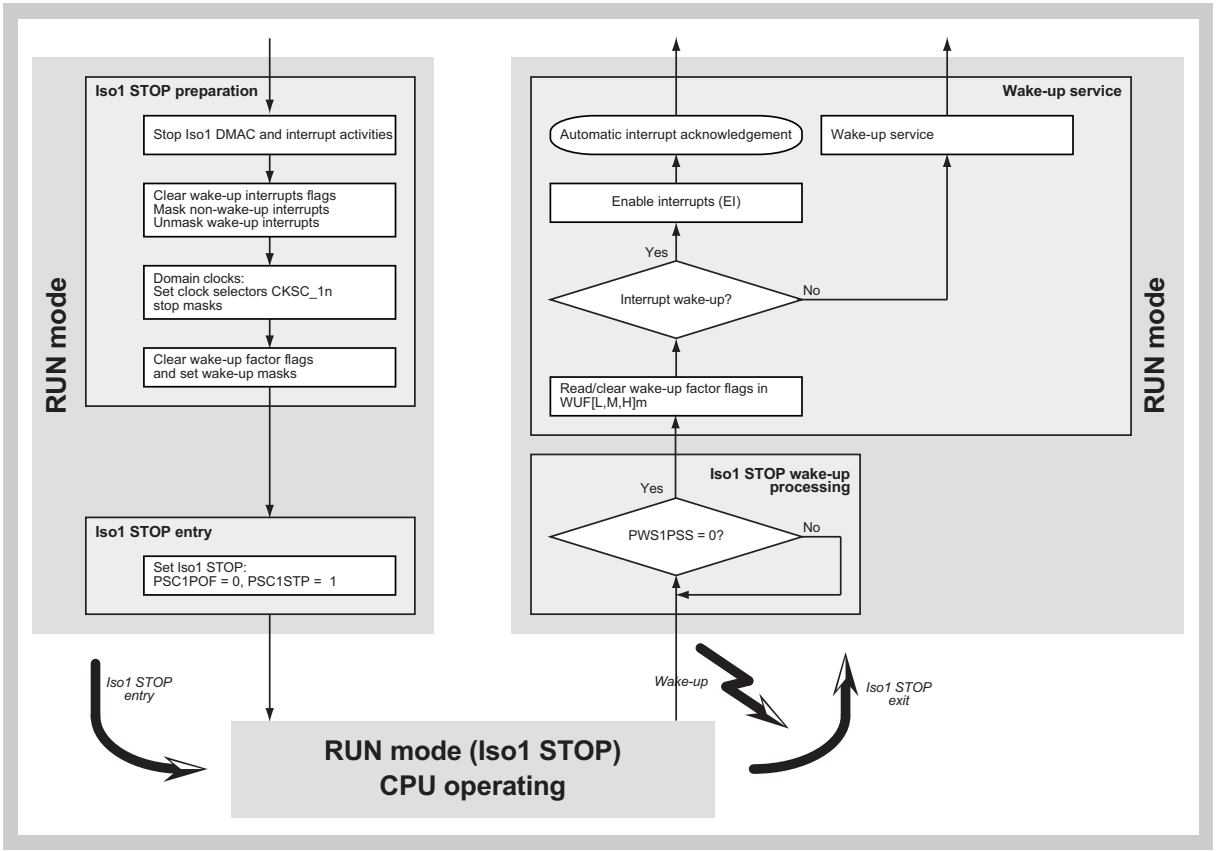


Figure 10-6 Recommended flow of RUN mode (Isolated-Area-1 STOP)

(5) S/W wake-up

The RUN mode (Iso1 STOP) can also be terminated by a S/W wake-up trigger PSC1.PSC1ISOWU = 1.

- Set PSC1.PSC1ISOWU = 1 to trigger Isolated-Area-1 wake-up from STOP mode.
- Wait until Isolated-Area-1 is woken up, indicated by PWS1.PWS1PSS = 0.

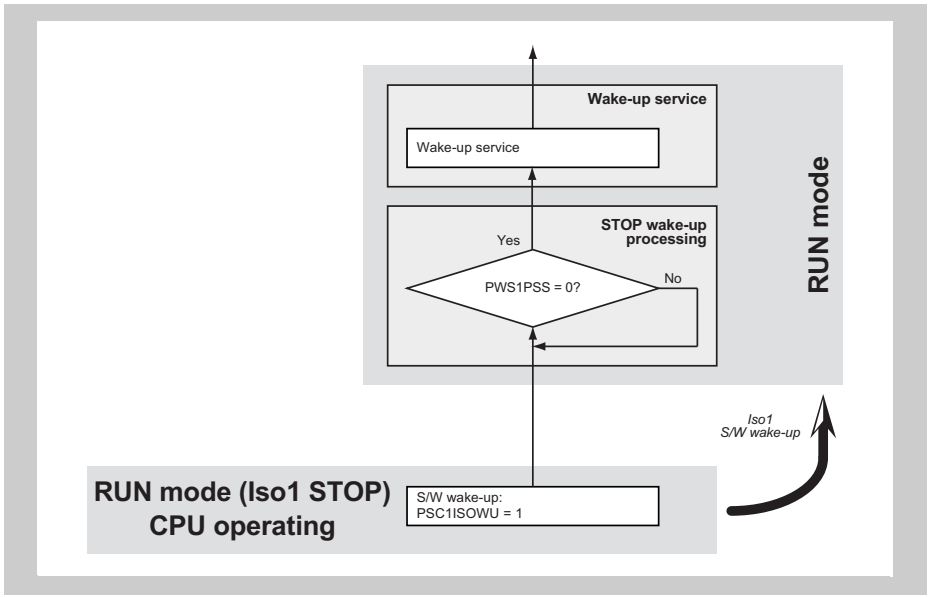


Figure 10-7 Isolated-Area-1 wake-up by software

10.3.3 DEEPSTOP mode

In DEEPSTOP mode the power supply of the isolated areas is switched off. Selectable Always-On-Area clock domains CKSCLK_An are stopped. Isolated-Area-1 must be switched off before Isolated-Area-0.

(1) DEEPSTOP preparation

Depending on the succession of the DEEPSTOP mode preparation steps, early occurring wake-up events, to be used as valid wake-up factors, may be discarded or saved as wake-up factors.
Here only one option for preparing the STOP mode is described. Refer to the section *"Application hint: Handling of wake-up events during stand-by mode preparation"* below for other options.

DMA/interrupt stop

Stop all DMA Controller (DMAC) and interrupt activities.

Interrupts

Disable interrupt acknowledgement by the CPU instruction "DI".

Interrupt handling, in particular interrupt masking, has no influence on the wake-up behaviour. However if wake-up interrupts shall launch automatic interrupt acknowledgement under certain wake-up conditions (refer to *3 "Wake-up before DEEPSTOP completion (early wake-up)" on page 607*), proceed as follows:

Clear the interrupt flags (ICn.RFn = 0) and

- mask non-wake-up interrupts (ICn.MKn = 1)
- unmask wake-up interrupts (ICn.MKn = 0).

Prepare Iso0 clocks

Change the clock source ID of the CPUCLK domain clock CPUCLK = CKSCLK_000 to the High Speed/Low Speed IntOsc, i.e. set CKSC_000.CKSCID_000 = $3A_H$.

Change all Isolated-Area-0 domain clocks CKSCLK_0n

- to a currently active clock sources, which is not derived from any PLL, or
- stop the respective clock domains by setting the CKSC_0n register to 0.

Prepare AWO clocks

- Select the Always-On-Area clock domains to stop or continue operation during DEEPSTOP. Set the clock stop masks:
  - CKSC_An.STPMK_mn = 1:
    domain clock CKSCLK_An operates during DEEPSTOP mode
  - CKSC_An.STPMK_mn = 0:
    domain clock CKSCLK_An stops during DEEPSTOP mode.
- Change all Always-On-Area domain clocks CKSCLK_An to currently active clock sources, which are not derived from any PLL, or stop the clock domains by CKSC_An.CKSCID_An = 0000_H.

Prepare clock generators

- Choose clock generators to operate respectively stop during stand-by mode (by MOSCE.MOSCESTPMK, SOSCE.SOSCESTPMK, ROSCE.ROSCESTPMK, PLLEk.PLLEkSTPMK bits).
- Choose if the MainOsc and SubOsc shall be automatically started upon a wake-up from selected isolated areas via the OSCWUFMSK register.

Note Make sure that clock generators, used as the clock source for domain clocks CKSCLK_mn, which shall not stop in stand-by mode (CKSC_mn.STPMK_mn = 1), must also not be stopped in stand-by.

Caution If a clock generator shall be stopped during stand-by mode and shall automatically resume operation after wake-up, restart of the clock generator may fail under certain conditions. Refer to the section *"Precaution: Clock generators and early wake-up"* below in this chapter for details.

Iso1 clocks Before the Isolated-Area-1 is set in DEEPSTOP mode, all modules on this power domain have to be stopped (CKSC_1n = 0) or supplied with a currently active clock CKSCLK_1n.
However since Iso1 DEEPSTOP does not affect any clock generator, the current clock configuration does not need to be changed.

Wake-up factors

- Clear wake-up factor flags in the WUF[L,M,H]m registers (by WUFC[L,M,H]mn = 1 in WUFC[L,M,H]m registers) and
  - mask non-wake-up factors (WUFMSK[L,M,H]mn = 1 in WUFMSK[L,M,H]m registers)
  - unmask wake-up factors to allow them to issue a wake-up (WUFMSK[L,M,H]mn = 0 in WUFMSK[L,M,H]m registers).

Caution If any previously set wake-up factor flag remains set in WUF[L,M,H]m when starting transition to stand-by mode, an early wake-up will immediately occur and the microcontroller will not enter stand-by mode.

(2) DEEPSTOP entry

Iso1 DEEPSTOP entry

- Set PSC1.PSC1POF = PSC1.PSC1STP = 1 to trigger Isolated-Area-1 transition to DEEPSTOP mode.
- Evaluate the wake-factor flags in the WUF[L,M,H]m registers.
  If any of the wake-up factor flags WUF[L,M,H]mn = 1, a wake-up has already occurred and wake-up processing can be performed.
- Evaluate PWS1.PWS1PSS and PWS1.PWS1ISO:
  - If PWS1.PWS1PSS ≠ 1 or PWS1.PWS1ISO ≠ 0, the microcontroller has not completely entered Iso1 DEEPSTOP mode.
  - If PWS1.PWS1PSS = 1 and PWS1.PWS1ISO = 0, the microcontroller has completely entered Iso1 DEEPSTOP mode.
    If no wake-up has occurred, Isolated-Area-1 is in DEEPSTOP mode.

Caution The CPU must not access

- any register of an Isolated-Area-1 module
- any Isolated-Area-1 clock selectors registers CKSC_1n and CSCSTAT_1n during Isolated-Area-1 DEEPSTOP mode.

Any attempt to access these registers will lead to a microcontroller deadlock.

Iso0 DEEPSTOP entry

- Set PSC0.PSC0POF = PSC0.PSC0STP = 1 to trigger Isolated-Area-0 transition to DEEPSTOP mode.
- Since PWS0.PWS0PSS turns immediately to "1" after PSC0STP = 1, run an infinite loop by waiting for PWS0.PWS0PSS = 0.
- While the CPU remains in the infinite loop, transition to Isolated-Area-0 DEEPSTOP starts.

(3) Wake-up before DEEPSTOP completion (early wake-up)

If a wake-up occurs before the transition to DEEPSTOP is completed, PWS0.PWS0PSS is set to 0 and the infinite loop is left, before the Isolated-Area-0 power supply is switched off.
Proceed as described in *5 “DEEPSTOP early wake-up processing” on page 608* .

Caution If a clock generator shall be stopped during stand-by mode and shall automatically resume operation after wake-up, restart of the clock generator may fail under certain conditions. Refer to the section *“Precaution: Clock generators and early wake-up”* below in this chapter for details.

(4) Wake-up after DEEPSTOP completion

When transition to DEEPSTOP mode is completed, the power supply for Isolated-Area-0, i.e. also for the CPU, is switched off. Thus the CPU starts from its reset state upon wake-up.

- The reset cause can be evaluated in the following succession:
  - If any reset flag in the RESF register is set, the reset flag in RESF indicates the reset cause.
  - If no wake-up factor flag is set in WUF[L,M,H]m registers, a power-up reset (Power-On-Clear or debugger reset) occurred.
  - If non of the above reset causes apply, the reset was caused by a DEEPSTOP wake-up.

(5) DEEPSTOP early wake-up processing

- Before proceeding make sure that the Isolated-Area-1 is completely woken up from DEEPSTOP mode by verifying that PWS1.PWS1PSS = 0 and PWS1.PWS1ISO = 1.
- Since the isolated areas were reset, the CPU conducts complete initialization of these areas.

Iso0 ports

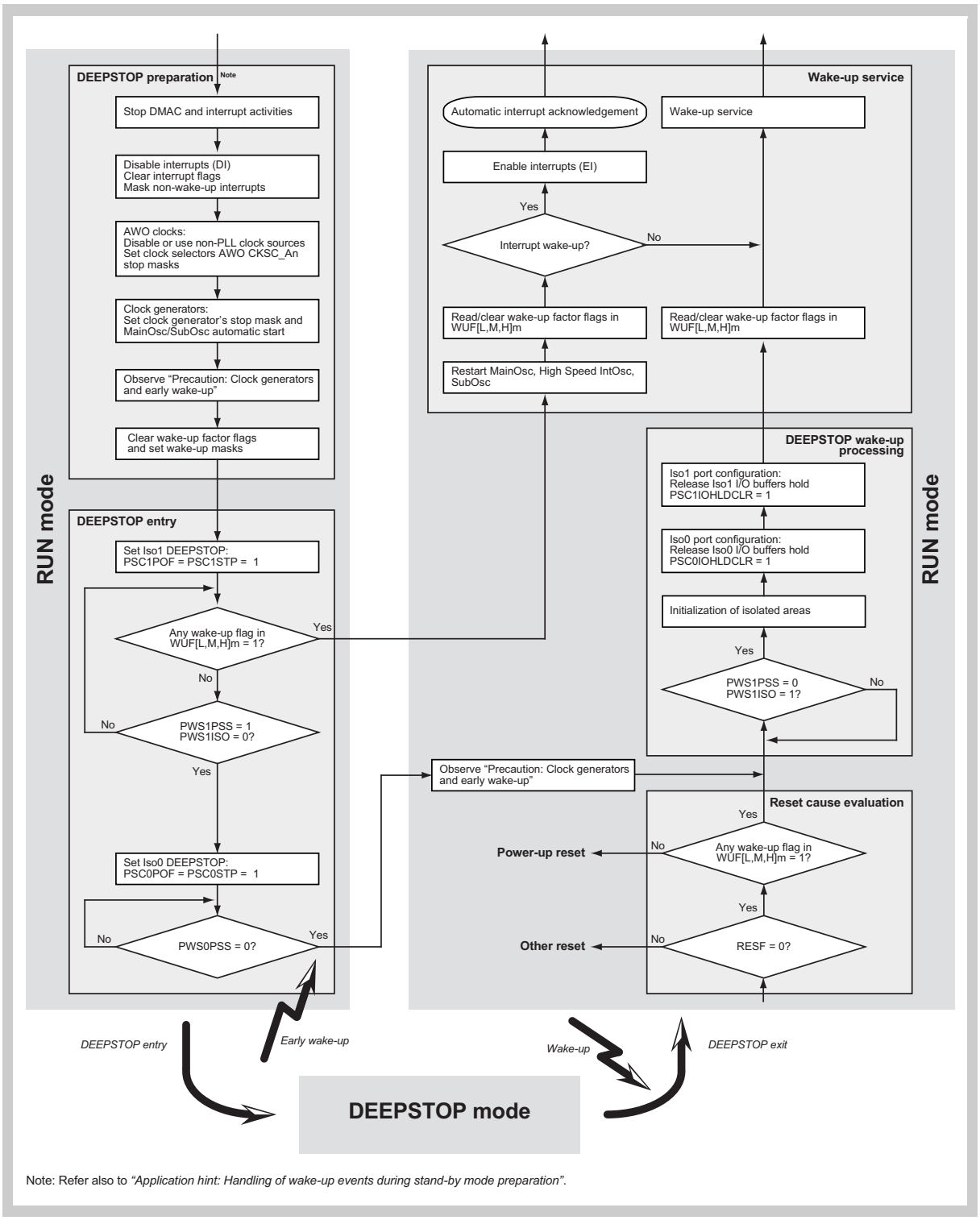
- Isolated-Area-0 port buffer operation is resumed by
  - configuration of the Iso0 ports
  - release of the Iso0 I/O buffers hold (PSC0.PSC0IOHLDCLR = 1)

Iso1 ports

- Isolated-Area-1 port buffer operation is resumed by
  - configuration of the Iso1 ports
  - release of the Iso1 I/O buffers hold (PSC1.PSC1IOHLDCLR = 1)

Wake-up service

- The wake-up factor is evaluated via the wake-up factor flags, which can be cleared afterwards.
- According to the evaluated wake-up factor from the WUF[L,M,H]m registers, the user program may have to serve the concerned wake-up.
  Note that also a wake-up interrupt does not generate an interrupt request towards the CPU after wake-up from DEEPSTOP mode, so an interrupt is not acknowledged automatically after enabling interrupt acknowledgement by the “EI” CPU instruction.



Note: Refer also to *"Application hint: Handling of wake-up events during stand-by mode preparation"*.

Figure 10-8 Recommended flow of DEEPSTOP mode

10.3.4 RUN mode (Isolated-Area-1 DEEPSTOP)

In RUN mode (Isolated-Area-1 DEEPSTOP) all CKSCLK_1n clocks and the power supply of Isolated-Area-1 are stopped.

(1) Iso1 DEEPSTOP preparation

Depending on the succession of the Iso1 DEEPSTOP mode preparation steps, early occurring wake-up events, to be used as valid wake-up factors, may be discarded or saved as wake-up factors.
Here only one option for preparing the Iso1 DEEPSTOP mode is described. Refer to the section *"Application hint: Handling of wake-up events during stand-by mode preparation"* below for other options.

DMA/interrupt stop

- Stop all DMA Controller (DMAC) and interrupt activities with regard to Isolated-Area-1 modules.

Interrupts

Interrupt handling, in particular interrupt masking, has no influence on the wake-up behaviour.
Since the CPU continues operation in RUN mode (Isolated-Area-1 STOP), all interrupts can be served during stand-by mode.
However if an interrupt shall not perform a wake-up, it must be masked by the wake-up factor mask registers WUFMSK[L,M,H]m.
If wake-up interrupts shall launch automatic interrupt acknowledgement at wake-up, proceed as follows:

- Clear the wake-up interrupt flags (ICn.RFn = 0) and
  - mask non-wake-up interrupts (ICn.MKn = 1)
  - unmask wake-up interrupts (ICn.MKn = 0).

Wake-up factors

- Clear wake-up factor flags in the WUF[L,M,H]m registers (by WUFC[L,M,H]mn = 1 in WUFC[L,M,H]m registers) and
  - mask non-wake-up factors (WUFMSK[L,M,H]mn = 1 in WUFMSK[L,M,H]m registers)
  - unmask wake-up factors to allow them to issue a wake-up (WUFMSK[L,M,H]mn = 0 in WUFMSK[L,M,H]m registers).

Caution If any previously set wake-up factor flag remains set in WUF[L,M,H]m when starting transition to stand-by mode, an early wake-up will immediately occur and the microcontroller will not enter stand-by mode.

(2) Iso1 DEEPSTOP entry

- Set PSC1.PSC1POF = 1 and PSC1.PSC1STP = 1 to trigger Isolated-Area-1 transition to DEEPSTOP mode.
- Wait until Isolated-Area-1 has completely entered DEEPSTOP mode, indicated by PWS1.PWS1PSS = 1 and PWS1,PWS1ISO = 0.

The CPU remains in operation.

Caution The CPU must not access

- any register of an Isolated-Area-1 module
- any Isolated-Area-1 clock selectors registers CKSC_1n and CSCSTAT_1n during Isolated-Area-1 DEEPSTOP mode.

Any attempt to access these registers will lead to a microcontroller deadlock.

(3) Iso1 DEEPSTOP Wake-up processing

- Wait until Isolated-Area-1 is woken up, indicated by PWS1.PWS1PSS = 0 and PWS1,PWS1ISO = 1.

Iso1 ports

- Isolated-Area-1 port buffer operation is resumed by
  - configuration of the Iso1 ports
  - release of the Iso1 I/O buffers hold (PSC1.PSC1IOHLDCLR = 1)

Since the Isolated-Area-1 was reset, it has to be completely re-initialized.

(4) Wake-up service

- The wake-up factor is evaluated via the wake-up factor flags, which can be cleared afterwards.
- If the wake-up factor was an interrupt, the interrupt request is stored in the Interrupt Controller and can be acknowledged after interrupt acknowledgement is enabled by the “EI” instruction, if it is unmasked.
- If the wake-up factor was not an interrupt, the wake-up service routine is called.

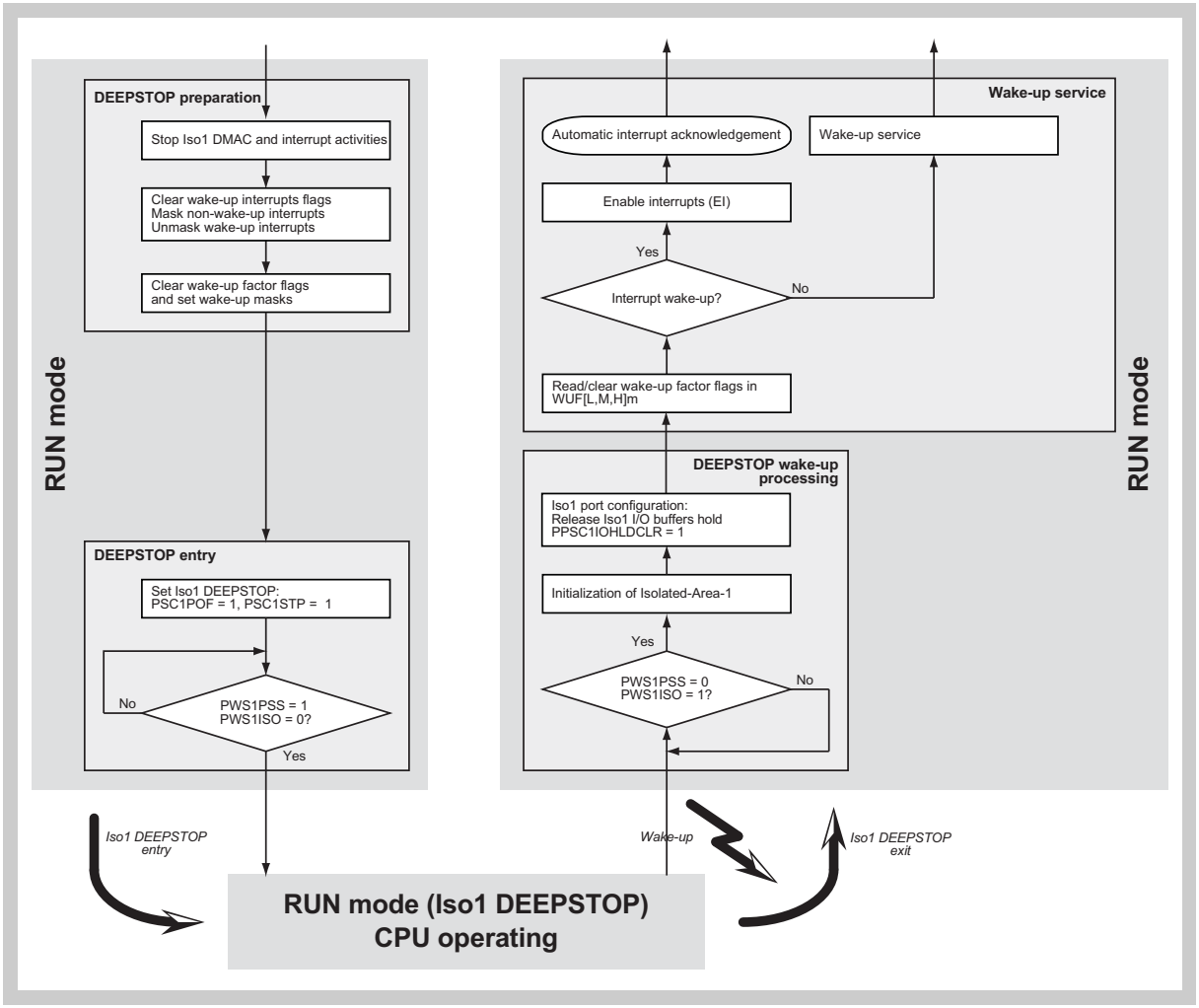


Figure 10-9 Recommended flow of RUN mode (Isolated-Area-1 DEEPSTOP)

(5) S/W wake-up

The RUN mode (Iso1 DEEPSTOP) can also be terminated by a S/W wake-up trigger PSC1.PSC1ISOWU = 1.

- Set PSC1.PSC1ISOWU = 1 to trigger Isolated-Area-1 wake-up from DEEPSTOP mode.
- Wait until Isolated-Area-1 is woken up, indicated by PWS1.PWS1PSS = 0 and PWS1,PWS1ISO = 1.

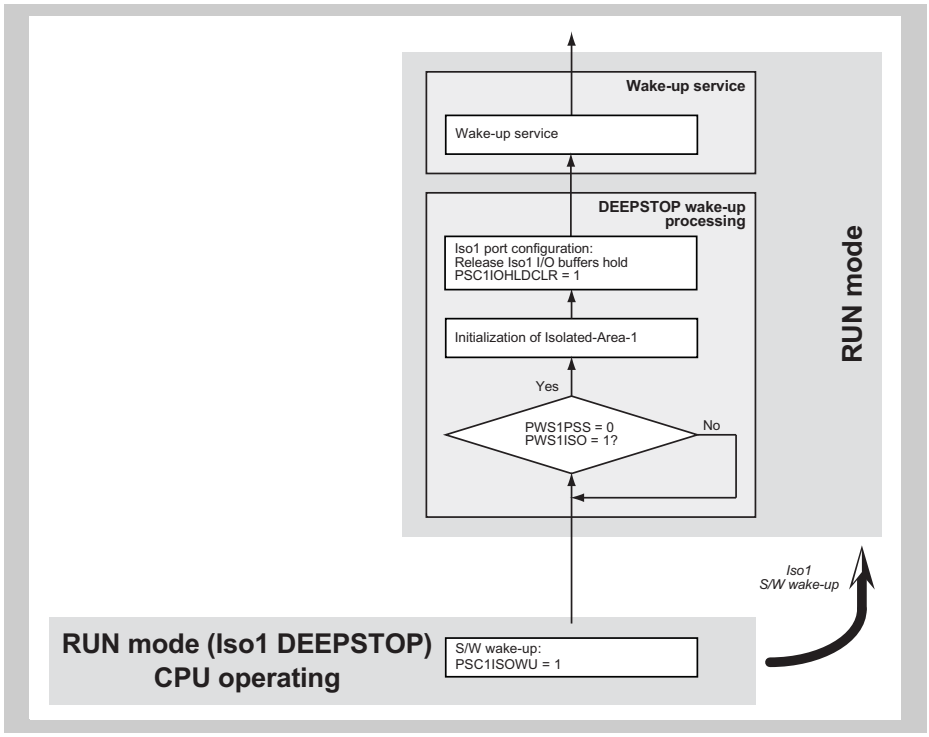


Figure 10-10 Isolated-Area-1 wake-up by software

10.3.5 Precaution: Clock generators and early wake-up

In case a clock generator shall be stopped in an Isolated-Area-0 stand-by mode and shall automatically resume operation after wake-up, restart of the clock generator may fail in case of an early wake-up, i.e. if a wake-up event occurs during the transition to the stand-by mode.

This generator start fail may happen under certain conditions, which basically differ in the intended behaviour of the Watchdog Timers WDTA0 (on the Always-On-Area) and WDTA1 (on the Isolated-Area-0) during stand-by mode.

If a Watchdog Timer shall continue operation during stand-by mode its WDTATCKI clock supply must be maintained during stand-by, thus the WDTATCKI clock selector's stop request must be masked:

- domain clock CKSCLK_A07 for WDTA0 WDTATCKI,
  stop request mask CKSC_A07.STPMK_A07 = 1 to maintain clock.
- domain clock CKSCLK_007 for WDTA1 WDTATCKI,
  stop request mask CKSC_007.STPMK_007 = 1 to maintain clock.

If a clock generator shall be stopped during stand-by, its stop request must not be masked:

- MainOsc stop in stand-by: MOSCE.MOSCESTPMK = 0
- SubOsc stop in stand-by: SOSCE.SOSCESTPMK = 0
- High Speed IntOsc stop in stand-by: ROSCE.ROSCESTPMK = 0
- PLLk stop in stand-by: PLLEk.PLLkSTPMK = 0

(1) Condition 1

The clock generator may fail to resume operation after early wake-up, if all following conditions are fulfilled:

1. Both Watchdog Timers shall continue operation during stand-by mode.
2. Any clock generator shall be stopped during stand-by.

(2) Condition 2

The clock generator may fail to resume operation after early wake-up, if all following conditions are fulfilled:

1. One or both of the Watchdog Timers WDTA0 and WDTA1 shall be stopped during stand-by mode.
   The WDTATCKI clock of the Watchdog Timer to be stopped has the frequency $f_{WDTAoff}$. In case both Watchdog Timers shall be stopped, $f_{WDTAoff}$ denotes the lower WDTATCKI frequency.
2. An Always-On-Area or Isolated-Area-0 clock domain CKSCLK_off has a frequency $f_{CKSCLK\_off} \leq f_{WDTAoff}$ and shall be stopped during stand-by mode.
3. The clock generator, which is the source of CKSCLK_off, shall be stopped during stand-by.

The following diagram summarizes, which countermeasures to apply under which conditions.

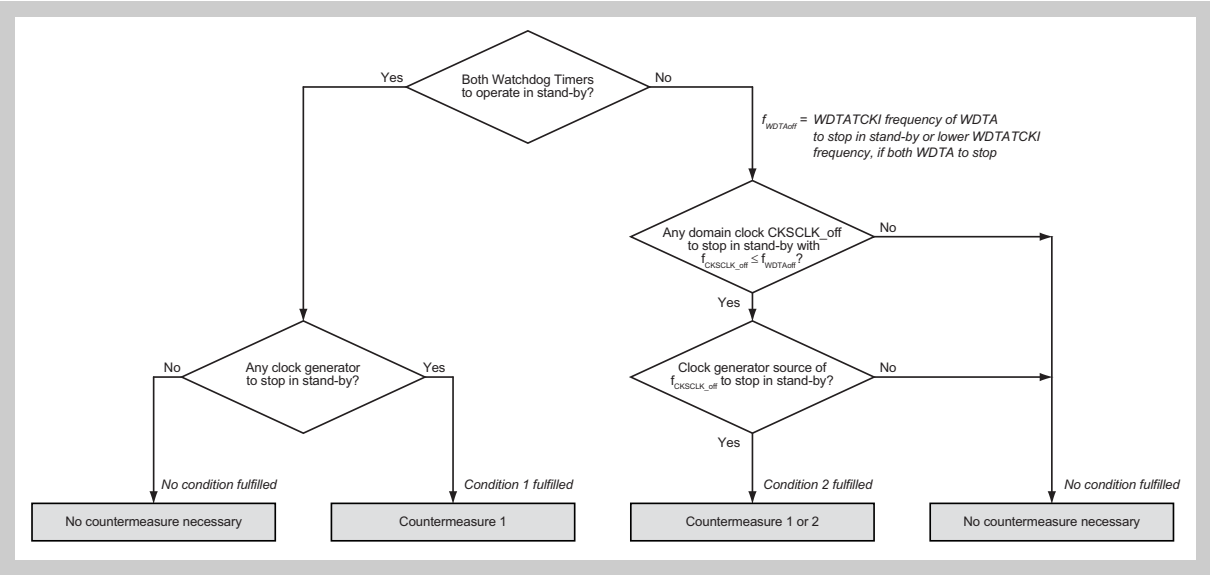


Figure 10-11 Clock generators and early wake-up

(3) Countermeasure 1

Before stand-by Change the clock source ID of the CPUCLK domain clock CKSCLK_000 to the High Speed/Low Speed IntOsc, i.e. set CKSC_000.CKSCID_000 = $3A_H$.

After early-wake-up All oscillators, which were in operation before stand-by mode entry, and have been stopped in stand-by mode must be re-started (MOSCE.MOSCEENTRG = 1, ROSCE.ROSCEENTRG = 1, SOSCE.SOSCEENTRG = 1 and PLLEk.PLLEkENTRG = 1, respectively).

This countermeasure can be applied under conditions 1 and 2.

(4) Countermeasure 2

Before stand-by

- Change the frequency of the domain clock CKSCLK_off to a clock with $f_{CKSCLK\_off} > f_{WDTAoff}$ before entering stand-by mode.

or

- Stop the domain clock CKSCLK_off by setting its ID CKSCID_mn = 0000_H (domain clock stopped).

After early wake-up Change the clock domain CKSCLK_off to its original frequency.

This countermeasure can be applied under condition 2.

10.3.6 Application hint: Handling of wake-up events during stand-by mode preparation

This application hint describes in more detail how the stand-by mode preparation determines, at which point of the flow wake-up events provoke an early wake-up or are discarded.

Wake-up interrupts set their interrupt request flag (the related interrupt control register bit ICn.RFn), however not all wake-up events are interrupts.

Additionally in case of DEEPSTOP mode, all interrupt request flags will be discarded, if the microcontroller completely enters DEEPSTOP mode.

Thus the occurrence of *all* wake-up events during stand-by mode preparation can only be saved as wake-up flags in the wake-up factor registers WUF[L,M,H]m. Wake-up servicing can then be performed after wake-up from stand-by mode by evaluating the wake-up factor registers.

It depends on the application requirements, after which step of the stand-by entry procedure wake-up events must be recorded and served. In the following two different flows are described as examples.

Non wake-up interrupts

All examples assume that interrupts, which shall not be used for wake-up, in particular all interrupts related to modules

- on an isolated area to enter DEEPSTOP
- which are to be stopped in a STOP mode,

are deactivated in the first step of the stand-by mode preparation.
As these interrupts will be masked as interrupts and wake-up events afterwards, they shall not occur.

(1) Discarding wake-up events during stand-by preparation

The following diagram shows a stand-by preparation flow, where all wake-up events, occurring during the stand-by preparation sequence, are discarded and will not be served after stand-by wake-up.

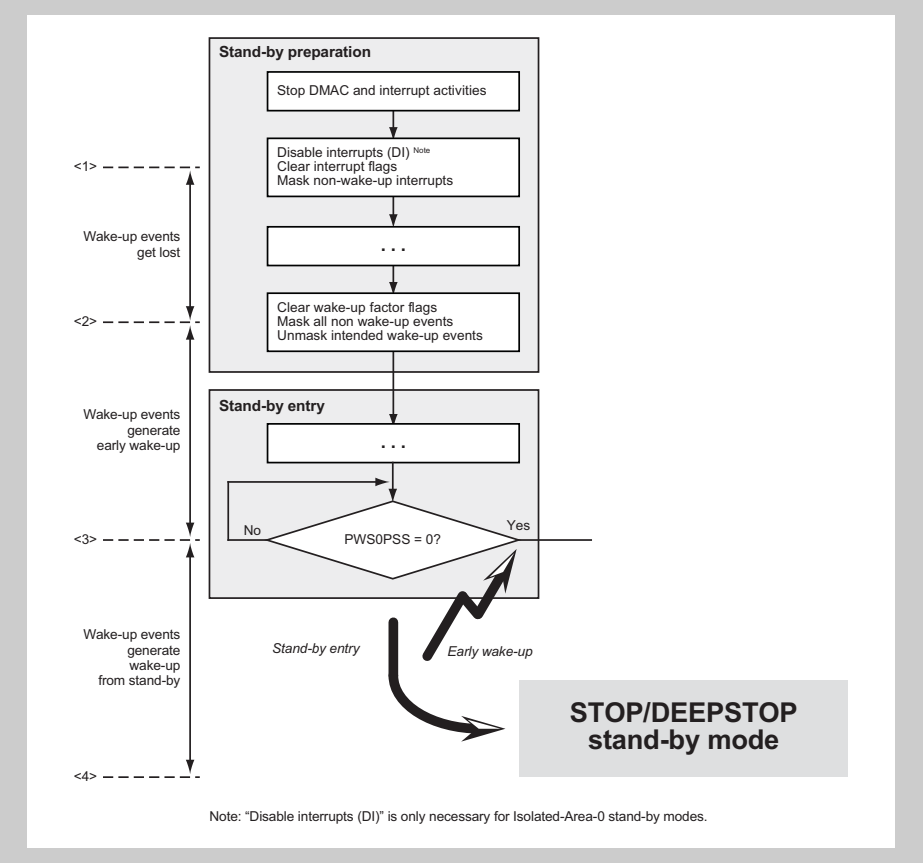


Figure 10-12 Discarding wake-up events during stand-by preparation

<1> Wake-up interrupts

- are not masked, but not acknowledged yet (because of "DI"),
- are also not saved as wake-up, because wake-up flags WUF[L,M,H]m are cleared later in <2>.

<1> – <2> During <1> and <2> all occurring wake-up events are discarded, thus will neither cause a wake-up nor any wake-up service routine after stand-by wake-up.

<2> Since wake-up events are unmasked, their occurrence will be saved in the wake-up flag registers WUF[L,M,H]m from now on.

<3> If an unmasked wake-up event is registered in WUF[L,M,H]m before stand-by mode is completely entered, an early wake-up will occur.

<4> All unmasked wake-up events occurring after stand-by mode was completely entered terminate the stand-by mode.

(2) Saving wake-up events during stand-by preparation

The following diagram shows a stand-by preparation flow, where wake-up events are registered as wake-up factors already during the stand-by preparation sequence.

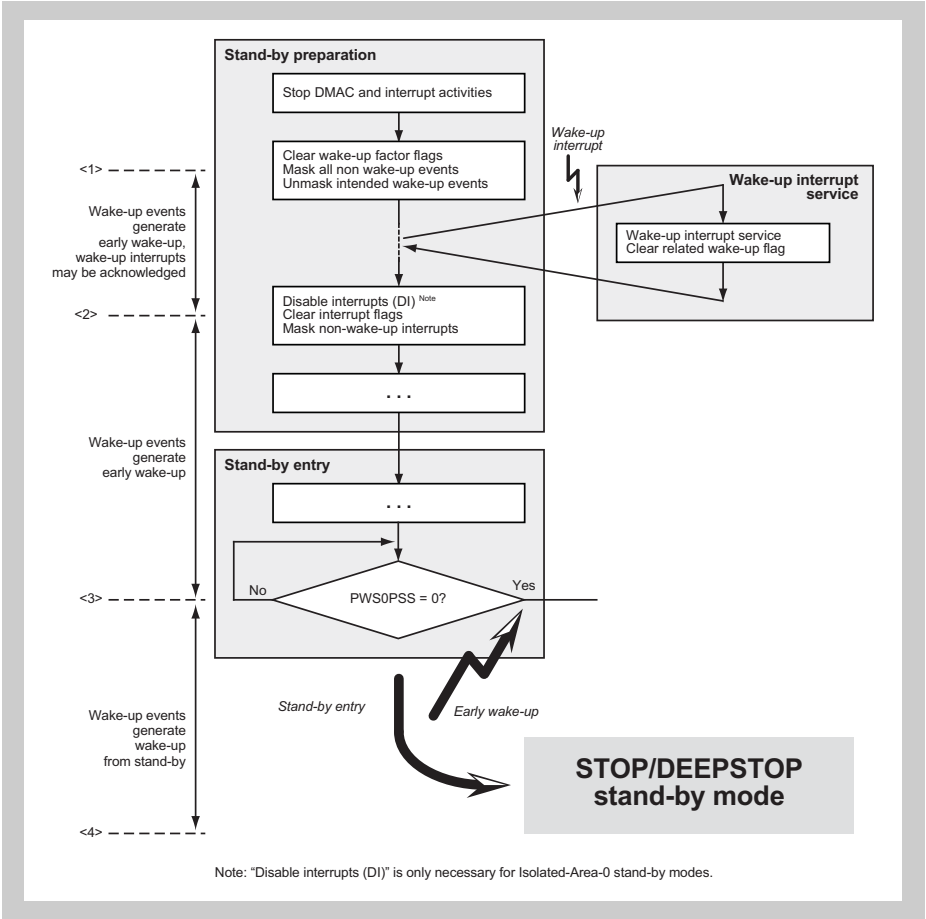


Figure 10-13 Saving wake-up events during stand-by preparation

<1> All intended wake-up events are unmasked here, thus they will be saved as wake-up factors in the WUF[L,M,H]m registers upon occurrence and will cause an early wake-up later on.

<1> – <2> In case wake-up interrupts are already enabled (ICn.MKn = 0), they will be served.
Since a wake-up interrupt is served here and is also registered as a wake-up factor, the related wake-up factor must be cleared via the WUFC[L,M,H]m registers in this interrupt service routine. Otherwise it remains registered as a wake-up factor and may be served a second time after stand-by wake-up.

<2> After disabling interrupt service in general (DI) and masking all non wake-up interrupts all wake-up events are only saved as wake-up factors in WUF[L,M,H]m and cause an early wake-up later on.

<3> All unmasked wake-up events occurring after stand-by mode was completely entered terminate stand-by mode.

10.4 Stand-by Controller Registers

This section contains a description of all the registers of the Stand-by Controller.

10.4.1 Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Following Stand-by Controller registers feature this special write protection:

- Power save control registers PSC0 and PSC1

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

10.4.2 Stand-by Controller registers overview

The Stand-by Controller is controlled and operated by the following registers:

Table 10-13 Stand-by Controller register overview (1/2)

Register Name	Shortcut	Address
Power save registers		
Power save control register 0	PSC0	FF42 0000$_H$
Power save control register 1	PSC1	FF42 0008$_H$
Power status register 0	PWS0	FF42 0004$_H$
Power status register 1	PWS1	FF42 000C$_H$
Isolated-Area-0 wake-up factor controller registers		
Wake-up factor register L	WUFL0	FF42 0100$_H$
Wake-up factor register M	WUFM0	FF42 0110$_H$
Wake-up factor register H	WUFH0	FF42 0120$_H$
Wake-up factor mask register L	WUFMSKL0	FF42 0104$_H$
Wake-up factor mask register M	WUFMSKM0	FF42 0114$_H$
Wake-up factor mask register H	WUFMSKH0	FF42 0124$_H$
Wake-up factor clear register L	WUFCL0	FF42 0108$_H$
Wake-up factor clear register M	WUFCM0	FF42 0118$_H$
Wake-up factor clear register H	WUFCH0	FF42 0128$_H$
Isolated-Area-1 wake-up factor controller registers		
Wake-up factor register L	WUFL1	FF42 0130$_H$
Wake-up factor register M	WUFM1	FF42 0140$_H$
Wake-up factor register H	WUFH1	FF42 0150$_H$
Wake-up factor mask register L	WUFMSKL1	FF42 0134$_H$
Wake-up factor mask register M	WUFMSKM1	FF42 0144$_H$
Wake-up factor mask register H	WUFMSKH1	FF42 0154$_H$
Wake-up factor clear register L	WUFCL1	FF42 0138$_H$

Table 10-13 Stand-by Controller register overview (2/2)

Register Name	Shortcut	Address
Wake-up factor clear register M	WUFCM1	$FF42\ 0148_H$
Wake-up factor clear register H	WUFCH1	$FF42\ 0158_H$
Oscillator wake-up mask registers:		
Oscillator wake-up mask register	OSCWUFMSK	$FF42\ 01A40_H$

10.4.3 Stand-by Controller control registers details

(1) PSC0 – Power save control register 0

This register controls the stand-by modes of the Always-On-Area and Isolated-Area-0.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 0000_H

Initial Value $0000\ 0000_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	PSC0 IOHLDSET	PSC0 IOHLDMSK	0[a]	PSC0 IOHLDCLR	0	PSC0 POF	PSC0 STP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) The default value "0" of bit 4 of PSC0 must be changed to "1" when using DEEPSTOP mode.

Caution The default value "0" of bit 4 of PSC0 must be changed to "1" when using DEEPSTOP mode.

Table 10-14 PSC0 register contents (1/2)

Bit position	Bit name	Function
6	PSC0 IOHLDSET	Isolated-Area-0 I/O buffer hold function trigger 0: terminate Isolated-Area-0 I/O buffer hold function 1: enter Isolated-Area-0 I/O buffer hold function Read value of this bit is always"0".
5	PSC0 IOHLDMSK	Isolated-Area-0 I/O buffer hold function mask 0: enable Isolated-Area-0 I/O buffer hold function 1: disable Isolated-Area-0 I/O buffer hold function If the PSC0IOHLDMSK = 1, the I/O buffers of Isolated-Area-0 do not go into hold mode during DEEPSTOP mode.
4	Bit 4	The default value "0" of bit 4 of PSC0 must be changed to "1" when using DEEPSTOP mode.

Table 10-14 PSC0 register contents (2/2)

Bit position	Bit name	Function
3	PSC0 IOHLDCLR	Isolated-Area-0 I/O buffer hold clear function trigger 0: no function 1: clear I/O buffer hold function Reading of this bit returns always 0. **Note:** Setting PSC0IOHLDCLR = 1 clears the I/O hold state of all Isolated-Area-0 buffers, independent of how I/O buffer hold was entered: either by setting PSC0IOHLDSET = 1 or automatically during transition to DEEPSTOP mode.
1	PSC0 POF	Isolated-Area-0 power-off, i.e. DEEPSTOP, selection 0: Isolated-Area-0 power remains on during stand-by mode (= STOP) 1: Isolated-Area-0 power switched off during stand-by mode (= DEEPSTOP)
0	PSC0 STP	Always-On-Area and Isolated-Area-0 stand-by mode trigger 0: no function 1: Always-On-Area and Isolated-Area-0 enter stand-by mode Reading of this bit returns always 0. Upon entering stand-by mode with PSC0STP = 1, • Always-On-Area always enters STOP mode • Isolated-Area-0 enters STOP (if PSC0POF = 0) or DEEPSTOP (if PSC0POF = 1) mode

(2) PSC1 – Power save control register 1

This register controls the stand-by modes of the Isolated-Area-1.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 0008$_H$

Initial Value 0000 0000$_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	PSC1 IOHLDSET	PSC1 IOHLDMSK	PSC1 REGSTP	PSC1 IOHLDCLR	PSC1I ISOWU	PSC1 POF	PSC1 STP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 10-15 PSC1 register contents (1/2)

Bit position	Bit name	Function
6	PSC1 IOHLDSET	Isolated-Area-1 I/O buffer hold function trigger 0: terminate Isolated-Area-1 I/O buffer hold function 1: enter Isolated-Area-1 I/O buffer hold function Read value of this bit is always "0". Writing to this bit is ignored, if Isolated-Area-1 is in stand-by mode (PWS1.PWS1PSS = 1).
5	PSC1 IOHLDMSK	Isolated-Area-1 I/O buffer hold function mask 0: enable Isolated-Area-1 I/O buffer hold function 1: disable Isolated-Area-1 I/O buffer hold function If PSC1IOHLDMSK = 1, the I/O buffers of Isolated-Area-1 do not go into hold mode during any Isolated-Area-1 DEEPSTOP mode.
4	PSC1 REGSTP	Control of the on-chip voltage regulators and WAKE signal behaviour in DEEPSTOP mode: 0: on-chip voltage regulators stay in normal operation and WAKE at high level during DEEPSTOP 1: on-chip voltage regulators enter power save mode and WAKE turns to low level in DEEPSTOP Refer to the *37.3 "Power-up and down procedures" on page 2610* for details.

Table 10-15 PSC1 register contents (2/2)

Bit position	Bit name	Function
3	PSC1 IOHLDCLR	I/O buffer hold function clear trigger 0: no function 1: clear I/O buffer hold function Writing to this bit is ignored, if Isolated-Area-1 is in stand-by mode (PWS1.PWS1PSS = 1). Reading of this bit returns always “0”. **Note:** Setting PSC1IOHLDCLR = 1 clears the I/O hold state of all Isolated-Area-1 buffers, independent of how I/O buffer hold was entered: either by setting PSC1IOHLDSET = 1 or automatically during transition to Isolated-Area-1 DEEPSTOP mode.
2	PSC1 ISOWU[a]	Wake-up trigger of Isolated-Area-1 0: no function 1: wake-up Isolated-Area-1 Reading of this bit returns always 0.
1	PSC1 POF	Isolated-Area-1 power-off, i.e. DEEPSTOP, selection 0: Isolated-Area-1 power remains on during stand-by mode (= STOP) 1: Isolated-Area-1 power switched off during stand-by mode (= DEEPSTOP)
0	PSC1 STP	Isolated-Area-1 stand-by mode trigger 0: no function 1: Isolated-Area-1enter stand-by mode Upon entering stand-by mode with PSC1STP = 1, the Isolated-Area-1 • enters STOP mode, if PSC1POF = 0 • enters DEEPSTOP mode, if PSC1POF = 1 Reading of this bit returns always 0.

a) Unlike wake-up by other wake-up factors, which occurrence can be recognized by evaluating the flags in the WUF[L,M,H]m registers, a S/W wake-up by PSC1ISOWU = 1 can not be recognized by any flag.

(3) PWS0 – Power status register 0

This register shows the stand-by mode status of the Isolated-Area-0.

Access This register can be read in 32-bit units.

Address FF42 0004_H

Initial Value $0000\ 0001_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PWS0 PSS	0	0	0	0	0	PWS0 IOHOLD	1
R	R	R	R	R	R	R	R

Table 10-16 PWS0 register contents

Bit position	Bit name	Function
7	PWS0 PSS	Stand-by status of Always-On-Area and Isolated-Area-0 0: Always-On-Area and Isolated-Area-0 is in normal mode 1: Always-On-Area and Isolated-Area-0 is in the transition to or is in stand-by mode PWS0PSS is set to 1 immediately after the Always-On-Area and Isolated-Area-0 stand-by mode trigger was issued (PSC0.PSC0STP = 1). Thus PWS0PSS = 1 indicates the state of transition to stand-by mode and the stand-by mode status. If the stand-by mode was entered, the CPU is also stopped and this register can not be read any more.
1	PWS0 IOHOLD	I/O buffer hold status of Isolated-Area-0 0: I/O buffers of Isolated-Area-0 are not in hold mode 1: I/O buffers of Isolated-Area-0 are in hold mode

Note This register can not be read in STOP and DEEPSTOP stand-by mode of Isolated-Area-0.

(4) PWS1 – Power status register 1

This register shows the stand-by mode status of the Isolated-Area-1.

Access This register can be read in 32-bit units.

Address FF42 000C$_H$

Initial Value 0000 0000$_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PWS1 PSS	0	0	0	0	0	PWS1 IOHOLD	PWS1 ISO
R	R	R	R	R	R	R	R

Table 10-17 PWS1 register contents

Bit position	Bit name	Function
7	PWS1 PSS	Stand-by status of Isolated-Area-1 0: Isolated-Area-1 is in normal operation mode 1: Isolated-Area-1 is in the transition to or is in stand-by mode PWS1PSS is set to 1 immediately after the Isolated-Area-1 stand-by mode trigger was issued (PSC1.PSC1STP = 1). Thus PWS0PSS = 1 indicates the state of transition to stand-by mode and the stand-by mode status.
1	PWS1 IOHOLD	I/O buffer hold status of Isolated-Area-1 0: I/O buffers of Isolated-Area-1 are not in hold mode 1: I/O buffers of Isolated-Area-1 are in hold mode
0	PWS1 ISO	Power status of Isolated-Area-1 0: Isolated-Area-1 is in power-off (DEEPSTOP) state 1: Isolated-Area-1 is not in power-off (DEEPSTOP) state

10.4.4 Wake-up factor controller registers details

(1) WUF[L,M,H]m – Wake-up factor registers (m = 0, 1)

This register informs about the wake-up factor of Isolated-Area-m.

Notes

1. For the assignment of the wake-up factors to wake-up factor register bits refer to the key word “Wake-up factors” in the first section of this chapter.
2. Wake-up factor bits WUFmn, which are not assigned to a valid wake-up factor, are fixed to 0.
3. To clear an asserted bit in this register use the WUFC[L,M,H]m registers.

Access This register can be read in 32-bit units.

Address WUFL0: FF42 0100_H
WUFM0: FF42 0110_H,
WUFH0: FF42 0120_H
WUFL1: FF42 0130_H,
WUFM1: FF42 0140_H
WUFH1: FF42 0150_H

Initial Value $0000\ 0000_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
WUF[L,M,H]m[31:24]							
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
WUF[L,M,H]m[23:16]							
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
WUF[L,M,H]m[15:8]							
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
WUF[L,M,H]m[7:0]							
R	R	R	R	R	R	R	R

Table 10-18 WUF[L,M,H]m registers contents

Bit position	Bit name	Function
31 to 0	WUF [L,M,H]mn	Indicates occurrence of wake-up factor WUF[L,M,H]mn (n = 0 to 31) 0: wake-up factor WUF[L,M,H]mn did not occur 1: wake-up factor WUF[L,M,H]mn occurred

(2) WUFMSK[L,M,H]m – Wake-up factor mask registers (m = 0 to 1)

This register enables wake-up factors of Isolated-Area-m.

Notes

1. For the assignment of the wake-up factors to wake-up factor register bits refer to the key word “Wake-up factors” in the first section of this chapter.
2. Wake-up mask bits WUFMSK[L,M,H]mn, which are not assigned to a valid wake-up factor, are fixed to 1.

Access This register can be read/written in 32-bit units.

Address WUFMSKL0: FF42 0104$_H$
WUFMSKM0: FF42 0114$_H$,
WUFMSKH0: FF42 0124$_H$
WUFMSKL1: FF42 0134$_H$,
WUFMSKM1: FF42 0144$_H$
WUFMSKH1: FF42 0154$_H$

Initial Value FFFF FFFF$_H$. This register is initialized by a system reset SYSRES.

31	30	29	28	27	26	25	24
WUFMSK[L,M,H]m[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
WUFMSK[L,M,H]m[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
WUFMSK[L,M,H]m[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
WUFMSK[L,M,H]m[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 10-19 WUFMSK[L,M,H]m registers contents

Bit position	Bit name	Function
31 to 0	WUFMSK [L,M,H]mn	Enable of wake-up factor WUF[L,M,H]mn 0: wake-up factor WUF[L,M,H]mn enabled 1: wake-up factor WUF[L,M,H]mn disabled

(3) WUFC[L,M,H]m – Wake-up factor clear registers (m = 0 to 1)

This register clears wake-up factor indicates in the WUF[L,M,H]mn register of Isolated-Area-m.

Notes

1. For the assignment of the wake-up factors to wake-up factor register bits refer to the key word “Wake-up factors” in the first section of this chapter.
2. Wake-up factor clear bits WUFC[L,M,H]mn, which are not assigned to a valid wake-up factor, must be written with “0”.

Access This register can be written in 32-bit units.

Address WUFCL0: FF42 0108$_H$
WUFCM0: FF42 0118$_H$,
WUFCH0: FF42 0128$_H$
WUFCL1: FF42 0138$_H$,
WUFCM1: FF42 0148$_H$
WUFCH1: FF42 0158$_H$

Initial Value Reading these registers returns always 0000 0000$_H$.

31	30	29	28	27	26	25	24
WUFC[L,M,H]m[31:24]							
W	W	W	W	W	W	W	W

23	22	21	20	19	18	17	16
WUFC[L,M,H]m[23:16]							
W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8
WUFC[L,M,H]m[15:8]							
W	W	W	W	W	W	W	W

7	6	5	4	3	2	1	0
WUFC[L,M,H]m[7:0]							
W	W	W	W	W	W	W	W

Table 10-20 WUFC[L,M,H]m registers contents

Bit position	Bit name	Function
31 to 0	WUFC [L,M,H]mn	Clear of wake-up factor WUF[L,M,H]mn of the wake-up factor registers 0: no function 1: clear WUF[L,M,H]mn

10.4.5 Oscillator wake-up mask registers details

(1) OSCWUFMSK – Oscillator wake-up mask register

This register controls the start of the MainOsc and SubOsc upon a wake-up factor of Isolated-Area-0 and Isolated-Area-1.

Access This register can be written/read in 32-bit units.

Address FF42 01A4$_H$

Initial Value 0000 0003$_H$. This register is initialized by a Power-up reset PURES.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
0	0	0	0	0	0	OSCWUF MSK01	OSCWUF MSK00
R	R	R	R	R	R	R/W	R/W

Table 10-21 OSCWUFMSK register contents

Bit position	Bit name	Function
1	OSCWUF MSK01	Enables Isolated-Area-1 wake-up factor for starting the MainOsc and SubOsc 0: oscillators starts by Isolated-Area-1 wake-up factor 1: oscillators do not start by Isolated-Area-1 wake-up factor
0	OSCWUF MSK00	Enables Isolated-Area-0 wake-up factor for starting the MainOsc and SubOsc 0: oscillators starts by Isolated-Area-0 wake-up factor 1: oscillators do not start by Isolated-Area-0 wake-up factor

Note If OSCWUFMSK0m = 0, the MainOsc and SubOsc are always started upon wake-up from the respective isolated area, even if any of the oscillators were stopped before the stand-by mode was entered.

Chapter 11 Code Protection and Security

11.1 Overview

The microcontroller supports various methods for protecting the program code in the flash memory from undesired access, such as illegal read-out or illegal reprogramming.

Some interfaces offer in general access to the internal flash memory: Nexus debug interface, external flash programmer interface, Self-Programming facilities and test interfaces.

In the following the security relevant items are listed. The features to protect the internal flash memory data from being read by unauthorized persons are described.

For more information about the flash memory, see chapter *"Flash Memory"*.

The following sections give an overview about supported code protection methods.

11.2 Flash Programmer and Self-Programming Protection

In general, illegal read-out and re-programming of the flash memory contents is possible via the flash programmer interface and the Self-Programming feature. For protection of the flash memory, the following flags provide various protection levels.

The flags can be set by flash programmers. For a description of flash memory programming see chapter *"Flash Memory"*.

(1) Chip erase protection flag (Chip erase protection function)

Set this flag to disable the chip erase function via the flash programmer interface. This flag does not affect the Self-Programming interface.

(2) Block erase protection flag (Block erase protection function)

Set this flag to disable the feature to erase single blocks via the flash programmer interface. This flag does not affect the Self-Programming interface.

This flag does not affect the chip erase function.

The flag is valid for the whole flash memory.

(3) Program protection flag (Program protection function)

Set this flag to disable the programming function via the flash programmer interface. This flag does not affect the Self-Programming interface.

The flag is valid for the whole flash memory.

(4) Read-out protection flag (Read-out protection function)

Set this flag to disable the feature that allows reading back the flash memory via the flash programmer interface. This flag does not affect the Self-Programming interface.

This flag is valid for the whole flash memory.

(5) Boot block cluster protection flag

Set this flag to disable erasure and rewrite of the boot block cluster.
The boot block cluster can not be manipulated in any way (no erase/write).

This applies in Serial- and Self-Programming mode.

Once this flag is set, it is impossible to reset this flag. Thus the boot block cluster content can not be changed any more.

(6) Flash shielding

Flash shielding allows to specify a random number of consecutive code flash memory blocks, the flash shield window, which can be erased and written. Erasure of and writing to all flash memory blocks outside the flash shield window is impossible.

11.3 On-Chip Debug Interface Protection

In general, illegal read-out of the flash memory contents is possible via the Nexus On-Chip Debug interface. For protection of the flash memory, the usage of the debug interface can be disabled.

The debug interface is protected via a 95-bit ID code and an internal control flag (On-Chip Debug enable control).

When the debugger is started, the status of the control flag in the internal flash memory is queried. Setting this flag to zero disables the On-Chip Debugger.

When debugging is enabled (On-Chip Debug enable flag is set), you have to enter the 95-bit ID code via the debugger. The code is compared with the ID code stored in the internal flash memory. If the codes do not match, debugging is not possible.

It is also possible to temporarily enable the On-Chip Debug interface by setting the On-Chip Debug enable flag via the user program, stored in the internal flash memory.
However after the device receives an external $\overline{\text{RESET}}$ or a Power-On-Clear reset POCRES, the On-Chip Debug enable flag is reset again and thus the On-Chip Debug interface is disabled, provided the On-Chip Debug enable flag in the flash memory is 0.

The On-Chip Debug enable flag and the ID code are stored in the flash memory extra area and can be accessed via the OCDIDL, OCDIDM, OCDIDH registers.

11.3.1 On-Chip Debug enable flag

On-Chip Debugging is controlled by the On-Chip Debug enable flag OCDIDH.OCDID[95].

(1) Permanent On-Chip Debug enable/disable

The On-Chip Debug enable flag can be permanently set or reset

- while reprogramming the flash memory by an external flash programmer
- by the user's program with the Self-Programming feature.

Both methods change the content of the internal flash memory.

Any modification of the On-Chip Debug control flag OCDIDH.OCDID[95] by re-programming of the flash memory becomes effective after next release from an external $\overline{\text{RESET}}$ or a Power-On-Clear reset POCRES.

(2) Temporary On-Chip Debug enable/disable

Temporarily enabling respectively disabling the On-Chip Debugging is done by the On-Chip Debug control register IDMODI:

- IDMODI.IDEN = 1 and IDMODI.IDDATA = 0 sets OCDIDH.OCDID[95] = 0 and disables On-Chip Debugging
- IDMODI.IDEN = 1 and IDMODI.IDDATA = 1 sets OCDIDH.OCDID[95] = 1 and enables On-Chip Debugging

Notes

1. After release from the next external $\overline{\text{RESET}}$ or Power-On-Clear reset POCRES, the On-Chip Debug control flag OCDIDH.OCDID[95] takes on the value from the internal flash memory.
2. Writing to IDMODI with IDMODI.IDEN = 0 does not modify the On-Chip Debug control flag.

11.3.2 On-Chip Debug ID code

The 95-bit ID code is accessible via the register bits OCDIDH.OCDID[94:64], OCDIDM.OCDID[63:32] and OCDIDL.OCDID[31:0].

The ID code can be specified

- while reprogramming the flash memory by an external flash programmer
- by the user's program with the Self-Programming feature.

Both methods change the content of the internal flash memory.

Any modification of the ID code OCDIDH.OCDID[94:64], OCDIDM.OCDID[63:32] and OCDIDL.OCDID[31:0] by re-programming of the flash memory becomes effective after next release from an external $\overline{\text{RESET}}$ or a Power-On-Clear reset POCRES.

11.3.3 On-Chip Debug protection levels summary

The following table summarizes the protection levels of the On-Chip Debug interface:

Table 11-1 On-Chip Debug protection levels

On-Chip Debug enable flag	ID code	Protection Level
0	X[a]	Level 2: Full protection On-Chip debug interface can not be used.
1	user-specific ID code	Level 1: ID code protection On-Chip Debug interface can only be used if the user enters the correct ID code.
	ID code is all ones[b]	Level 0: No protection On-Chip Debug interface can be used.

a) ID codes are not compared
b) This is the default state after the flash memory has been erased.

Notes

1. Once the On-Chip Debug interface has been set as "use-prohibited", it cannot be used until the On-Chip Debug enable flag is set to 1 by the user's program or by Self-Programming.
2. After you have set protection levels 1 or 2, set the "block erase disable flag" in the flash extra area. Otherwise, an unauthorized person could erase the block that contains the ID code or the "On-Chip Debug enable flag", respectively, and thus suspend the protection.

11.3.4 On-Chip Debug control registers

Following registers are dedicated to the On-Chip Debugger:

Table 11-2 On-Chip Debug control registers overview

Register Name	Shortcut	Address
On-Chip Debug ID register L	OCDIDL	FF47 0000_H
On-Chip Debug ID register M	OCDIDM	FF47 0004_H
On-Chip Debug ID register H	OCDIDH	FF47 0008_H
On-Chip Debug control register	IDMODI	FF47 0000_H

(1) OCDIDL/M/H - On-Chip Debug ID registers

These registers hold the 95-bit ID code, the user is requested to enter upon start of a debug session.
By use of the OCDID[95] bit On-Chip Debugging can generally be disable respectively enabled.

Access In normal operation mode these registers can be read in 32-bit units.
Writing to this register is only possible in flash programming and self-programming mode.
On-chip Debug control bit OCDID[95] can be temporarily modified via the IDMODI register also in normal operation mode.

Address OCDIDL: FF47 0000_H, OCDIDM: FF47 0004_H, OCDIDH: FF47 0008_H

Initial Value User defined

OCDIDH:

31	30	...		0
OCDID [95]	OCDID [94]	...	OCDID [65]	OCDID [64]
R	R	...	R	R

OCDIDM:

31	30	...		0
OCDID [63]	OCDID [62]	...	OCDID [33]	OCDID [32]
R	R	...	R	R

OCDIDL:

31	30	...		0
OCDID [31]	OCDID [30]	...	OCDID [1]	OCDID [0]
R	R	...	R	R

Table 11-3 OCDIDH/M/L registers contents

Register	Bit position	Bit name	Function
OCDIDH	31	OCDID[95]	Enable/disable On-Chip Debug: 0: On-Chip Debug disabled 1: On-Chip Debug enabled
OCDIDH	30 - 0	OCDID[94:64]	95-bit On-Chip Debug ID code
OCDIDM	31 - 0	OCDID[63:32]	
OCDIDL	31 - 0	OCDID[31:0]	

(2) IDMODI - On-Chip Debug control register

This register allows to temporarily enable/disable On-Chip Debugging in normal operation mode, i.e. by the user's software.
This is performed by modifying the OCDIDH.OCDID[95] On-Chip Debug control bit.

Note Any modification of the On-Chip Debug control bit OCDIDH.OCDID[95] by the user's software becomes invalid after the next release from an external $\overline{\text{RESET}}$ or a Power-On-Clear reset POCRES.

Protection Writing to this register is protected by a special sequence of instructions by using a protection command register.
Refer to the section *"Write protected Registers"* in chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be written in 32-bit units.
Reading this register returns the value of OCDIDL.

Address FF47 0000_H

Initial Value User defined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	IDEN	IDDATA
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 11-4 IDMODI register contents

Bit position	Bit name	Function
1 0	IDEN IDDATA	Modify enable/disable On-Chip Debug control bit OCDIH.OCDI[95]

IDEN	IDDATA	OCDIH.OCDI[95] modification
0	X	not modified
1	0	set to 0 (On-Chip Debug disabled)
	1	set to 1 (On-Chip Debug enabled)

Chapter 12 Reset Controller

12.1 Functional Overview

Several system reset functions are provided in order to initialize the microcontroller hardware and its registers.

Features summary

A reset can be caused by the following events:

- External reset signal $\overline{\text{RESET}}$
  Noise in the external reset signal is eliminated by an analog filter.
- Power-On-Clear (POCRES)
- Overflow of the Watchdog Timers (WDTA0RES, WDTA1RES)
- Clock Monitors reset ($\overline{\text{CLMA0RES}}$, $\overline{\text{CLMA2RES}}$, $\overline{\text{CLMA3RES}}$)
- Low-Voltage Indicator reset (LVIRES)
- Software reset (SWRES)
- Debugger reset (DBRES)

The following block diagram shows the main components of the Reset Controller.

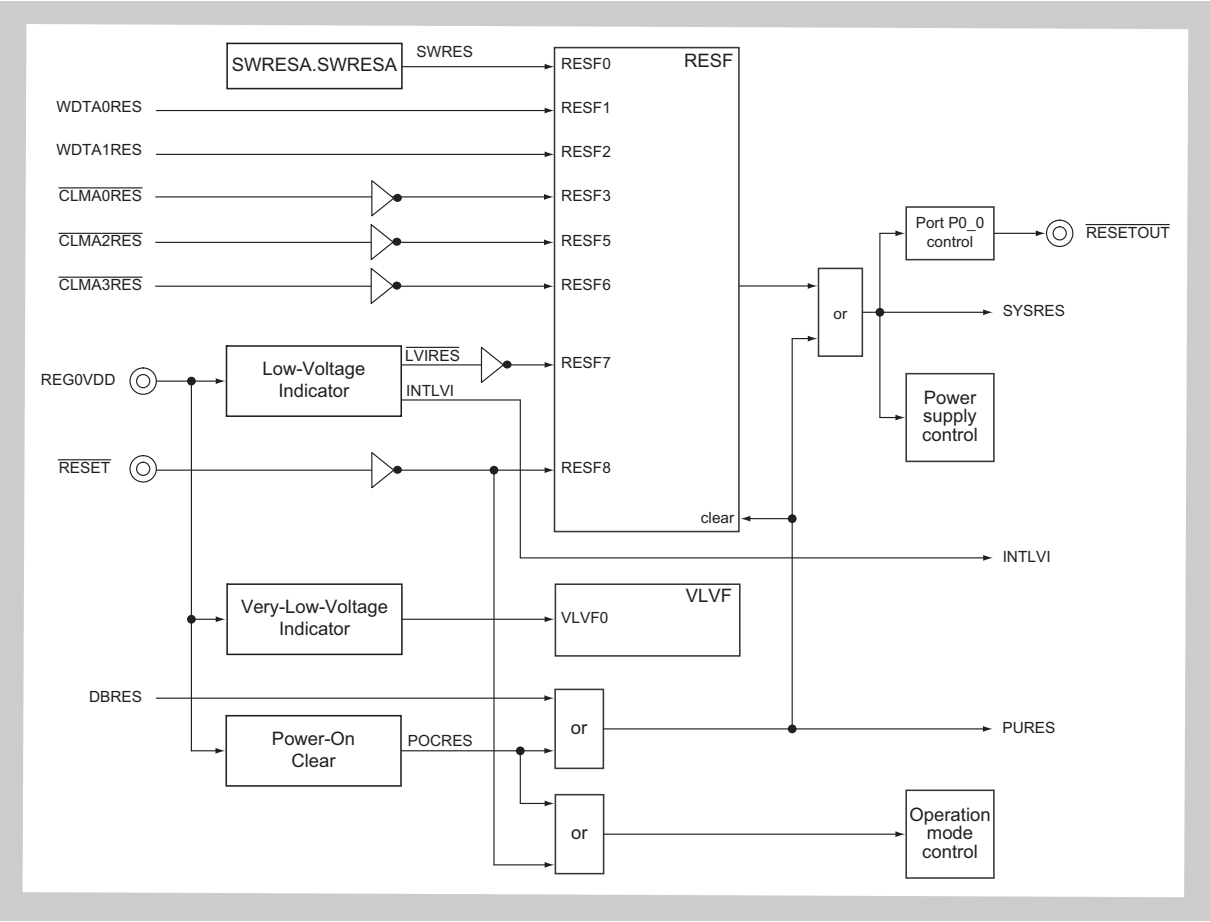


Figure 12-1 Block diagram of the Reset Controller

(1) Internal reset signals

The Reset Controller manages the generation of two reset signals upon occurrence of reset signals from various reset sources:

- system reset SYSRES
  The system reset is generated by all reset sources. SYSRES is applied to all microcontroller components, except the clock generator circuits. Consequently all clock generators continue operation, provided they were operating before the SYSRES.
- power-up reset PURES
  The power-up reset PURES is activated by the Power-On-Clear reset POCRES and the debugger reset DBRES. It resets some clock generators. Hence these clock generators stop operation and must be restarted. Refer to chapter *"Clock Controller"* detailed information, which clock generators are stopped by PURES.
  Assertion of PURES generates also a SYSRES, thus all microcontroller components are reset as well.

(2) Operation mode control

The microcontroller's operation mode is determined upon release of the Power-On-Clear reset or the external $\overline{\text{RESET}}$.
Refer to the section *"Operation modes"* in the *"CPU System Functions"* chapter for details about the operation modes.

(3) Isolated areas power supply

Any reset switches off the power supplies of the isolated areas 0 and 1. Thus after a reset the isolated areas have the same status as after initial power-up or after wake-up from DEEPSTOP mode, i.e. they have to be completely re-initialized.

Note Be aware that also after any reset the internal Data RAM content is undefined.

(4) Reset flags

The reset source flag register RESF holds a flag for each reset source, that is set, when the respective reset was asserted.
All reset flags are only cleared by a power-up reset PURES or by software.
Refer to the section *12.2.1 "Reset flags" on page 642* for details.

(5) Interrupts

The Low-Voltage Indicator generates the interrupt INTLVI.
Refer to the section *12.2.3 "Low-Voltage Indicator (LVI)" on page 644* for details.

(6) On-chip modules resets

Watchdog Timers reset

The Watchdog Timers can generate the resets WDTA0RES and WDTA1RES.
Refer to the section *12.2.6 “Watchdog Timers reset (WDTAnRES)” on page 650* for details.

Clock Monitor resets

The Clock Monitors can generate the resets $\overline{\text{CLMA0RES}}$, $\overline{\text{CLMA2RES}}$ and $\overline{\text{CLMA3RES}}$.
Refer to the section *12.2.8 “Clock Monitors reset (CLMAnRES)” on page 650* for details.

Debugger reset

In case a debugger is connected, it can generate the reset DBRES.
Refer to the section *12.2.9 “Debugger reset (DBRES)” on page 650* for details.

(7) Software controlled resets

SWRES

A software reset can be generated by use of the software reset control register SWRESA.
Refer to the section *12.2.7 “Software reset (SWRES)” on page 650* for details.

(8) Reset output signal

After release of SYSRES P0_0 outputs a $\overline{\text{RESETOUT}}$ signal, which is low level after reset release.

Any change of the P0_0 configuration terminates the $\overline{\text{RESETOUT}}$ output.

Note

Since the $\overline{\text{RESETOUT}}$ signal is activated by all reset events, thus also when an internal reset is applied, it can be used for reset of external devices.

(9) Power supply supervision

Several circuits observe the level of the external power supply REGVDD and generate different actions, depending on REGVDD level.

Power-On Clear

The Power-On-Clear circuit (POC) permanently compares the power supply voltage V_{DD} with an internal reference voltage. It ensures that the microcontroller only operates as long as the power supply exceeds a well-defined limit.
Refer to the section *12.2.2 “Power-On-Clear reset (POCRES)” on page 643* for details.

Low-Voltage Indicator

The Low-Voltage Indicator (LVI) generates the LVIRES reset, if the voltage level of REGVDD drops below a certain level. The level can be adjusted and the LVIRES can be masked.
Refer to the section *12.2.3 “Low-Voltage Indicator (LVI)” on page 644* for details.

Very-Low-Voltage Indicator

The Very-Low-Voltage Indicator (VLVI) flag VLVF.VLVF0 indicates, that REGVDD dropped below a certain level and retention of the microcontroller’s on-chip Back-up RAMs (BURAM) content is not guaranteed.
Refer to the section *12.2.4 “Very-Low-Voltage Indicator (VLVI)” on page 646* for details.

12.2 Functional Description

12.2.1 Reset flags

The reset flag register RESF provides reset flags for each reset source.

If a reset has occurred, the assigned flag is set. This way the source of the reset can be evaluated.

All flags in RESF are not cleared, except by a power-up reset PURES (POCRES or DBRES). Thus this registers behaves cumulative: each reset source sets its own flag, independent of all others.

12.2.2 Power-On-Clear reset (POCRES)

The Power-On-Clear circuit (POC) permanently compares the power supply voltage REG0VDD with the internal reference voltage V_{POC}. It ensures that the microcontroller only operates as long as the power supply exceeds a well-defined limit.

If REG0VDD falls below the internal reference voltage (REG0VDD < V_{POC}), the internal reset signal POCRES, power-up reset PURES and the system reset SYSRES are generated.

Refer to the Electrical Target Specification for the specification of the internal voltage reference level V_{POC}.

Clock generator reset

The Power-On-Clear reset POCRES resets also some clock generators via PURES. Hence these clock generators stop operation and must be restarted. Refer to chapter *"Clock Controller"* for detailed information, which clock generators are stopped by POCRES, respectively PURES.

At Power-On Clear reset, the reset status flag register RESF and the PowerGood fail indicator PWGD are cleared.

The Power-On-Clear function holds the microcontroller in reset state as long as the power supply voltage REG0VDD does not exceed the threshold level V_{POC}.

The following figure illustrates the timing of a POCRES.

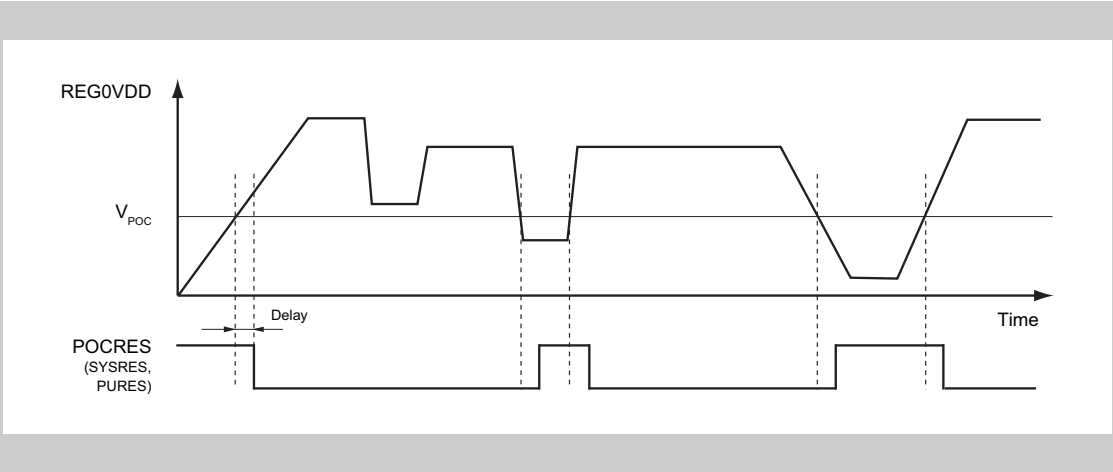


Figure 12-2 POC reset timing

Delay

A delay time is induced between REG0VDD crossing the V_{POC} level and assertion of POCRES.
Refer to the Electrical Target Specification for the delay time specification.

12.2.3 Low-Voltage Indicator (LVI)

The Low-Voltage Indicator circuit (LVI) permanently compares the power supply voltage REG0VDD with the LVI internal reference voltage V_{LVI}.

If REG0VDD falls below the internal reference voltage (REG0VDD < V_{LVI}), the internal reset signal $\overline{\text{LVIRES}}$, the interrupt INTLVI, and the system reset SYSRES are generated.

Additionally the $\overline{\text{LVIRES}}$ flag RESF.RESF7 is set.
RESF.RESF7 is not cleared automatically, if REG0VDD exceeds V_{LVI}. It is cleared by

- setting RESFC.RESFC7 = 1
- power-up reset PURES (POCRES or DBRES)

LVI reference voltage

The LVI reference voltage V_{LVI} can be selected from 6 different levels by LVICNT.LVICNT[2:0].
If LVICNT.LVICNT[2:0] is set to 00_B, the LVI is disabled.
Refer to the Electrical Target Specification for the specification of the internal voltage reference levels V_{LVI}.

LVI interrupt

The LVI interrupt INTLVI is asserted if

- REG0VDD falls below the reference voltage (REG0VDD < V_{LVI})
- REG0VDD rises above the reference voltage (REG0VDD > V_{LVI})

The INTLVI interrupt can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

$\overline{\text{LVIRES}}$ mask

Generation of the LVIRES can be disabled:

- LVICNT.LVIRESMSK = 0: LVIRES not masked (enabled)
- LVICNT.LVIRESMSK = 1: LVIRES masked (disabled)

The following figure illustrates the timing of a $\overline{\text{LVIRES}}$ and RESF.RESF7.

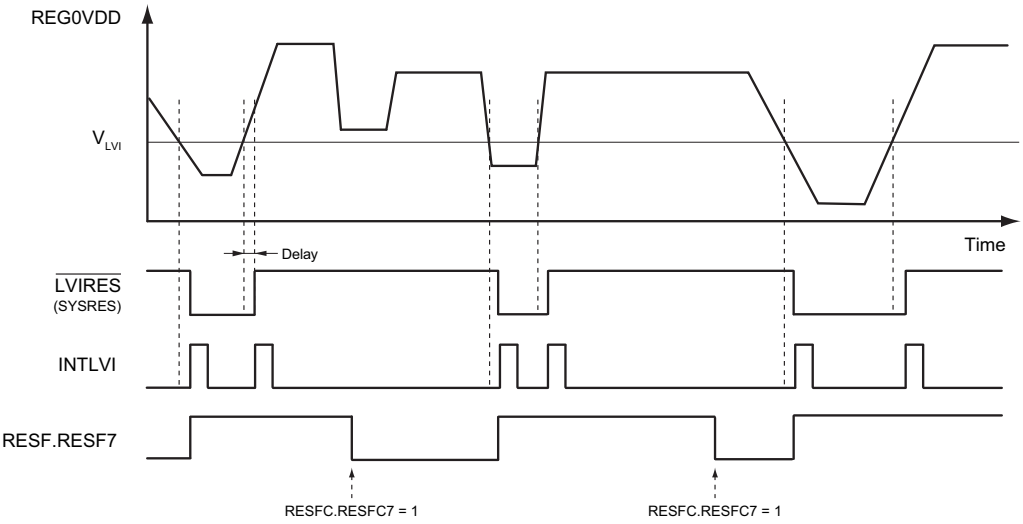


Figure 12-3 LVI reset and interrupt timing

Delay A delay time is induced between REG0VDD crossing the V_{LVI} level and assertion of $\overline{\text{LVIRES}}$ respectively setting of RESF.RESF7.
Refer to the Electrical Target Specification for the delay time specification.

12.2.4 Very-Low-Voltage Indicator (VLVI)

The Very-Low-Voltage Indicator circuit (VLVI) permanently compares the power supply voltage REG0VDD with the VLVI internal reference voltage V_{VLVI}.

Refer to the Electrical Target Specification for the specification of the internal voltage reference level V_{VLVI}.

BURAM content retention

If the power supply voltage REG0VDD does not fall below V_{VLVI}, the content of the on-chip Back-up RAMs (BURAM) is retained and does not need to be restored.
If REG0VDD falls below V_{VLVI} the BURAM content must be assumed to have altered. Thus the entire BURAM must be restored before continuing.

If REG0VDD falls below the internal reference voltage (REG0VDD < V_{VLVI}), the flag VLVF.VLVF0 is set.
VLVF.VLVF0 is not cleared automatically, if REG0VDD exceeds V_{VLVI}. It is cleared by

- setting VLVFC.VLVFC0 = 1

Note VLVF.VLVF0 is no affected by any reset.

The following figure illustrates the timing of VLVF.VLVF0.

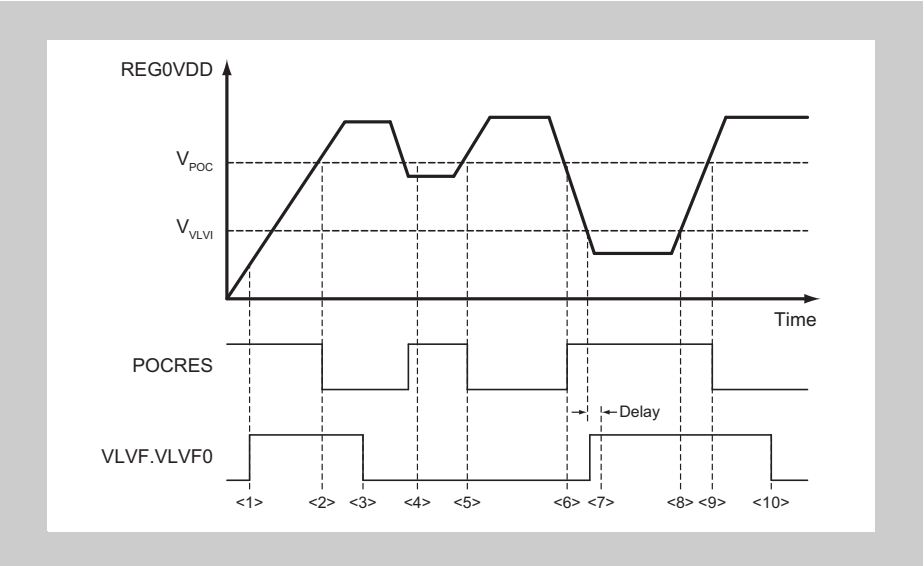


Figure 12-4 VLVI timing

<1> REG0VDD is below V_{VLVI} -> VLVF.VLVF0 = 1

<2> POCRES is released, because REG0VDD exceeds V_{POC}, VLVF.VLVF0 = 1 indicates, that the BURAM is not initialized

<3> VLVF.VLVF0 = 0 must be set by software

<4> to <5> While REG0VDD remains below Power-On-Clear voltage V_{POC}, POCRES is asserted

<6> REG0VDD drops below Power-On-Clear voltage V_{POC}, POCRES is asserted

<7> REG0VDD drops below V_{VLVI} -> VLVF.VLVF0 = 1

<8> Though REG0VDD rises above V_{VLVI}, VLVF.VLVF0 stays 1

<9> REG0VDD rises above V_{POC}, POCRES is de-asserted

<10> VLVF.VLVF0 = 0 must be set by software

Delay A delay time is induced between REG0VDD crossing the V_{VLVI} level setting of VLVF.VLVF0.
Refer to the Electrical Target Specification for the delay time specification.

12.2.5 External $\overline{\text{RESET}}$

Reset is performed when a low level signal is applied to the $\overline{\text{RESET}}$ pin.

Additionally the $\overline{\text{RESET}}$ flag RESF.RESF8 is set.
RESF.RESF8 is not cleared automatically, if $\overline{\text{RESET}}$ is de-asserted. It is cleared by

- setting RESFC.RESFC8 = 1
- power-up reset PURES (POCRES or DBRES)

An external $\overline{\text{RESET}}$ generates the system reset SYSRES.

The $\overline{\text{RESET}}$ is passed through an analog noise filter to prevent erroneous resets due to noise.

The following figure shows the timing when an external reset is performed. It explains the effect of the noise eliminator.

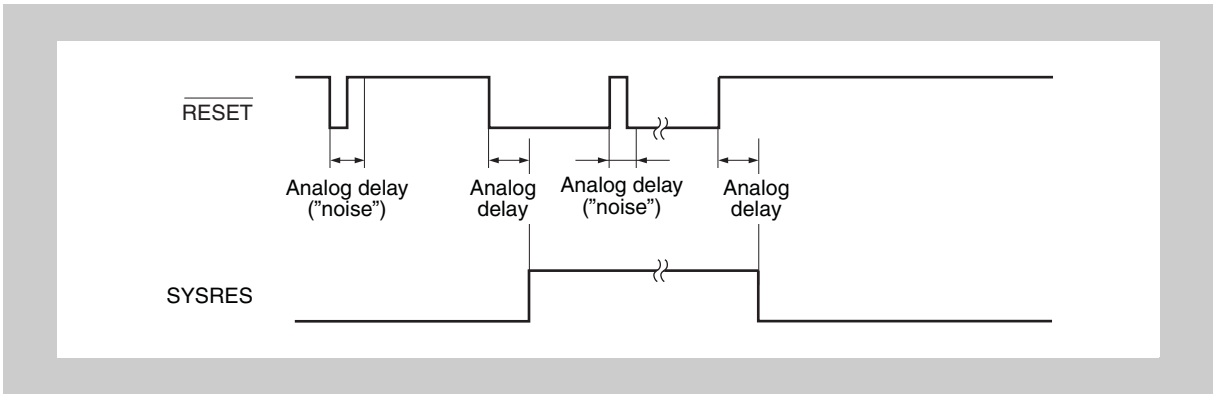


Figure 12-5 External $\overline{\text{RESET}}$ timing

The analog delay is caused by the analog filter. The filter regards pulses up to a certain width as noise and suppresses them.
For the minimum $\overline{\text{RESET}}$ pulse width refer to the Electrical Target Specification.

(1) MainOsc during $\overline{\text{RESET}}$

During assertion of the external $\overline{\text{RESET}}$ the MainOsc is operating with its maximum amplification gain, provided its power supply is active.

The figure below shows the behaviour of the MainOsc under different circumstances, in particular in relation to the power-up reset PURES.

PURES The power-up reset PURES is activated by the Power-On-Clear reset POCRES or the debugger reset DBRES.
Note that the MainOsc is disabled after a PURES.

MainOsc status The figure shows the MainOsc status as follows:

- max = maximum amplification gain, which means short stabilization mode (MainOsc control register MOSCC.MOSCCSHTSTBY = 1)
- AMPSEL = selected amplification gain, set by the MainOsc control register MOSCC.MOSCCAMPSEL[1:0]
- off = MainOsc is stopped

MainOsc start/stop The MainOsc is started respectively stop by software via the MainOsc enable register MOSCE as follows:

- S/W start: MOSCEENTRG = 1
- S/W stop: MOSCEDISTRG = 1

For details concerning the MainOsc control and amplification gain refer to the description of the MainOsc and its control registers in the *"Clock Controller"* chapter.

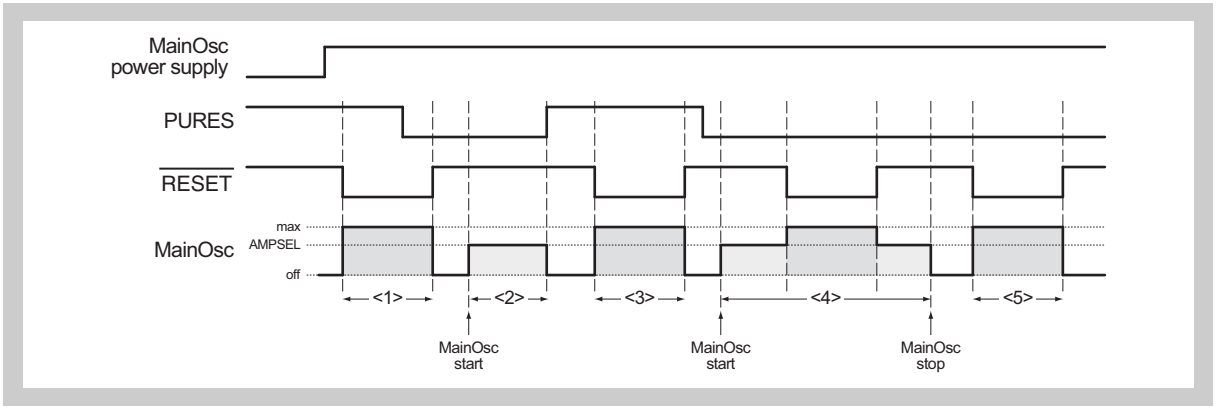


Figure 12-6 MainOsc behaviour during $\overline{\text{RESET}}$

<1> The MainOsc operates, as long as $\overline{\text{RESET}}$ stays active, even if PURES is deactivated before $\overline{\text{RESET}}$. However the MainOsc stops upon de-assertion of $\overline{\text{RESET}}$.

<2> The MainOsc is started by S/W and stopped by PURES.

<3> MainOsc is operating with maximum amplification gain during active $\overline{\text{RESET}}$.

<4> The MainOsc is operating with MOSCCAMPSEL[1:0] amplification gain and it is set to maximum during active $\overline{\text{RESET}}$. After de-assertion of $\overline{\text{RESET}}$ the amplification gain returns to its previous level.

<5> If the MainOsc was disabled, it operates with maximum amplification gain during active $\overline{\text{RESET}}$ and is stopped after de-assertion of $\overline{\text{RESET}}$.

12.2.6 Watchdog Timers reset (WDTAnRES)

The Watchdog Timers can be configured to generate a reset if the watchdog time expires. After watchdog reset, the Watchdog Timer reset flags RESF.RESF1 for WDTA0RES respectively RESF.RESF2 for WDTA1RES are set and the system reset SYSRES is asserted.

RESF.RESF1 (RESF.RESF2) is not cleared automatically, if WDTA0RES (WDTA1RES) is de-asserted. It is cleared by

- setting RESFC.RESFC1 = 1 (RESFC.RESFC2 = 1)
- power-up reset PURES (POCRES or DBRES)

12.2.7 Software reset (SWRES)

The software reset SWRES can be asserted by setting SWRESA.SWRESA0 = 1.

This generates a system reset SYSRES and sets the reset flag RESF.RESF0 = 1.

RESF.RESF0 is not cleared automatically. It is cleared by

- setting RESFC.RESFC0 = 1
- power-up reset PURES (POCRES or DBRES)

12.2.8 Clock Monitors reset ($\overline{\text{CLMAnRES}}$)

The Clock Monitors can generate the resets:

- $\overline{\text{CLMA0RES}}$, if a MainOsc fail is detected
- $\overline{\text{CLMA2RES}}$, if a High Speed IntOsc fail is detected
- $\overline{\text{CLMA3RES}}$, if a PLL0 fail is detected

Upon a Clock Monitor reset, the system reset SYSRES is generated and the respective reset flag in the RESF register is set.
These flags are not cleared automatically. They are cleared by

- setting RESFC.RESFC3 = 1 for $\overline{\text{CLMA0RES}}$, (RESFC.RESFC5 = 1 for $\overline{\text{CLMA2RES}}$, RESFC.RESFC6 = 1 for $\overline{\text{CLMA3RES}}$)
- power-up reset PURES (POCRES or DBRES)

12.2.9 Debugger reset (DBRES)

In case a debugger is connected, it can generate the reset DBRES.

DBRES activates the power-up reset PURES, thus operates in the same way like the Power-On-Clear reset POCRES:

- It resets some clock generators. Hence these clock generators stop operation and must be restarted.
- The reset status flag register RESF is cleared.

12.3 Registers

This section contains a description of all registers of the Reset Controller.

12.3.1 Writing to protected registers

Write protected registers are protected from inadvertent write access due to erroneous program execution, etc.

Following Reset Controller registers feature this special write protection:

- Software reset register SWRESA

Refer to the section *"Write protected Registers"* in the chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

12.3.2 Reset Controller registers overview

The Reset Controller is controlled and operated by the following registers:

Table 12-1 Reset Controller registers overview

Register name	Shortcut	Address
General reset flags registers		
Reset factor register	RESF	FF42 0160_H
Reset factor clear register	RESFC	FF42 0168_H
Software reset control registers		
Software reset register	SWRESA	FF42 0204_H
Low-Voltage Indicator reset control registers		
LVI control register	LVICNT	FF42 0200_H
Very-Low-Voltage flag control registers		
VLVF register	VLVF	FF42 0180_H
VLVF clear register	VLVFC	FF42 0188_H

12.3.3 Reset Controller general control registers details

(1) RESF- Reset factor register

This register contains information about which type of resets occurred since the last Power-On Clear reset.

Each reset condition sets the corresponding flag in the register.
For example, if a Clock Monitor $\overline{\text{CLMA0RES}}$ occurs after a Watchdog Timer reset WDTA0RES, RESF reads 0000 000A$_H$.

Access This register can be read in 32-bit units.

Address FF42 0160$_H$

Initial Value 0000 0000$_H$. This register is initialized by a power-up reset PURES (POCRES and DBRES).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RESF8	RESF7	RESF6	RESF5	0	RESF3	RESF2	RESF1	RESF0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 12-2 RESF register contents

Bit position	Bit name	Function
8	RESF8	External reset flag 0: no $\overline{\text{RESET}}$ reset occurred 1: $\overline{\text{RESET}}$ reset has occurred
7	RESF7	Low-Voltage Indicator reset flag 0: no LVIRES reset occurred 1: LVIRES reset has occurred
6	RESF6	Clock Monitor CLMA3 reset flag 0: no $\overline{\text{CLMA3RES}}$ reset occurred 1: $\overline{\text{CLMA3RES}}$ reset has occurred
5	RESF5	Clock Monitor CLMA2 reset flag 0: no $\overline{\text{CLMA2RES}}$ reset occurred 1: $\overline{\text{CLMA2RES}}$ reset has occurred
3	RESF3	PLL0 Clock Monitor CLMA0 reset flag 0: no $\overline{\text{CLMA0RES}}$ reset occurred 1: $\overline{\text{CLMA0RES}}$ reset has occurred
2	RESF2	Watchdog Timer WDTA1 reset flag 0: no WDTA1RES reset occurred 1: WDTA1RES reset has occurred
1	RESF1	Watchdog Timer WDTA0 reset flag 0: no WDTA0RES reset occurred 1: WDTA0RES reset has occurred
0	RESF0	Software reset flag 0: no SWRES reset occurred 1: SWRES reset has occurred

(2) RESFC - Reset factor clear register

This register clears the reset flags of the RESF register.

Access This register can be read/written in 32-bit units.

Address FF42 0168$_H$

Initial Value Reading this registers returns always 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	RESFC8	RESFC7	RESFC6	RESFC5	0	RESFC3	RESFC2	RESFC1	RESFC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 12-3 RESFC register contents

Bit position	Bit name	Function
8	RESFC8	External reset flag RESF.RESF8 flag clear 0: no function 1: clear RESF.RESF8
7	RESFC7	Low-Voltage Indicator reset flag RESF.RESF7 flag clear 0: no function 1: clear RESF.RESF7
6	RESFC6	PLL0 Clock Monitor CLMA3 reset flag RESF.RESF6 flag clear 0: no function 1: clear RESF.RESF6
5	RESFC5	High-Speed IntOsc Clock Monitor CLMA2 reset flag RESF.RESF5 flag clear 0: no function 1: clear RESF.RESF5
3	RESFC3	MainOsc Clock Monitor CLMA0 reset flag RESF.RESF3 flag clear 0: no function 1: clear RESF.RESF3
2	RESFC2	Watchdog Timer WDTA1 reset flag RESF.RESF3 flag clear 0: no function 1: clear RESF.RESF2
1	RESFC1	Watchdog Timer WDTA0 reset flag RESF.RESF2 flag clear 0: no function 1: clear RESF.RESF1
0	RESFC0	Software reset flag RESF.RESF0 flag clear 0: no function 1: clear RESF.RESF0

12.3.4 Software reset control registers details

(1) SWRESA - Software reset register

This register is used to generate a software reset SWRES.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 0204_H

Initial Value Reading this registers returns always 0000 0000_H.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SWRESA0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 12-4 SWRESA register contents

Bit position	Bit name	Function
0	SWRESA0	Software reset control 0: no function 1: generate software reset SWRES

12.3.5 Low-Voltage Indicator reset control registers

(1) LVICNT - LVI control register

This register is used to control the Low-Voltage Indicator and to select the LVI detection level.

Protection Writing to this register is protected by a special sequence of instructions by using the protection command register PROTCMD2.
Refer to the section *"Write protected Registers"* in chapter *"CPU System Functions"* for a detailed description how to write to write protected registers.

Access This register can be read/written in 32-bit units.

Address FF42 0200_H

Initial Value $0000\ 0000_H$. This register is initialized by a power-up reset PURES (POCRES and DBRES).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	LVIRESMK	LVICNT[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

Table 12-5 LVICNT register contents

Bit position	Bit name	Function
3	LVIRESMK	LVI reset LVIRES mask 0: LVIRES unmasked 1: LVIRES masked (LVIRES is not generated)
2 to 0	LVICNT[2:0]	LVI detection level **Note:** Refer to the Electrical Target Specification for the specification of the LVI detection levels.

LVICNT[2:0]	Detection level
000_B	LVI is deactivated
001_B	LVI level 1
010_B	LVI level 2
011_B	LVI level 3
$1XX00_B$	setting prohibited

Note If the selected LVI detection level is close to the Power-On-Clear detection level, both may detect low voltage at the same time.
In this case the Power-On-Clear reset POCRES is performed and the reset factor register RESF is cleared. Thus the LVI reset flag RESF.RESF7 does not indicate the LVI detection.

12.3.6 Very-Low-Voltage flag control registers

(1) VLVF - Very-Low-Voltage flag register

This register shows the status of the Very-Low-Voltage detection.

Access This register can be read in 32-bit units.

Address FF42 0180$_H$

Initial Value 0000 0001$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VLVF0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 12-6 VLVF register contents

Bit position	Bit name	Function
0	VLVF0	VLVI status 0: Very low voltage not detected 1: Very low voltage detected

(2) VLVFC - Very-Low Voltage flag clear register

This register clears the VLVF0 flag of the VLVF register.

Access This register can be read/written in 32-bit units.

Address

FF42 0188$_H$

Initial Value Reading this registers returns always 0000 0000$_H$.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VLV FC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 12-7 VLVFC register contents

Bit position	Bit name	Function
0	VLVFC	VLVF flag VLVF.VLVF0 flag clear 0: no function 1: clear VLVF.VLVF0

Chapter 13 OS Timer (OSTM)

This chapter contains a generic description of the OS Timer.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

13.1 V850E2/Fx4-H OSTM Features

Instances

This microcontroller has the following number of instances of the OS Timer.

Table 13-1 Instances of OS Timer

OS Timer	
Instances	1
Names	OSTM0

Instances index n

Throughout this chapter, the individual instances of the OS Timer are identified by the index “n” (n = 0), for example OSTMnCTL for the OS Timer n control register.

Register addresses

All OS Timer register addresses are given as address offsets from the individual base addresses <OSTMn_base>.
The <OSTMn_base> addresses of each OSTMn are listed in the following table:

Table 13-2 Register base addresses <OSTMn_base>

OSTMn instance	<OSTMn_base> address
OSTM0	FF80 0000_H

Clock supply All OS Timers provide one clock input.

Table 13-3 OSTM clock supply

OSTMn instance	OSTMn clock	Connected to
OSTM0	PCLK	Clock Controller CKSCLK_112

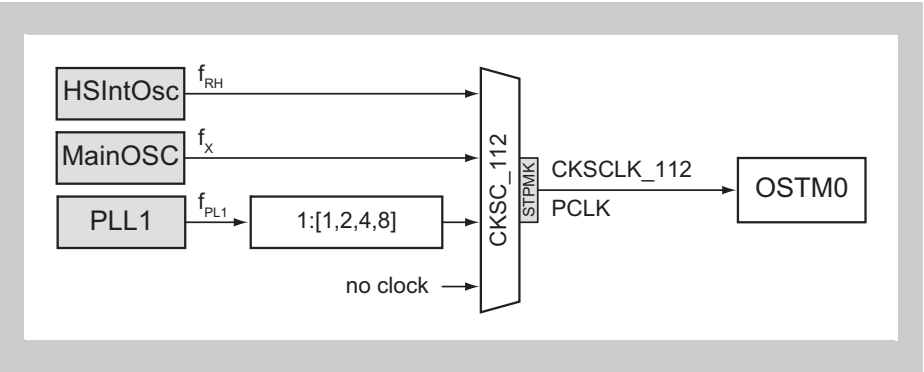


Figure 13-1 OSTM clock supply

Interrupts The OS Timers can generate the following interrupt requests:

Table 13-4 OSTMn interrupt requests

OSTMn signals	Function	Connected to
OSTM0TINT	OSTMn interrupt	Interrupt Controller INTOSTM0

OSTM H/W reset The OS Timers and their registers are initialized by the following reset signal:

Table 13-5 OSTMn reset signal

OSTMn	Reset signal
OSTM0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

13.2 Functional Overview

Features summary The OS Timer has two operation modes

– Interval timer mode

– Free-run compare mode

The following block diagram shows the main components of the OS Timer.

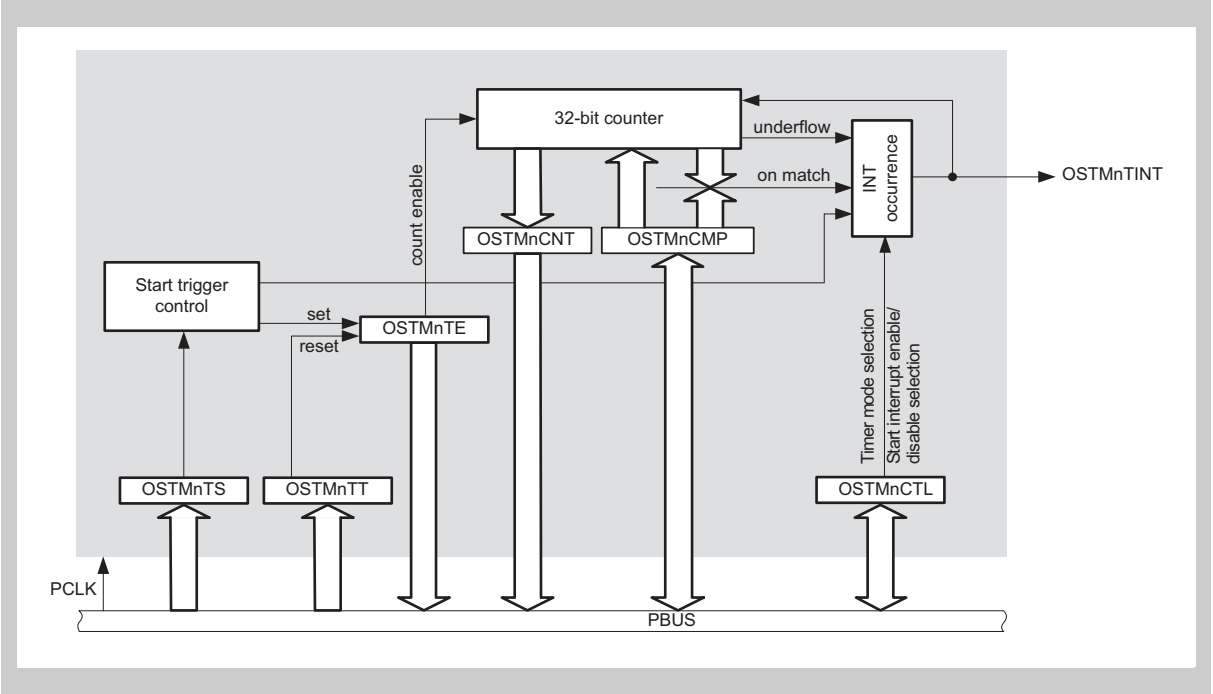


Figure 13-2 Block diagram of the OS Timer

13.3 Functional Description

The OS Timer is a 32-bit timer/counter.

It can be used as an interval timer or in free-run compare mode. The selected operation mode specifies the count direction and controls the interrupt request generation.

13.3.1 Count clock

The count clock of the OS Timer is defined by PCLK clock input.

This is illustrated in the following figures.

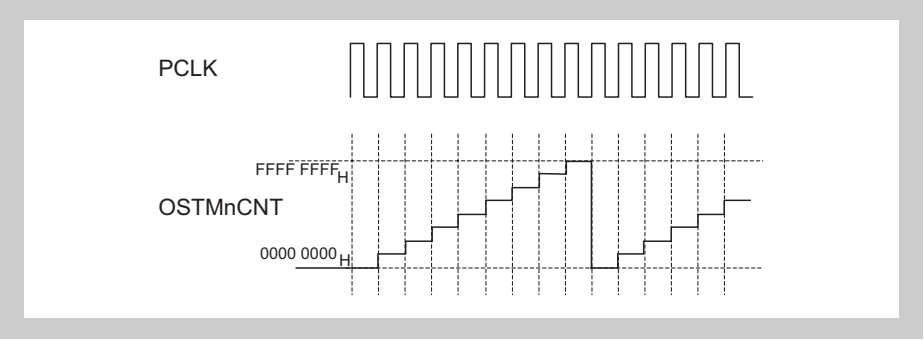


Figure 13-3 Counter operation

13.3.2 Interrupt request generation

By default, interrupt request OSTMnTINT is generated on counter underflow (interval timer mode) or when the counter matches the compare value (free-run compare mode).

Additionally, an interrupt request can be generated at counter start or counter restart. This is controlled by bit OSTMnCTL.OSTMnMD0.

This is illustrated in the following figure.

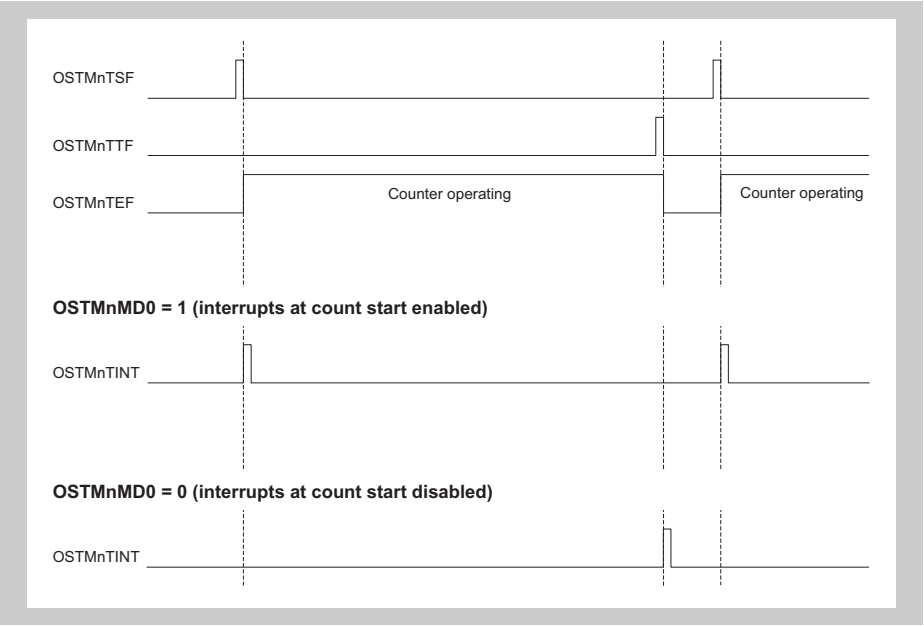


Figure 13-4 Interrupt generation at count start

13.3.3 Starting and stopping the timer

The OS Timer is started and stopped as follows:

Start The timer is started by setting the bit OSTMnTS.OSTMnTSF = 1 or

Status bit OSTMnTE.OSTMnTEF is set to 1 and bit OSTMnTS.OSTMnTSF returns to zero.

Depending on the operation mode, the counter starts to count down or to count up. Refer to *13.3.4 “Interval timer mode” on page 662* and *13.3.5 “Free-run compare mode” on page 666* for details.

Stop The timer is stopped by setting the bit OSTMnTT.OSTMnTTF = 1.

Status bit OSTMnTE.OSTMnTEF is cleared and bit OSTMnTT.OSTMnTTF immediately returns to zero.

When the counter is stopped, the register OSTMnCNT holds its current value until a new count operation starts.

13.3.4 Interval timer mode

In interval timer mode, the OSTM can be used as a reference timer generating interrupt requests at fixed intervals.

(1) Basic operation in interval timer mode

In interval timer mode, the timer counts down, starting from the value specified in the OSTMnCMP register. When the counter underflows (0000 0000_H is reached), an interrupt request OSTMnTINT is generated.

The interval timer mode is set by OSTMnCTL.OSTMnMD1 = 0.

The OSTMnCMP register can be rewritten at any time. If it is rewritten during count operation, the counter loads the new OSTMnCMP value when the next 0000 0000_H is reached. Then the counter continues with the new value.

OSTMnTINT period

The period of OSTMnTINT is:

- OSTMnTINT occurrence period = count clock period * (OSTMnCMP + 1)

The following figure shows the basic operation of the OS Timer in interval timer mode with counter start interrupt enabled (OSTMnCTL.OSTMnMD0 = 1):

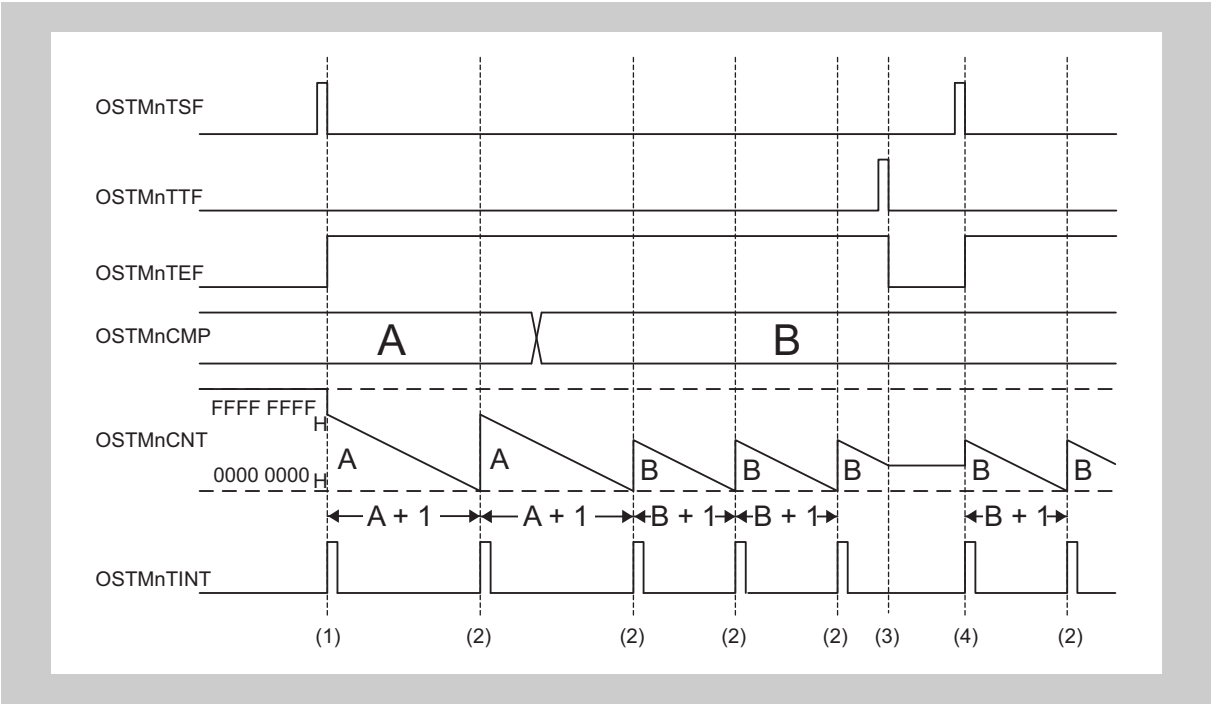


Figure 13-5 Timing diagram of OSTM in interval timer mode

(1) After counter start (OSTMnTS.OSTMnTSF = 1), the OSTMnTE.OSTMnTEF bit is set to indicate that the counter is enabled. OSTMnTS.OSTMnTSF returns to 0 in order to start the timer.
The counter counts down starting from the value of OSTMnCMP. The counter value is indicated in register OSTMnCNT. OSTMnTINT is asserted.

(2) When the counter reaches 0000 0000_H, the interrupt request OSTMnTINT is asserted. The counter loads the new start value from OSTMnCMP and continues to count down.

(3) At counter stop (OSTMnTT.OSTMnTTF = 1), the OSTMnTE.OSTMnTEF bit is cleared to indicate that the counter is disabled. OSTMnTT.OSTMnTTF returns to 0.The counter holds its current value until the counter is restarted.

(4) If the counter is restarted (OSTMnTS.OSTMnTSF = 1), the counter loads the new start value from OSTMnCMP and continues to count down.

Forced restart A forced restart of the counter is performed by setting OSTMnTS.OSTMnTSF = 1 during the count operation.

The counter loads the start value from the OSTMnCMP register and continues to count down.

The following figure shows the forced restart of the OS Timer in interval timer mode, with counter start interrupt enabled (OSTMnCTL.OSTMnMD0 = 1):

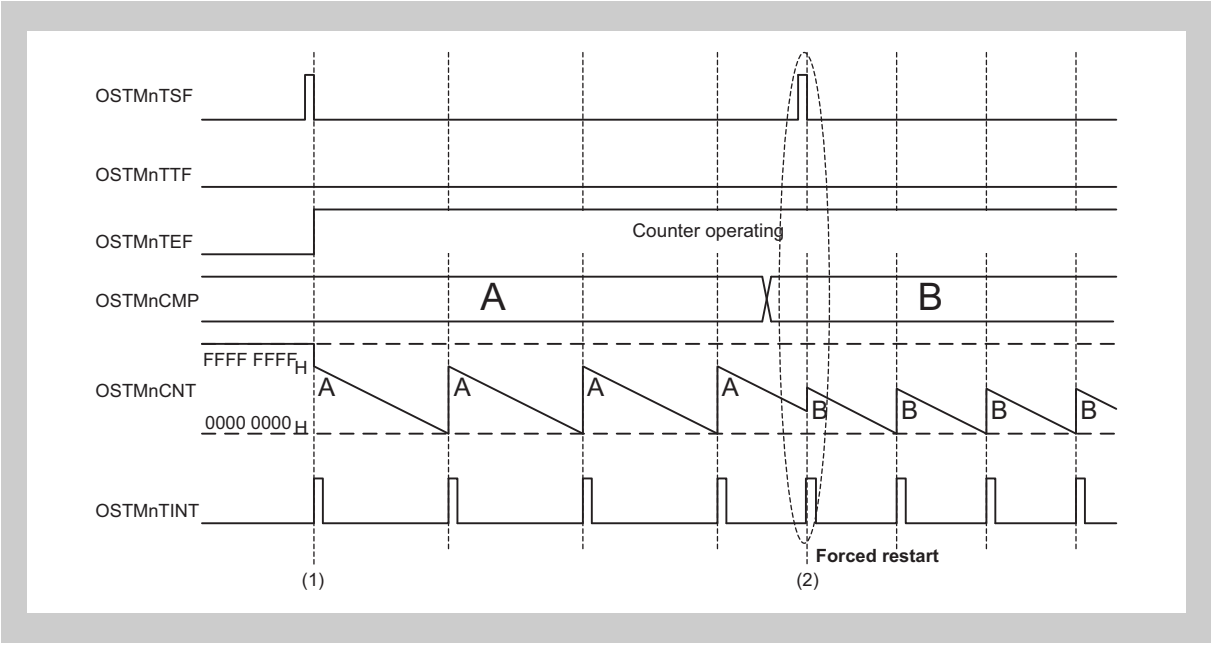


Figure 13-6 Timing diagram of forced restart in interval timer mode

The timing diagram above shows the following:

(1) The counter is started and stopped as shown and explained in *Figure 13-5 “Timing diagram of OSTM in interval timer mode” on page 663* .

(2) The counter is started again (OSTMnTS.OSTMnTSF = 1), while it is operating (OSTMnTE.OSTMnTEF = 1).
The counter immediately restarts counting down, starting with the current value of OSTMnCMP.
The interrupt request OSTMnTINT is asserted.

(2) Operation when OSTMnCMP = 0000 0000$_H$

When OSTMnCMP = 0000 0000$_H$ the OSTMnTINT interrupt request is set to 1, while the counter is enabled.

The following figure shows the operation of the OS Timer, when OSTMnCMP = 0000 0000$_H$, counter start interrupt is enabled (OSTMnCTL.OSTMnMD0 = 1):

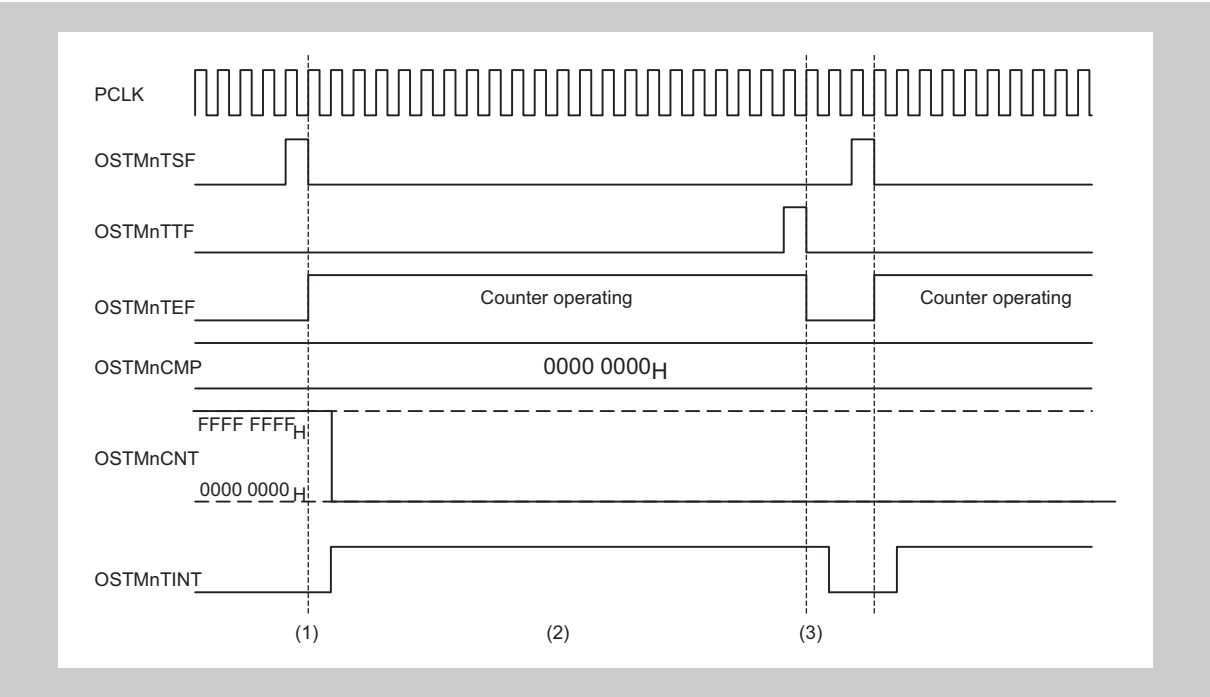


Figure 13-7 Timing diagram when OSTMnCMP = 0000 0000$_H$ in interval timer mode

The timing diagram above shows the following:

(1) After counter start, the counter starts counting, but will be reloaded with the OSTMnCMP value, and thus remains 0000 0000$_H$.

(2) The interrupt request OSTMnTINT is continuously asserted.

(3) After counter stop, the interrupt request OSTMnTINT is deasserted.

(3) Initialization for interval timer mode

The setting procedure in interval timer mode after a reset release is described below:

Initialization

1. Set the start value of the down-counter in the OSTMnCMP register.
2. Select the interval timer mode by clearing bit OSTMnCTL.OSTMnMD1.
3. Select the interrupt mode at counter start (OSTMnCTL.OSTMnMD0).

13.3.5 Free-run compare mode

(1) Basic operation in free-run compare mode

In free-run compare mode, the counter counts up from 0000 0000$_H$ to FFFF FFFF$_H$. When the value of the OSTMnCMP register matches the current count value, the OSTMnTINT interrupt request is output.

The free-run compare mode is selected by setting OSTMnCTL.OSTMnMD1 = 1.

The OSTMnCMP register can be rewritten at any time.

The following figure shows the basic operation of the OS Timer in free-run compare mode with counter start enabled (OSTMnCTL.OSTMnMD0 = 1):

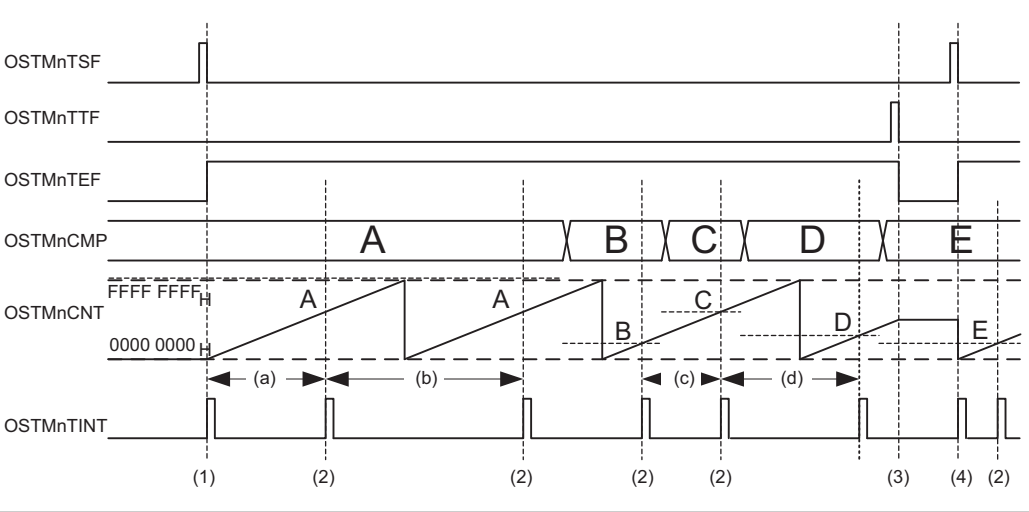


Figure 13-8 Timing diagram of OSTM in free-run compare mode

The timing diagram above shows the following:

(1) After counter start (OSTMnTS.OSTMnTSF = 1), the OSTMnTE.OSTMnTEF bit is set to indicate that the counter is enabled. OSTMnTS.OSTMnTSF returns to 0 in order to start the counter.
The counter counts up from 0000 0000$_H$ to FFFF FFFF$_H$. The counter value is indicated in register OSTMnCNT.

(2) When the value of the OSTMnCMP register matches the current count value, the interrupt request OSTMnTINT is asserted.

(3) At counter stop (OSTMnTT.OSTMnTTF = 1), the OSTMnTE.OSTMnTEF bit is cleared to indicate that the counter is disabled. OSTMnTT.OSTMnTTF returns to 0.
The counter holds its current value until the counter is restarted.

(4) If the counter is restarted (OSTMnTS.OSTMnTSF = 1), the counter starts counting from 0000 0000$_H$.

The OSTMnTINT occurrence period is different at count start and depends on the old and new compare value if OSTMnCMP is rewritten during operation:

Table 13-6 OSTMnTINT occurrence timing

Old compare	New compare	Counter value at time of rewrite	OSTMnTINT occurrence period	Label in timing diagram
Counter start			(A + 1) * count clock period	(a)
A	A	No rewrite	(FFFF FFFF$_H$ + 1) * count clock period	(b)
B	C > B	B < counter value < C	(C - B) * count clock period	(c)
C	D < C	Counter value > D,C	(FFFF FFFF$_H$ - C + D + 1) * count clock period	(d)

Forced restart A forced restart operation is not performed even if the bit OSTMnTS.OSTMnTSF is set during the count operation. The counter ignores this setting and continues counting.

(2) Operation when OSTMnCMP = 0000 0000$_H$

The following figure shows the operation of the OS Timer when OSTMnCMP = 0000 0000$_H$, counter start interrupt is enabled (OSTMnCTL.OSTMnMD0 = 1).

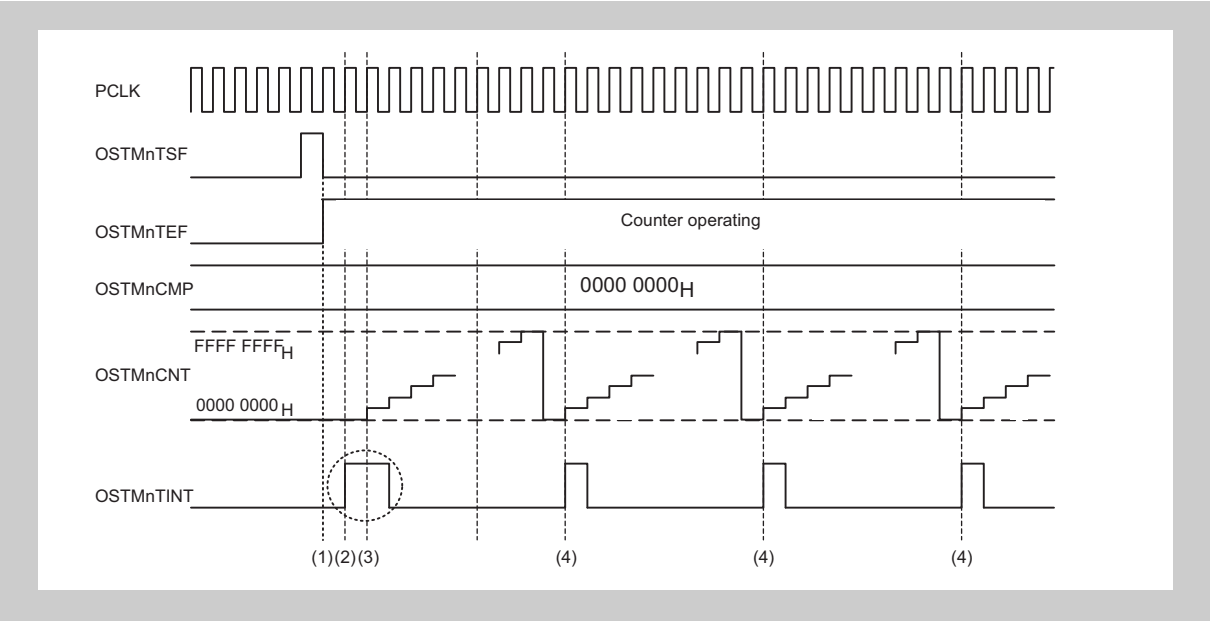


Figure 13-9 Timing diagram when OSTMnCMP = 0000 0000$_H$ in free-run compare mode

The timing diagram above shows the following:

(1) After counter start, the counter starts counting up from 0000 0000$_H$ to FFFF FFFF$_H$.

(2) The interrupt request OSTMnTINT at counter start is generated.

(3) If the current count value matches OSTMnCMP, the compare interrupt is generated. In the above case with OSTMnCMP = 0000 0000$_H$, OSTMnTINT stays active for 2 PCLK periods.

(4) Every FFFF FFFF$_H$ clock cycles the interrupt request OSTMnTINT is asserted.

(3) Initialization for free-run compare mode

The setting procedure in free-run compare mode after a reset release is described below:

Initialization

1. Set the compare value in the OSTMnCMP register.
2. Select the free-run compare mode by setting the bit OSTMnCTL.OSTMnMD1.
3. Select the interrupt mode at counter start by the bit OSTMnCTL.OSTMnMD0.

13.4 OS Timer Registers

This section contains a description of all registers of the OS Timer.

13.4.1 OS Timer registers overview

The OS Timer is controlled and operated by the following registers:

Table 13-7 OS Timer registers overview

Register name	Shortcut	Address
OSTM compare register	OSTMnCMP	<OSTMn_base>
OSTM counter register	OSTMnCNT	<OSTMn_base> + 4_H
OSTM count enable status register	OSTMnTE	<OSTMn_base> + 10_H
OSTM count start trigger register	OSTMnTS	<OSTMn_base> + 14_H
OSTM count stop trigger register	OSTMnTT	<OSTMn_base> + 18_H
OSTM control register	OSTMnCTL	<OSTMn_base> + 20_H
OSTM emulation register	OSTMnEMU	<OSTMn_base> + 24_H

<OSTMn_base> The base addresses <OSTMn_base> of the OSTMn is defined in the first section of this chapter under the key word “Register addresses”.

13.4.2 OS Timer registers details

(1) OSTMnCMP - OSTM compare register

This register stores the start value of the down-counter or the value with which the counter is compared, depending on the operation mode.

Access This register can be read/written in 32-bit units.

Address <OSTMn_base>

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSTMnCMP[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTMnCMP[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 13-8 OSTMnCMP register contents

Bit position	Bit name	Function
31 to 0	OSTMnCMP[31:0]	• In interval timer mode: start value of the down-counter • In free-run compare mode: compare value

(2) OSTMnCNT - OSTM counter register

This register indicates the count value of the timer.

Access This register can be read in 32-bit units.

Address <OSTMn_base> + 4_H

Initial Value The initial value depends on the operation mode of the OS Timer, see *Table 13-10 "Correlation between operation mode, counting direction and initial value" on page 670* .

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OSTMnCNT[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OSTMnCNT[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 13-9 OSTMnCNT register contents

Bit position	Bit name	Function
31 to 0	OSTMnCNT[31:0]	32-bit counter value

The following table shows the correlation between operation mode, counting direction and initial value. The initial value is the value that is read after the operating mode has been changed.

Table 13-10 Correlation between operation mode, counting direction and initial value

Timer operation mode	OSTMnCTL.OSTMnMD1	Counting direction	Initial value
Interval Timer Mode	0[a]	Down	FFFF $FFFF_H$
Free Running Compare Mode	1	Up	0000 0000_H

a) Value after reset.

(3) OSTMnTE - OSTM count enable status register

This register indicates whether the counter is enabled or disabled.

Access This register can be read in 8-bit units.

Address <OSTMn_base> + 10_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTEF
R	R	R	R	R	R	R	R

Table 13-11 OSTMnTE register contents

Bit position	Bit name	Function
0	OSTMnTEF	Indicates, whether the counter is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting OSTMnTS.OSTMnTSF to 1 sets this bit to 1. Setting OSTMnTT.OSTMnTTF to 1 resets this bit to 0.

Note If the counter is disabled, the counter value OSTMnCNT remains its value.

If the counter is restarted again, it

- continues with this value in interval timer mode.
- restarts with count value 0000 0000_H in free-run compare mode.

(4) OSTMnTS - OSTM count start trigger register

This register starts the counter.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <OSTMn_base> + 14_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMnTSF
R	R	R	R	R	R	R	W

Table 13-12 OSTMnTS register contents

Bit position	Bit name	Function
0	OSTMnTSF	Starts the counter: 0: No function 1: Starts the counter and sets OSTMnTE.OSTMnTEF = 1. When the counter is enabled, this bit returns to 0. • In interval timer mode, forced restart is executed when this bit is set while OSTMnTE.OSTMnTEF = 1. • In free-run compare mode, setting this bit is ignored while OSTMnTE.OSTMnTEF = 1.

(5) OSTMnTT - OSTM count stop trigger register

This register stops the counter.

Access This register can be written in 8-bit units. It is always read as 00_H.

Address <OSTMn_base> + 18_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	OSTMn TTF
R	R	R	R	R	R	R	W

Table 13-13 OSTMnTT register contents

Bit position	Bit name	Function
0	OSTMn TTF	Stops the counter: 0: No function 1: Stops the counter and clears the bit OSTMnTE.OSTMnTEF. When the counter has stopped, this bit returns to 0.

(6) OSTMnCTL - OSTM control register

This register specifies the counter operation mode and controls the generation of the interrupt request OSTMnTINT at counter start.

Access This register can be read/written in 8-bit units. It can only be written when the counter is disabled (OSTMnTE.OSTMnTEF = 0).

Address <OSTMn_base> + 20_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	OSTMn MD1	OSTMn MD0
R	R	R	R	R	R	R/W	R/W

Table 13-14 OSTMnCTL register contents

Bit position	Bit name	Function
1	OSTMnMD1	Specifies the counter operation mode: 0: Interval timer mode 1: Free-run compare mode
0	OSTMnMD0	Controls the OSTMnTINT interrupt request at counter start: 0: Disables interrupt at counter start 1: Enables interrupt at counter start

(7) OSTMnEMU - OSTMn emulation register

This register controls whether the OSTMn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <OSTMn_base> + 24_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
OSTMn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 13-15 OSTMnEMU register contents

Bit position	Bit name	Function
7	OSTMn SVSDIS	Emulation control 0: OSTMn can be stopped during emulation 1: OSTMn continuous operating during emulation

Chapter 14 Window Watchdog Timer A (WDTA)

This chapter contains a generic description of the Window Watchdog Timer A (WDTA).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

14.1 V850E2/Fx4-H WDTA Features

Instances

This microcontroller has the following number of instances of the Window Watchdog Timer A.

Table 14-1 Instances of WDTA

Window Watchdog Timer A	
Instances	2
Names	WDTA0, WDTA1

Instances index n

Throughout this chapter, the individual instances of a Window Watchdog Timer A are identified by the index “n” (n = 0 to 1), for example, WDTAnWDTE for the WDTAn Watchdog Timer enable register.

Register addresses

All WDTAn register addresses are given as address offsets from the individual base address <WDTAn_base>.
The base address <WDTAn_base> of each WDTAn is listed in the following table:

Table 14-2 Register base addresses <WDTAn_base>

WDTAn instance	<WDTAn_base> address
WDTA0	FF80 6000_H
WDTA1	FF80 7000_H

Clock supply All Window Watchdog Timers A provide two clock inputs.

Table 14-3 WDTAn clock supply

WDTAn instance	WDTAn clock	Connected to
WDTA0	PCLK	Clock Generator CKSCLK_A02
	WDTATCKI	Clock Generator CKSCLK_A07
WDTA1	PCLK	Clock Generator CKSCLK_005
	WDTATCKI	Clock Generator CKSCLK_007

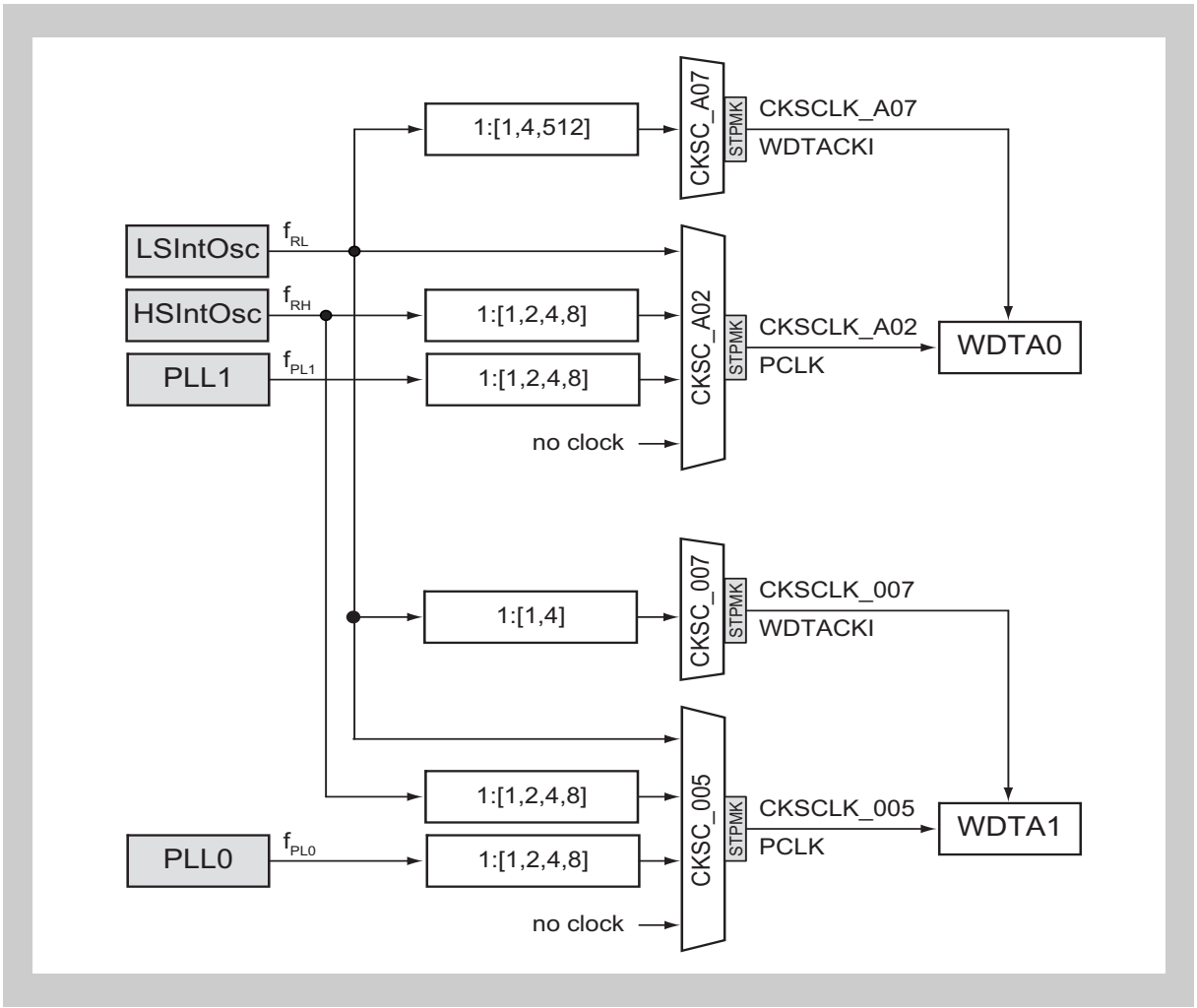


Figure 14-1 WDTA clock supply

Interrupts and reset outputs

The interrupts and reset outputs of the WDTAn are listed in the table below.

Table 14-4 WDTA interrupts and reset outputs

WDTAn signals	Function	Connected to
WDTA0:		
WDTA0RES	Error reset	Reset Controller WDTA0RES
WDTA0TNMI	Error non-maskable interrupt	Interrupt Controller WDTA0NMI[a], Motor control signals Hi-Z control[b]
INTWDT0	75% interrupt	Interrupt Controller INTWDTA0[a]
WDTA1:		
WDTA1RES	Error reset	Reset Controller WDTA1RES
WDTA1TNMI	Error non-maskable interrupt	Interrupt Controller WDTA1NMI[a]
INTWDT1	75% interrupt	Interrupt Controller INTWDTA1[a]

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

b) Refer to cahpter *20 "Motor Control" on page 1576* for details.

WDTA H/W reset

The Window Watchdog Timers A and their registers are initialized by the following reset signal:

Table 14-5 WDTAn reset signal

WDTAn	Reset signal
WDTA0	• Reset Controller: SYSRES
WDTA1	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

WDTA during DEEPSTOP mode

If the V850E2/Fx4-H enters the DEEPSTOP stand-by mode, WDTA0 remains in operation and would need to be retriggered, although the CPU is not operating.
In order to stop also WDTA0 during DEEPSTOP, its operation clock CKSCLK_A07 can be stopped by setting CKSC_A07.STPMK_A07 = 0.

WDTA during debugging

When operating the V850E2/Fx4-H microcontroller under control of a debugger and the microcontroller is stopped, e.g. because of a breakpoint hit, the WDTA is stopped and all WDTA registers can not be written. In particular it is not possible to re-trigger the WDTA by the debugger's single-step execution.

14.2 WDTA Start-up Options

The start-up options determine the start-up configuration of the WDTA after reset release. They are described in the following table.

Table 14-6 WDTA start-up options (1/2)

Start-up option	Function	Description	Connected to
OPWDEN	WDTA enable/disable	Enables/disables the WDTA: 0: WDTA is disabled 1: WDTA is enabled	• WDTA0: Flash mask option OPBT0.OPBT0[19] • WDTA1: Flash mask option OPBT0.OPBT0[23]
OPWDOVF[2:0]	Count clock setting	Specifies the reset value of the count clock control bits WDTAnMD.WDTAnOVF[2:0].	WDTA0/WDTA1: Flash mask option OPBT0.OPBT0[18:16]
OPWDTPR	Start mode signal selector	Specifies the signal that sets the start mode: 0: OPWDRUN start-up option 1: WDTATRTYP input signal If WDTATRTYP is selected (OPWDTPR = 1), the start mode depends on the reset type. Refer to *14.2.1 "V850E2/Fx4-H WDTAn start modes" on page 678* and *"WDTA after reset release"* in the section *"Functional Description"* for details.	fixed to 0
OPWDRUN	Automatic start enable	Specifies the start mode: 0: Software trigger start mode 1: Automatic start mode Refer to *14.2.1 "V850E2/Fx4-H WDTAn start modes" on page 678* and *"WDTA after reset release"* in the section *"Functional Description"* for details.	• WDTA0: Flash mask option OPBT0.OPBT0[20] • WDTA1: Flash mask option OPBT0.OPBT0[24]
OPWDVAC	Varying Activation Code (VAC) enable	Enables/disables the Varying Activation Code function (VAC) 0: VAC is disabled 1: VAC is enabled Refer to *"WDTA trigger"* in the section *"Functional Description"* for details.	• WDTA0: Flash mask option OPBT0.OPBT0[22] • WDTA1: Flash mask option OPBT0.OPBT0[26]

Table 14-6 WDTA start-up options (2/2)

Start-up option	Function	Description	Connected to
OPWDWS[1:0]	Initial open window size setting	Specifies the reset value of the open window size control bits WDTAnMD.WDTAnWS[1:0]. The open window size control bits only apply after the first WDTA trigger and not after reset release. After reset release the open window size is 100%. Refer to *"Window function"* in the section *"Functional Description"* for details.	fixed to 11_B
OPWDINT	INTWDTn (75% interrupt) request generation	Specifies the reset value of control bit WDTAnMD.WDTAnWIE. This bit enables/ disables the output of the 75% interrupt request INTWDTn. Refer to *"75% interrupt output"* in the section *"Functional Description"* for details.	fixed to 0

14.2.1 V850E2/Fx4-H WDTAn start modes

If a Watchdog Timer is enabled (OPWDEN = 1), its start mode depends on OPWDRUN:

- OPWDRUN = 0: software trigger mode
- OPWDRUN = 0: automatic start mode

Caution If the Watchdog Timer is disabled by the flash mask option OPWDEN = 0, it can not be enabled afterwards.

14.3 Functional Overview

Features summary

The Window Watchdog Timer A has the following functions:

- Operation mode after reset selectable by using start-up options
- Fixed activation code and variable activation code (VAC) selectable
- Two start modes available:
  - Automatic start mode
  - Software trigger mode
- Reset-dependent start mode selection
- Operation upon error detection selectable:
  - Generation of NMI request WDTAnTNMI on error detection
  - Generation of reset WDTAnTRES on error detection
- Interrupt request generation at 75% of the counter overflow value
- Window function

The following figure shows the main components of the Window Watchdog Timer A:

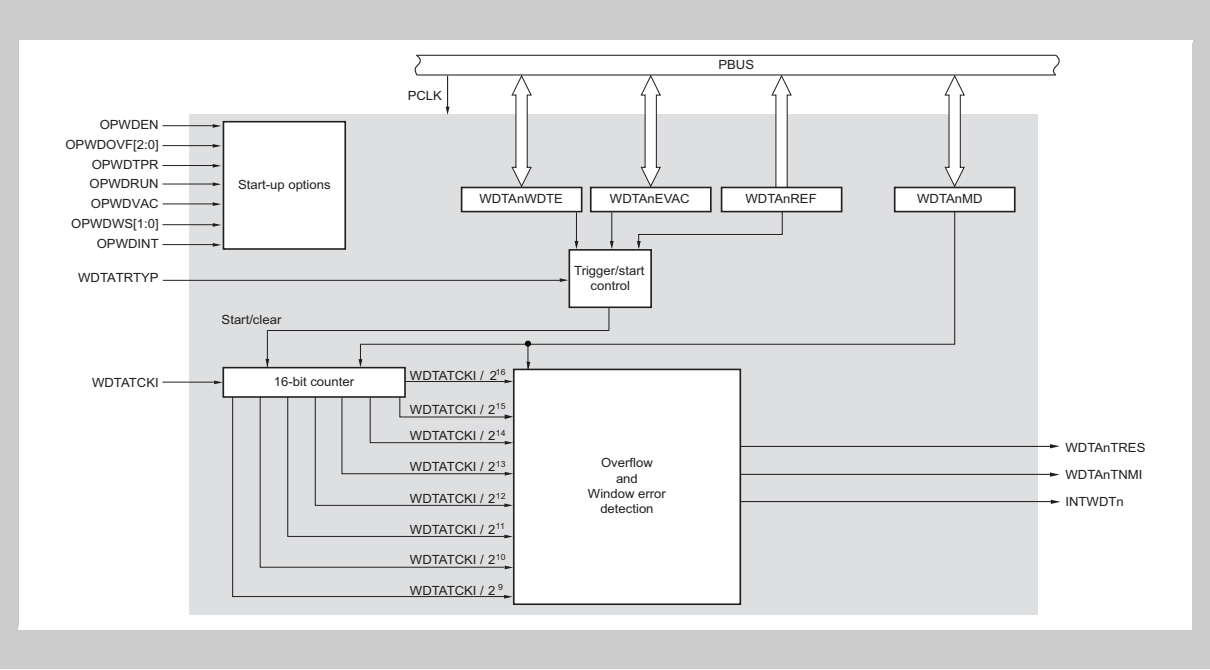


Figure 14-2 Block diagram of the Window Watchdog Timer A

14.4 Functional Description

The Window Watchdog Timer A generates a reset or a non-maskable interrupt if the 16-bit counter overflows or if any other error condition is fulfilled.
For a description of all error conditions, refer to the section *"Error detection"*.

The counter is cleared and restarted every time a WDTA trigger occurs in the open window period.
Refer to the sections *"WDTA trigger"* and *"Window function"* for details.

At 75% of the maximum counter value, the WDTA can generate an interrupt request INTWDTn.
Refer to the section *"75% interrupt output"* for details.

After reset release, the start-up options specify the start mode and the WDTA settings. The settings can be modified by writing the Watchdog Timer mode register WDTAnMD.
Refer to the section *"WDTA after reset release"* for details.

14.4.1 WDTA after reset release

(1) Start modes

The WDTA provides two modes for the counter start after reset release:

- Software trigger start mode

  The counter value remains 0000_H after reset release.

  The counter is started with the first WDTA trigger.
  The first trigger can occur any time after reset release.

- Automatic start mode

  The counter starts automatically after reset release.
  The first trigger must occur before the counter overflows.

(2) Start mode selection

The start mode can be selected as follows:

- By start-up options
- By the WDTATRTYP input signal

  This signal indicates the reset type. Thus, the selected start mode after reset release depends on the reset type.

The start mode selection is listed in the following table.

Table 14-7 Start mode selection

Start-up options		Input signal	Reset type	Start mode
OPWDTPR	OPWDRUN	WDTATRTYP		
0	0	Ignored	Ignored	Software trigger
	1			Automatic
1	0	Ignored	Ignored	Software trigger
	1	0	Any apart from automatic start reset source	Software trigger
		1	Automatic start reset source	Automatic

(3) WDTA settings after reset release

The WDTA settings are as follows between reset release and the first trigger:

Function	Setting	Remark
Start mode	Specified by start-up options	For a description of the start modes, refer to the section *"WDTA after reset release"*.
Count clock		
75% interrupt mode		
Error mode	Reset mode	Any error condition before the first trigger generates a reset.
Open window size	100%	If automatic start mode is specified, the first trigger is valid any time before the counter overflows.

Change WDTA settings

After the first trigger, the WDTA continues according to the settings of the Watchdog Timer mode register WDTAnMD.

To change the WDTA settings, WDTAnMD must be written *before* the first trigger. Changing the value of WDTAnMD *after* the first trigger leads to an error.

If WDTAnMD is not changed before the first trigger, the WDTA mode is specified by the initial value of WDTAnMD.

The new or initial value of WDTAnMD applies after the first trigger.

Automatic start mode timing

The automatic start mode timing and the changes to the WDTA settings are illustrated in the following figure.

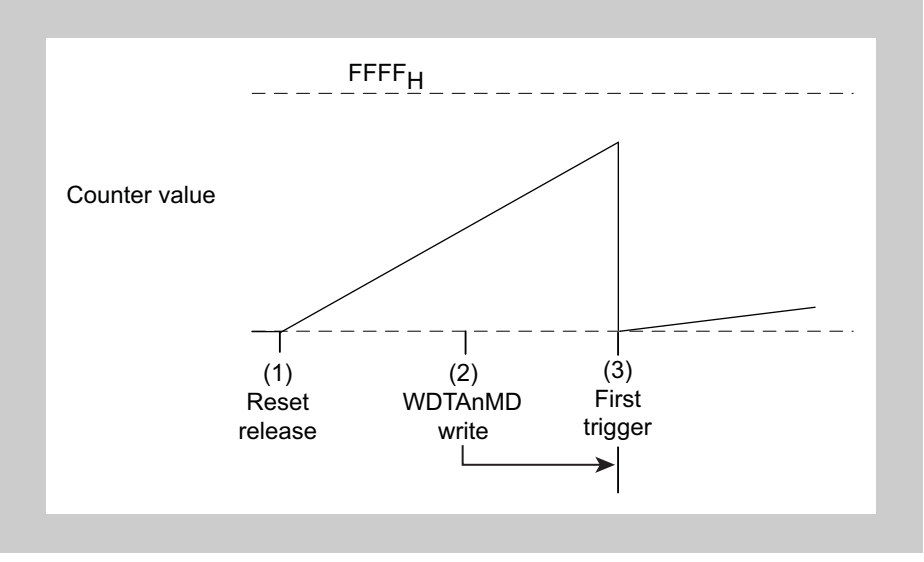


Figure 14-3 Timing diagram of WDTA start in automatic start mode

The timing diagram above shows the following:

1. After reset release, the counter starts immediately.
   The count clock is specified by the start-up options, for example:
   – Count clock after reset release = WDTATCKI / 2^{13}
   (OPWDOVF[2:0] = 100_B)

2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The first trigger must occur before the counter overflows.
   After the first trigger, the settings specified in WDTAnMD are applied, for example a new count clock:
   – Count clock after first trigger = WDTATCKI / 2^{16}
   (WDTAnMD.WDTAnOVF[2:0] = 111_B

   With the decreased count clock, the counter value rises more slowly over time.

Software trigger start mode timing

The software trigger start mode timing and the changes to the WDTA settings are illustrated in the following figure.

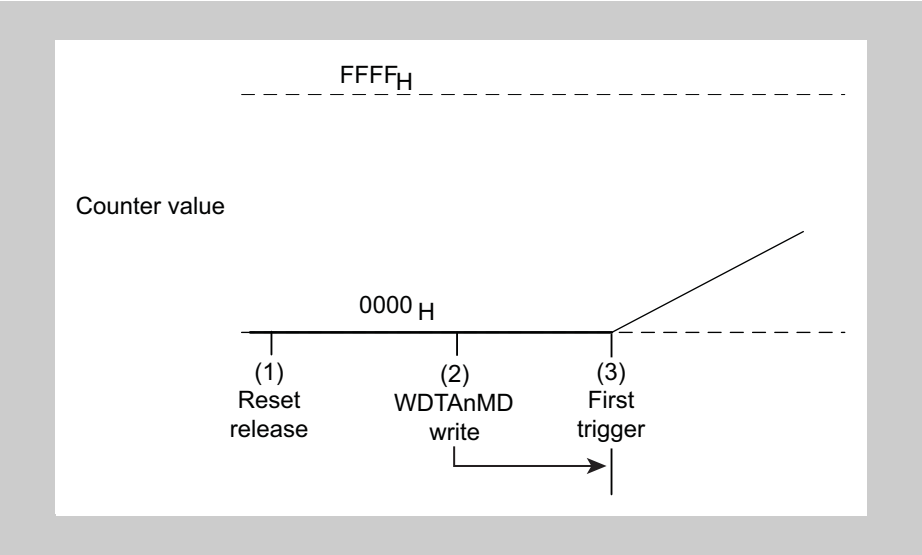


Figure 14-4 Timing diagram of WDTA start in software trigger start mode

The timing diagram above shows the following:

1. After reset release, the counter remains 0000_H until the first trigger.
   The count clock is specified by the start-up options, but it does not have any effect.
2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The counter starts at the first trigger.
   The count clock and other settings specified in WDTAnMD are applied.

14.4.2 WDTA trigger

The WDTA trigger has the following functions:

- Counter start trigger in software trigger start mode
- Counter restart trigger to avoid counter overflow

The trigger register to be used differs depending on whether the activation code is fixed or variable. The type of activation code and the associated trigger register are specified by using the start-up option OPWDVAC.

Table 14-8 Trigger register and activation code

Type of activation code	Trigger register	Activation code
Fixed	WDTAnWDTE	AC_H
Variable	WDTAnEVAC	Refer to the section *"Varying Activation Code calculation"* for details.

(1) Variable activation code calculation

The variable activation code (ExpectWDTE) is calculated using a reference value in register WDTAnREF. The reference value in WDTAnREF is updated each time the trigger register WDTAnEVAC is written.

- Use the expression below to calculate the variable activation code (ExpectWDTE):
  ExpectWDTE = AC_H – WDTAnREF (old)
- Use the expression below to calculate how the WDTAnREF value is updated:
  WDTAnREF (new) = rotate left 1 bit (ExpectWDTE)

Note WDTAnREF is also updated upon an erroneous WDTA trigger, i.e.:

- When the value written to WDTAnEVAC differs from the expected activation code.
- When writing to WDTAnEVAC outside the open window.

In either case, a WDTAnTRES reset or WDTAnTNMI interrupt request is also generated.

The table below lists the variable activation codes according to the number of triggers.

Table 14-9 Expected activation code development

No[a]	WDTAnREF (old)		ExpectWDTE (AC_H - WDTAnREF)		WDTAnREF (new)	
0	0000 0000	00_H	1010 1100	AC_H	0101 1001	59_H
1	0101 1001	59_H	0101 0011	53_H	1010 0110	$A6_H$
2	1010 0110	$A6_H$	0000 0110	06_H	0000 1100	$0C_H$
…	…	…	…	…	…	…

a) Number of triggers after reset

This generates a sequence of 122 different numbers that have to be used to re-trigger the Watchdog Timer.

Note Bit 7 of the WDTAnEVAC register (WDTAnEVAC7) cannot be cleared to 0 after the WDTA has been started. Thus even if bit 7 of the activation code is 0, the WDTA will not stop.

14.4.3 Error detection

The conditions for error detection are:

- Overflow interval time is exceeded (counter overflow)
- Wrong activation code is written to the trigger register
- Writing to the trigger register outside the open window.
- Illegal update of Watchdog Timer mode register WDTAnMD:
  - Writing a *new* value to WDTAnMD after the first trigger leads to an error detection.
  - Writing the same value to WDTAnMD after the first trigger does *not* lead to an error detection.

Error mode When an error is detected, either an NMI request (WDTAnTNMI) or a reset (WDTAnTRES) is generated.
WDTAnMD.WDTAnERM selects the error mode:

- WDTAnMD.WDTAnERM = 0: NMI mode
- WDTAnMD.WDTAnERM = 1: reset mode

Caution Due to the fact, that any modification of the WDTAnMD register becomes effective with the first WDTA trigger, the initial value of WDTAnMD remains active until the first WDTA trigger.

This means in particular, that a WDTAnTRES is generated in the following case:

- WDTA is changed to NMI mode by WDTAnMD.WDTAnERM = 0
- WDTAnMD is written again before the first WDTA trigger (error condition)

After error detection

After detection of an error the Watchdog Timer operation is stopped. For restarting the Watchdog Timer a reset is necessary.

- In NMI mode:
  The NMI service routine has to initiate assertion of a reset, e.g. by applying a software reset.
- In reset mode:
  The Watchdog Timer generates the reset by itself.

The following figure shows the reset or NMI request generation when the counter overflows and automatic start mode is selected.

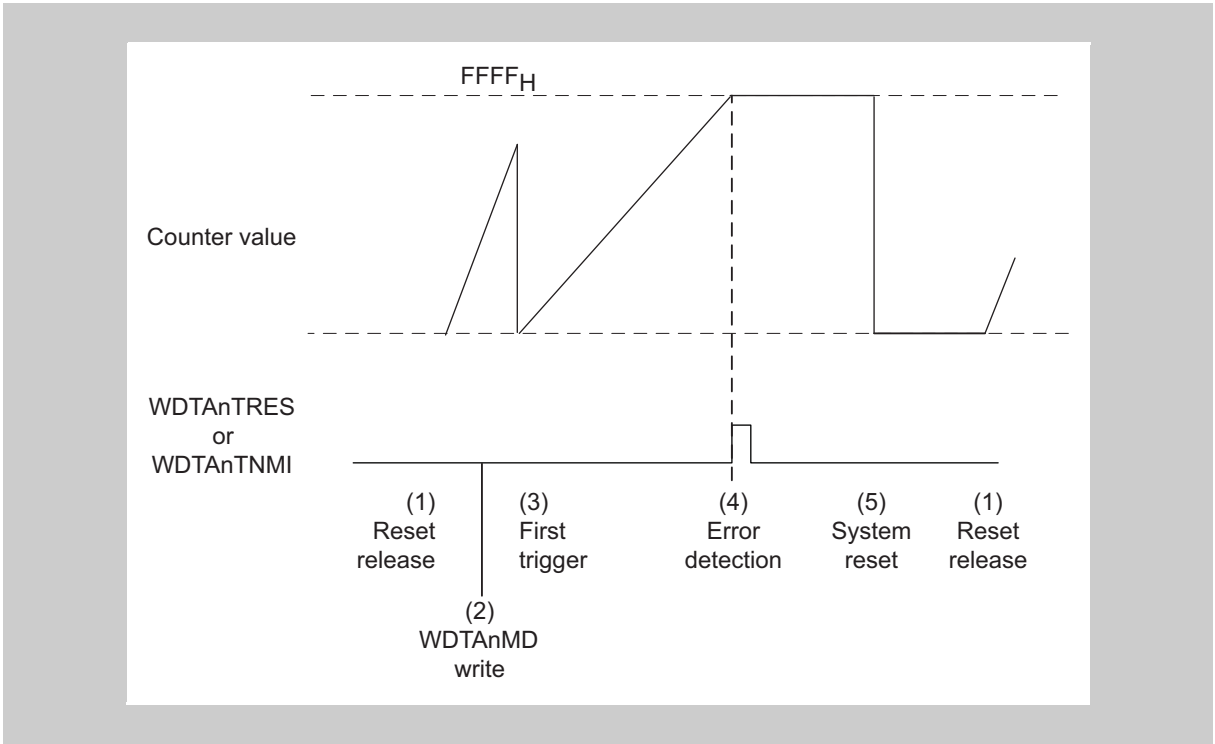


Figure 14-5 Timing diagram of WDTA NMI request or reset generation

The timing diagram above shows the following:

1. After reset release, the counter starts (automatic start mode is selected).
2. WDTAnMD is written before the first trigger. However, the settings are not applied immediately.
3. The counter is cleared at the first trigger and the new WDTA settings are applied.
4. When the counter overflows, an error is detected. Depending on the error mode, either interrupt request WDTAnTNMI or reset WDTAnTRES is generated.
   The counter value remains, the Watchdog Timer is stopped and can not be restarted, until a system reset is performed.
5. When the system is reset, the counter is cleared and stopped until reset release.

14.4.4 75% interrupt output

When the counter reaches 75% of the maximum counter value, the interrupt request INTWDTn is generated.

This function can be automatically enabled with the start-up option OPWDINT = 1.
By use of WDTAnMD.WDTAnWIE this function can be enabled respectively disabled afterwards.

The following figure shows the 75% interrupt request generation under following conditions:

- Automatic start mode selected
- Count clock changes after first trigger

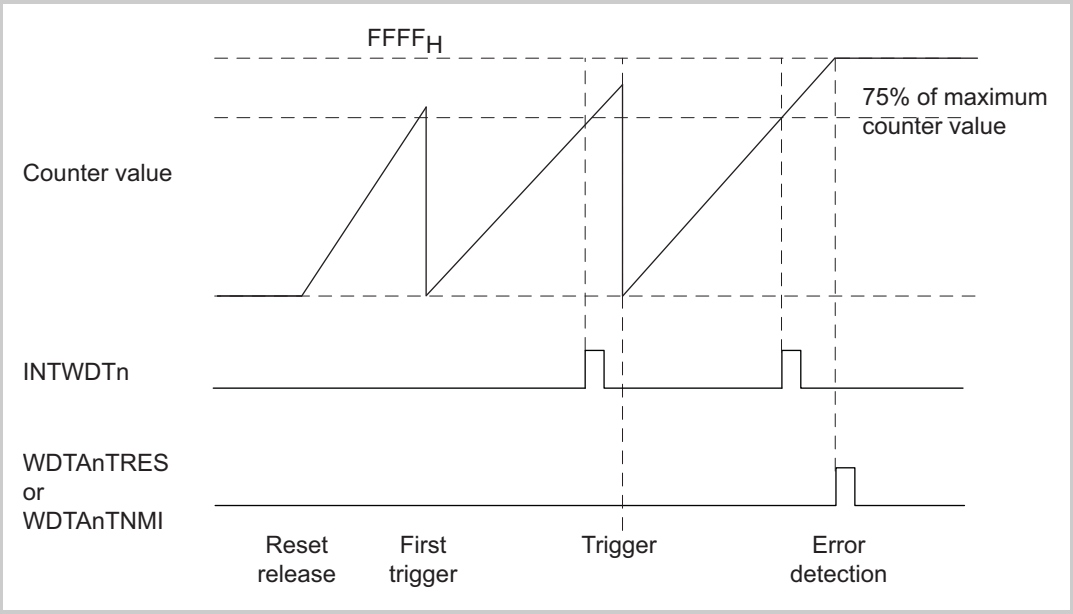


Figure 14-6 Timing diagram of WDTA 75% interrupt output

14.4.5 Window function

When the open window size is set to less than 100%, an error is detected if the trigger occurs outside the open window.

The definition of the open window size differs before and after the first trigger:

- After reset release, the open window size is 100%.

  OPWDWS[1:0] and bits WDTAnMD.WDTAnWS[1:0] are not applied.
- After the first trigger, the open window size is specified by bits WDTAnMD.WDTAnWS[1:0].

The following figure shows WDTA operation with an open window size of 25% and with automatic start mode selected.

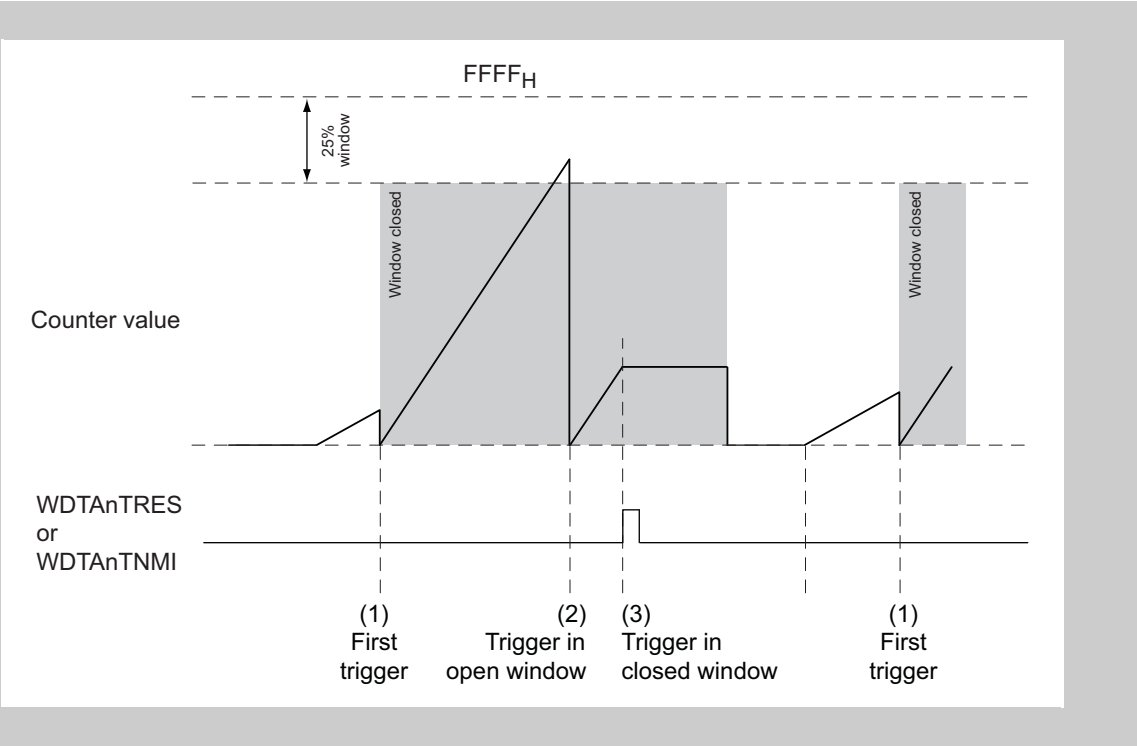


Figure 14-7 Timing diagram of WDTA window function

The timing diagram above shows the following:

1. The open window size is fixed to 100% for the first trigger.
2. A trigger that occurs in the open window does not generate an error.
3. A trigger that occurs in the closed window generates a WDTAnTNMI request or a WDTAnTRES reset, depending on the selected error mode.

14.5 Application hint: Evaluation of the Watchdog status

In case the Watchdog Timer status needs to be evaluated, the following procedures could for example be used.

Variable activation code disabled

- if WDTAnWDTE = $2C_H$: Watchdog was not activated
- if WDTAnWDTE = AC_H: Watchdog was activated

Variable activation code enabled

- if WDTAnREF ≠ 0: Watchdog was activated
- if WDTAnREF = 0:
  - if WDTAnEVAC = $2C_H$: Watchdog was not activated
  - if WDTAnEVAC ≠ $2C_H$: Watchdog was activated

14.6 WDTA registers

This section contains a description of all registers of the WDTA.

14.6.1 WDTA registers overview

The WDTA is controlled and operated by the following registers:

Table 14-10 WDTA register overview

Register name	Shortcut	Address
WDTA enable register	WDTAnWDTE	<WDTAn_base> + 0000_H
WDTA VAC enable register	WDTAnEVAC	<WDTAn_base> + 0004_H
WDTA reference value register	WDTAnREF	<WDTAn_base> + 0008_H
WDTA mode register	WDTAnMD	<WDTAn_base> + $000C_H$

<WDTAn_base> The base addresses <WDTAn_base> of the WDTAn is defined in the first section of this chapter under the key word “Register addresses”.

14.6.2 WDTA registers details

(1) WDTAnWDTE – WDTA enable register

This register is the WDTA start control and trigger register if the VAC function is not used (start-up option OPWDVAC = 0).

WDTA trigger Writing AC_H to this register restarts the counter.
Refer to the section *"WDTA trigger"* for details.

The behaviour of this register depends on activation of the VAC function, refer to the table *"WDTAnWDTE behaviour"* below.

Access This register can be read/written in 8-bit units.

Address <WDTAn_base> + 0000_H

Initial Value x010 1100_B.The initial value of the "x" bit depend on the start-up options OPWDEN, OPWDTPR, WDTATRTYP, OPWDRUN and OPWDVAC.
Refer to the table *"WDTAnRUN initial value"* below.

7	6	5	4	3	2	1	0
WDTAnRUN	0	1	0	1	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 14-11 WDTAnWDTE register contents

Bit position	Bit name	Function
7	WDTAnRUN	Enables/disables the WDTAn: 0: WDTAn disabled 1: WDTAn enabled Since the WDTA can not be stopped once it was started, this bit can only be cleared by a reset.

Initial value WDTAnRUN is only valid if WDTA is enabled (OPWDEN = 1) and VAC is disabled (OPWDVAC = 0). In this case, the initial value of bit WDTAnRUN depends on other start-up options is listed below:

Table 14-12 WDTAnRUN initial value

Start-up options		Input signal	Start mode	Initial value of WDTAnRUN
OPWD TPR	OPWD RUN	WDTATRTYP		
0	0	Ignored	Software trigger	0
0	1	Ignored	Automatic	1
1	Ignored	0	Software trigger	0
1	Ignored	1	Automatic	1

The behaviour of WDTAnWDTE during read/write accesses depends on activation of the VAC mode, as shown in the table below.

Table 14-13 WDTAnWDTE behaviour

OPWDVAC	WDTAnWDTE		Remark
	Read	Write	
0	AC_H	WDTA trigger AC_H[a]	VAC disabled: WDTAnWDTE enabled
1	$2C_H$	ignored	VAC enabled: WDTAnWDTE disabled

a) Any other write value will lead to an error detection.

(2) WDTAnEVAC – WDTA enable VAC register

This register is the start control and trigger register if the VAC function is used (start-up option OPWDVAC = 1).

WDTA trigger Writing the correct activation code to this register restarts the counter. Refer to the section *"WDTA trigger"*.

The behaviour of this register depends on activation of the VAC function, refer to the table *"WDTAnEVAC behaviour"* below.

Access This register can be read/written in 8-bit units.

Address <WDTAn_base> + 0004_H

Initial Value $x010\ 1100_B$.The initial value of the "x" bit depend on the start-up options OPWDEN, OPWDTPR, WDTATRTYP, OPWDRUN and OPWDVAC. Refer to the table *"WDTAnRUN initial value"* below.

7	6	5	4	3	2	1	0
WDTAnEVAC7	0	1	0	1	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 14-14 WDTAnEVAC register contents

Bit position	Bit name	Function
7	WDTAnEVAC7	Enables/disables the WDTAn: 0: WDTAn disabled 1: WDTAn enabled Since the WDTA can not be stopped once it was started, this bit can only be cleared by a reset. Thus even if bit 7 of the activation code is 0, the WDTA will not stop.

Initial value WDTAnEVAC7 is only valid if WDTA is enabled (OPWDEN = 1) and VAC is enabled (OPWDVAC = 1). In this case, the initial value of bit WDTAnEVAC7 depends on other start-up options is listed below:

Table 14-15 WDTAnRUN initial value

Start-up options		Input signal	Start mode	Initial value of WDTAnEVAC7
OPWD TPR	OPWD RUN	WDTATRTYP		
0	0	Ignored	Software trigger	0
0	1	Ignored	Automatic	1
1	Ignored	0	Software trigger	0
1	Ignored	1	Automatic	1

The behaviour of WDTAnEVAC during read/write accesses depends on activation of the VAC mode, as shown in the table below.

Table 14-16 WDTAnEVAC behaviour

OPWDVAC	WDTAnEVAC		Remark
	Read	Write	
0	$2C_H$	ignored	VAC disabled: WDTAnEVAC disabled
1	last written VAC	WDTA trigger VAC[a]	VAC enabled: WDTAnEVAC enabled

a) Any other write value will lead to an error detection.

(3) WDTAnREF – WDTA reference value register

This register contains the reference value for calculating the activation code of the VAC function. It is automatically updated after every trigger operation. Refer to the section *"WDTA trigger"*.

If VAC is disabled (OPWDVAC = 0), reading this register returns 00_H.

Access This register can be read in 8-bit units.

Address <WDTAn_base> + 0008_H

Initial Value 00_H

7	6	5	4	3	2	1	0
WDTAnREF[7:0]							
R	R	R	R	R	R	R	R

Table 14-17 WDTAnREF register contents

Bit position	Bit name	Function
7 to 0	WDTAnREF[7:0]	Reference value for activation code calculation.

(4) WDTAnMD – WDTA mode register

This register specifies the overflow interval time, the 75% interrupt output mode, the error mode, and the open window size.

It can be updated only once after reset release and before the first trigger.

Caution The updated WDTAnMD value is effective after the next WDTA trigger.

Updating this register after the WDTA has been started leads to error detection, but the read value of this register can be written without generating an error.

Access This register can be read/written in 8-bit units.

Address <WDTAn_base> + $000C_H$

Initial Value $0xxx\ x1xx_B$. The initial value of the "x" bits depend on the start-up options OPWDOVF[2:0], OPWDINT and OPWDWS[1:0].
Refer to *"WDTA Start-up Options"* in the first section of this chapter.

7	6	5	4	3	2	1	0
0	WDTAnOVF[2:0]			WDTAnWIE	WDTAnERM	WDTAnWS[1:0]	
R/W[a]	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) Writing to this bit is ignored, reading returns 0.

Table 14-18 WDTAnMD register contents (1/2)

Bit position	Bit name	Function
6 to 4	WDTAnOVF[2:0]	Selects the count clock and thus the overflow interval time: WDTAnOVF2 / WDTAnOVF1 / WDTAnOVF0 / Count clock 0 / 0 / 0 / WDTATCKI / 2^9 0 / 0 / 1 / WDTATCKI / 2^{10} 0 / 1 / 0 / WDTATCKI / 2^{11} 0 / 1 / 1 / WDTATCKI / 2^{12} 1 / 0 / 0 / WDTATCKI / 2^{13} 1 / 0 / 1 / WDTATCKI / 2^{14} 1 / 1 / 0 / WDTATCKI / 2^{15} 1 / 1 / 1 / WDTATCKI / 2^{16} The overflow interval time is calculated from the count clock: Overflow interval time = 1 / count clock frequency The reset values of WDTAnOVF[2:0] depend on start-up option OPWDOVF[2:0].
3	WDTAnWIE	Enables/disables the 75% interrupt request INTWDTn: 0: INTWDTn disabled 1: INTWDTn enabled The reset value of WDTAnWIE depends on start-up option OPWDINT.

Table 14-18 WDTAnMD register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>2</td><td>WDTAnERM</td><td>Specifies the error mode:
0: NMI request mode
1: Reset mode</td></tr>
<tr><td>1 to 0</td><td>WDTAnWS[1:0]</td><td>Selects the open window size:
<table>
<tr><th>WDTAnWS1</th><th>WDTAnWS0</th><th>Open window size</th></tr>
<tr><td>0</td><td>0</td><td>25%</td></tr>
<tr><td>0</td><td>1</td><td>50%</td></tr>
<tr><td>1</td><td>0</td><td>75%</td></tr>
<tr><td>1</td><td>1</td><td>100%</td></tr>
</table>
The reset values of WDTAnWS[1:0] depend on start-up option OPWDWS[1:0].</td></tr>
</table>

Chapter 15 Timer Array Unit A (TAUA)

This chapter contains a generic description of the Timer Array Unit A (TAUA).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

15.1 V850E2/Fx4-H TAUA Features

Instances

This microcontroller has following number of instances of the Timer Array Unit A.

Table 15-1 Instances of TAUA

Timer Array Unit A	
Instance	1
Name	TAUA0

Instances index n

Throughout this chapter, the individual instances of a Timer Array Unit A is identified by the index "n" (n = 0), for example, TAUAnTOM for the TAUAn channel output mode register.

Channel index m

The Timer Array Unit A has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7 , 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All TAUAn register addresses are given as address offsets to the individual base address <TAUAn_base>.
The base address <TAUAn_base> of each TAUAn is listed in the following table:

Table 15-2 Register base addresses <TAUAn_base>

TAUAn instance	<TAUAn_base> address
TAUA0	FF80 8000_H

Clock supply All Timer Array Units A provide one clock input:

Table 15-3 TAUAn clock supply

TAUAn instance	TAUAn clock	Connected to
TAUA0	PCLK	Clock Generator CKSCLK_006

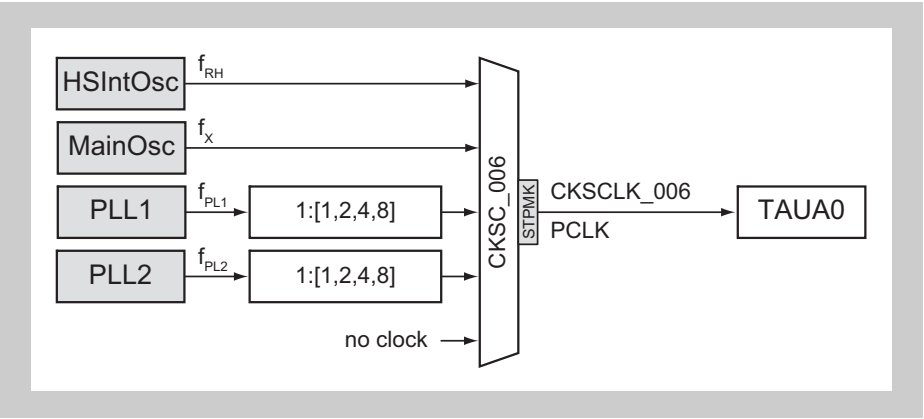


Figure 15-1 TAUA clock supply

Interrupts and DMA/DTS The Timer Array Unit A can generate the following interrupt and DMA/DTS requests:

Table 15-4 TAUAn interrupt and DMA/DTS requests (1/2)

TAUAn signals	Function	Connected to
TAUA0:		
INTTAUA0I0 to INTTAUA0I7[a]	Channel 0 to 7 interrupt	Interrupt Controller INTTAUA0I0 to INTTAUA0I7[b]
INTTAUA0I8[a]	Channel 8 interrupt	Interrupt Controller INTTAUA0I8[b] DMA Controller trigger 12
INTTAUA0I9[a]	Channel 9 interrupt	Interrupt Controller INTTAUA0I9[b] DMA Controller trigger 13
INTTAUA0I10[a]	Channel 10 interrupt	Interrupt Controller INTTAUA0I10[b] DMA Controller trigger 14 DTS Controller trigger 40
INTTAUA0I11[a]	Channel 11 interrupt	Interrupt Controller INTTAUA0I11[b] DMA Controller trigger 15 DTS Controller trigger 41
INTTAUA0I12[a]	Channel 12 interrupt	Interrupt Controller INTTAUA0I12[b] DMA Controller trigger 16 DTS Controller trigger 42
INTTAUA0I13[a]	Channel 13 interrupt	Interrupt Controller INTTAUA0I13[b] DMA Controller trigger 17 DTS Controller trigger 43

Table 15-4 TAUAn interrupt and DMA/DTS requests (2/2)

TAUAn signals	Function	Connected to
INTTAUA0I14[a]	Channel 14 interrupt	Interrupt Controller INTTAUA0I14[b] DMA Controller trigger 18 DTS Controller trigger 44
INTTAUA0I15[a]	Channel 15 interrupt	Interrupt Controller INTTAUA0I15[b] DMA Controller trigger 19 DTS Controller trigger 45

a) These interrupts can be used to as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

b) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

TAUA H/W reset

The Timer Array Units A and their registers are initialized by the following reset signal:

Table 15-5 TAUAn reset signal

TAUAn	Reset signal
TAUA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

I/O signals

The I/O signals of the Timer Array Unit A are used for various purposes, as listed in the following table.

Note √ / – : used / not used for the function

Table 15-6 TAUA0 I/O signals (1/2)

TAUA0 signal	Connected to				
	Miscellaneous[a]	Port	Encoder Timer[b]	Motor Control[c]	PWM delay[d]
TAUA0 input channels: All TAUA0 port input signals are passed through a noise filter, refer to the section *"Port Filters"* in chapter *"Port Functions"* further details.					
TAUA0TTIN0	FCN0 TSOUT, URTE10RX	TAUA0I0	√	–	–
TAUA0TTIN1	FCN1 TSOUT, URTE11RX	TAUA0I1	√	–	–
TAUA0TTIN2	URTE8RX	TAUA0I2	√	–	–
TAUA0TTIN3	URTE9RX	TAUA0I3	–	–	–
TAUA0TTIN4	–	TAUA0I4	–	–	–
TAUA0TTIN5	–	TAUA0I5	–	–	–
TAUA0TTIN6	–	TAUA0I6	–	–	–
TAUA0TTIN7	–	TAUA0I7	–	–	–
TAUA0TTIN8	–	TAUA0I8	–	–	–
TAUA0TTIN9	–	TAUA0I9	–	–	–
TAUA0TTIN10	FCN0 TSOUT	TAUA0I10	–	√	–

Table 15-6 TAUA0 I/O signals (2/2)

TAUA0 signal	Connected to				
	Miscellaneous[a]	Port	Encoder Timer[b]	Motor Control[c]	PWM delay[d]
TAUA0TTIN11	FCN1 TSOUT	TAUA0I11	–	√	–
TAUA0TTIN12	FCN2 TSOUT	TAUA0I12	–	√	–
TAUA0TTIN13	FCN3 TSOUT	TAUA0I13	–	√	–
TAUA0TTIN14	–	TAUA0I14	–	√	–
TAUA0TTIN15	DCN0 TSOUT	TAUA0I15	–	√	–
TAUA0 output channels:					
TAUA0TTOUT0	–	TAUA0O0	–	–	–
TAUA0TTOUT1	–	TAUA0O1	–	–	√
TAUA0TTOUT2	–	TAUA0O2	–	√	√
TAUA0TTOUT3	–	TAUA0O3	–	–	–
TAUA0TTOUT4	–	TAUA0O4	–	√	–
TAUA0TTOUT5	–	TAUA0O5	–	√	√
TAUA0TTOUT6	–	TAUA0O6	–	√	√
TAUA0TTOUT7	–	TAUA0O7	–	√	–
TAUA0TTOUT8	–	TAUA0O8	–	√	–
TAUA0TTOUT9	–	TAUA0O9	–	√	√
TAUA0TTOUT10	–	TAUA0O10	–	√	√
TAUA0TTOUT11	–	TAUA0O11	–	√	–
TAUA0TTOUT12	–	TAUA0O12	–	√	–
TAUA0TTOUT13	–	TAUA0O13	–	√	√
TAUA0TTOUT14	–	TAUA0O14	–	√	√
TAUA0TTOUT15	–	TAUA0O15	–	√	–

a) Refer to the section *"TAUA Input Selection"* below for further details.
b) Refer to the chapter "Encoder Timer (ENCA)" for further details.
c) Refer to the chapter *"Motor Control"* for further details.
d) Refer to the chapter *"PWM Diagnostic"* for further details.

Caution The initial state of the input signals noise filters block the input signals. Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

15.2 TAUA Input Selection

15.2.1 TAUA0 input selection

The TAUA0 has several options to connect its input signals:

- FCN0, FCN1, FCN2, FCN3, FCN4, DCN0 time stamp output signals TSOUT for timing measurements
- URTE8 to URTE11 data receive signals URTEnRX for baud rate measurement
- odd numbered TAUA0 input signals TAUA0TTINm can be connected to even numbered input signals

The following figure depicts the TAUA0 input selection scheme:

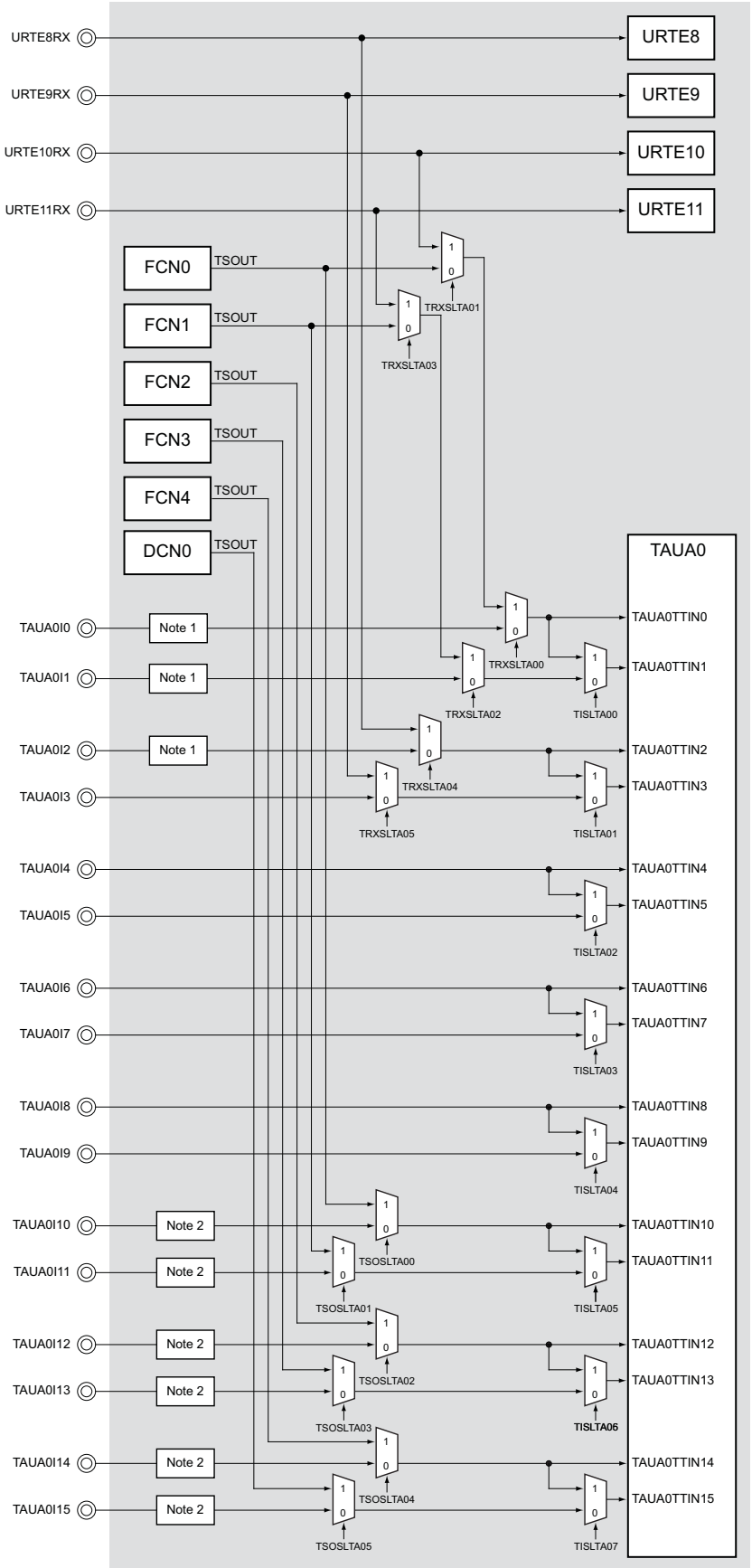


Figure 15-2 TAUA0 input selector scheme

Notes
1. See chapter *"Encoder Timer"* for additional input selector settings.
2. See chapter *"Motor Control"* for additional input selector settings.

The following tables show the optional inputs to several TAUAn inputs:

Table 15-7 TAUA0 input selections - TAUA0TTIN0 to TAUA0TTIN3

TAUA0 input	Input options	Selection control	
		TISLTA0 register bit	TRXSLTA0 register bits
TAUA0TTIN0	Port TAUA0I0	X	TRXSLTA0[1:0] = $x0_B$
	FCN0 TSOUT (CAN I/F 0 time stamp output)		TRXSLTA0[1:0] = 01_B
	Port URTE10RX (URTE10 data receive signal)		TRXSLTA0[1:0] = 11_B
TAUA0TTIN1	Port TAUA0I1	TISLTA00 = 0	TRXSLTA0[3:2] = $x0_B$
	FCN1 TSOUT (CAN I/F 1 time stamp output)		TRXSLTA0[3:2] = 01_B
	Port URTE11RX (URTE11 data receive signal)		TRXSLTA0[3:2] = 11_B
	Input to TAUA0TTIN0	TISLTA00 = 1	X
TAUA0TTIN2	Port TAUA0I2	X	TRXSLTA0[4] = 0_B
	Port URTE8RX (URTE8 data receive signal)		TRXSLTA0[4] = 1_B
TAUA0TTIN3	Port TAUA0I3	TISLTA01 = 0	TRXSLTA0[5] = 0_B
	Port URTE9RX (URTE9 data receive signal)		TRXSLTA0[5] = 1_B
	Input to TAUA0TTIN2	TISLTA01 = 1	X

Table 15-8 TAUA0 input selections - TAUA0TTIN4 to TAUA0TTIN9

TAUA0 input	Input options	Selection control	
TAUA0TTIN4	Port TAUA0I4	TISLTA0.TISLTA02 =	x
TAUA0TTIN5	Port TAUA0I5		0
	Port TAUA0I4		1
TAUA0TTIN6	Port TAUA0I6	TISLTA0.TISLTA03 =	x
TAUA0TTIN7	Port TAUA0I7		0
	Port TAUA0I6		1
TAUA0TTIN8	Port TAUA0I8	TISLTA0.TISLTA04 =	x
TAUA0TTIN9	Port TAUA0I9		0
	Port TAUA0I8		1

Table 15-9 TAUA0 input selections - TAUA0TTIN10 to TAUA0TTIN15

TAUA0 input	Input options	Selection control	
		TISLTA0 register bit	TSOSLTA0 register bits
TAUA0TTIN10	Port TAUA0I10	X	TSOSLTA0[0] = 0_B
	FCN0 TSOUT (FCN0 time stamp output)		TSOSLTA0[0] = 1_B
TAUA0TTIN11	Port TAUA0I11	TISLTA05 = 0	TSOSLTA0[1] = 0_B
	FCN1 TSOUT (FCN1 time stamp output)		TSOSLTA0[1] = 1_B
	Input to TAUA0TTIN10	TISLTA05 = 1	X
TAUA0TTIN12	Port TAUA0I12	X	TSOSLTA0[2] = 0_B
	FCN2 TSOUT (FCN0 time stamp output)		TSOSLTA0[2] = 1_B
TAUA0TTIN13	Port TAUA0I13	TISLTA06 = 0	TSOSLTA0[3] = 0_B
	FCN1 TSOUT (FCN3 time stamp output)		TSOSLTA0[3] = 1_B
	Input to TAUA0TTIN12	TISLTA06 = 1	X
TAUA0TTIN14	Port TAUA0I14	X	TSOSLTA0[4] = 0_B
	FCN4 TSOUT (FCN4 time stamp output)		TSOSLTA0[4] = 1_B
TAUA0TTIN15	Port TAUA0I15	TISLTA07 = 0	TSOSLTA0[5] = 0_B
	DCN0 TSOUT (DCN0 time stamp output)		TSOSLTA0[5] = 1_B
	Input to TAUA0TTIN14	TISLTA07 = 1	X

(1) TISLTA0 - TAUA0 odd inputs selection register

This register selects the input signals to the odd TAUA0 inputs.

Access This register can be read/written in 8-bit units.

Address FF77 1000_H

Initial Value 00_H

7	6	5	4	3	2	1	0
TISLTA07	TISLTA06	TISLTA05	TISLTA04	TISLTA03	TISLTA02	TISLTA01	TISLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-10 TISLTA0 register contents

Bit position	Bit name	Function
7	TISLTA07	Selection of TAUA0TTIN15: 0: Port TAUA0I15 or DCN0 time stamp TSOUT[a] 1: TAUA0TTIN14 input
6	TISLTA06	Selection of TAUA0TTIN13: 0: Port TAUA0I13 or FCN3 time stamp TSOUT[a] 1: TAUA0TTIN12 input
5	TISLTA05	Selection of TAUA0TTIN11: 0: Port TAUA0I11 or FCN1 time stamp TSOUT[a] 1: TAUA0TTIN10 input
4	TISLTA04	Selection of TAUA0TTIN9: 0: Port TAUA0I9 1: TAUA0TTIN8 input
3	TISLTA03	Selection of TAUA0TTIN7: 0: Port TAUA0I7 1: TAUA0TTIN6 input
2	TISLTA02	Selection of TAUA0TTIN5: 0: Port TAUA0I5 1: TAUA0TTIN4 input
1	TISLTA01	Selection of TAUA0TTIN3: 0: Port TAUA0I3 or port URTE9RX[b] 1: TAUA0TTIN2 input
0	TISLTA00	Selection of TAUA0TTIN1: 0: Port TAUA0I1 or FCN1 time stamp TSOUT[bc] 1: TAUA0TTIN0 input

a) Refer also to the TSOSLTA0 register description.
b) Refer also to the TRXSLTA0 register description.
c) See chapter *"Encoder Timer (ENCA)"* for input selectors

(2) TRXSLTA0 - TAUA0 receive input selection register

This register selects the input signals to several TAUAn inputs out of signals related to other functional blocks (FCN, URTE).

Access This register can be read/written in 8-bit units.

Address FF77 1004_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	TRXSLTA05	TRXSLTA04	TRXSLTA0[3:2]		TRXSLTA0[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-11 TRXSLTA0 register contents

Bit position	Bit name	Function
5	TRXSLTA05	Selection of TAUA0TTIN3 selector TISLTA0.TISLTA01 input:
4	TRXSLTA04	Selection of TAUA0TTIN2:
3, 2	TRXSLTA0[3:2]	Selection of TAUA0TTIN1 selector TISLTA0.TISLTA00 input:
1, 0	TRXSLTA0[1:0]	Selection of TAUA0TTIN0:

TRXSLTA05	TAUA0TTIN3 selector input
0_B	Port TAUA0I3[a]
1_B	Port URTE9RX

TRXSLTA04	TAUA0TTIN2
0_B	Port TAUA0I2[a]
1_B	Port URTE8RX

TRXSLTA0[3:2]	TAUA0TTIN1 selector input
00_B	Port TAUA0I1[a]
10_B	
01_B	FCN1 time stampTSOUT
11_B	Port URTE11RX

TRXSLTA0[1:0]	TAUA0TTIN0
00_B	Port TAUA0I0[a]
10_B	
01_B	FCN0 time stamp TSOUT
11_B	Port URTE10RX

a) See chapter *"Encoder Timer (ENCA)"* for additional input selector settings.

(3) TSOSLTA0 - TAUA0 input selection register

This register selects the input signals to several TAUAn inputs out of signals related to other functional blocks (FCN, DCN, URTE).

Access This register can be read/written in 8-bit units.

Address FF77 2014$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	TSOSLTA05	TSOSLTA04	TSOSLTA03	TSOSLTA02	TSOSLTA01	TSOSLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-12 TSOSLTA0 register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>5</td><td>TSOSLTA05</td><td>Selection of TAUA0TTIN15 selector TISLTA0.TISLTA07 input:
<table>
<tr><th>TSOSLTA05</th><th>TAUA0TTIN15 selector input</th></tr>
<tr><td>0</td><td>Port TAUA0I15</td></tr>
<tr><td>1</td><td>DCN0 time stamp TSOUT</td></tr>
</table></td></tr>
<tr><td>4</td><td>TSOSLTA04</td><td>Selection of TAUA0TTIN14:
<table>
<tr><th>TSOSLTA04</th><th>TAUA0TTIN14</th></tr>
<tr><td>0</td><td>Port TAUA0I14</td></tr>
<tr><td>1</td><td>FCN4 time stamp TSOUT</td></tr>
</table></td></tr>
<tr><td>3</td><td>TSOSLTA03</td><td>Selection of TAUA0TTIN13 selector TISLTA0.TISLTA06 input:
<table>
<tr><th>TSOSLTA03</th><th>TAUA0TTIN13 selector input</th></tr>
<tr><td>0</td><td>Port TAUA0I13</td></tr>
<tr><td>1</td><td>FCN3 time stamp TSOUT</td></tr>
</table></td></tr>
<tr><td>2</td><td>TSOSLTA02</td><td>Selection of TAUA0TTIN12:
<table>
<tr><th>TSOSLTA02</th><th>TAUA0TTIN12</th></tr>
<tr><td>0</td><td>Port TAUA0I12</td></tr>
<tr><td>1</td><td>FCN2 time stamp TSOUT</td></tr>
</table></td></tr>
<tr><td>1</td><td>TSOSLTA01</td><td>Selection of TAUA0TTIN11 selector TISLTA0.TISLTA05 input:
<table>
<tr><th>TSOSLTA01</th><th>TAUA0TTIN11 selector input</th></tr>
<tr><td>0</td><td>Port TAUA0I11</td></tr>
<tr><td>1</td><td>FCN1 time stamp TSOUT</td></tr>
</table></td></tr>
</table>

Table 15-12 TSOSLTA0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>0</td><td>TSOSLTA00</td><td>Selection of TAUA0TTIN12:
<table>
<tr><th>TSOSLTA00</th><th>TAUA0TTIN12</th></tr>
<tr><td>0</td><td>Port TAUA0I12</td></tr>
<tr><td>1</td><td>FCN0 time stamp TSOUT</td></tr>
</table></td></tr>
</table>

15.3 Functional Overview

Features summary

The TAUA has the following functions:

- 16 channels
- 16-bit counter and 16-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Real-time output
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUA:

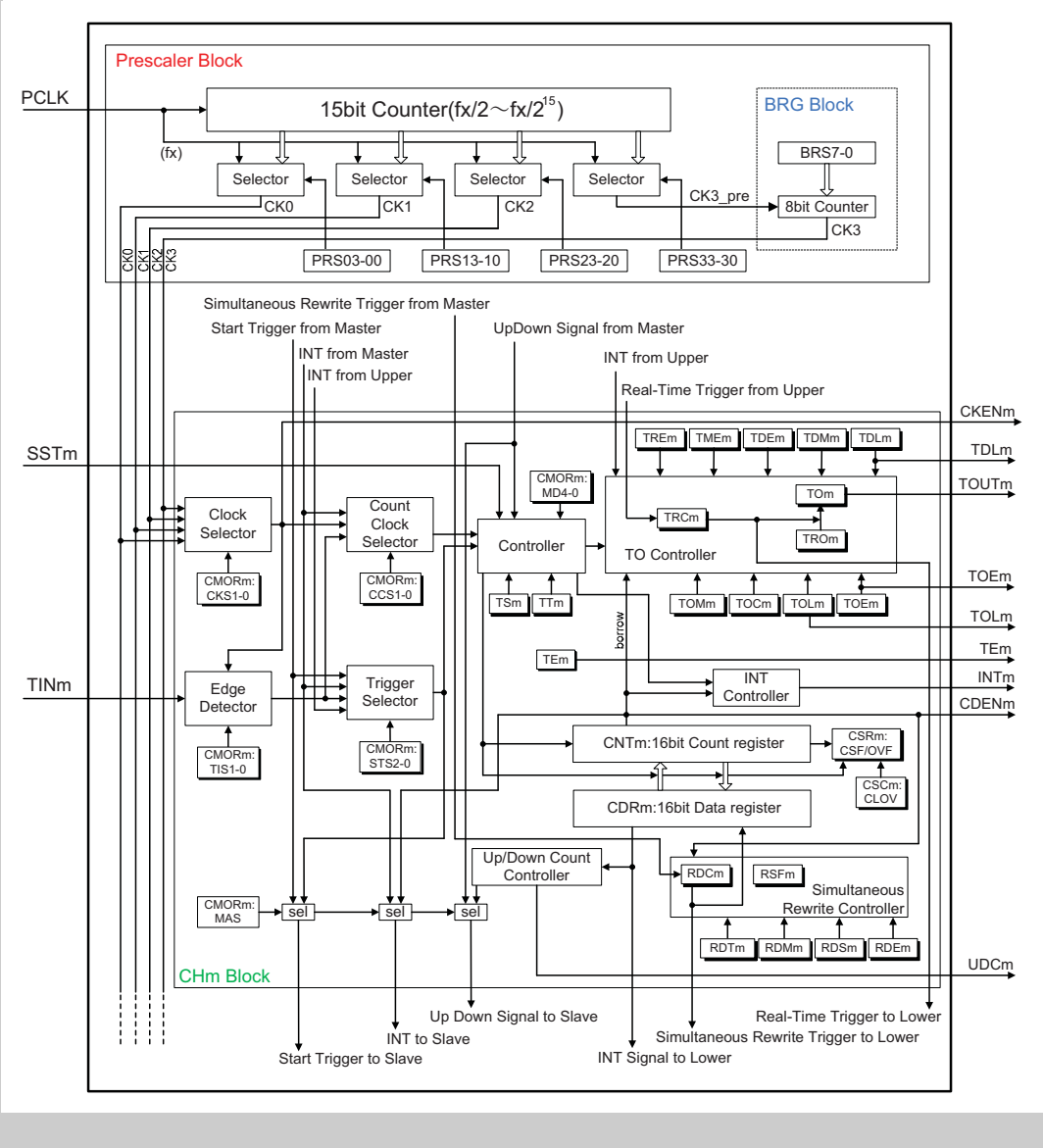


Figure 15-3 Block diagram of the TAUA

The prefix “TAUAn” has been omitted from the register names for the sake of clarity in the above figure.

15.3.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.

  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.

  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.

  Examples are “Capture Mode”, “Event Count Mode”, and “Interval Timer Mode”.

- **Channel output mode**

  The channel output mode defines the operation of TAUAnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Examples are “Independent Channel Output Mode 1” and “Synchronous Channel Output Mode 2 with Dead Time Output”.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

  It defines the operation mode, start and capture trigger, count clock, the real-time trigger generation, and the channel output mode.

  Examples are “Divider Function”, “One-Phase PWM Output Function”, and “Triangle PWM Output Function”

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 5, channel 3 is an upper channel and channel 9 is a lower channel.

15.4 Functional Description

The Timer Array Unit A is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler block for count clock generation and 16 channels, each equipped with a 16 bit counter TAUAnCNTm and a 16-bit data register TAUAnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in different operation modes, either independently or in combination with other channels (synchronously), i.e. multiple channels depend on each other with one master and one or more slave channels.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler block

The prescaler block provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

Count clocks CK0 to CK2 are derived from PCLK by a configurable prescaler division factor of 2^0 to 2^{15}. The fourth count clock CK3 can be adjusted more precisely by an additional division factor that is not a power of 2.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUAnIm from master channel
- Edge detected TAUAnTTINm input signal

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUAnCMORm.MD[4:0])
- Counter start enable (TAUAnTS.TSm) and counter stop (TAUAnTT.TTm)

  When counter start is enabled, status flag TAUAnTE.TEm is set.
- Count direction (can be controlled by master channel)

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUAnTE.TEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUAnTSSTm
- TAUAnTTINm input valid edge
- INTTAUAnIm from the master or any upper channel
- Up/down output trigger signal TAUAnTUDSm of the master channel
- Dead-time output signal TAUAnTDL.TDLm of the TAUAnTTOUTm generation unit.

Simultaneous rewrite controller

Simultaneous rewrite control is a special function that can be used in synchronous operation modes. The data registers of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller

ensures that new data register values of all channels become effective at the same time.

TAUAnTO Controller The output control of every channel enables the generation of various output signal forms such as PWM signals or triangular waves.

Signals The TAUA has various input and output signals. A full list can be found in the first section of this chapter under the keyword “I/O signals”.

15.5 General Operating Procedure

The following lists the general operation procedure for the TAUAn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. All circuits and registers of all channels are initialized. The control register of TAUAnTTOUTm is also initialized and outputs a low level.

1. Set the TAUAnTPS and TAUAnBRS registers to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUAn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUAnTS.TSm bit to 1.
   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
   The function is in operation.
4. If desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUAnTT.TTm bit to 0.

Note A detailed description of the required control bits and the operation of the individual functions is given in *15.15 “Independent Channel Operation Functions” on page 751* and *15.22 “Synchronous Channel Operation Functions” on page 875* .

15.6 Operation Modes

The TAUA contains 14 operation modes. These determine the basic behavior of a channel, for example whether a timer counts up or down, whether the data register TAUAnCDRm acts as a compare register or stores the initial value of the counter, etc.

One operation mode can be set for each channel. It is specified using the TAUAnCMOR.MD[4:0] bits.

When choosing a function, these settings cannot be specified individually, but are grouped under the term operation mode. If a function uses multiple channels, the operation mode of each channel must be set correctly for the function to work correctly.

For more information about the operation modes required by each function, refer to the required function in *15.15 “Independent Channel Operation Functions” on page 751* and *15.22 “Synchronous Channel Operation Functions” on page 875* .

15.7 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *15.7.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *15.7.2 "Simultaneous start and stop of synchronous channel counters" on page 718*
- *15.8 "Simultaneous Rewrite" on page 719*

15.7.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2, CH4, …) can be set as master channels. Any channel apart from CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 and the lower channels (CH4, CH5, …) can be set as slave channels.
- If multiple master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH4 are master channels, CH1 to CH3 can be set as slave channels for CH0, but CH5 to CH7 cannot.

Operation clock

- The same operation clock must be set for the slave channel and the master channel. This is achieved using the TAUAnCMORm.CKS[1:0] bits of the slave and master channel.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

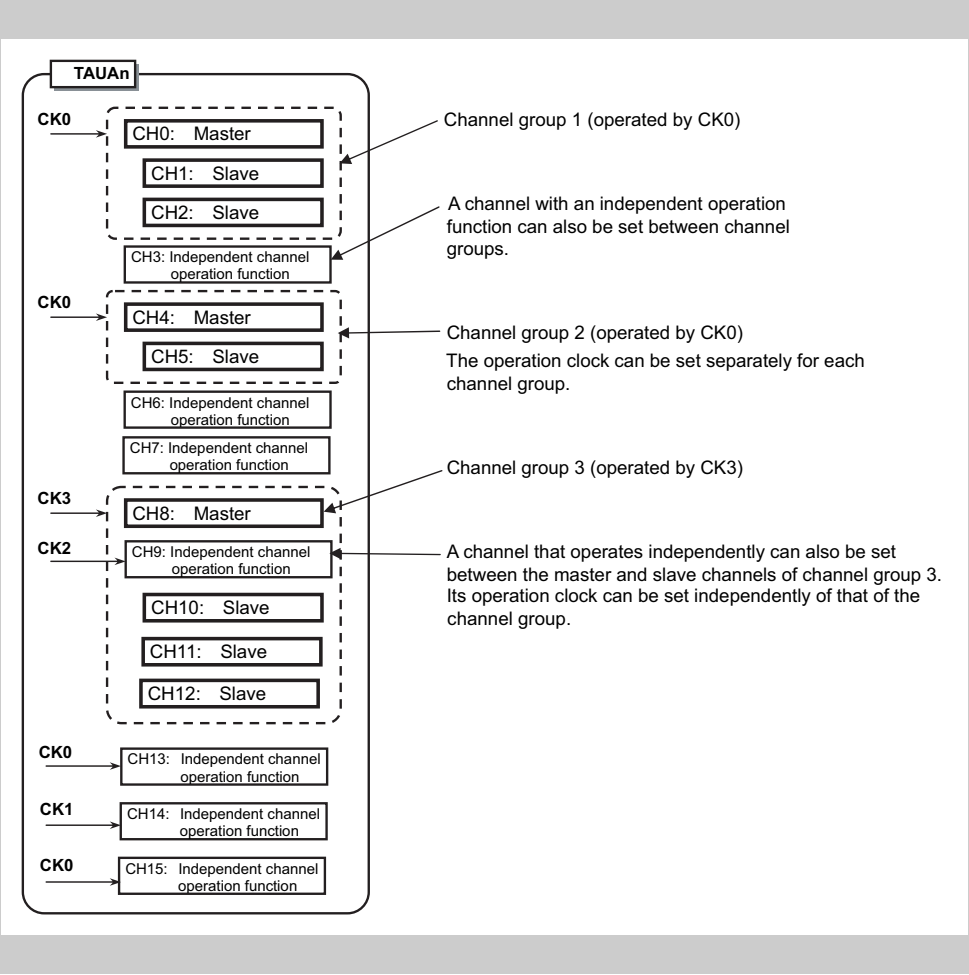


Figure 15-4 Grouping of the channels and assignment of operation clocks

INTTAUAnIm, start trigger, and count clock

- Master channels can transfer an interrupt request (INTTAUAnI), the start trigger, and the count clock to slave channels.
- Slave channels can use INTTAUAnI, the start trigger, and the count clock of the master channels but cannot transfer their INTTAUAnI, start trigger, or count clock to the lower channels.
- A master channel cannot use INTTAUAnI, the start trigger, or the count clock of the higher master channels.

15.7.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a TAUA unit, and between TAUA units.

(1) Simultaneous start and stop within a TAUA unit

- To simultaneously start synchronized channels, the TAUAnTS.TSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUAnTT.TTm bits of the channels must be set at the same time.

Writing to the TAUAnTS.TSm bits sets the corresponding TAUAnTE.TEm bits to 1, enabling counting. TAUAnTS.TSm = 1 only enables the corresponding counter to start; the exact time that it starts depends on the operation mode.

(2) Simultaneous start between TAUA units

Counters in different TAUA units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal. The simultaneous start trigger register is then sent to the TAUAnTSSTm input.

15.8 Simultaneous Rewrite

15.8.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUAnCDRm and TAUAnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by:

- The counter on the master channel or upper channel (depending on the selected operation mode) reaching a certain value
- INTTAUAnI being issued on the upper channel specified by TAUAnRDC.RDCm

There are four methods for simultaneous rewrite. These are listed in the following table, along with how to specify them and when they cause simultaneous rewrite to be triggered.

Table 15-13 Simultaneous rewrite methods and when they are triggered

Method	Simultaneous rewrite triggered when	TAUAn RDE. RDEm	TAUAn RDS. RDSm	TAUAn RDM. RDMm
-	No simultaneous rewrite	0	0	0
A	The master channel (re)starts counting	1	0	0
B	The slave channel starts counting down at the upper peak of a triangular cycle	1	0	1
C1	INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm	1	1	0
C2	INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm that in turn is triggered by an external signal	1	1	1

The following table lists which of these four methods is available for each channel operation function. For more information about the individual channel operation functions, see the corresponding sections in *15.15 “Independent Channel Operation Functions” on page 751* and *15.22 “Synchronous Channel Operation Functions” on page 875* .

Table 15-14 Channel operation functions and methods they use (1/2)

Function	A	B	C1	C2
Simultaneous Rewrite Trigger Output Function Type 1			X	
PWM Output Function	X		X	
One-Shot Pulse Output Function	X			
Trigger Start PWM Output Function	X			X
Offset Trigger Output Function	X			
Delay Pulse Output Function	X			

Table 15-14 Channel operation functions and methods they use (2/2)

Function	A	B	C1	C2
Triangle PWM Output Function		X	X	
Triangle PWM Output Function with Dead Time		X	X	
Interrupt Culling Function	X	X	X	
AD Conversion Trigger Output Function Type 1	X		X	
AD Conversion Trigger Output Function Type 2		X	X	
Synchronous Real-Time Output Function Type 1	X		X	
Synchronous Real-Time Output Function Type 2		X	X	
Synchronous Real-Time Output Function Type 3				
Non-Complementary Modulation Output Function Type 1	X		X	
Non-Complementary Modulation Output Function Type 2		X	X	
Complementary Modulation Output Function		X	X	

15.8.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

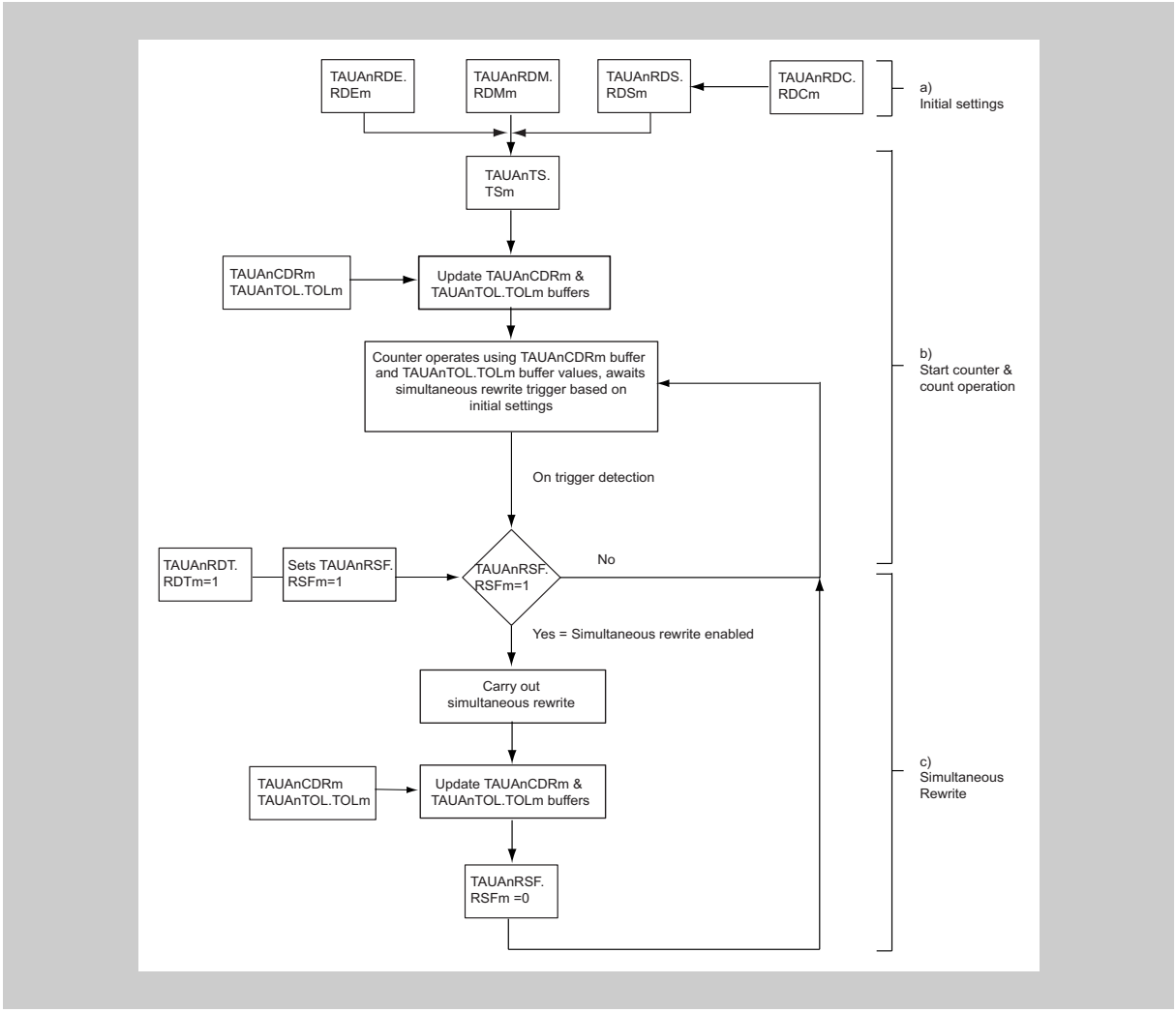


Figure 15-5 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUAnRDE.RDEm = 1
- To select the type of simultaneous rewrite, set TAUAnRDM.RDMm and TAUAnRDS.RDSm according to the values in *Table 15-13 “Simultaneous rewrite methods and when they are triggered” on page 719*
- To select which upper channel is monitored for the simultaneous rewrite trigger use TAUAnRDC.RDCm (prerequisite: TAUAnRDS.RDSm is set to upper channel)

(2) Start counter and count operation

- To start all the TAUAnCNTm counters in the channel group, set the corresponding TAUAnTS.TSm bits to 1. TAUAnTOL.TOLm and the values in the data registers (TAUAnCDRm) are written to the corresponding TAUAnTOL.TOLm buffer (TAUAnTOL.TOLm buf) and data buffer registers (TAUAnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUAnRDT.RDTm) to 1 sets the reload flag (TAUAnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUAnRDT.RDTm then immediately returns to 0, but TAUAnRSF.RSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUAnRSF.RSFm bit is checked to see if simultaneous rewrite is enabled (TAUAnRSF.RSFm = 1). If it is, simultaneous rewrite is carried out. Otherwise the value of the TAUAnRSF.RSFm bit is re-evaluated the next time the trigger is detected.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUAnRSF.RSFm = 1), the current values of the data registers are copied to their buffers. These values are then written to the corresponding counters and the values are applied the next time the counter starts or restarts.
- The TAUAnRSF.RSFm bit is set to 0, and the system awaits the next simultaneous rewrite trigger.

15.8.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUAnRDE.RDEm, TAUAnRDS.RDSm, TAUAnRDM.RDMm, and TAUAnRDC.RDCm cannot be changed while the counter is in operation (TAUAnTE.TEm = 1).
- TAUAnTOL.TOLm can only be rewritten during operation when in PWM output function or triangle PWM output function. For all other output functions, TAUAnTOL.TOLm must be written before the counter starts. If it is rewritten in another function, TAUAnTTOUTm outputs an invalid wave.
- When an upper channel is used as the channel issuing the simultaneous rewrite trigger (TAUAnRDS.RDSm = 1), the TAUAnRDC.RDCm bit controls all the lower channels. This means that if the TAUAnRDC.RDCm bits of CH2 and CH7 are set to 1 and the TAUAnRDC.RDCm bits of other channels are set to 0, CH2 and CH7 serve as simultaneous rewrite trigger generation channels. CH2 controls the lower channels CH3 to CH6, and CH7 controls the lower channels CH8 to CH15.
- If simultaneous rewrite is enabled and an upper channel is selected for the simultaneous rewrite trigger (TAUAnRDE.RDEm and TAUAnRDS.RDSm = 1) but no upper channel is set (TAUAnRDC.RDC[15:0] = 0), simultaneous rewrite cannot take place.

15.8.4 Types of simultaneous rewrite

In the following section the four simultaneous rewrite methods are explained using timing diagrams.

(1) Simultaneous rewrite when the master channel (re)starts counting (method A)

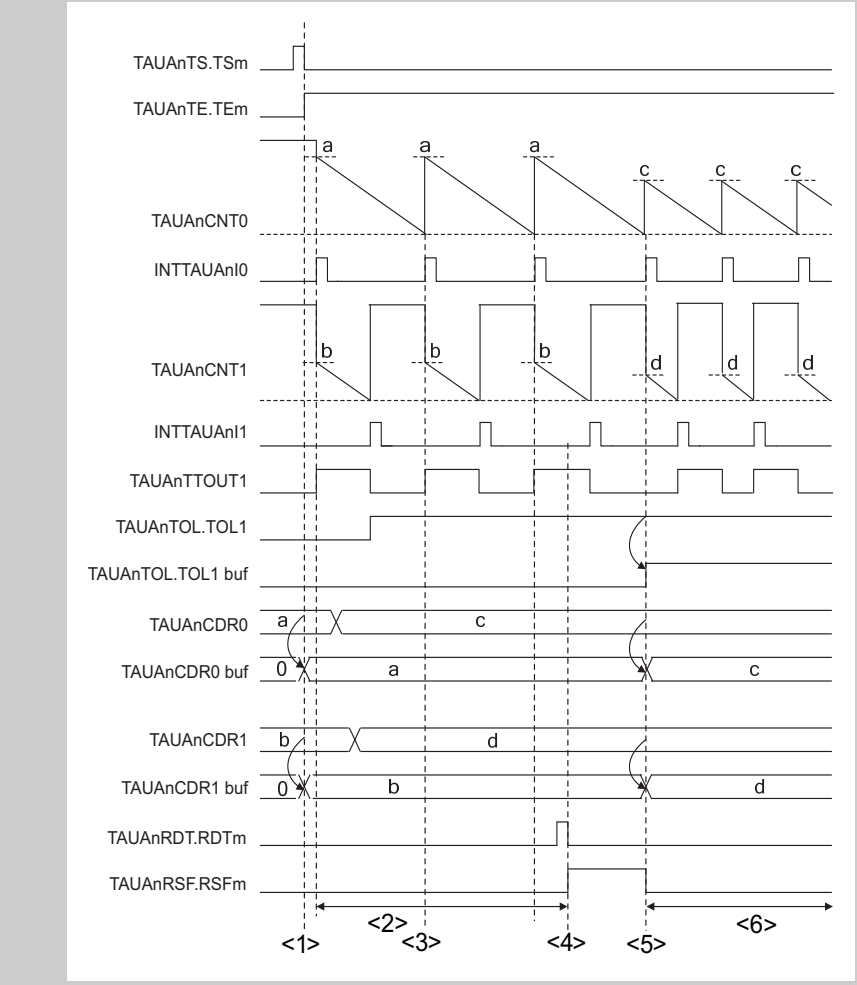


Figure 15-6 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method A is applied.

Description:

1. When the counter starts, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer and the value of TAUAnTOL.TOLm is copied to the TAUAnTOL.TOLm buffer. The TAUAnCDRm buffer value is written to the counter.
2. The TAUAnCDRm and TAUAnTOL.TOLm registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUAnRSF.RSFm = 0).
4. The reload data trigger bit (TAUAnRDT.RDTm) is set to 1 which sets the status flag (TAUAnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewrite is triggered by counter TAUAnCNT0 starting to count down. The TAUAnCDRm value is written to the TAUAnCDRm buffer and the TAUAnTOL.TOLm value is written to the TAUAnTOL.TOLm buffer.
   – The counter starts to count down from the value in the TAUAnCDRm buffer and the TAUAnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUAnTOL.TOLm becomes effective.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUAnCDRm and TAUAnTOL.TOLm can be changed again.

(2) Simultaneous rewrite at the peak of a triangular cycle of the slave channel (method B)

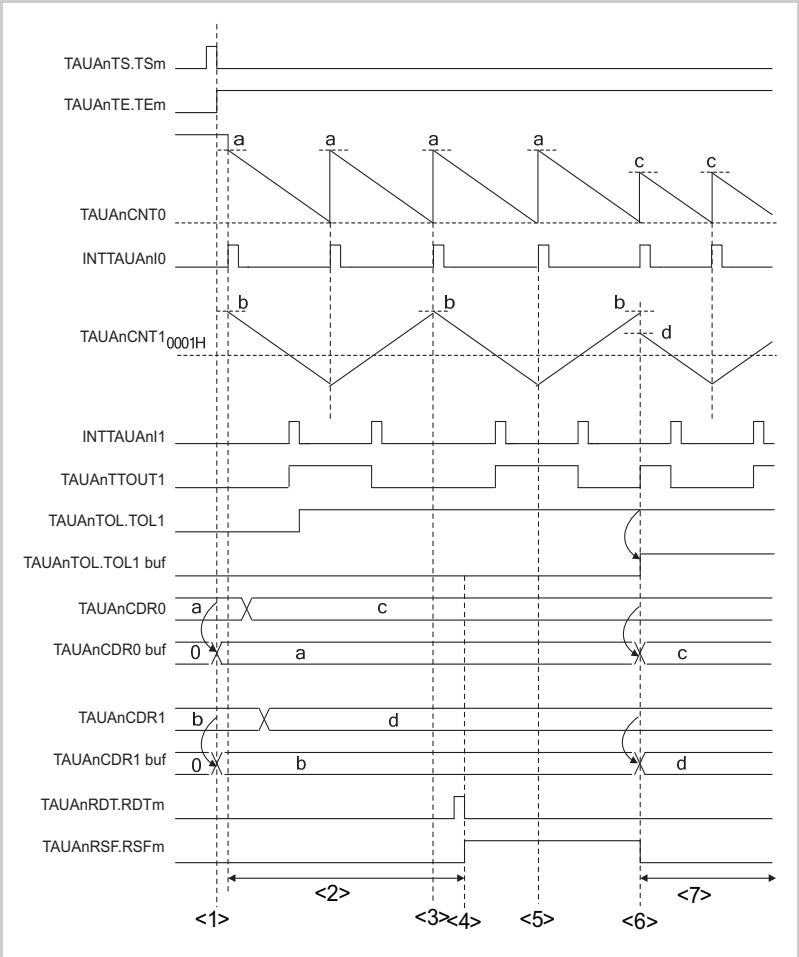


Figure 15-7 Simultaneous rewrite at the peak of a triangular cycle of the slave channel

Setup:

- CH0 is the master channel, counting up and down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method B is applied.

Description:

1. When the counter starts, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer. The buffer value is written to the counter.
2. The TAUAnCDRm and TAUAnTOL registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. Simultaneous rewrite does not occur because it is disabled (TAUAnRSF.RSFm = 0).
4. The reload data trigger bit (TAUAnRDT.RDTm) is set to 1 which sets the status flag (TAUAnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by the slave channel starting to count down at an upper peak.
6. Simultaneous rewrite is triggered; the TAUAnCDRm value is written to the TAUAnCDRm buffer, the TAUAnTOL.TOLm value is written to the TAUAnTOL.TOLm buffer.
   – The counters start to count down from the value in the TAUAnCDRm buffer and the TAUAnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUAnTOL.TOLm becomes effective.
7. The counters count down and await the next simultaneous rewrite trigger. The values of TAUAnCDRm and TAUAnTOL.TOLm can be changed again.

(3) Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm (method C1)

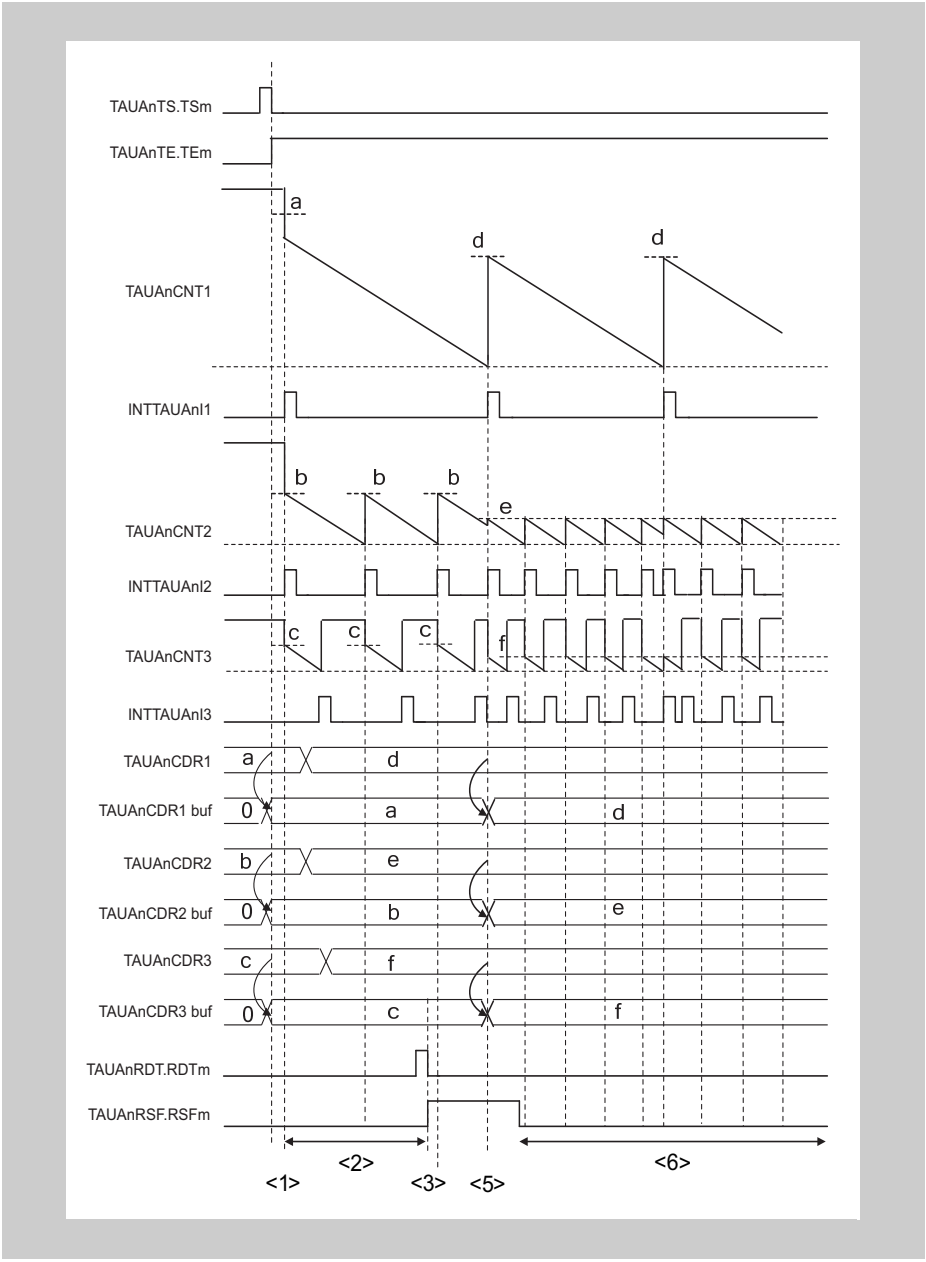


Figure 15-8 Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm

Setup:

- CH1 is an upper channel, counting down. CH2 is a master channel, CH3 is the slave channel, and simultaneous rewrite method C1 is applied. The TAUAnRDC register specifies which upper channel is monitored for an INTTAUAnI trigger.

Description:

1. When the counter starts, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer. The buffer value is written to the counter.
2. The TAUAnCDRm register can be written at any time, but the value does not affect the counter as the counter reads the buffer value.
3. The reload data trigger bit (TAUAnRDT.RDTm) is set to 1 which sets the status flag (TAUAnRSF.RSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewrite is triggered by INT1 which is caused by counter 1 reaching 0000_H. The TAUAnCDRm values are written to the corresponding TAUAnCDRm buffers, the counters start to count down from the values in the TAUAnCDRm buffers, and the TAUAnRSF.RSFm bit is reset to 0.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUAnCDRm registers can be changed again.

(4) Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm that in turn is triggered by an external signal (method C2)

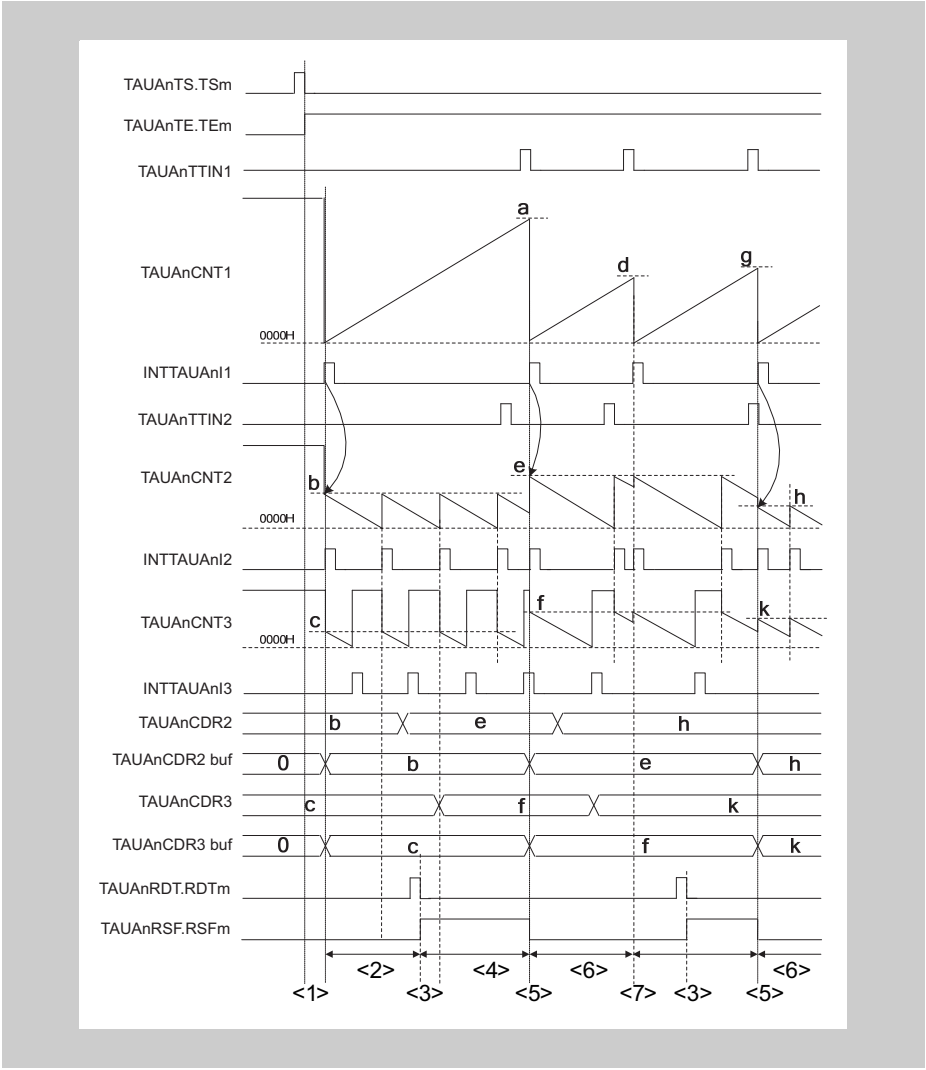


Figure 15-9 Simultaneous rewrite when INTTAUAnIm is generated on an upper channel specified by TAUAnRDC.RDCm that in turn is triggered by an external signal

Setup:

- CH1 is an upper channel, counting up. CH2 is a master channel, CH3 is the slave channel, and synchronous channel operation method C2 is applied. The TAUAnRDC register specifies which upper channel is monitored for an INTTAUAnIm trigger.

Description:

1. When the counter starts, the value of TAUAnCDRm is copied to the TAUAnCDRm buffer. The buffer value is written to the counter.
2. The TAUAnCDRm register can be written at any time, but the value does not affect the counter as the counters read the buffer values.
3. The reload data trigger bit (TAUAnRDT.RDTm) is set to 1 which sets the status flag (TAUAnRSF.RSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewrite is triggered by INT1 which is caused by external signal TIN1. The TAUAnCDRm values are written to the corresponding TAUAnCDRm buffers. The counters then start to count from the values in the corresponding TAUAnCDRm buffers. The TAUAnRSF.RSFm bit is reset to 0.
6. The counters count down and await the next simultaneous rewrite trigger. The values of the TAUAnCDRm registers can be changed again.
7. An external signal occurs at TIN2 but simultaneous rewrite does not take place because it is disabled (TAUAnRSF.RSFm = 0).

15.9 Channel Output Modes

The output of the TAUAnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (Direct Channel Output Mode, TAUAnTOE.TOEm = 0)

  When controlled by software, the output register bit (TAUAnTO.TOm) can be written and the value of the bit is transferred to the output pin (TAUAnTTOUTm).

- By TAUA signals (TAUAnTOE.TOEm = 1)

  When operated by TAUA signals, the output level of TAUAnTTOUTm is set or reset or toggled by internal signals. The value of TAUAnTO.TOm is updated accordingly to reflect the value of TAUAnTTOUTm.

  – Independently (Independent Channel Output Mode, TAUAnTOM.TOMm = 0)

    When operated independently, the output of the TAUAnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUAnTOM.TOMm = 0).

  – Synchronously (Synchronous Channel Output Mode, TAUAnTOM.TOMm = 1)

    When operated synchronously, the output of the TAUAnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUAnTOM.TOMm = 1).

The TAUAnTO.TOm bit can always be read to determine the current value of TAUAnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 15-15 “Channel output modes” on page 732* .

The channel output modes are described in detail in

- *15.9.2 “Channel output modes controlled independently by TAUAn signals” on page 734*
- *15.9.3 “Channel output modes controlled synchronously by TAUAn signals” on page 736* .

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUAnTOL.TOLm.

The value of the TAUAnTOL.TOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function or triangle PWM output function. Otherwise, changes to TAUAnTOL.TOLm result in an invalid TAUAnTTOUTm signal.

Refer to *15.8 “Simultaneous Rewrite” on page 719* .

The various channel output modes and the channel output control bits are listed in the following table.

Multiple outputs

For a function on channel m that uses its output and the outputs of other channels (q):

- If channel m requires a certain operation mode for channel q, set the operation mode on channel q.
- If channel m does not require a certain operation mode for channel q, the counter on channel q must be disabled (TAUAnTE.TEq = 0), i.e. no operation mode can be used on channel q, even one that does not generate an output.

Table 15-15 Channel output modes

Channel output mode	TAUAn TOE. TOEm	TAUAn TOM. TOMm	TAUAn TOC. TOCm	TAUAn TDE. TDEm	TAUAn TRE. TREm	TAUAn TME. TMEm	TAUAn TDM. TDMm
By software							
Direct Channel Output Mode	0	x					
By timer signals, independently (Independent Channel Output Mode)							
Independent Channel Output Mode 1	1	0	0	0	0	0	0
with Real-Time Output					1		
Independent Channel Output Mode 2			1		0		
By timer signals, synchronously (Synchronous Channel Output Mode)							
Synchronous Channel Output Mode 1	1	1	0	0	0	0	0
with Non-Complementary Modulation Output					1	1	
Synchronous Channel Output Mode 2			1	0	0	0	0
with Dead Time Output				1			
with One-Phase PWM Output							1
with Complementary Modulation Output					1	1	0
with Non-Complementary Modulation Output			1	0			

- All combinations not listed in this table are forbidden.
- Bits marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUAnTE.TE = 1):

- TAUAnTOM.TOMm,
- TAUAnTOC.TOCm,
- TAUAnTDE.TDEm,
- TAUAnTRE.TREm, and
- TAUAnTDM.TDMm

The following bits cannot be changed during count operation (TAUAnTE.TEm = 1) except in channel output modes with modulation output:

- TAUAnTME.TMEm,
- TAUAnTDL.TDLm

15.9.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUAnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUAnTOE.TOEm = 0).

1. Set TAUAnTO.TOm to specify the initial level of the TAUAnTTOUTm output.
2. Set the channel output mode using *Table 15-15 “Channel output modes” on page 732* and the output logic using the TAUAnTOL.TOLm bit.
3. Start the counter (TAUAnTS.TSm = 1).

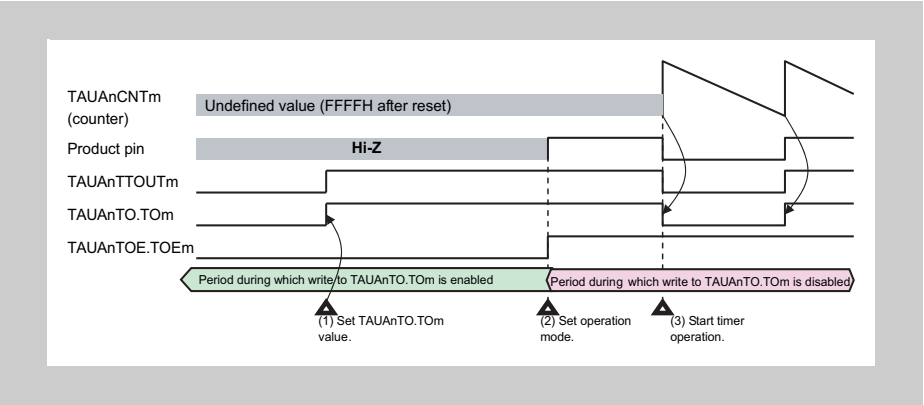
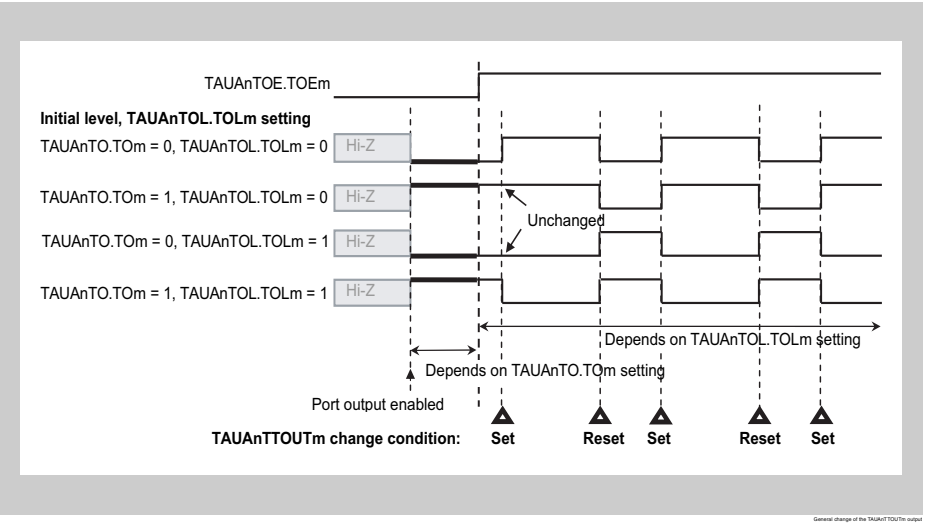


Figure 15-10 General procedure for specifying a TAUAnTTOUTm channel output mode

The following figure shows a general illustration of how the output changes when the counter is enabled:



- TAUAnTO.TOm sets the initial value of TAUAnTTOUTm and can be changed while TAUAnTOE.TOEm = 0.
- TAUAnTOL.TOLm specifies whether the set signal sets TAUAnTO.TOm to high (TAUAnTOL.TOLm = 0) or low (inverted logic, TAUAnTOL.TOLm = 1).

15.9.2 Channel output modes controlled independently by TAUAn signals

This chapter lists the channel output modes that are controlled independently by TAUAn signals. The control bits used to specify a mode are listed in *Table 15-15 “Channel output modes” on page 732* .

(1) Independent Channel Output Mode 1

Set/reset conditions In this output mode, TAUAnTTOUTm toggles when INTTAUAnIm is detected. The value of TAUAnTOL.TOLm is ignored.

Prerequisites None, other than those in *Table 15-15 “Channel output modes” on page 732* .

(2) Independent Channel Output Mode 1 with Real-Time Output

In this output mode, the value of the TAUAnTRO.TROm bit of the trigger channel is output to TAUAnTTOUTm. The trigger channel is specified by setting the corresponding TAUAnTRC.TRCm bit to 1. It controls all lower channels for which TAUAnTRC.TRCm = 0.

Set/reset conditions The value of the TAUAnTRO.TROm bit is only sent to TAUAnTTOUTm when an interrupt INTTAUAnIm occurs on the trigger channel. The interrupt is generated either:

- at certain specified intervals or
- on detection of a valid TAUAnTTINm input edge / counter start

The type of trigger is set using the TAUAnCMORm.MD[4:1] bits.

Prerequisites Both master and slave channels can be set as a trigger generation channel. A channel for which TAUAnTRC.TRCm is set to 1 serves as the trigger generation channel even if TAUAnTRE.TREm is set to 0. Real-time output cannot take place if no channel is set to 1 or if TAUAnTRCm.TRC0 = 0.

This can be seen in the following figure.

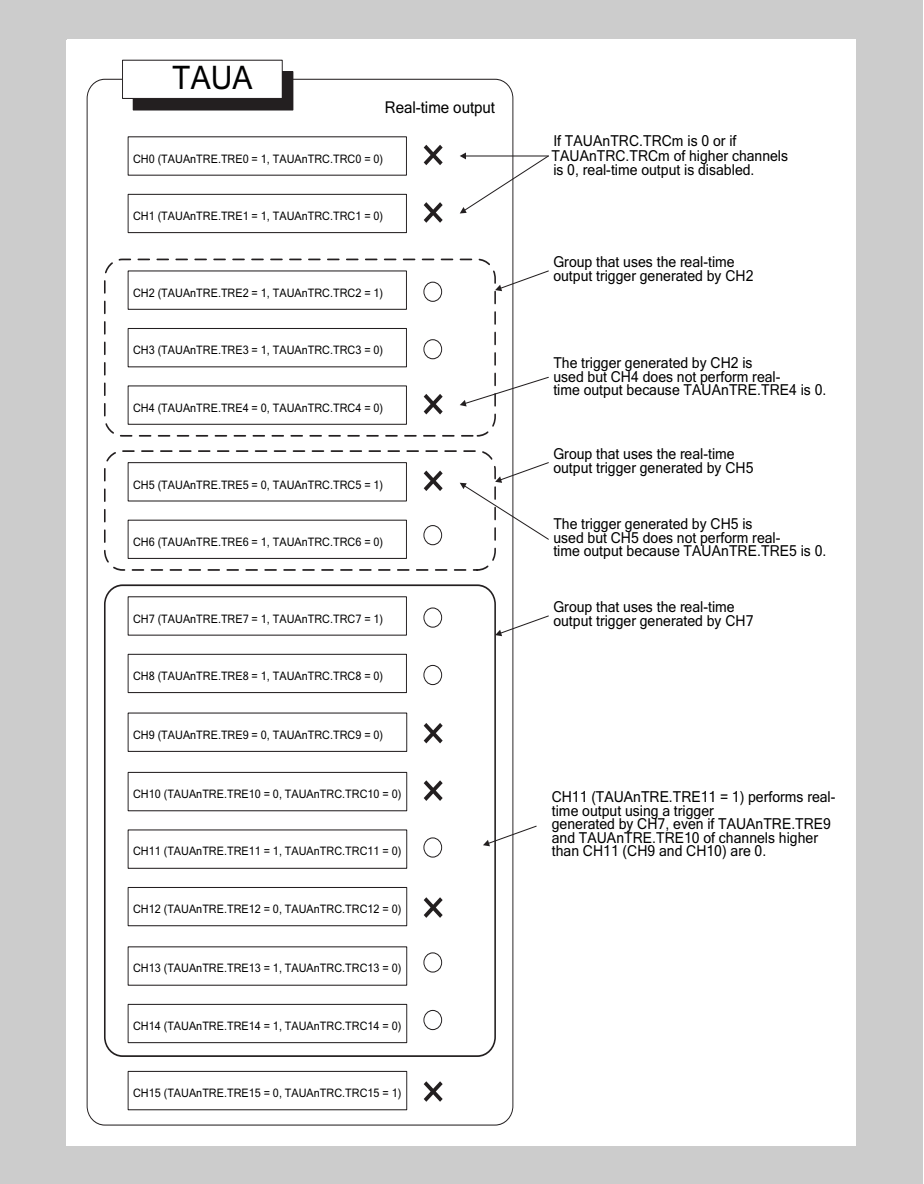


Figure 15-11 Real-time output

(3) Independent Channel Output Mode 2

Set/reset conditions In this output mode, TAUAnTTOUTm is set when INTTAUAnIm occurs upon count start and reset when INTTAUAnIm occurs due to a match between TAUAnCNTm and TAUAnCDRm.

Prerequisites None, other than those in *Table 15-15 “Channel output modes” on page 732* .

15.9.3 Channel output modes controlled synchronously by TAUAn signals

This chapter lists the channel output modes that are controlled synchronously by TAUAn signals. The control bits used to specify a mode are listed in *Table 15-15 “Channel output modes” on page 732* .

(1) Synchronous Channel Output Mode 1

Set/reset conditions

In this output mode, INTTAUAnIm of the master channel serves as the set signal and INTTAUAnIm of the slave channel as the reset signal. If INTTAUAnIm of the master channel and INTTAUAnIm of the slave channel are generated at the same time, INTTAUAnIm of the slave channel (reset signal) has priority over INTTAUAnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites

None, other than those in *Table 15-15 “Channel output modes” on page 732* .

(2) Synchronous Channel Output Mode 1 with Non-Complementary Modulation Output

Set/reset conditions

In this output mode, TAUAnTTOUTm outputs the result of an AND operation between the PWM output of a channel and the real-time output bit (TAUAnTRO.TROm).

The phase period to which the dead time is added is specified using the TAUAnTDL.TDLm bit; for positive phase set TAUAnTDL.TDLm = 0 and for negative phase set TAUAnTDL.TDLm = 1.

Prerequisites

A set of at least three channels is required to generate the PWM output. The master channel and slave 1 generate the period, and slave 2 generates the duty cycle. In typical applications, a further 5 slave channels are also used that operate in the same manner as slave 2.

Only the PWM output and the real-time output bit of the same channel can be combined.

TAUAnTRO.TROm, TAUAnTME.TMEm, and TAUAnTDL.TDLm can only be changed during count operation.

- If TAUAnTME.TMEm is changed, the new value of TAUAnTME.TMEm is applied upon detection of INTTAUAnIm on the specified channel.
- If TAUAnTME.TMEm and TAUAnTDL.TDLm are changed, the new values are applied upon detection of INTTAUAnIm on the master channel.

(3) Synchronous Channel Output Mode 2

In this output mode, the operation mode must be set to Up Down Count mode. The result is a triangle PWM wave at TAUAnTTOUTm. For details refer to *15.25.1 “Triangle PWM Output Function” on page 940* .

Set/reset conditions TAUAnCNTm of the slave channel counts down and up alternatively. When it passes 0001_H it generates an interrupt, causing TAUAnTTOUTm to toggle.

Prerequisites A set of two channels is required to generate the triangle PWM output. TAUAnTTOUTm must be set to 0 before the function starts.

(4) Synchronous Channel Output Mode 2 with Dead Time Output

In this output mode, a dead time delay is added to TAUAnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

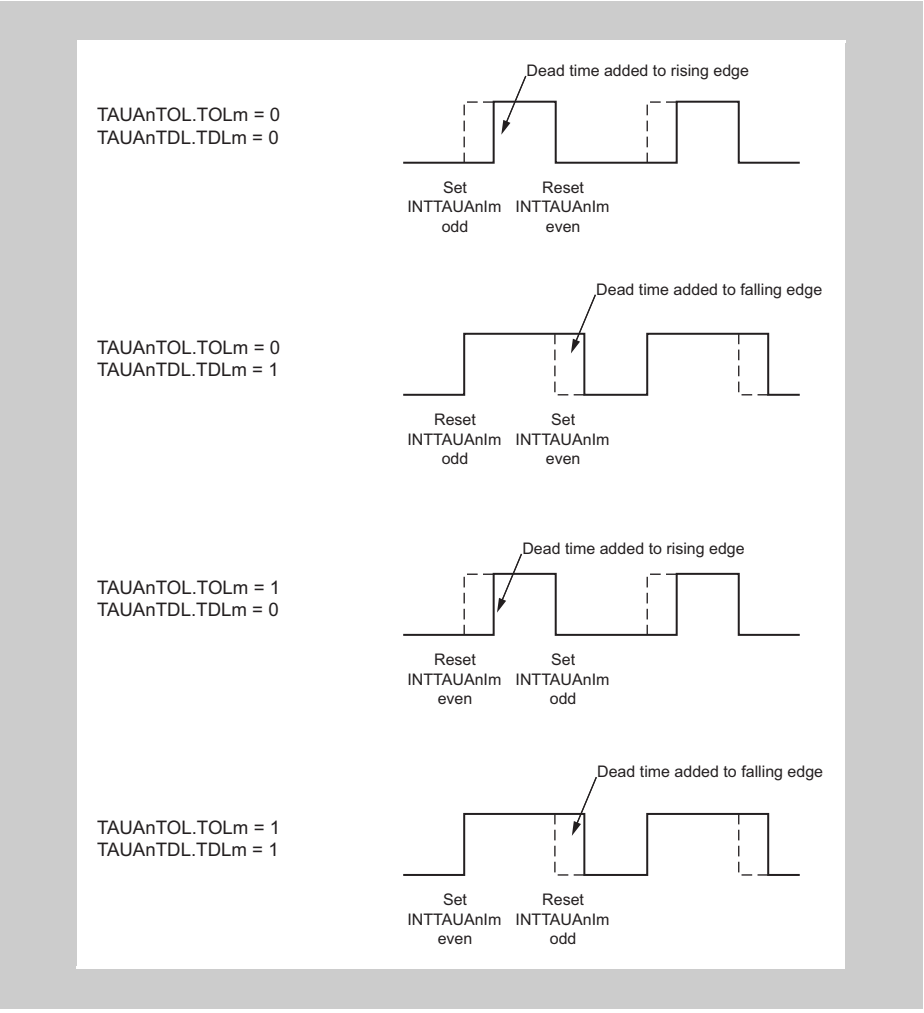


Figure 15-12 Set/reset conditions for Synchronous Channel Output Mode 2 with Dead Time Output

The edge to which the dead time is added is specified using the TAUAnTDL.TDLm bit; for rising edge set TAUAnTDL.TDLm = 0 and for falling edge set TAUAnTDL.TDLm = 1.

Prerequisites Dead time control requires a set of three channels, each operating in the following modes:

- One master channel

  The master channel must be set to Interval Timer Mode

- One even slave channel

  The even slave channel must be set to Up Down Count Mode

- One odd slave channel (even channel + 1)

  The odd slave channel must be set to One Count Mode

The values of the following bits must be the same for the odd channel and the even channel:

- TAUAnTOE.TOEm,
- TAUAnTME.TMEm,
- TAUAnTRE.TREm,
- TAUAnTOM.TOMm,
- TAUAnTOC.TOCm,
- TAUAnTDE.TDEm, and
- TAUAnTDM.TDMm

(5) Synchronous Channel Output Mode 2 with One-Phase PWM Output

In this output mode, a dead time delay is added to TAUAnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

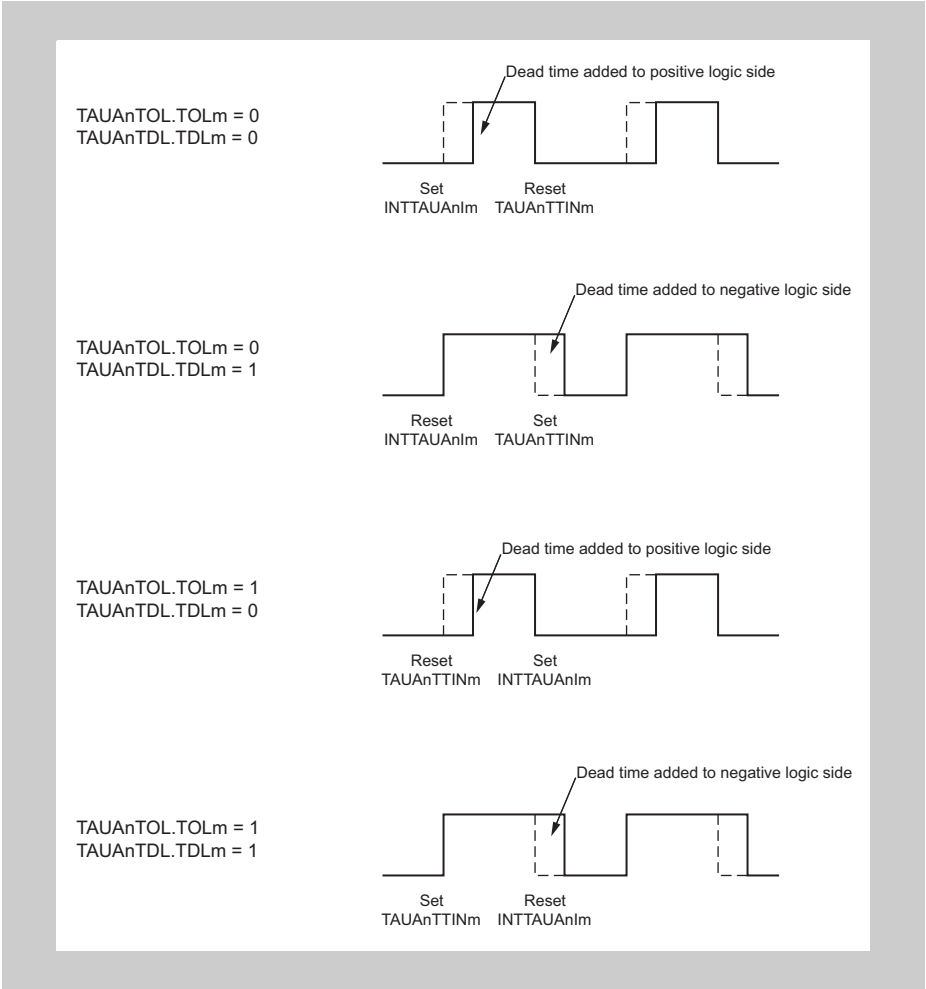


Figure 15-13 Set/reset conditions for Synchronous Channel Output Mode 2 with One-Phase PWM Output

The edge to which the dead time is added is specified using the TAUAnTDL.TDLm bit; for rising edge set TAUAnTDL.TDLm = 0 and for falling edge set TAUAnTDL.TDLm = 1.

Prerequisites

One-phase PWM output control requires a set of two channels:

- One even slave channel
- One odd slave channel (even channel + 1)

  The odd slave channel must be set to “One Count Mode”

The values of the following bits must be the same for the odd channel and the even channel:

- TAUAnTOE.TOEm,
- TAUAnTME.TMEm,

- TAUAnTRE.TREm,
- TAUAnTOM.TOMm,
- TAUAnTOC.TOCm,
- TAUAnTDE.TDEm, and
- TAUAnTDM.TDMm

(6) Synchronous Channel Output Mode 2 with Complementary Modulation Output

Set/reset conditions

In this output mode, TAUAnTTOUTm outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bit (TAUAnTRO.TROm), the modulation output bit (TAUAnTME.TMEm), and the output level bit (TAUAnTOL.TOLm) of a pair of slave channels.

For details refer to *15.27.3 “Complementary Modulation Output Function” on page 1038* .

Prerequisites

A set of at least four channels is required for this mode. The master channel and slave 1 generate the period, slave channel 2 generates the duty cycle, and slave 3 generates the dead time. Slave 2 and 3 are a pair. In typical applications, a further 4 channels are also used that operate in the same manner as slaves 2 and 3 respectively.

TAUAnTRO.TROm, TAUAnTME.TMEm, and TAUAnTDL.TDLm can only be changed during count operation.

- If TAUAnTME.TMEm is changed during operation, the new TAUAnTME.TMEm value is applied upon detection of INTTAUAnIm at the specified channel.
- If TAUAnTME.TMEm and TAUAnTDL.TDLm are changed, the new values are applied upon detection of INTTAUAnIm on an even slave channel.

(7) Synchronous Channel Output Mode 2 with Non-Complementary Modulation Output

The difference to "Synchronous Channel Output Mode 1 with Non-Complementary Modulation Output" is the PWM wave shape.

It is a rectangular wave with mode 1 while it is a triangular wave with mode 2 (see *15.26.1 “Synchronous Real-Time Output Function Type 1” on page 977*).

15.10 Start Timing of Operating Modes

This chapter describes when the counters of the different operating modes start after the TAUAnTS.TSm bit is set to 1.

In all modes, the value of the data register and whether or not an interrupt is issued depends on the individual mode and corresponding register settings.

15.10.1 Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode

The counter starts at the start of the next count clock cycle after TAUAnTS.TSm is set to 1. The value of data register is also loaded at the point the counter starts.

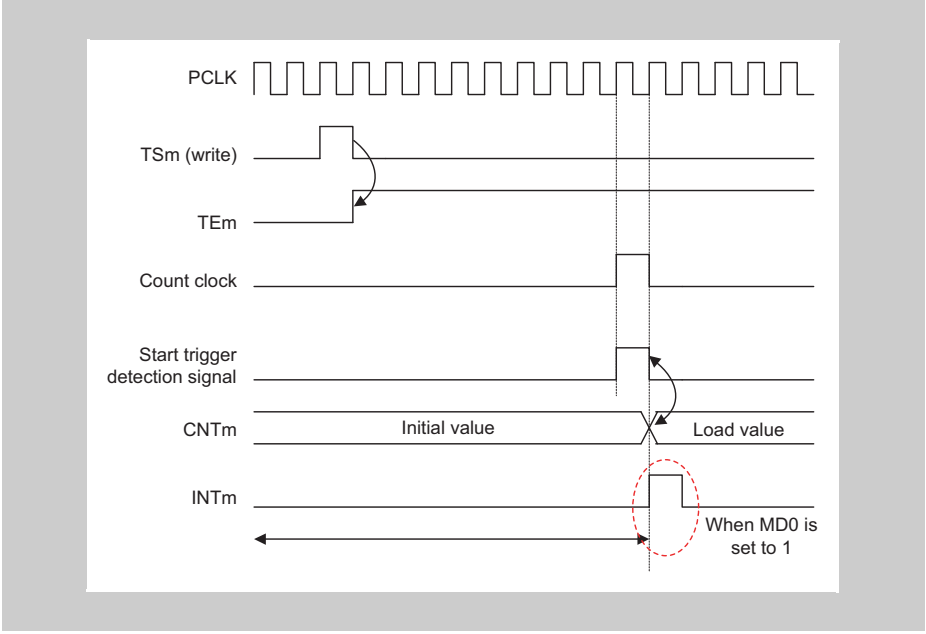


Figure 15-14 Start timing of Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode

15.10.2 Event Mode

The value of the data register is loaded as soon as TAUAnTS.TSm is set to 1. The counter also starts immediately. The value of the data register increments at the start of each subsequent count clock cycle.

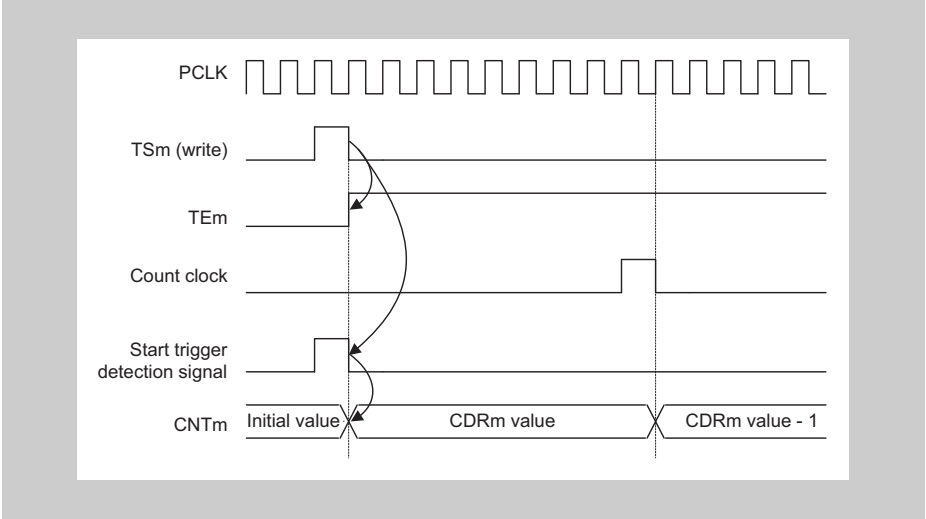


Figure 15-15 Start timing of Event Mode

15.10.3 All other operating modes

In all other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter is only triggered by detection of a valid TAUAnTTINm edge. The value of data register is also loaded at the point the counter starts. Nevertheless, the count clock cycles determine the frequency with which all operations take place.

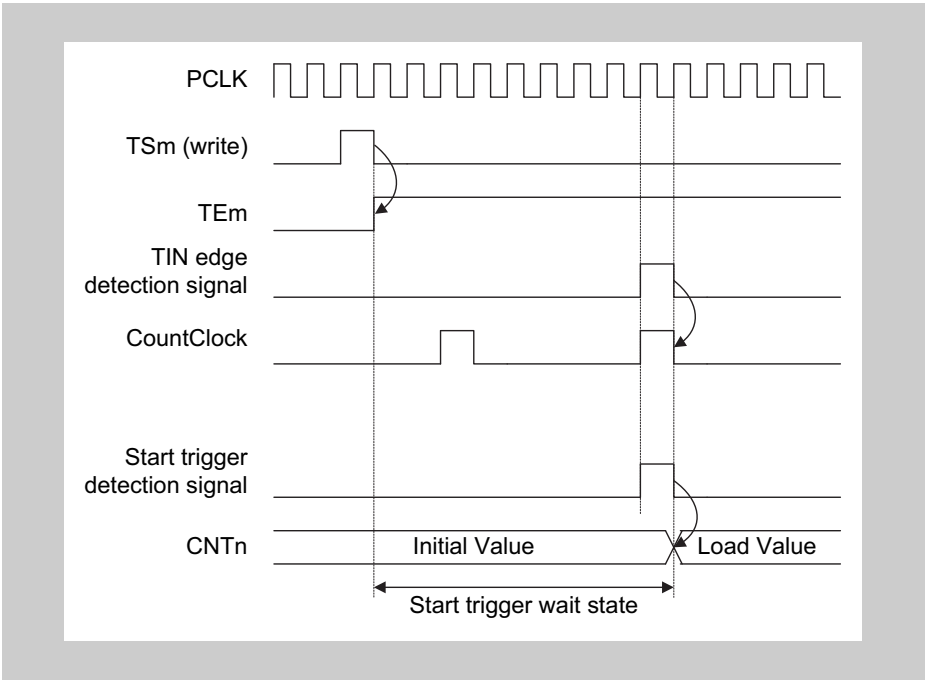


Figure 15-16 Start timing of all other operating modes

15.11 TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts

When the counter starts, it is possible to specify whether an INTTAUAnIm is generated using the TAUAnCMOR.MD0 bit. The effect of the bit depends on the selected mode, as shown in the following table. The effects of INTTAUAnIm on TAUAnTTOUTm depend on the selected channel operation function.

Table 15-16 Effect of CMOR.MD0 bit on generation of INTTAUAnIm when counter is triggered

Mode	TAUAnCMOR. MD0 bit	INTTAUAnIm generated when counter starts
Interval Timer Mode Capture Mode Count Capture Mode	0	No
	1	Yes
Capture & One Count Mode Capture & Gate Count Mode Event Count Mode Up Down Count Mode	0	No
One Count Mode Gate Count Mode	0/1	No, regardless of setting of TAUAnCMOR.MD0 bit.
Pulse One Count Mode		Yes, regardless of setting of TAUAnCMOR.MD0 bit.

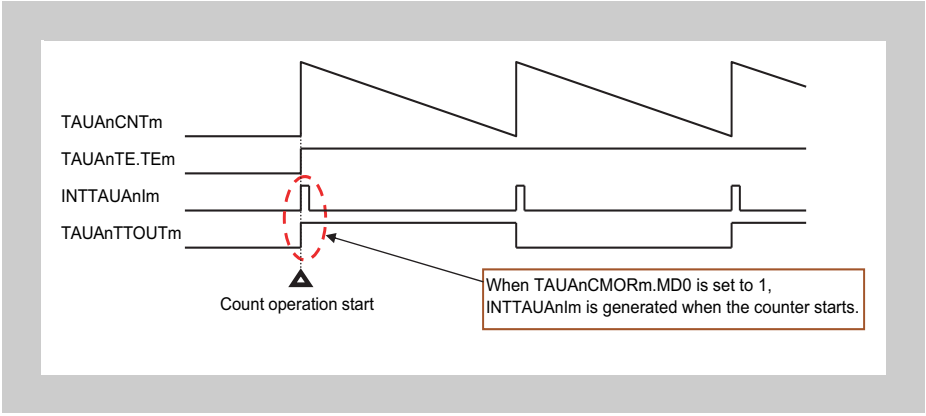


Figure 15-17 INTTAUAnIm generated when counter starts

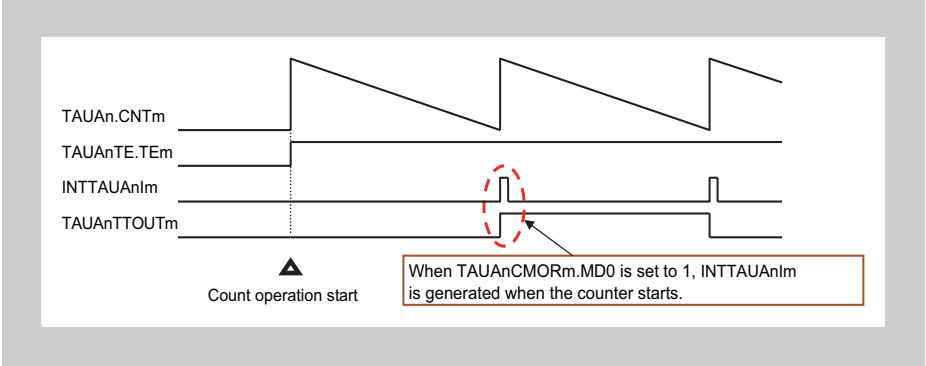


Figure 15-18 INTTAUAnIm not generated when counter starts

15.12 Interrupt Generation upon Overflow

Certain independent functions that count up, overflow without generating an interrupt when they reach $FFFF_H$. This section describes how it is possible to generate an interrupt, by combining a channel operating in one of these modes with a channel in a different operation mode which counts down.

The appropriate operation mode for the second channel depends on the operation mode of the first channel. Nevertheless, the principle is the same for all combinations:

- Find a operation mode for the second channel that counts down in such a manner, that it reaches 0000_H at the same time as the first channel overflows (TAUAnCNTm = $FFFF_H$).
- Set TAUAnCDRm of the second channel to $FFFF_H$
- The two channels must count at the same speed (i.e. they must have the same count clock)
- Both channels are triggered by the same TAUAnTTINm input
- The trigger detection settings (TAUAnCMORm.STS[2:0] and TAUAnCMURm.TIS[1:0]) must be identical for both channels
- This is only possible for TAUA0 and TAUA4 by using configurable "input selectors" (on system level)

Result: the down-counter of the second channel reaches 0000_H at exactly the same time as the up-counter of the first channel overflows (TAUAnCNTm = $FFFF_H$). Thus the second channel generates the desired interrupt.

The following sections list the operating modes that count down that are required to match specific operating modes that count up, as well as example timing diagrams.

15.12.1 Capture Mode

Applies to

- TAUAnTTINm Input Pulse Interval Measurement Function
- Real-Time Output Function Type 2
- Simultaneous Rewrite Trigger Generation Function Type 2

Combine with Interval Timer Mode

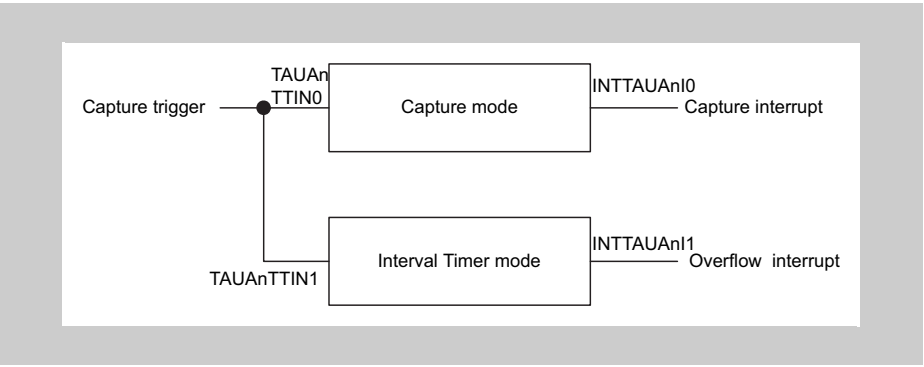


Figure 15-19 Combination of Capture Mode and Interval Timer Mode

Timing diagram

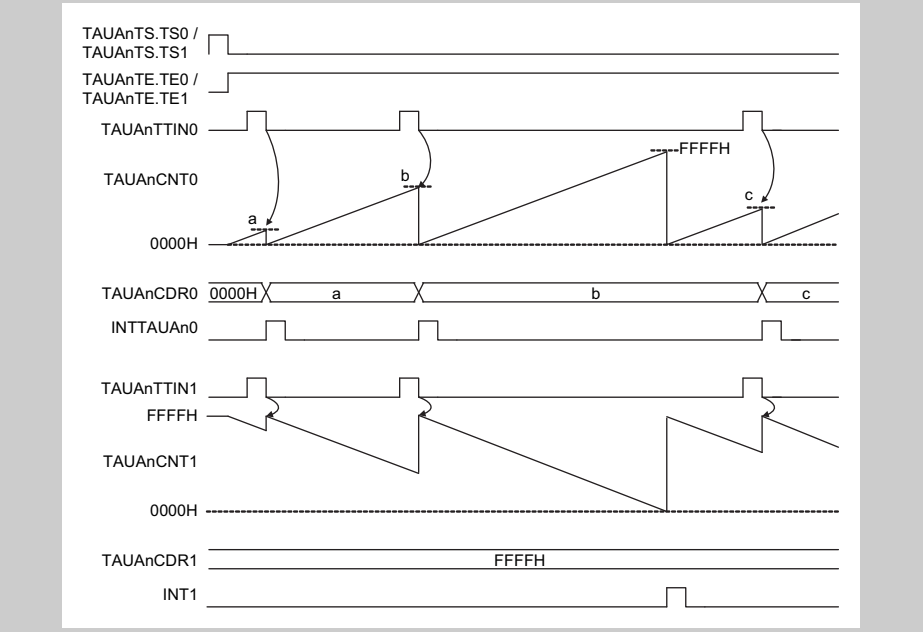


Figure 15-20 Interrupt generation via combination of Capture Mode and Interval Timer Mode

15.12.2 Capture and One Count Mode

Applies to • TAUAnTTINm Input Signal Width Measurement Function

Combine with One Count Mode

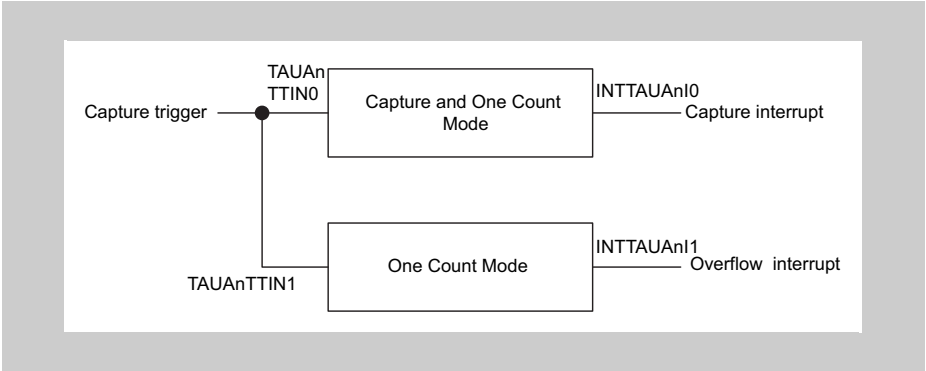


Figure 15-21 Combination of Capture and One Count Mode and One Count Mode

Timing diagram

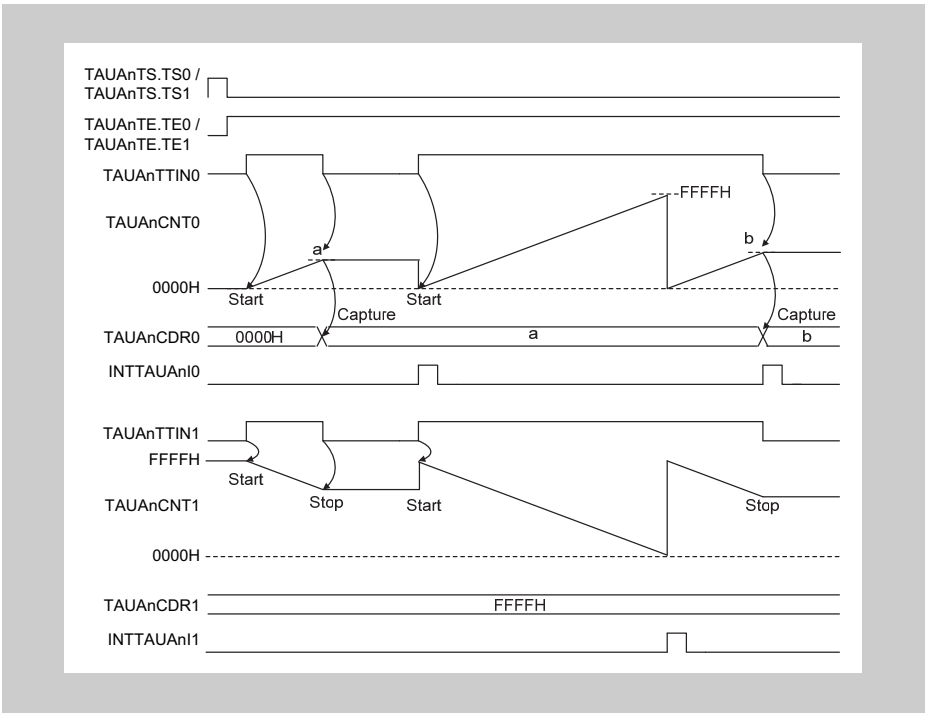


Figure 15-22 Interrupt generation via combination of Capture and One Count Mode and One Count Mode

15.12.3 Count Capture Mode

Applies to
- TAUAnTTINm Input Position Detection Function

Combine with
Interval Timer Mode

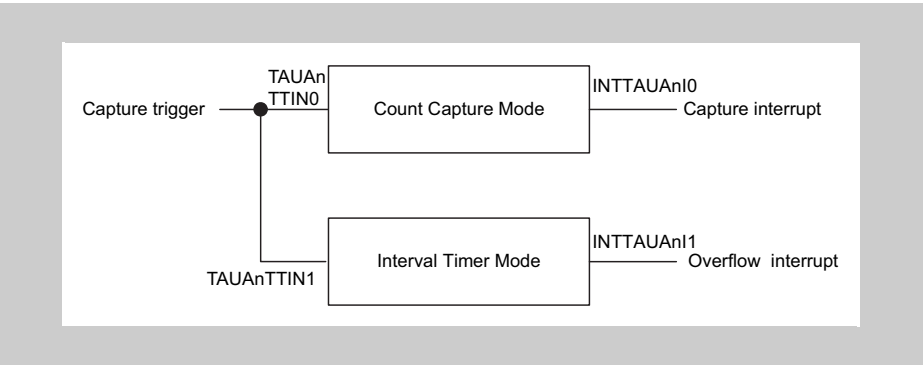


Figure 15-23 Combination of Count Capture Mode and Interval Timer Mode

Timing diagram

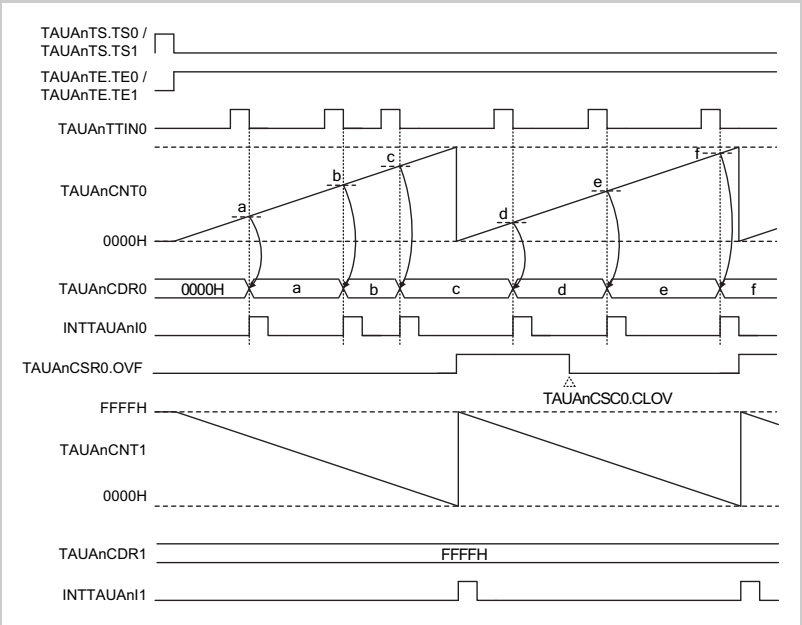


Figure 15-24 Interrupt generation via combination of Count Capture Mode and Interval Timer Mode

In the above timing diagram, TAUAnCSRm.OVF is set to 1 when TAUAnCNTm overflows. It is reset by a software command (TAUAnCSCm.CLOV = 1).

15.12.4 Capture and Gate Count Mode

Applies to • TAUAnTTINm Input Period Count Detection Function

Combine with Gate Count Mode

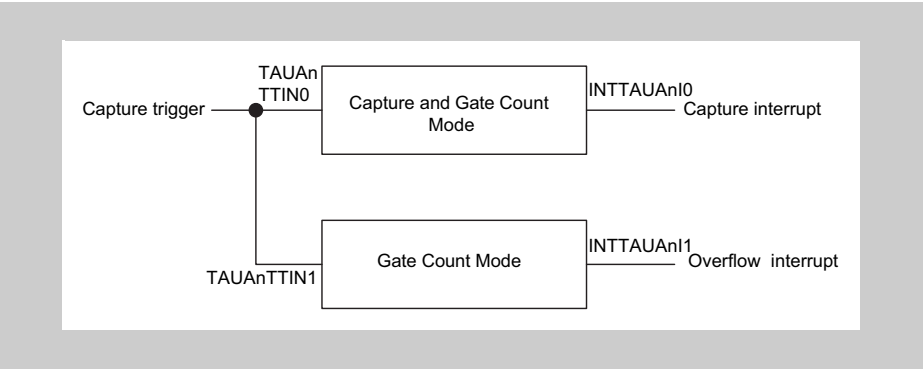


Figure 15-25 Combination of Capture and Gate Count Mode and Gate Count Mode

Timing diagram

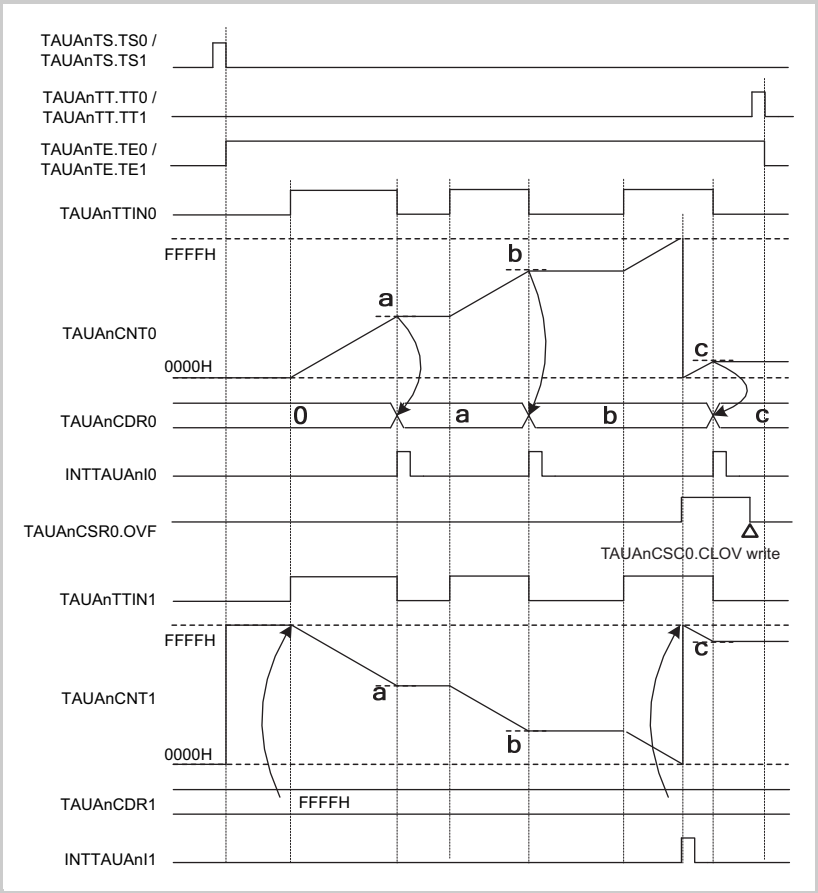


Figure 15-26 Interrupt generation via combination of Capture and Gate Count Mode and Gate Count Mode

In the above timing diagram, TAUAnCSRm.OVF is set to 1 when TAUAnCNTm overflows. It is reset by a software command (TAUAnCSCm.CLOV = 1).

15.13 TAUAnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows when edge detection takes place.

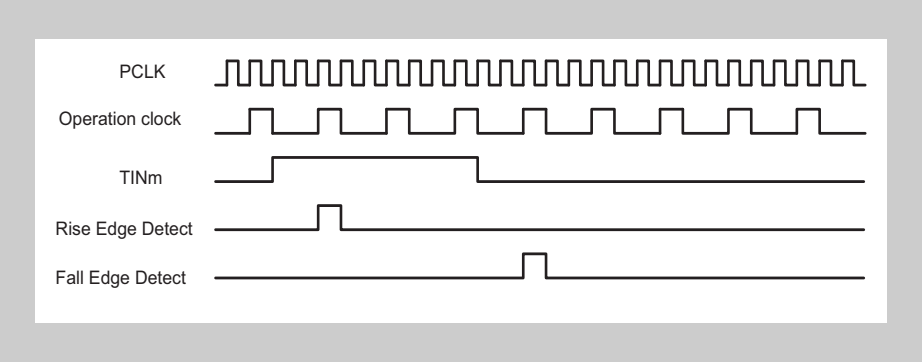


Figure 15-27 Basic edge detection timing

15.14 Assigning DMA Window Addresses

DMA (direct memory access) can be used to store values in the TAUAnDWR registers, for example the current values of the TAUAnCDRm register and the TAUAnTOL register. The 16 TAUAnDWR registers of the TAUAn are controlled by the 8 TAUAnDAS registers, i.e. each TAUAnDAS register controls two TAUAnDWR registers.

The following figure shows how to rewrite the period register, duty registers, and TAUAnTOL register to the TAUAnDWR registers. The addresses of the selected registers (TAUAnCDR0, TAUAnCDR2, TAUAnCDR4, TAUAnCDR6, TAUAnCDR9, TAUAnCDR13, TAUAnTOL.TOLm, and TAUAnRDT.RDTm) are written to the TAUAnDAS registers. The TAUAnDMA window address function then writes the values of the selected registers to the corresponding TAUAnDWR registers.

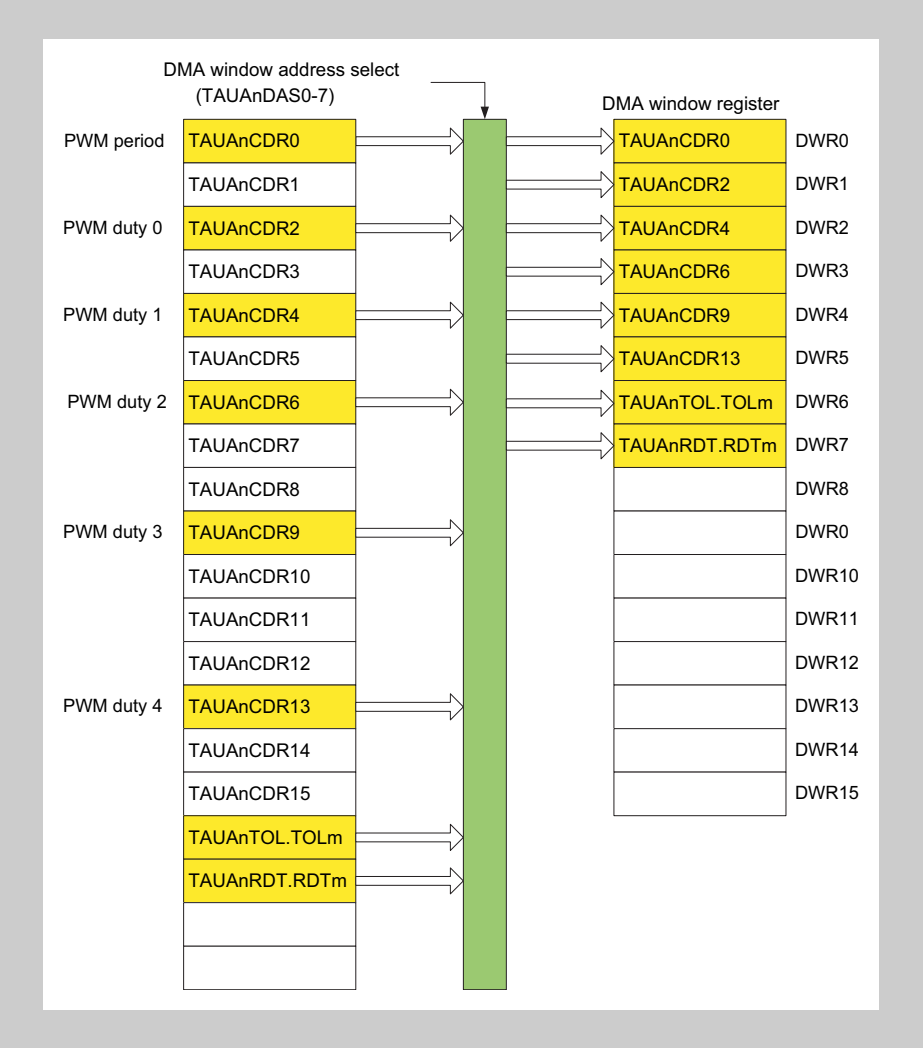


Figure 15-28 Assigning DMA window addresses

Note When using the simultaneous rewrite function for a data register, it is recommended to allocate the RDT register at the end of the specified area.

15.15 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by the Timer Array Unit A. For a general overview of independent channel operation, see *15.4 “Functional Description” on page 712* .

15.16 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals or with a specified delay.

- *15.16.1 “Interval Timer Function”*
- *15.16.2 “TAUAnTTINm Input Interval Timer Function”*
- *15.16.3 “Delay Count Function”*
- *15.16.4 “One-Pulse Output Function”*

15.16.1 Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUAnIm) at regular intervals. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 15-17 "TAUAnCMORm settings for Interval Timer Function" on page 754*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *15.9 "Channel Output Modes" on page 731*

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value.

When the counter reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm stop but retain their values. The counter can be reset by setting TAUAnTS.TSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TSm to 1 during operation.

Conditions If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUAnTTOUTm does not toggle. This results in an inverted TAUAnTTOUTm signal compared to when TAUAnCMORm.MD0 is set to 1. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

(2) Equations

INTTAUAnIm cycle = count clock cycle x (TAUAnCDRm + 1)

TAUAnTTOUTm square wave cycle = count clock cycle x (TAUAnCDRm + 1) x 2

(3) Block diagram and general timing diagram

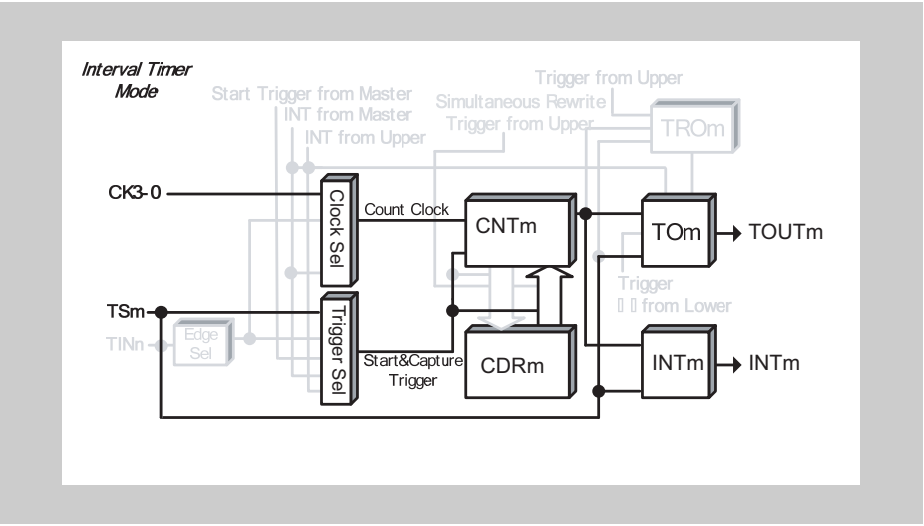


Figure 15-29 Block diagram for Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)

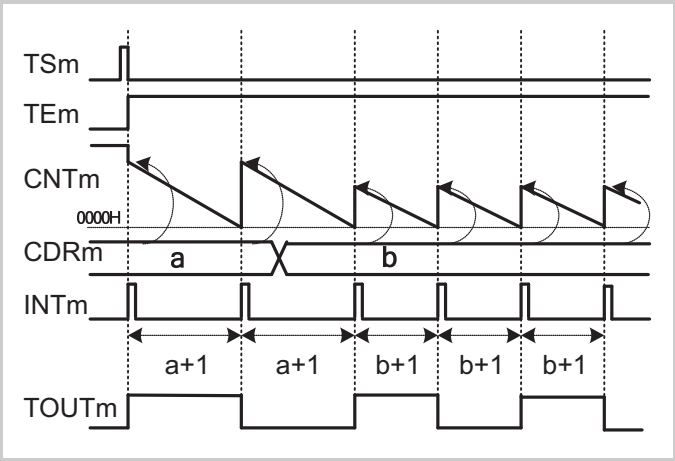


Figure 15-30 General timing diagram for Interval Timer Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-17 TAUAnCMORm settings for Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-18 TAUAnCMURm settings for Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 15-19 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Interval Timer Function. Therefore, these registers must be set to 0.

Table 15-20 Simultaneous rewrite settings for Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for Interval Timer Function

Table 15-21 Operating procedure for Interval Timer Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-17 "TAUAnCMORm settings for Interval Timer Function" on page 754* and *Table 15-18 "TAUAnCMURm settings for Interval Timer Function" on page 754* Set the value of the TAUAnCDRm register Set the channel output mode by setting the control bits as described in *Table 15-19 "Control bit settings for Independent Channel Output Mode 1" on page 755*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.MD0 = 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The TAUAnCDRm register value can be changed at any time. The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation • INTTAUAnIm is generated and TAUAnTTOUTm toggles.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H, count clock = PCLK/2

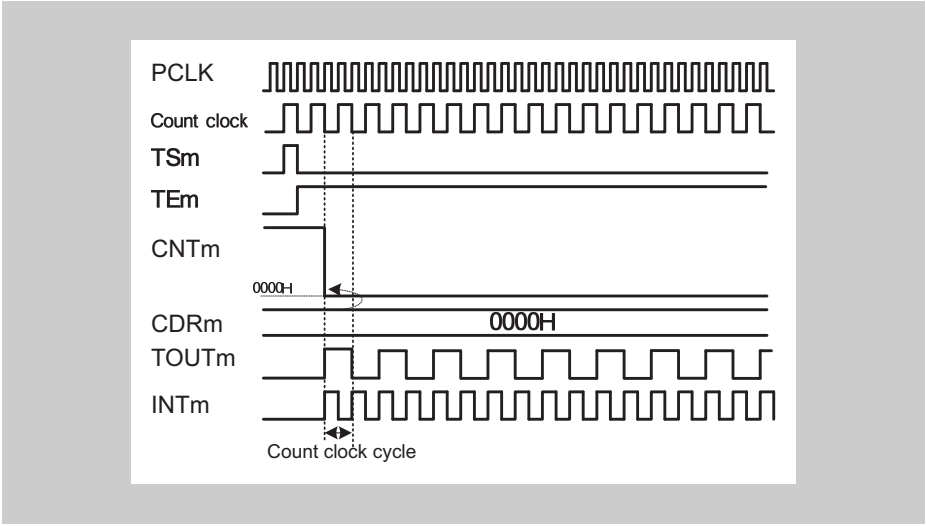


Figure 15-31 TAUAnCDRm = 0000_H, count clock = PCLK/2

- If TAUAnCDRm = 0000_H and the count clock = PCLK/2^1, the TAUAnCDRm value is written to TAUAnCNTm every count clock, meaning that TAUAnCNTm is always 0000_H.
- INTTAUAnIm is generated every count clock, resulting in TAUAnTTOUTm toggling every count clock.

(b) TAUAnCDRm = 0000_H, count clock = PCLK

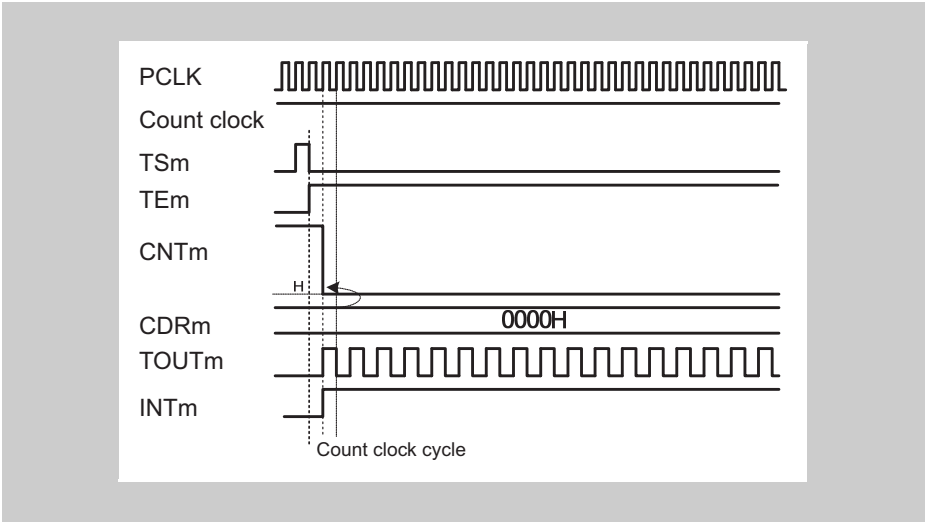


Figure 15-32 TAUAnCDRm = 0000_H, count clock = PCLK

- If TAUAnCDRm = 0000_H and the count clock = PCLK, the TAUAnCDRm value is written to TAUAnCNTm every PCLK clock, meaning that TAUAnCNTm is always 0000_H.
- INTTAUAnIm is generated continuously, resulting in TAUAnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

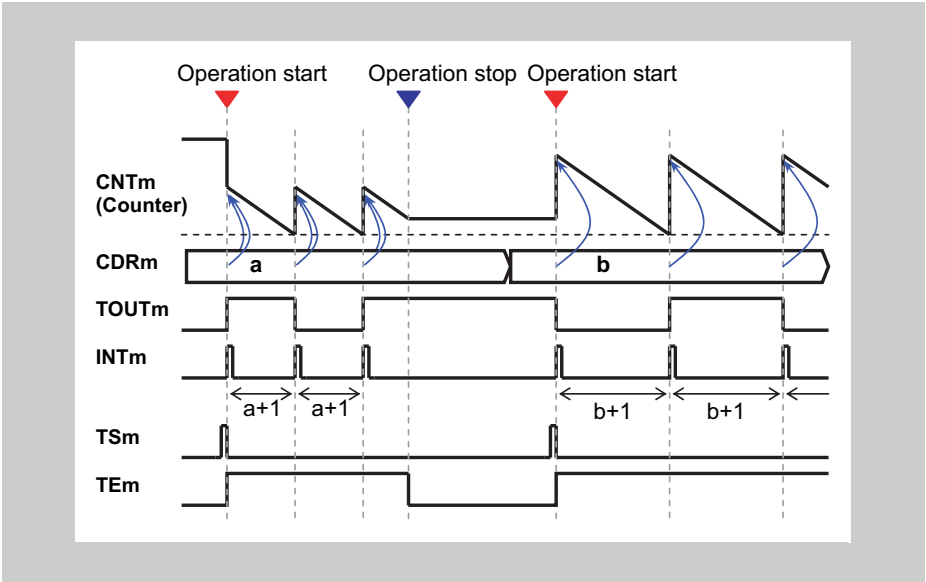


Figure 15-33 Operation stop and restart, TAUAnCMORm.MD0 = 1

- The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0.
- TAUAnCNTm and TAUAnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUAnTS.TSm to 1.

(d) Forced restart

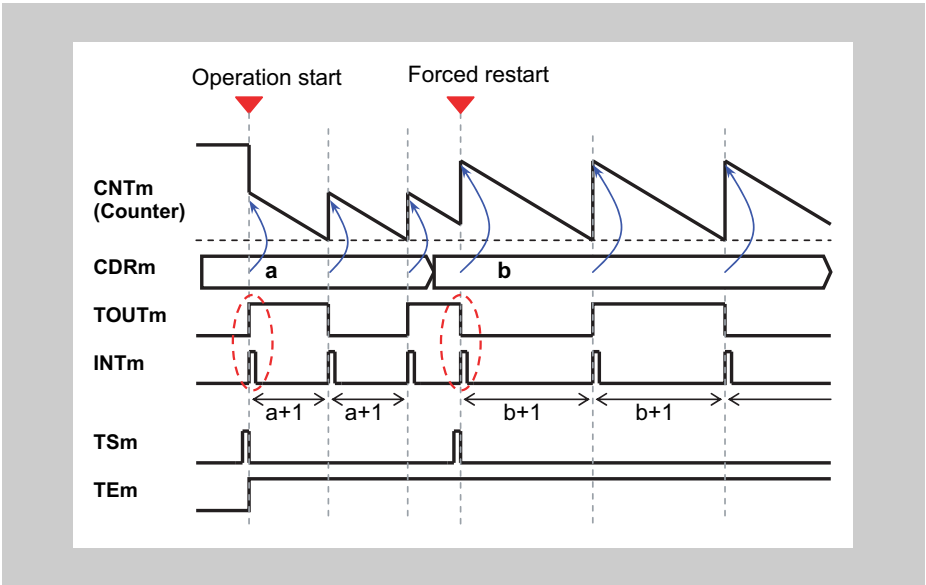


Figure 15-34 Forced restart operation, TAUAnCMORm.MD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUAnTS.TSm to 1 during operation.
- If the TAUAnCMORm.MD0 bit is set to 1, the first interrupt after a start or restart is generated.

15.16.2 TAUAnTTINm Input Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUAnIm) at regular intervals or when a valid TAUAnTTINm input edge is detected. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 15-22 "TAUAnCMORm settings for TAUAnTTINm Input Interval Timer Function" on page 761*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *15.9 "Channel Output Modes" on page 731*

Description This function operates in an identical manner to the Interval Timer Function (see *15.16.1 "Interval Timer Function" on page 752*), except that this function is restarted by a valid TAUAnTTINm input edge. The type of edge used as the trigger is specified using the TAUAnCMURm.TIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUAnIm cycle = count clock cycle x (TAUAnCDRm + 1)

TAUAnTTOUTm square wave cycle = count clock cycle x (TAUAnCDRm + 1) x 2

(3) Block diagram and general timing diagram

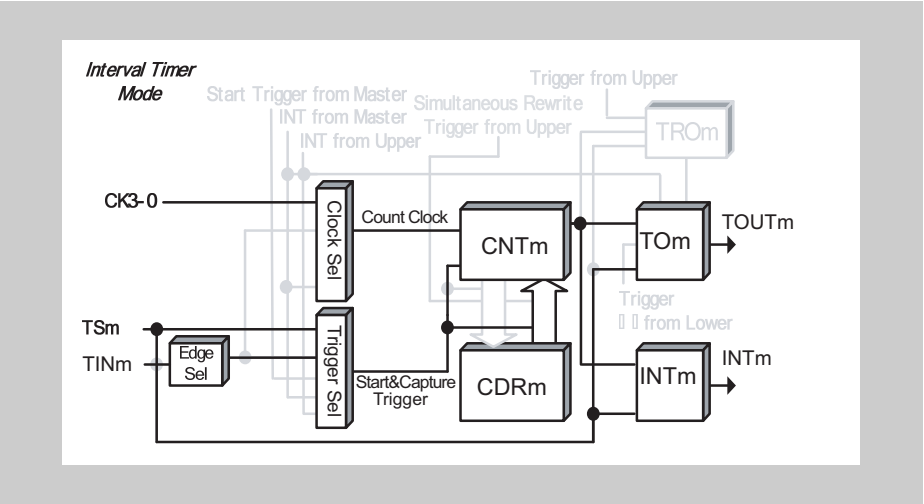


Figure 15-35 Block diagram for TAUAnTTINm Input Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)
- Rising edge detection (TAUAnCMURm.TIS[1:0] = 01_B)

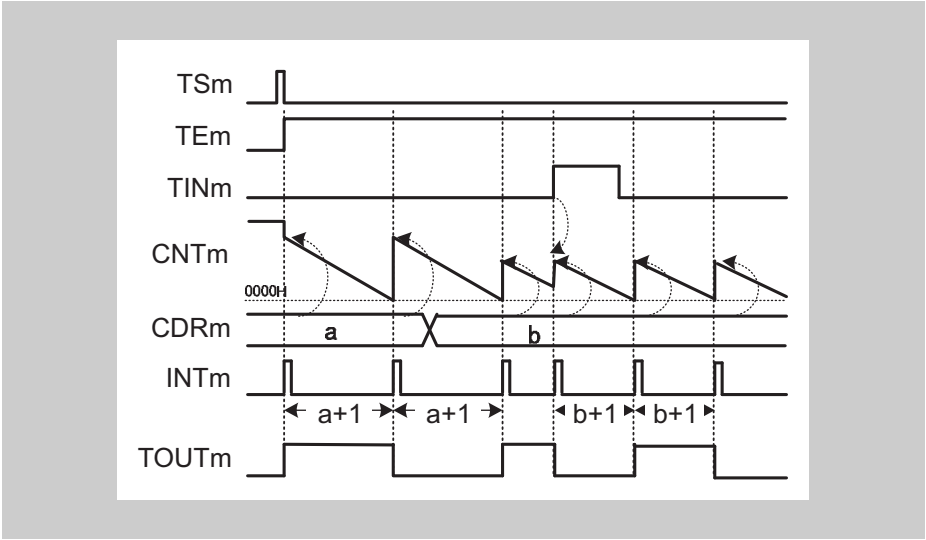


Figure 15-36 General timing diagram for TAUAnTTINm Input Interval Timer Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-22 TAUAnCMORm settings for TAUAnTTINm Input Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-23 TAUAnCMURm settings for TAUAnTTINm Input Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 15-24 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Interval Timer Function. Therefore, these registers must be set to 0.

Table 15-25 Simultaneous rewrite settings for TAUAnTTINm Input Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUAnTTINm Input Interval Timer Function

Table 15-26 Operating procedure for TAUAnTTINm Input Interval Timer Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-22 "TAUAnCMORm settings for TAUAnTTINm Input Interval Timer Function" on page 761* and *Table 15-23 "TAUAnCMURm settings for TAUAnTTINm Input Interval Timer Function" on page 761* Set the value of the TAUAnCDRm register Set the channel output mode by setting the control bits as described in *Table 15-24 "Control bit settings for Independent Channel Output Mode 1" on page 762*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.MD0 = 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The values of the TAUAnCMURm.TIS[1:0] and TAUAnTO.TOm bits and the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at all times. Detection of TAUAnTTINm edge	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation • INTTAUAnIm is generated and TAUAnTTOUTm toggles When a TAUAnTTINm input valid edge is detected during count operation, TAUAnCNTm reloads the TAUAnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *15.16.1 "Interval Timer Function" on page 752* also apply, except for this function the counter can also be restarted by a valid TAUAnTTINm input edge.

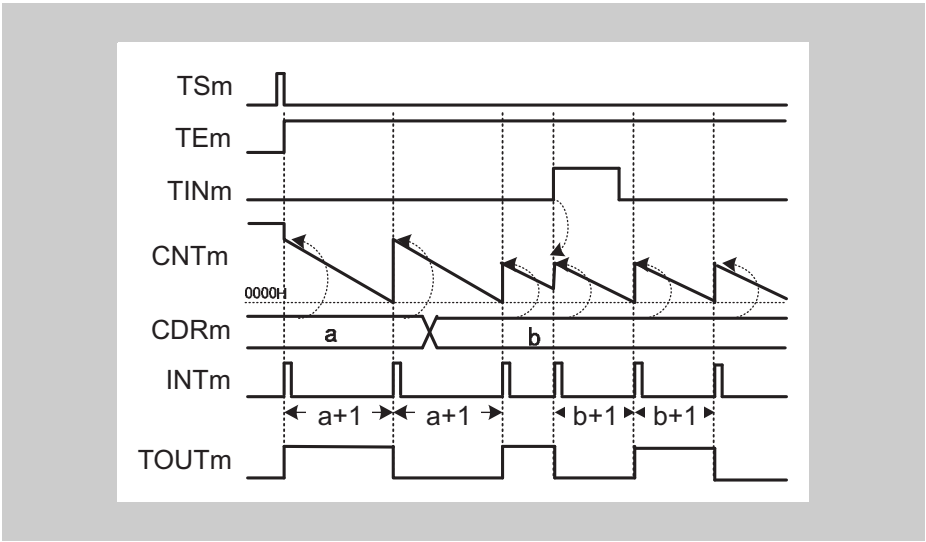


Figure 15-37 Counter triggered by rising TAUAnTTINm input edge (TAUAnCMURm.TIS[1:0] = 01_B), TAUAnCMORm.MD0 = 1

- If a valid TAUAnTTINm input edge is detected, an interrupt is generated which causes TAUAnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUAnCMURm.TIS[1:0] = 01_B).

15.16.3 Delay Count Function

(1) Overview

Summary This function generates interrupts (INTTAUAnIm), which have a defined delay to the TAUAnTTINm input signal. TAUAnTTINm input signal pulses that occur within the delay period are ignored.

Prerequisites
- The operation mode must be set to One Count Mode, refer to *Table 15-27 "TAUAnCMORm settings for Delay Count Function" on page 767*
- TAUAnTTOUTm is not used for this function
- Trigger detection must be disabled during counting (TAUAnCMORn.MD0 = 0)

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. The value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value.

When the counter reaches 0000_H an interrupt is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input edge.

When the counter is counting down, further TAUAnTTINm input signals are ignored, i. e. the counter does not reset.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TIS[1:0] bits:
- If TAUAnCMURm.TIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUAnCMURm.TIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUAnCMURm.TIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Delay between TAUAnTTINm and INTTAUAnIm =

count clock cycle × (TAUAnCDRm + 1)

(3) Block diagram and general timing diagram

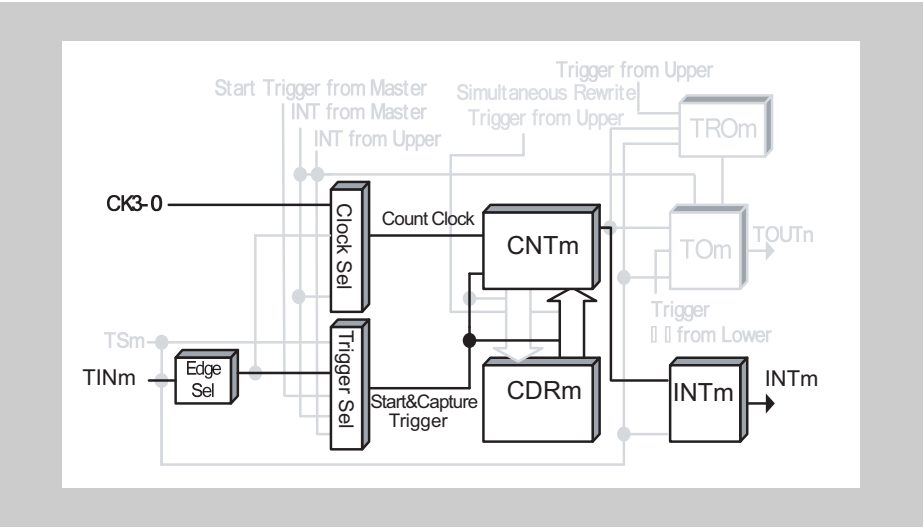


Figure 15-38 Block diagram for Delay Count Function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

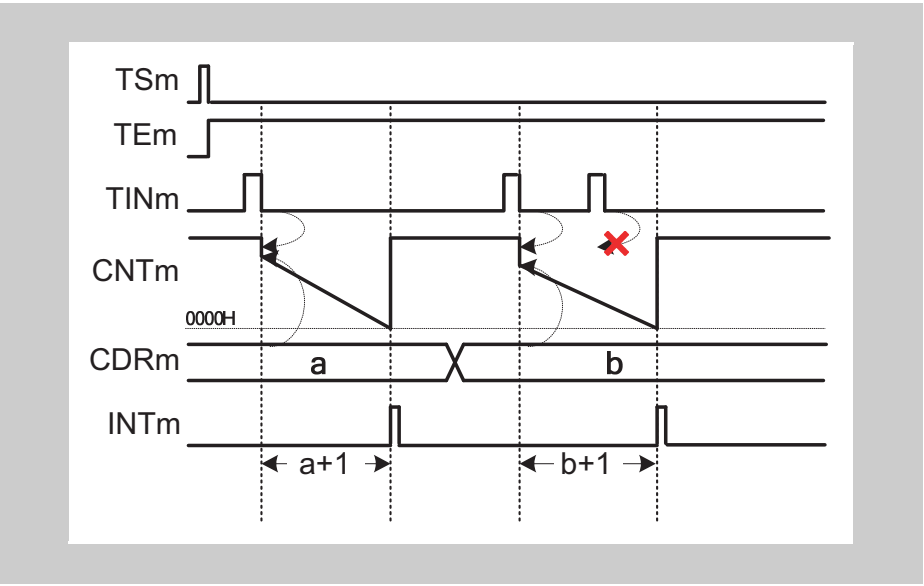


Figure 15-39 General timing diagram for Delay Count Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-27 TAUAnCMORm settings for Delay Count Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-28 TAUAnCMURm settings for Delay Count Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Delay Count Function. Therefore, these registers must be set to 0.

Table 15-29 Simultaneous rewrite settings for Delay Count Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for Delay Count Function

Table 15-30 Operating procedure for Delay Count Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-27 "TAUAnCMORm settings for Delay Count Function" on page 767* and *Table 15-28 "TAUAnCMURm settings for Delay Count Function" on page 767* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	The values set in the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: INTTAUAnIm is generated TAUAnCNTm stops counting, returns to $FFFF_H$, and waits for a trigger. If a trigger occurs while TAUAnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its value.

Restart

15.16.4 One-Pulse Output Function

(1) Overview

Summary This function generates an interrupt (INTTAUAnIm) when a valid TAUAnTTINm input edge is detected and also a specific interval later. TAUAnTTINm input signal pulses that occur within the defined interval are ignored. When an interrupt is generated, the TAUAnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Pulse One Count Mode, refer to *Table 15-31 "TAUAnCMORm settings for One-Pulse Output Function" on page 772*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *15.9 "Channel Output Modes" on page 731*
- Trigger detection must be disabled during counting (TAUAnCMORn.MD0 = 0).

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input edge is detected. The value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value. An interrupt is generated and TAUAnTTOUTm toggles.

When the counter reaches 0001_H an interrupt is generated and TAUAnTTOUTm toggles. The counter stops at 0000_H and awaits the next valid TAUAnTTINm input edge.

When the counter is counting down, further TAUAnTTINm input signals are ignored, i.e. the counter does not reset.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TIS[1:0] bits:

- If TAUAnCMURm.TIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUAnCMURm.TIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUAnCMURm.TIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Interval between TAUAnTTINm and INTTAUAnIm = TAUAnTTOUTm (timer output) width = count clock cycle × TAUAnCDRm

(3) Block diagram and general timing diagram

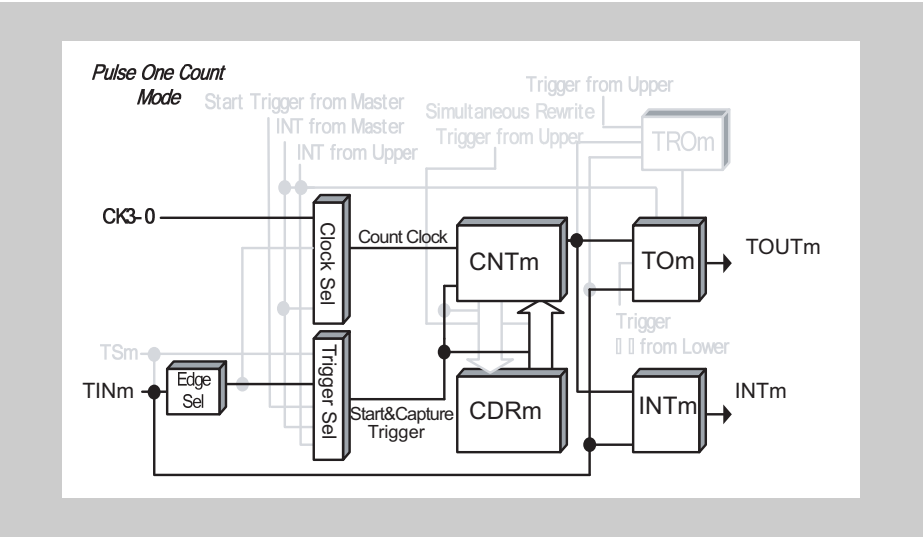


Figure 15-40 Block diagram for One-Pulse Output Function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

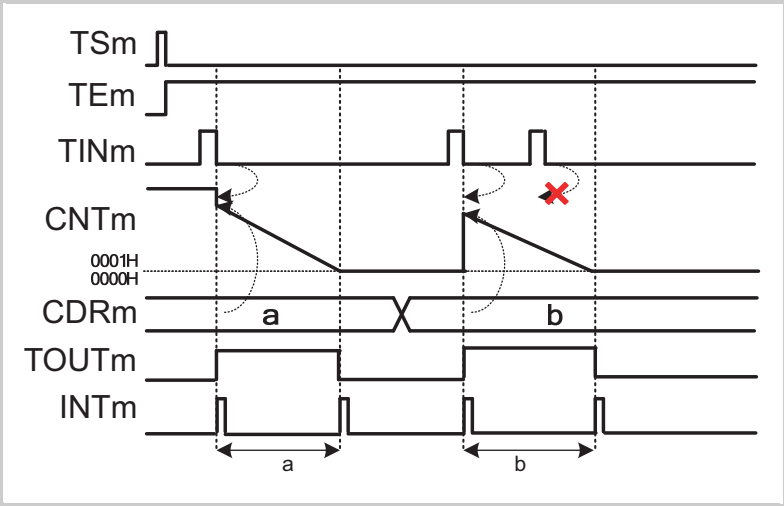


Figure 15-41 General timing diagram for One-Pulse Output Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-31 TAUAnCMORm settings for One-Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-32 TAUAnCMURm settings for One-Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement)

(c) Channel output mode

Table 15-33 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the One-Pulse Output Function. Therefore, these registers must be set to 0.

Table 15-34 Simultaneous rewrite settings for One-Pulse Output Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for One-Pulse Output Function

Table 15-35 Operating procedure for One-Pulse Output Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-31 "TAUAnCMORm settings for One-Pulse Output Function" on page 772* and *Table 15-32 "TAUAnCMURm settings for One-Pulse Output Function" on page 772* Set the value of the TAUAnCDRm register Set the channel output mode by setting the control bits as described in *Table 15-33 "Control bit settings for Independent Channel Output Mode 1" on page 773*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at all times.	INTTAUAnIm is generated when TAUAnCNTm starts and TAUAnTTOUTm is set to its active level. TAUAnCNTm counts down. When the counter reaches 0001_H: • INTTAUAnIm is generated • TAUAnTTOUTm is set to its inactive level. TAUAnCNTm stops counting and waits for a trigger. If a trigger occurs while TAUAnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values.

Restart

15.17 Independent Channel Signal Measurement Functions

This chapter describes functions that measure the widths of an individual TAUAnTTINm pulse or the total width of successive TAUAnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *15.17.1 “TAUAnTTINm Input Pulse Interval Measurement Function”*
- *15.17.2 “TAUAnTTINm Input Signal Width Measurement Function”*
- *15.17.3 “Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)”*
- *15.17.4 “TAUAnTTINm Input Period Count Detection Function”*
- *15.17.5 “Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)”*
- *15.17.6 “TAUAnTTINm Input Pulse Interval Judgment Function”*
- *15.17.7 “TAUAnTTINm Input Signal Width Judgment Function”*

15.17.1 TAUAnTTINm Input Pulse Interval Measurement Function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUAnCSRm.OVF to measure the interval of the TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture Mode, refer to *Table 15-37 "TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Measurement Function" on page 778*
- TAUAnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The counter TAUAnCNTm starts counting up from 0000_H. When a valid TAUAnTTINm edge is detected, the value of TAUAnCNTm is captured, transferred to TAUAnCDRm, and an interrupt INTTAUAnIm is generated. The counter resets to 0000_H and subsequently continues operation.

If the counter reaches $FFFF_H$ before a valid TAUAnTTINm edge is detected, it overflows to 0000_H. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUAnCDRm and TAUAnCSRm.OVF respectively depend on the values of bits TAUAnCMORm.COS[1:0]:

Table 15-36 Effects of an overflow

TAUAnCMORm.COS[1:0]	When overflow occurs		When a valid TAUAnTTINm input is then detected	
	TAUAnCDRm	TAUAnCSRm.OVF	TAUAnCDRm and TAUAnCNTm	TAUAnCSRm.OVF
00	Unchanged	0	TAUAnCNTm written to TAUAnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUAnCNTm set to 0, TAUAnCDRm unchanged	0
11		1		

If an overflow is set (TAUAnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUAnCSCm.CLOV = 1.

The combination of the value of TAUAnCDRm and TAUAnCSRm.OVF can be used to deduce the interval of the TAUAnTTINm signal. However, if an overflow occurs multiple times before a valid TAUAnTTINm input is detected, the overflow bit TAUAnCSRm.OVF cannot indicate this.

The function can be stopped by setting TAUAnTT.TTm = 1, which in turn sets TAUAnTE.TEm = 0. TAUAnCNTm stops but retains its value. While the function is stopped, TAUAnTTINm input valid edge detection and TAUAnCNTm capture are not performed.

The function can be restarted by setting TAUAnTS.TSm = 1. The counter is reset to 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TSm = 1 during operation.

Conditions If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

Note When TAUAnCMORm.COS[1:0] = 11_B, the value of TAUAnCNTm is *not* written to TAUAnCDRm when the first valid TAUAnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUAnTTINm input pulse interval = count clock cycle x
[(TAUAnCSRm.OVF x ($FFFF_H$ + 1)) + TAUAnCDRm capture value + 1]

(3) Block diagram and general timing diagram

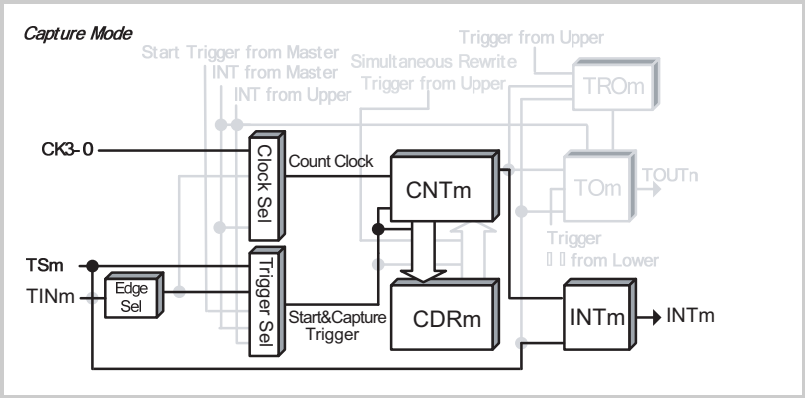


Figure 15-42 Block diagram for TAUAnTTINm Input Pulse Interval Measurement Function

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)
- When a valid TAUAnTTINm input is detected after an overflow TAUAnCDRm is changed and TAUAnCSRm.OVF is set to 1 (TAUAnCMORm.COS[1:0] = 00_B)

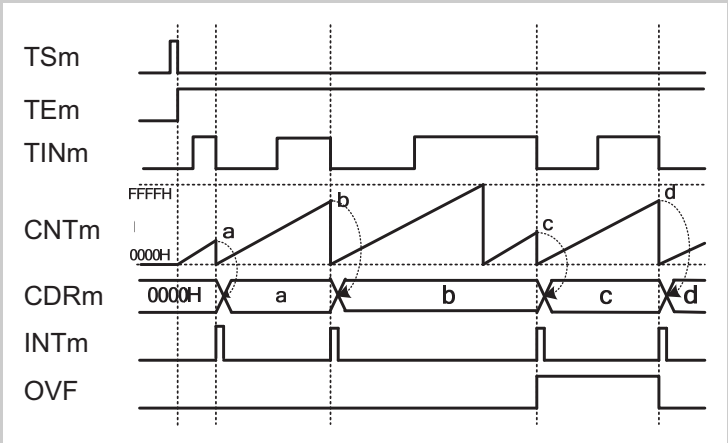


Figure 15-43 General timing diagram for TAUAnTTINm Input Pulse Interval Measurement Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-37 TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external capture trigger
COS[1:0]	See *Table 15-36 "Effects of an overflow" on page 776*
MD[4:1]	0010: Capture Mode
MD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-38 TAUAnCMURm settings for TAUAnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Pulse Interval Measurement Function. Therefore, these registers must be set to 0.

Table 15-39 Simultaneous rewrite settings for TAUAnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUAnTTINm Input Pulse Interval Measurement Function

Table 15-40 Operating procedure for TAUAnTTINm Input Pulse Interval Measurement Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-37 "TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Measurement Function" on page 778* and *Table 15-38 "TAUAnCMURm settings for TAUAnTTINm Input Pulse Interval Measurement Function" on page 778* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. INTTAUAnIm is generated when TAUAnCMORm.MD0 is set to 1.
During operation	Detection of TAUAnTTINm edges. The TAUAnCMURm.TIS[1:0] bits can be changed at any time. The TAUAnCDRm and TAUAnCSRm registers can be read at any time.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm, and returns to 0000_H • INTTAUAnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUAnCMORm.COS[1:0] = 00_B

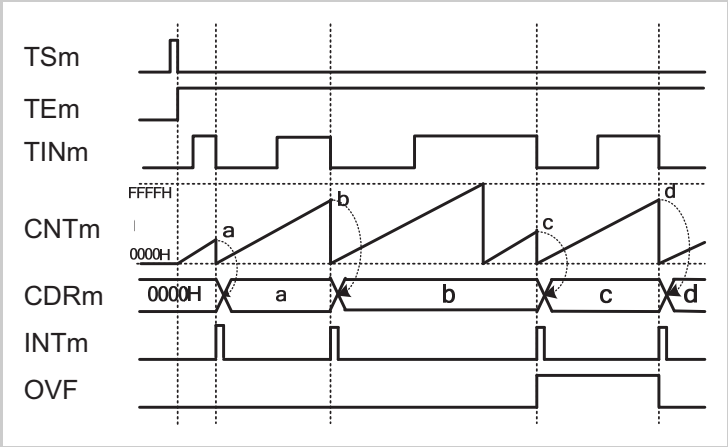


Figure 15-44 TAUAnCMORm.COS[1:0] = 00_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is written to TAUAnCDRm and TAUAnCSRm.OVF is set to 1.

(b) TAUAnCMORm.COS[1:0] = 01_B

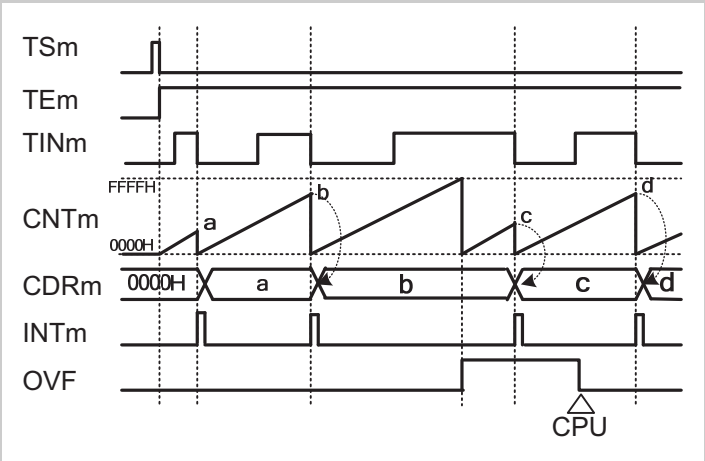


Figure 15-45 TAUAnCMORm.COS[1:0] = 01_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is written to TAUAnCDRm.

- TAUAnCSRm.OVF is only cleared by a CPU command.

(c) TAUAnCMORm.COS[1:0] = 10_B

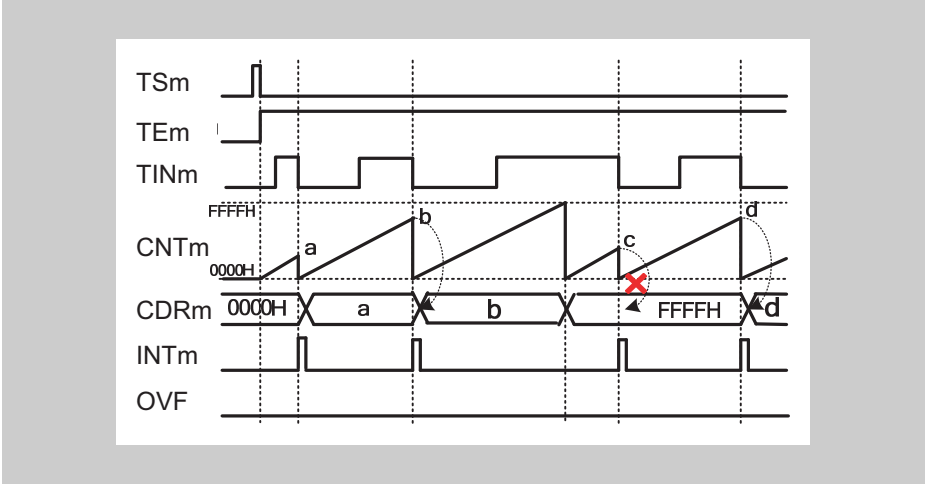


Figure 15-46 TAUAnCMORm.COS[1:0] = 10_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$ and TAUAnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.OVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.

(d) TAUAnCMORm.COS[1:0] = 11_B

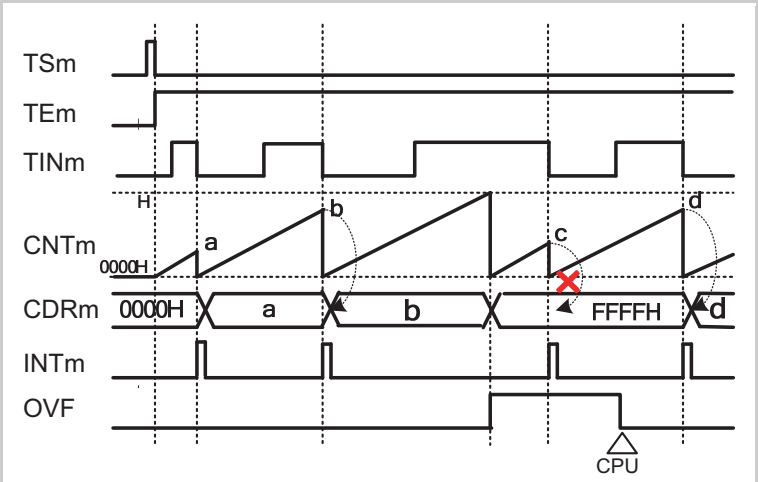


Figure 15-47 TAUAnCMORm.COS[1:0] = 11_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$, and TAUAnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.OVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.
- TAUAnCSRm.OVF is cleared by a CPU command.

15.17.2 TAUAnTTINm Input Signal Width Measurement Function

(1) Overview

Summary This function measures the width of a TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & One Count Mode, refer to *Table 15-42 "TAUAnCMORm settings for TAUAnTTINm Input Signal Width Measurement Function" on page 786*
- TAUAnTTOUTm is not used for this function
- TAUAnCMORm.MD0 must be set to 0

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. When a valid TAUAnTTINm start edge is detected, the counter TAUAnCNTm starts counting up from 0000_H. When a valid TAUAnTTINm stop edge is detected, the value of TAUAnCNTm is captured, transferred to TAUAnCDRm, and an interrupt INTTAUAnIm is generated. The counter retains its value and awaits the next valid TAUAnTTINm input start edge.

If the counter reaches $FFFF_H$ before a valid TAUAnTTINm stop edge is detected, it overflows. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUAnCDRm and TAUAnCSRm.OVF respectively depend on the values of bits TAUAnCMORm.COS[1:0]:

Table 15-41 Effects of an overflow

TAUAnCMORm.COS[1:0]	When overflow occurs		When a valid TAUAnTTINm input stop edge is detected	
	TAUAnCDRm	TAUAnCSRm.OVF	TAUAnCDRm and TAUAnCNTm	TAUAnCSRm.OVF
00	Unchanged	0	TAUAnCNTm written to TAUAnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUAnCNTm stops counting TAUAnCDRm unchanged	0
11		1		

If an overflow is set (TAUAnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUAnCSCm.CLOV = 1.

The combination of the value of TAUAnCDRm and TAUAnCSRm.OVF can be used to deduce the width of the TAUAnTTINm signal. However, if an overflow occurs multiple times before a valid TAUAnTTINm input is detected, the overflow bit TAUAnCSRm.OVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUAnCMORm.COS[1:0] = 11_B, the value of TAUAnCNTm is *not* written to TAUAnCDRm when the first valid TAUAnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUAnTTINm input signal width = count clock cycle x [(TAUAnCSRm.OVF x ($FFFF_H$ + 1)) + TAUAnCDRm capture value + 1]

(3) Block diagram and general timing diagram

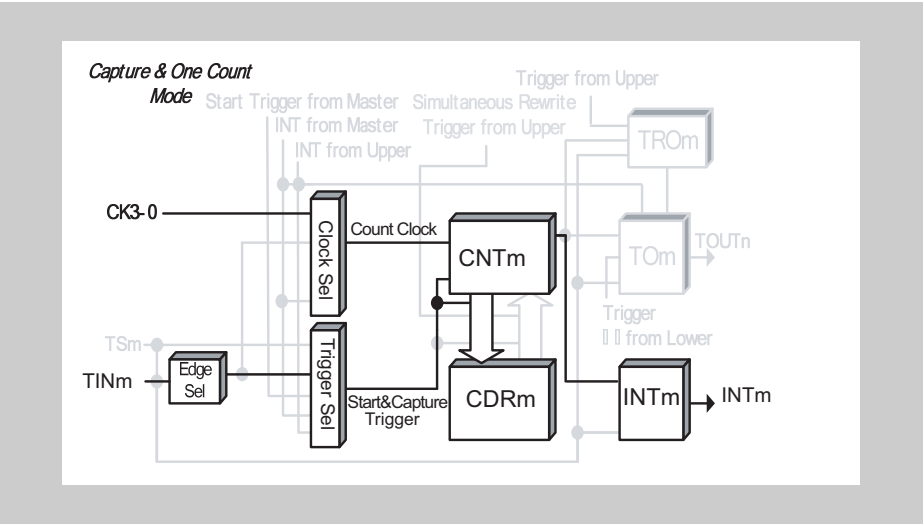


Figure 15-48 Block diagram for TAUAnTTINm Input Signal Width Measurement Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TIS[1:0] = 11_B)
- When a valid TAUAnTTINm input is detected after an overflow TAUAnCDRm is changed and TAUAnCSRm.OVF is set to 1 (TAUAnCMORm.COS[1:0] = 00_B)

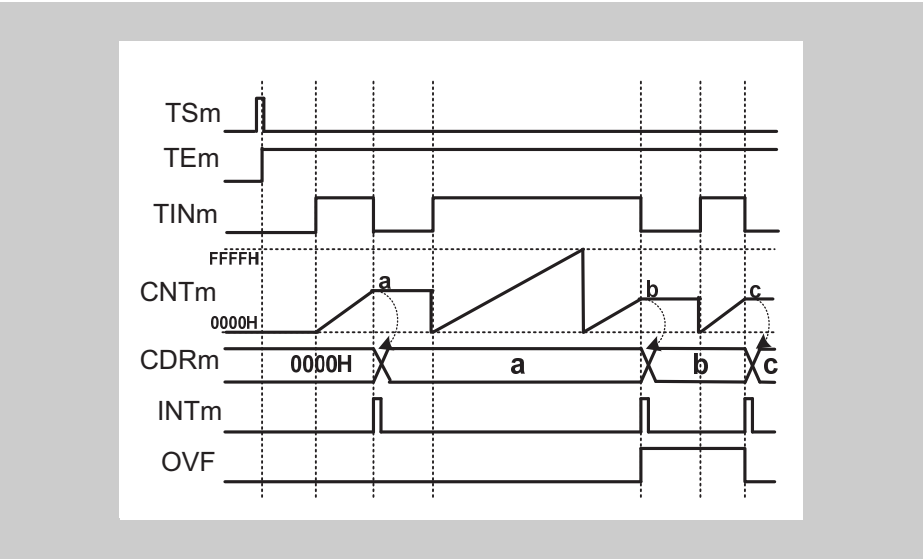


Figure 15-49 General timing diagram for TAUAnTTINm Input Signal Width Measurement Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-42 TAUAnCMORm settings for TAUAnTTINm Input Signal Width Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	See *Table 15-41 “Effects of an overflow” on page 784*
MD[4:1]	0110: Capture & One Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-43 TAUAnCMURm settings for TAUAnTTINm Input Signal Width Measurement Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Signal Width Measurement Function. Therefore, these registers must be set to 0.

Table 15-44 Simultaneous rewrite settings for TAUAnTTINm Input Signal Width Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUAnTTINm Input Signal Width Measurement Function

Table 15-45 Operating procedure for TAUAnTTINm Input Signal Width Measurement Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-42 "TAUAnCMORm settings for TAUAnTTINm Input Signal Width Measurement Function" on page 786* and *Table 15-43 "TAUAnCMURm settings for TAUAnTTINm Input Signal Width Measurement Function" on page 786* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a TAUAnTTINm start is detected, TAUAnCNTm starts to count up.
During operation	Detection of TAUAnTTINm edges. The TAUAnCMURm.TIS[1:0] bits can be changed at any time. The TAUAnCDRm and TAUAnCSRm registers can be read at any time.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm, and retains its value • INTTAUAnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUAnCMORm.COS[1:0] = 00_B

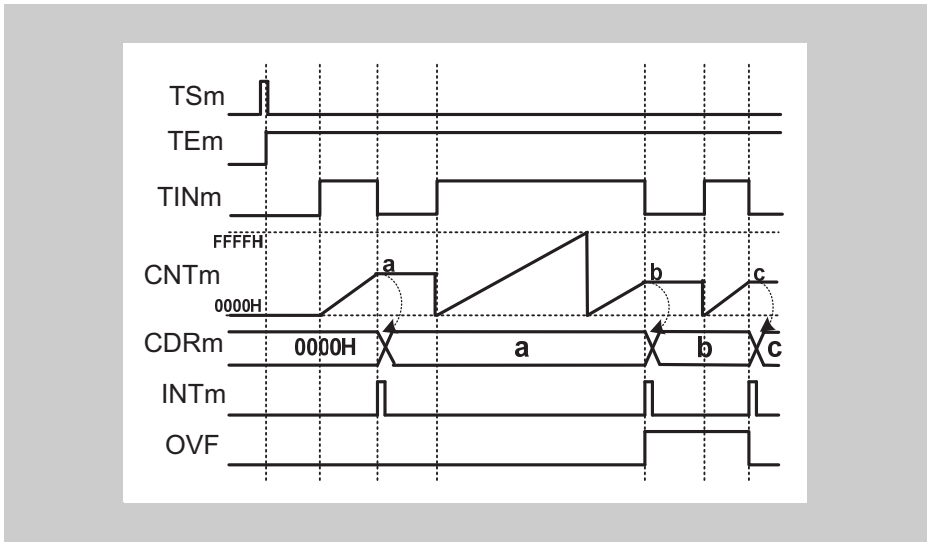


Figure 15-50 TAUAnCMORm.COS[1:0] = 00_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is written to TAUAnCDRm and TAUAnCSRm.OVF is set to 1.

(b) TAUAnCMORm.COS[1:0] = 01_B

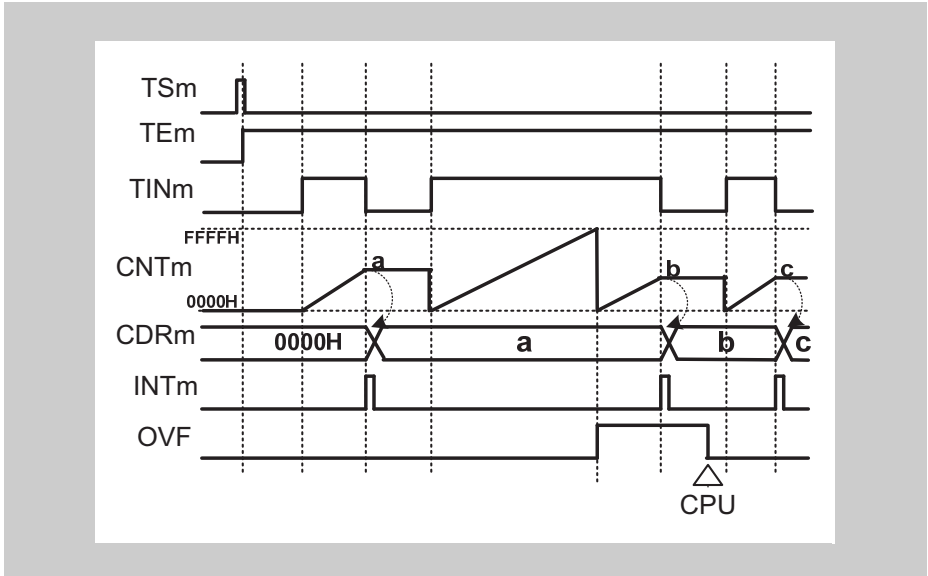


Figure 15-51 TAUAnCMORm.COS[1:0] = 01_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUAnCDRm remains unchanged and TAUAnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, the value of TAUAnCNTm is written to TAUAnCDRm.

- TAUAnCSRm.OVF is only cleared by a CPU command.

(c) TAUAnCMORm.COS[1:0] = 10_B

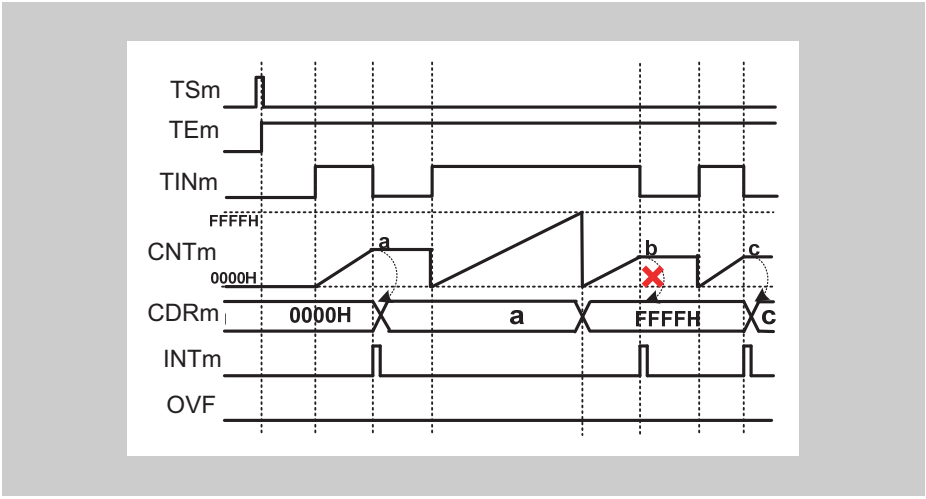


Figure 15-52 TAUAnCMORm.COS[1:0] = 10_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$ and TAUAnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.OVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.

(d) TAUAnCMORm.COS[1:0] = 11_B

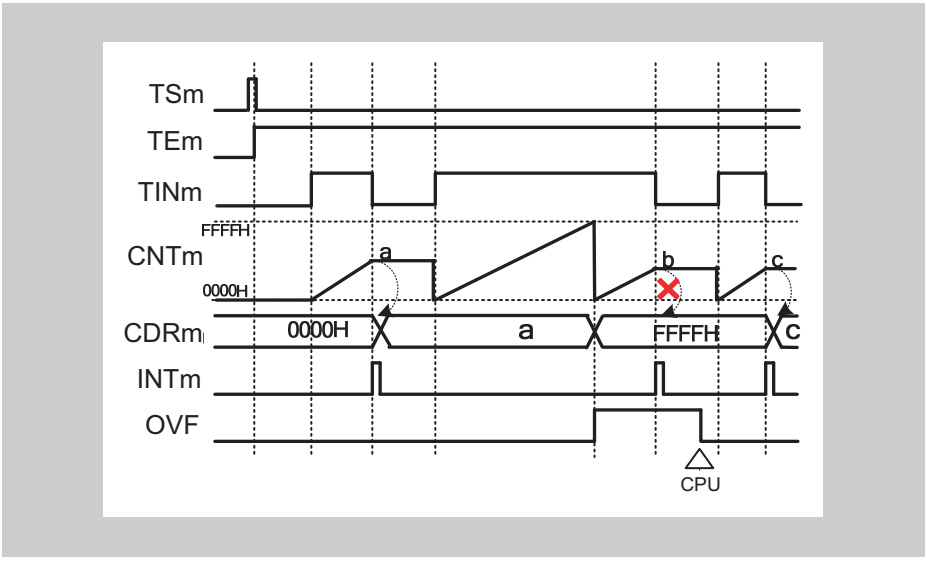


Figure 15-53 TAUAnCMORm.COS[1:0] = 11_B, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUAnCDRm is set to $FFFF_H$, and TAUAnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUAnTTINm input edge, TAUAnCNTm is reset to 0, but TAUAnCDRm and TAUAnCSRm.OVF remain unchanged.
- Thus, the next TAUAnTTINm input valid edge after the overflow is ignored.
- TAUAnCSRm.OVF is cleared by a CPU command.

15.17.3 Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

(1) Overview

Summary This function measures the width of an individual TAUAnTTINm input signal. An interrupt is generated if the TAUAnTTINm input width is longer than $FFFF_H$.

Prerequisites
- The operation mode must be set to One Count Mode, refer to *Table 15-46 "TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)" on page 794*
- TAUAnTTOUTm is not used for this function
- The value of TAUAnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. $FFFF_H$ is written to TAUAnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUAnTTINm input start edge is detected, TAUAnCNTm reloads $FFFF_H$ and starts to count down.

If the counter reaches 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TIS[1:0] bits:
- If TAUAnCMURm.TIS[1:0] = 10_B, the TAUAnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TIS[1:0] = 11_B, the TAUAnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

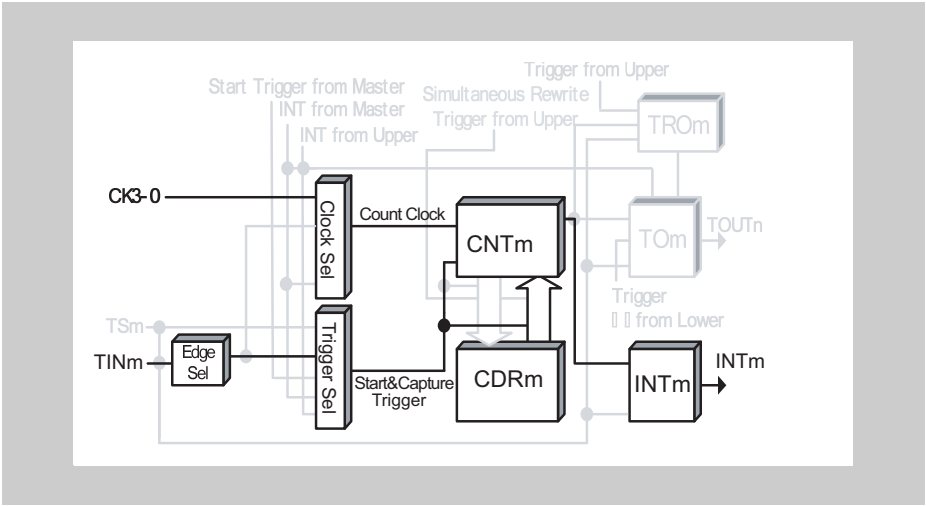


Figure 15-54 Block diagram for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TIS[1:0] = 11_B)

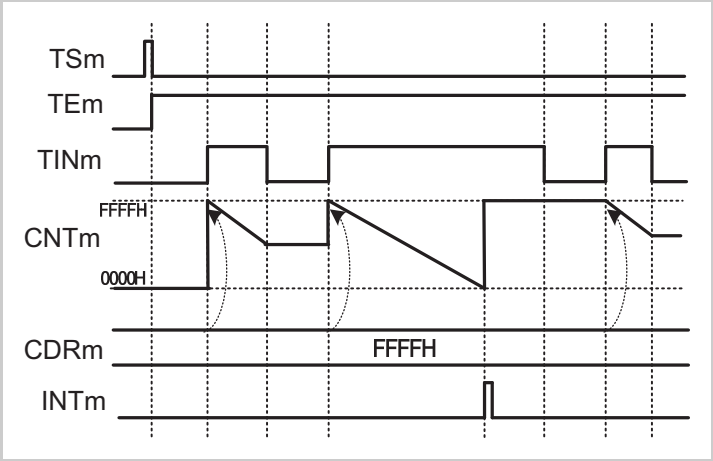


Figure 15-55 General timing diagram for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-46 TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-47 TAUAnCMURm settings Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement). Therefore, these registers must be set to 0.

Table 15-48 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(4) Operating procedure for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

Table 15-49 Operating procedure for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-46 "TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)" on page 794* and *Table 15-47 "TAUAnCMURm settings Overflow Interrupt Output Function (During TAUAnTTINm Width Measurement)" on page 794* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$).
During operation	The TAUAnCNTm register can be read at any time. Detection of TAUAnTTINm edges.	TAUAnCNTm counts down. When the counter reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm stops counting at $FFFF_H$ and waits for a trigger. When a reverse edge of TAUAnTTINm is detected during count operation: • TAUAnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.17.4 TAUAnTTINm Input Period Count Detection Function

(1) Overview

Summary This function measures the cumulative width of a TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & Gate Count Mode, refer to *Table 15-50 "TAUAnCMORm settings for TAUAnTTINm Input Period Count Detection Function" on page 799*
- TAUAnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The counter awaits a valid TAUAnTTINm input edge.

When a valid TAUAnTTINm input start edge is detected, the counter starts to count from 0000_H.

When a valid TAUAnTTINm input stop edge is detected, the current TAUAnCNTm value is written to TAUAnCDRm and an interrupt (INTTAUAnIm) is generated. The counter stops and retains its value until the next valid TAUAnTTINm input start edge is detected.

If the counter reaches $FFFF_H$ the bit TAUAnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUAnCSRm.OVF is reset by the CPU by setting TAUAnCSCm.CLOV = 1.

Note The input TAUAnTTINm is sampled at the frequency of the operation clock, specified by the TAUAnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUAnTTOUTm has an error of ± 1 operation clock cycle.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TIS[1:0] bits:
- If TAUAnCMURm.TIS[1:0] = 10_B, the TAUAnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TIS[1:0] = 11_B, the TAUAnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

Cumulative TAUAnTTINm input width =
count clock cycle × (($FFFF_H$ × TAUAnCSRm.OVF) + (TAUAnCDRm capture value + 1))

(3) Block diagram and general timing diagram

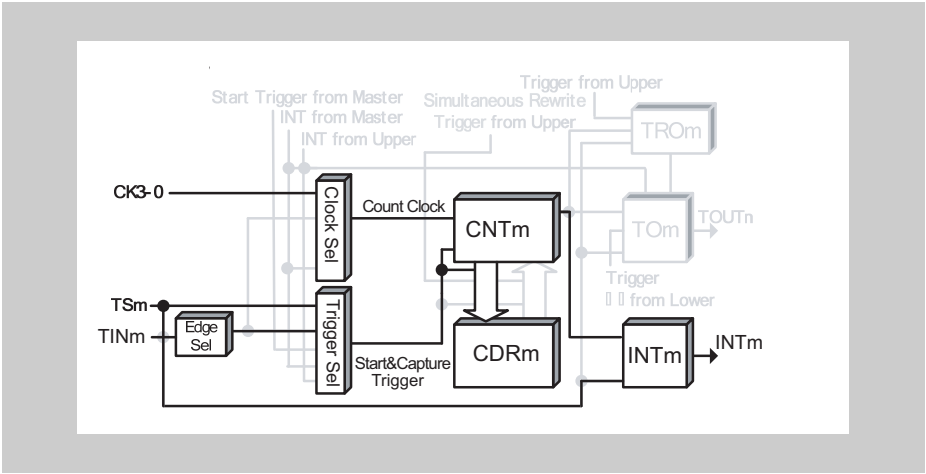


Figure 15-56 Block diagram for TAUAnTTINm Input Period Count Detection Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TIS[1:0] = 11_B)

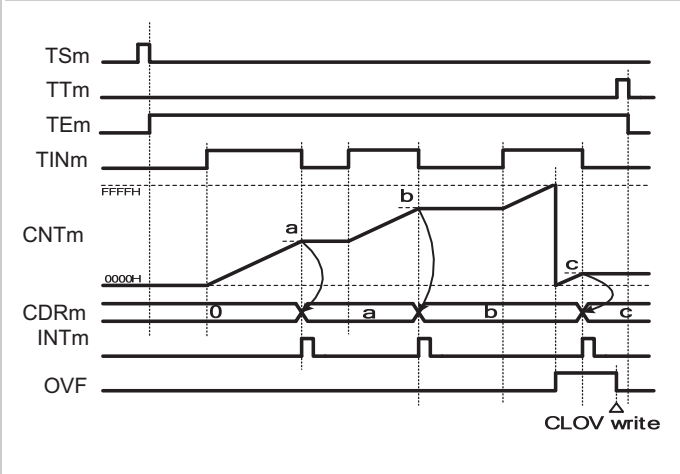


Figure 15-57 General timing diagram for TAUAnTTINm Input Period Count Detection Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-50 TAUAnCMORm settings for TAUAnTTINm Input Period Count Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	01: Overflow (TAUAnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1101: Capture & Gate Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-51 TAUAnCMURm settings for TAUAnTTINm Input Period Count Detection Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Period Count Detection Function. Therefore, these registers must be set to 0.

Table 15-52 Simultaneous rewrite settings for TAUAnTTINm Input Period Count Detection Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUAnTTINm Input Period Count Detection Function

Table 15-53 Operating procedure for TAUAnTTINm Input Period Count Detection Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-50 "TAUAnCMORm settings for TAUAnTTINm Input Period Count Detection Function" on page 799* and *Table 15-51 "TAUAnCMURm settings for TAUAnTTINm Input Period Count Detection Function" on page 799* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a start edge is detected, TAUAnCNTm is cleared to 0000_H and TAUAnCNTm starts to count up.
During operation	Detection of TAUAnTTINm edges. The TAUAnCDRm, TAUAnCNTm, and TAUAnCSRm registers can be read at any time. The TAUAnCSC.CLOV bit can be set to 1.	When a TAUAnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUAnCNTm starts to count up from the stop value. When TAUAnCNTm detects a capture edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUAnCDRm and INTTAUAnIm is generated. Counting stops at the "value transferred to TAUAnCDRm + 1" value and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. If the TAUAnCNTm reaches $FFFF_H$, the counter overflows and TAUAnCSR.OVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and it and TAUAnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

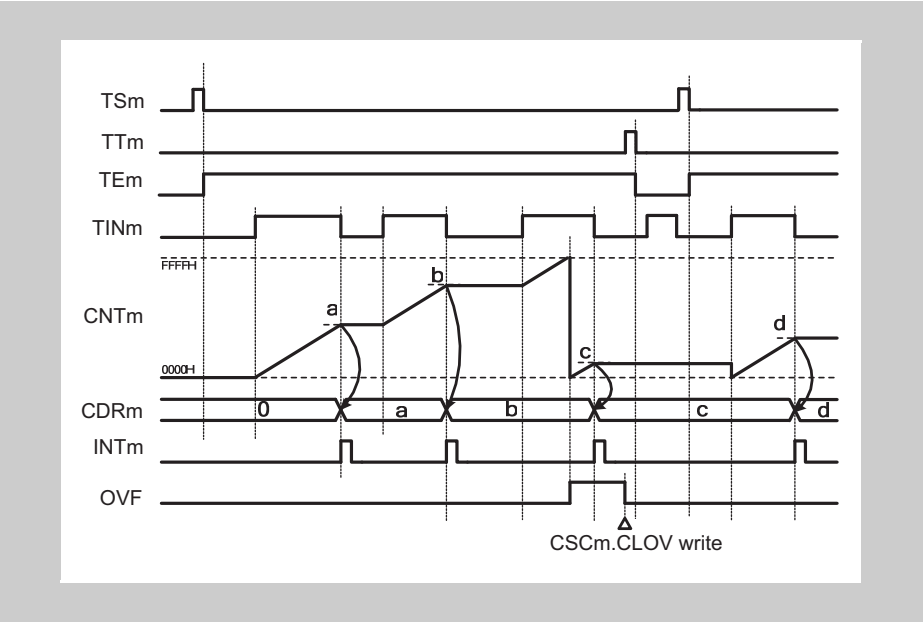


Figure 15-58 Operation stop and restart, TAUAnCMURm.TIS[1:0] = 11_B

- The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0.
- TAUAnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUAnTTINm input edges are ignored.
- The counter can be restarted by setting TAUAnTS.TSm to 1. TAUAnCNTm restarts to count from 0000_H.

15.17.5 Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

(1) Overview

Summary This function measures the cumulative width of a TAUAnTTINm input signal. An interrupt is generated if the cumulative TAUAnTTINm input width is longer than $FFFF_H$.

Prerequisites

- The operation mode must be set to Gate Count Mode, refer to *Table 15-54 "TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)" on page 805*
- TAUAnTTOUTm is not used for this function
- The value of TAUAnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. $FFFF_H$ is written to TAUAnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUAnTTINm input start edge and then continues to count down from the current value.

When the counter reaches 0000_H an interrupt is generated. $FFFF_H$ is written to TAUAnCNTm and the counter continues to count down until a TAUAnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUAnCMURm.TIS[1:0] bits:

- If TAUAnCMURm.TIS[1:0] = 10_B, the TAUAnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUAnCMURm.TIS[1:0] = 11_B, the TAUAnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

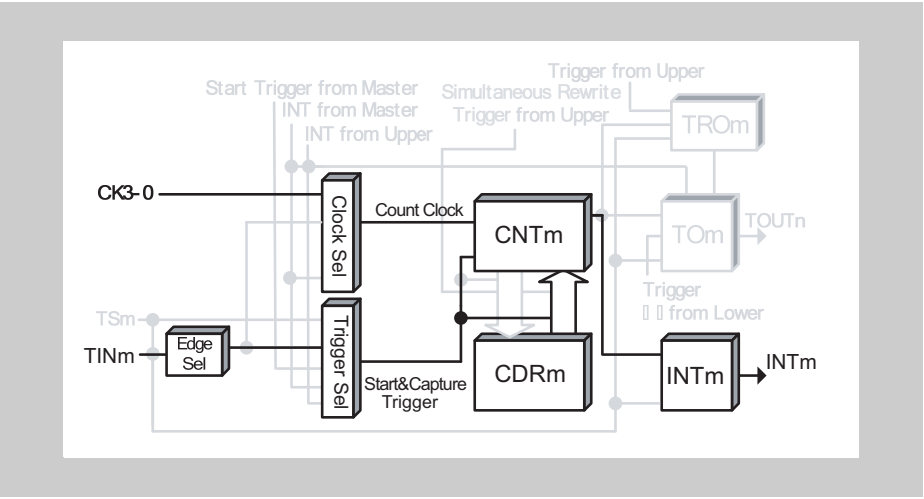


Figure 15-59 Block diagram for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TIS[1:0] = 11_B)

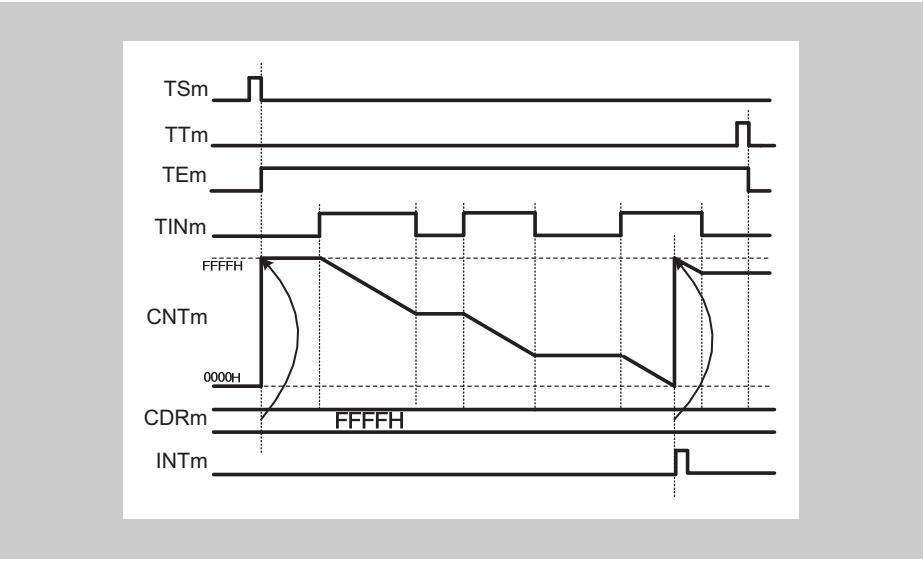


Figure 15-60 General timing diagram for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-54 TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1100: Gate Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-55 TAUAnCMURm settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection). Therefore, these registers must be set to 0.

Table 15-56 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(4) Operating procedure for Overflow Interrupt Output Function (during TAUAnTTINm input period count detection)

Table 15-57 Operating procedure for Overflow Interrupt Output Function (during TAUAnTTINm input period count detection)

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-54 "TAUAnCMORm settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)" on page 805* and *Table 15-55 "TAUAnCMURm settings for Overflow Interrupt Output Function (During TAUAnTTINm Input Period Count Detection)" on page 805* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the start edge. When a start edge is detected, TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$).
During operation	The TAUAnCNTm register can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm loads the TAUAnCDRm value ($FFFF_H$) and continues to count down. When a reverse edge of TAUAnTTINm is detected during count operation: • TAUAnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.17.6 TAUAnTTINm Input Pulse Interval Judgment Function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUAnCNTm) and the value in the channel data register (TAUAnCDRm) when a TAUAnTTINm input pulse occurs. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true.

Prerequisites
- The operation mode must be set to Judge Mode, refer to *Table 15-58 "TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Judgment Function" on page 810*
- TAUAnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value.

When a TAUAnTTINm valid edge is detected or TAUAnTS.TSm is set to 1, the function compares the current values of TAUAnCNTm and TAUAnCDRm. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true. TAUAnCNTm reloads the value of TAUAnCDRm and subsequently continues operation, regardless of the result of the comparison.

If the counter reaches 0000_H before a TAUAnTTINm valid edge is detected, TAUAnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

Conditions The TAUAnCMORm.MD0 bit specifies the type of comparison:
- If TAUAnCMORm.MD0 = 0, INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm.
- If TAUAnCMORm.MD0 = 1, INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm.

(2) Block diagram and general timing diagram

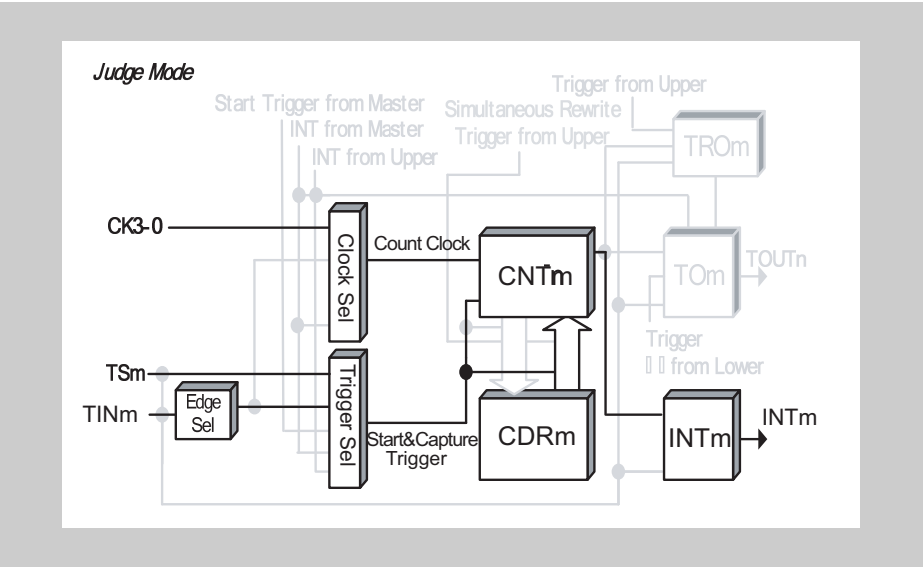


Figure 15-61 Block diagram for TAUAnTTINm Input Pulse Interval Judgment Function

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

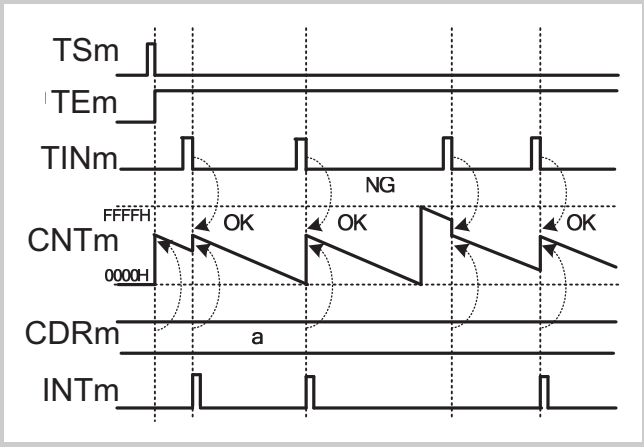


Figure 15-62 General timing diagram for TAUAnTTINm Input Pulse Interval Judgment Function

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-58 TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0001: Judge Mode
MD0	0: INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm 1: INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-59 TAUAnCMURm settings for TAUAnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Pulse Interval Judgment Function. Therefore, these registers must be set to 0.

Table 15-60 Simultaneous rewrite settings for TAUAnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for for TAUAnTTINm Input Pulse Interval Judgment Function

Table 15-61 Operating procedure for TAUAnTTINm Input Pulse Interval Judgment Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-58 "TAUAnCMORm settings for TAUAnTTINm Input Pulse Interval Judgment Function" on page 810* and *Table 15-59 "TAUAnCMURm settings for TAUAnTTINm Input Pulse Interval Judgment Function" on page 810* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value.
During operation	Detection of TAUAnTTINm edges. The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When a TAUAnTTINm input edge is detected • TAUAnCNTm reloads TAUAnCDRm and continues count operation. • TAUAnCNTm compares the values and judges the condition according to the TAUAnCMORm.MD0 setting. • If the condition is satisfied, INTTAUAnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.17.7 TAUAnTTINm Input Signal Width Judgment Function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUAnCNTm) and the value in the channel data register (TAUAnCDRm) at a valid stop edge of a TAUAnTTINm input signal. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true.

Prerequisites

- The operation mode must be set to Judge & One Count Mode, refer to *Table 15-62 "TAUAnCMORm settings for TAUAnTTINm Input Signal Width Judgment Function" on page 815*
- TAUAnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. When a valid TAUAnTTINm input start edge is detected, the current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value.

When a TAUAnTTINm valid stop edge is detected, the function compares the current values of TAUAnCNTm and TAUAnCDRm. An interrupt signal INTTAUAnIm is generated if the result of the comparison is true. The counter TAUAnCNTm retains its value until the next TAUAnTTINm valid start edge is detected, regardless of the result of the comparison.

If the counter reaches 0000_H before a valid TAUAnTTINm stop edge is detected, TAUAnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

Conditions

- The TAUAnCMORm.MD0 bit specifies the type of comparison:
  - If TAUAnCMORm.MD0 = 0, INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm.
  - If TAUAnCMORm.MD0 = 1, INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm.
- The TAUAnCMURm.TIS[1:0] bits specify the type of width measurement:
  - For high width measurement, the start edge is a rising TAUAnTTINm edge and the stop edge is a falling TAUAnTTINm edge.
  - For low width measurement, the start edge is a falling TAUAnTTINm edge and the stop edge is a rising TAUAnTTINm edge.
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

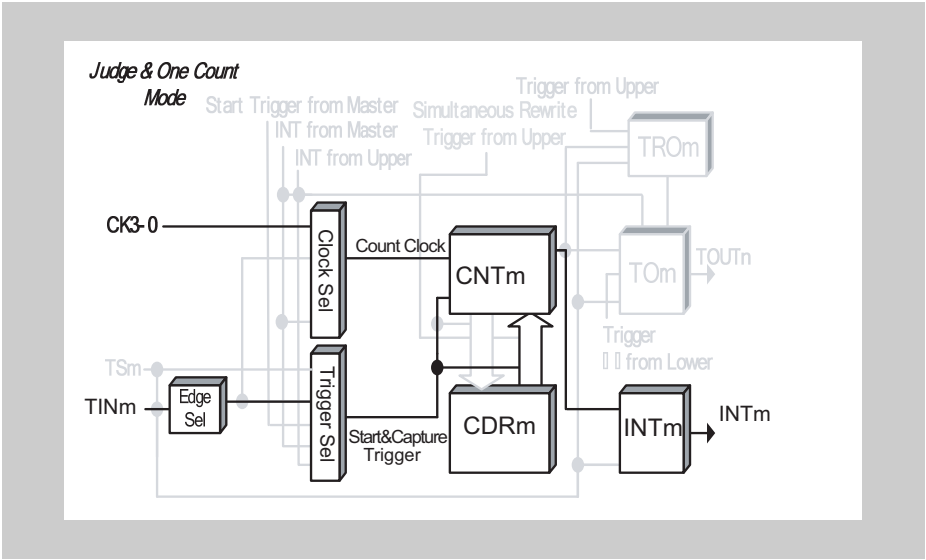


Figure 15-63 Block diagram for TAUAnTTINm Input Signal Width Judgment Function

The following settings apply to the general timing diagram:

- INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm (TAUAnCMORm.MD0 = 0)
- TAUAnTTINm valid start edge = rising edge, TAUAnTTINm valid stop edge = falling edge (TAUAnCMURm.TIS[1:0] = 11_B)

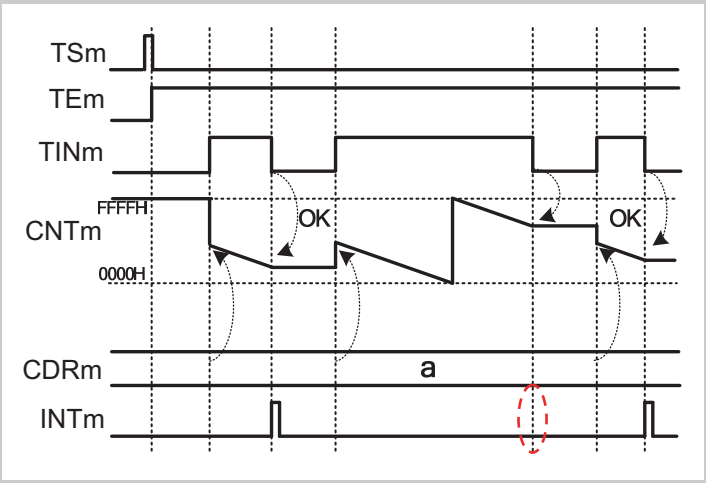


Figure 15-64 General timing diagram for TAUAnTTINm Input Signal Width Judgment Function

(3) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-62 TAUAnCMORm settings for TAUAnTTINm Input Signal Width Judgment Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUAnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0111: Judge & One Count Mode
MD0	0: INTTAUAnIm is generated when TAUAnCNTm ≤ TAUAnCDRm 1: INTTAUAnIm is generated when TAUAnCNTm > TAUAnCDRm

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-63 TAUAnCMURm settings for TAUAnTTINm Input Signal Width Judgment Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Signal Width Judgment Function. Therefore, these registers must be set to 0.

Table 15-64 Simultaneous rewrite settings for TAUAnTTINm Input Signal Width Judgment Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for TAUAnTTINm Input Signal Width Judgment Function

Table 15-65 Operating procedure for TAUAnTTINm Input Signal Width Judgment Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-62 "TAUAnCMORm settings for TAUAnTTINm Input Signal Width Judgment Function" on page 815* and *Table 15-63 "TAUAnCMURm settings for TAUAnTTINm Input Signal Width Judgment Function" on page 815* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 and TAUAnCNTm waits for detection of the TAUAnTTINm start edge. When a TAUAnTTINm start edge is detected, TAUAnCNTm loads the TAUAnCDRm value.
During operation	Detection of TAUAnTTINm edges The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When a TAUAnTTINm stop edge is detected: • TAUAnCNTm stops and waits for detection of the TAUAnTTINm start edge. • TAUAnCNTm compares the values and judges the condition according to the TAUAnCMORm.MD0. • If the condition is satisfied, INTTAUAnIm is generated Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

15.18 Independent Channel Real-Time Functions

This chapter describes functions that output the value of the TAUAnTRO.TROm bit in real time:

- *15.18.1 “Real-Time Output Function Type 1”*
- *15.18.2 “Real-Time Output Function Type 2”*

15.18.1 Real-Time Output Function Type 1

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). In this function, the interrupt is generated at certain specified intervals.

Prerequisites

- One channel using the TAUAnTTOUTm control of one of more other channels
- The operation mode of the upper channel must be set to Interval Timer Mode, refer to *Table 15-66 “TAUAnCMORm settings for Real-Time Output Function Type 1” on page 821*
- The operation mode of the lower channel(s) can be set as desired
- The channel output mode of all the channels must be set to Independent Channel Output Mode 1 with Real-Time Output, refer to *15.9 “Channel Output Modes” on page 731*
- Real-time output must be enabled for the upper channel (TAUAnTRE.TREm = 1)

Description The counter of the upper channel is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The current value of the data register of the upper channel (TAUAnCDRm) is written to the counter (TAUAnCNTm) and the counter starts to count down from this value.

When the counter of the upper channel reaches 0000_H, INTTAUAnIm is generated and TAUAnTTOUTm outputs the current value of the real-time output bit (TAUAnTRO.TROm) in all channels. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.

Conditions

- The channel which is monitored for INTTAUAnIm is specified by setting TAUAnTRC.TRCm = 1 for the corresponding channel. The TAUAnTRC.TRCm bit must be 0 for all other channels.
- If real-time output of a lower channel is disabled (TAUAnTRE.TREm = 0) or the channel itself is used as the rewrite trigger (TAUAnTRC.TRCm = 1), the value of that channel’s TAUAnTRO.TROm bit is output when INTTAUAnIm is generated by that channel.
- If real-time output of a lower channel is enabled (TAUAnTRE.TREm = 1) and TAUAnTRC.TRCm = 0, the value of that channel’s TAUAnTRO.TROm bit is output when INTTAUAnIm is generated by an upper channel.
- If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not output. For details refer to *15.11 “TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts” on page 743* .

(2) Equations

INTTAUAnIm generation cycle = count clock cycle × (TAUAnCDRm value + 1)

(3) Block diagram and general timing diagram

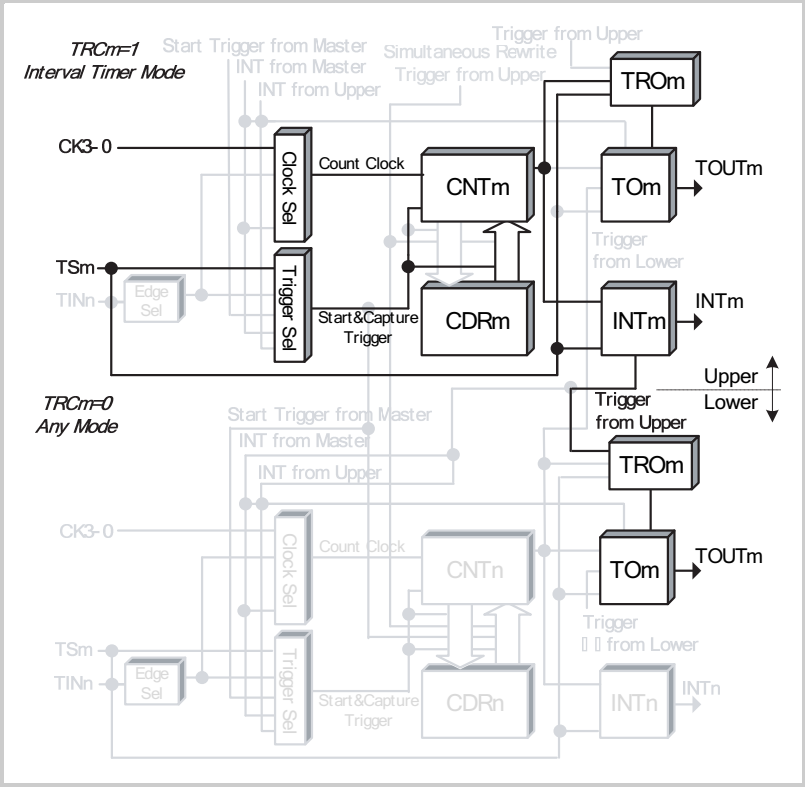


Figure 15-65 Block diagram for Real-Time Output Function Type 1

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)

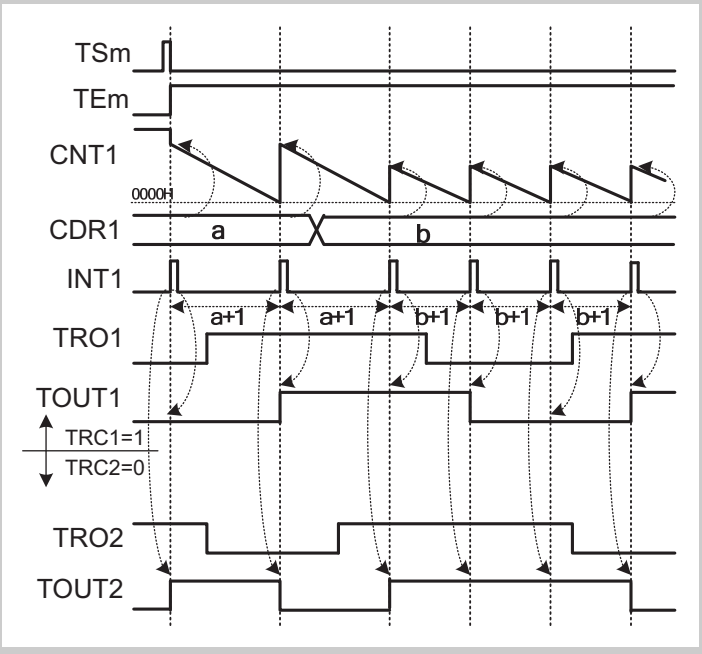


Figure 15-66 General timing diagram for Real-Time Output Function Type 1

(4) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-66 TAUAnCMORm settings for Real-Time Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-67 TAUAnCMURm settings for Real-Time Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

Table 15-68 Control bit settings for Independent Channel Output Mode 1 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite for the upper channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Real-Time Output Function Type 1. Therefore, these registers must be set to 0.

Table 15-69 Simultaneous rewrite settings for Real-Time Output Function Type 1

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

Table 15-70 Control bit settings for the lower channel in Independent Channel Output Mode 1 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TRE.TREm	0: Disables real-time output 1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m 1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite for the lower channel(s)

The simultaneous rewrite registers of the lower channel(s) can be set arbitrarily.

(6) Operating procedure for Real-Time Output Function Type 1

Table 15-71 Operating procedure for Real-Time Output Function Type 1

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-66 "TAUAnCMORm settings for Real-Time Output Function Type 1" on page 821* and *Table 15-67 "TAUAnCMURm settings for Real-Time Output Function Type 1" on page 821* Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *5 "Register settings for the lower channel(s)"* Set the value of the TAUAnCDRm register Set the channel output mode for the upper channel by setting the control bits as described in *Table 15-68 "Control bit settings for Independent Channel Output Mode 1 with Real-Time Output" on page 822* Set the channel output mode for the lower channel by setting the control bits as described in *Table 15-70 "Control bit settings for the lower channel in Independent Channel Output Mode 1 with Real-Time Output" on page 823*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm = 1 in the channel where TAUAnTRC.TRCm is set to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	[Channel where TAUAnTRC.TRCm is set to 1] TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. INTTAUAnIm is generated when TAUAnCMORm.MD0 is set to 1.
During operation	The TAUAnCDRm register and TAUAnTRO.TROm can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation. • INTTAUAnIm is generated. • TAUAnTTOUTm outputs the current value of the real-time output bit TAUAnTRO.TROm. The operation is then repeated
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(7) Specific timing diagrams

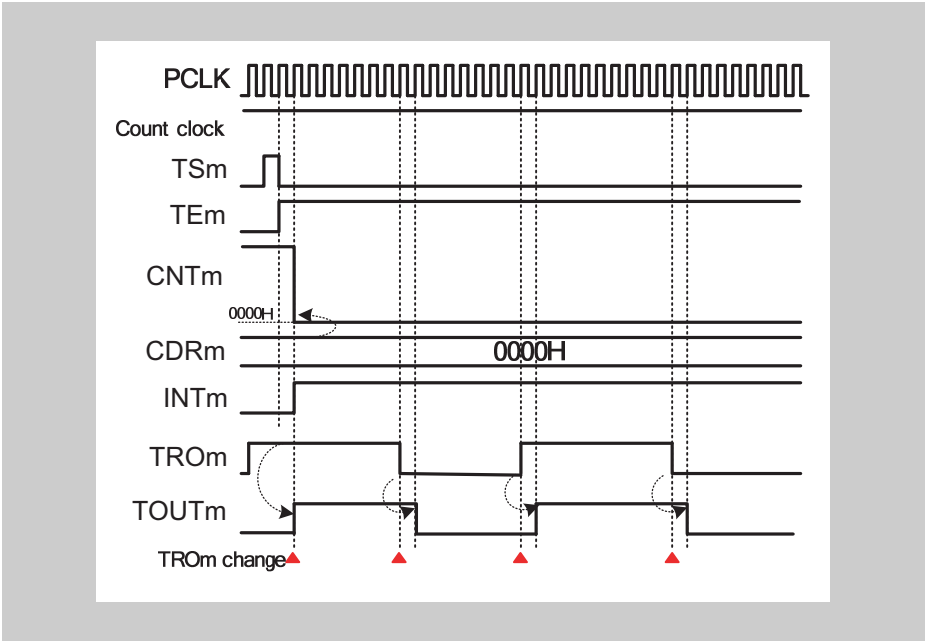


Figure 15-67 TAUAnCDRm = 0000_H, TAUAnCMORm.MD0 = 1

- The value of TAUAnTTOUTm follows the TAUAnTRO.TROm settings with once PCLK cycle delay.

15.18.2 Real-Time Output Function Type 2

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). In this function, the interrupt is generated when a valid TAUAnTTINm input edge is detected or the function starts.

Prerequisites

- Two (or more) channels
- The operation mode of the upper channel must be set to Capture Mode, refer to *Table 15-72 "TAUAnCMORm settings for Real-Time Output Function Type 2" on page 828*
- The operation mode of the lower channel(s) can be set as desired
- The channel output mode of all the channels must be set to Independent Channel Output Mode 1 with Real-Time Output, refer to *15.9 "Channel Output Modes" on page 731*
- Real-time output must be enabled for the upper channel (TAUAnTRE.TREm = 1)

Description The counter of the upper channel is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The counter of the upper channel starts to count up.

When a valid TAUAnTTINm input edge occurs on the upper channel, an interrupt is generated, and the counter restarts. The value of the counter is copied to the data register (TAUAnCDRm) but it is meaningless. TAUAnTTOUTm outputs the current value of the real-time output bit (TAUAnTRO.TROm) in all channels.

The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.

Conditions

- The channel which is monitored for INTTAUAnIm is specified by setting TAUAnTRC.TRCm = 1 for the corresponding channel. The TAUAnTRC.TRCm bit must be 0 for all other channels.
- If real-time output of a lower channel is disabled (TAUAnTRE.TREm = 0) or the channel itself is used as the rewrite trigger (TAUAnTRC.TRCm = 1), the value of that channel's TAUAnTRO.TROm bit is output when INTTAUAnIm is generated by that channel.
- If real-time output of a lower channel is enabled (TAUAnTRE.TREm = 1) and TAUAnTRC.TRCm = 0, the value of that channel's TAUAnTRO.TROm bit is output when INTTAUAnIm is generated by an upper channel.
- If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not output. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

(2) Block diagram and general timing diagram

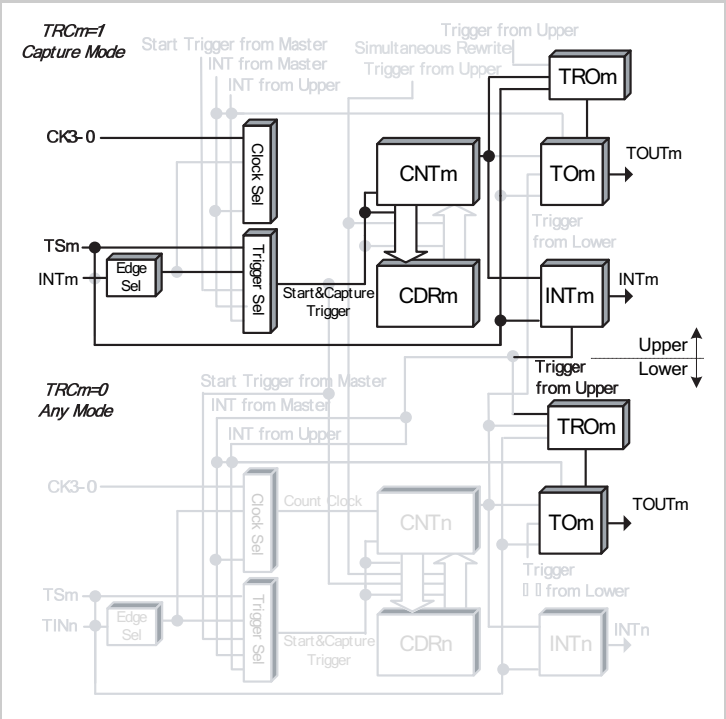


Figure 15-68 Block diagram for Real-Time Output Function Type 2

The following settings apply to the general timing diagram:

- INTTAUAnIm not output at operation start (TAUAnCMORm.MD0 = 0)

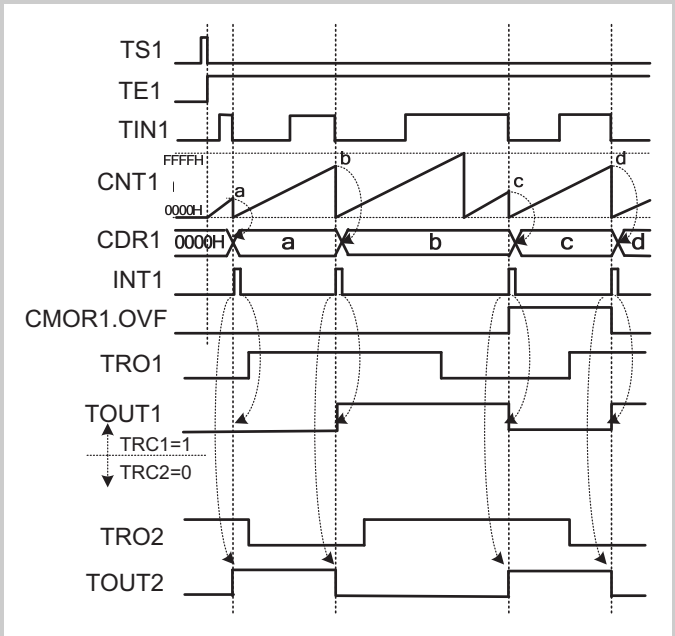


Figure 15-69 General timing diagram for Real-Time Output Function Type 2

(3) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-72 TAUAnCMORm settings for Real-Time Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Valid edge of the TAUAnTTINm input signal is the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0010: Capture Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-73 TAUAnCMURm settings for Real-Time Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode for the upper channel

Table 15-74 Control bit settings for Independent Channel Output Mode 2 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite for the upper channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Real-Time Output Function Type 2. Therefore, these registers must be set to 0.

Table 15-75 Simultaneous rewrite settings for Real-Time Output Function Type 2

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

Table 15-76 Control bit settings for lower channel in Independent Channel Output Mode 2 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	0: Disables real-time output 1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m 1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite for the lower channel(s)

The simultaneous rewrite registers of the lower channel(s) can be set arbitrarily.

(5) Operating procedure for Real-Time Output Function Type 2

Table 15-77 Operating procedure for Real-Time Output Function Type 2

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-72 "TAUAnCMORm settings for Real-Time Output Function Type 2" on page 828* and *Table 15-73 "TAUAnCMURm settings for Real-Time Output Function Type 2" on page 828* Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *4 "Register settings for the lower channel(s)"* Set the value of the TAUAnCDRm register Set the channel output mode for the upper channel by setting the control bits as described in *Table 15-74 "Control bit settings for Independent Channel Output Mode 2 with Real-Time Output" on page 829* Set the channel output mode for the lower channel by setting the control bits as described in *Table 15-76 "Control bit settings for lower channel in Independent Channel Output Mode 2 with Real-Time Output" on page 830*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the channel where TAUAnTRC.TRCm is set to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	[Channel where TAUAnTRC.TRCm is set to 1] TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. When TAUAnCMORm.MD0 is set to 1, INTTAUAnIm is generated.
During operation	TAUAnTRO.TROm can be changed at any time.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm input valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm and returns to 0000_H • INTTAUAnIm is generated If an overflow has occurred at this time: • TAUAnCSRm.OVF is set to 1. If no overflow has occurred: • TAUAnCSRm.OVF is cleared to 0. Afterwards, this procedure is repeated. TAUAnCNTm outputs the current value of the real-time output bit TAUAnTRO.TROm.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it, TAUAnCSRm.OVF, and TAUAnTTOUTm retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(6) Specific timing diagrams

(a) Operation start and stop

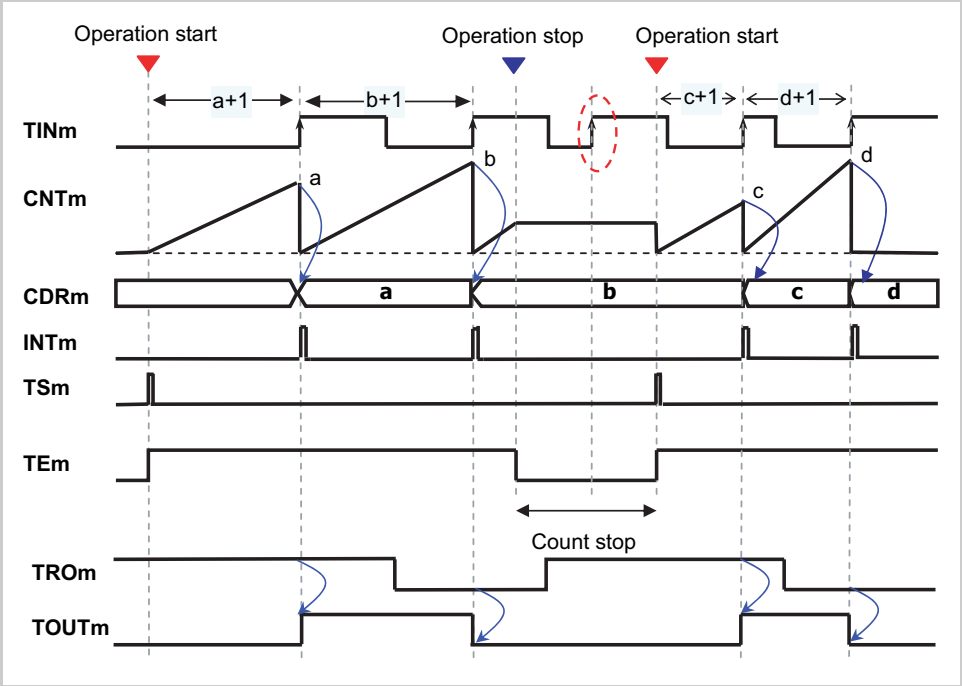


Figure 15-70 Operation start and stop, TAUAnCMORm.MD0 = 0

- When TAUAnTS.TSm is set to 1, the counter starts counting up.
- When a valid input edge is detected, the current value of the counter is written to the data register (TAUAnCDRm) and an interrupt is generated.
- TAUAnTTOUTm outputs the current value of the real-time output bit (TAUAnTRO.TROm) and the counter resets and starts to count up again.
- The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.
- If the counter is stopped (TAUAnTT.TTm is set to 1), valid input edges are ignored and no interrupt is generated.

15.19 Independent Channel Simultaneous Rewrite Functions

This chapter describes functions that carry out simultaneous rewrite:

- *15.19.1 “Simultaneous Rewrite Trigger Generation Function Type 1”*
- *15.19.2 “Simultaneous Rewrite Trigger Generation Function Type 2”*

15.19.1 Simultaneous Rewrite Trigger Generation Function Type 1

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is generated at regular intervals.

Prerequisites

- Two (or more) channels, each with simultaneous rewrite enabled (TAUAnRDE.RDEm = 1)
- The operation mode of the upper channel must be set to Interval Timer Mode, refer to *Table 15-78 "TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 837*
- The operation mode of the lower channel(s) can be set as desired
- TAUAnTTOUTm is not used for any channel in this function

Description The counters are started by setting the channel trigger bits (TAUAnTS.TSm and TAUAnTS.TSm_lower) to 1. This in turn sets TAUAnTE.TEm and TAUAnTE.TEm_lower = 1, enabling count operation. The current value of the data register buffer of the upper channel (TAUAnCDRm buf) is written to the counter (TAUAnCNTm) and the counter starts to count down from this value. The counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a counter reaches 0000_H, an interrupt is generated from the channel. The corresponding TAUAnCNTm then reloads the current TAUAnCDRm buffer value and subsequently continues operation.

If the channel where the interrupt occurs is specified as the trigger channel for simultaneous rewrite (TAUAnRDC.RDCm = 1) and is an upper channel, simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUAnRSF.RSFm = 1).

The values of the data registers are copied to the corresponding data register buffers. Each time a counter starts to count down, it reads the value in the data register buffer and counts down from this value.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel which is monitored for INTTAUAnIm is specified by setting TAUAnRDC.RDCm = 1 for the corresponding channel. The TAUAnRDC.RDCm bit must be 0 for all other channels in which simultaneous rewrite should take place.
- If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

(2) Equations

Simultaneous rewrite trigger generation cycle = count clock cycle × (TAUAnCDRm + 1)

To control simultaneous rewrite, the following condition must be satisfied:

TAUAnCDRm = [(value of TAUAnCDRm of master channel subject to simultaneous rewrite + 1) × number of interrupts] - 1

That is, the ratio of TAUAnCDRm + 1 and TAUAnCDRm_master + 1 must be an integer. This integer corresponds to the number of interrupts.

(3) Block diagram and general timing diagram

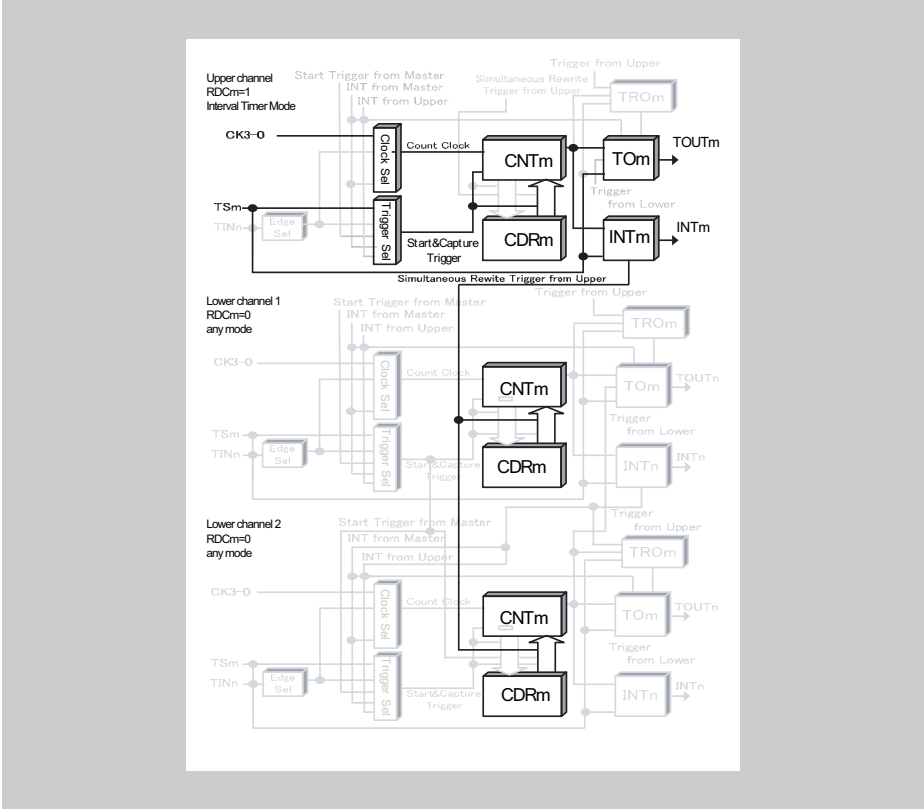


Figure 15-71 Block diagram for Simultaneous Rewrite Trigger Generation Function Type 1

The following settings apply to the general timing diagram:

- INTTAUAnIm generated at operation start (TAUAnCMORm.MD0 = 1)

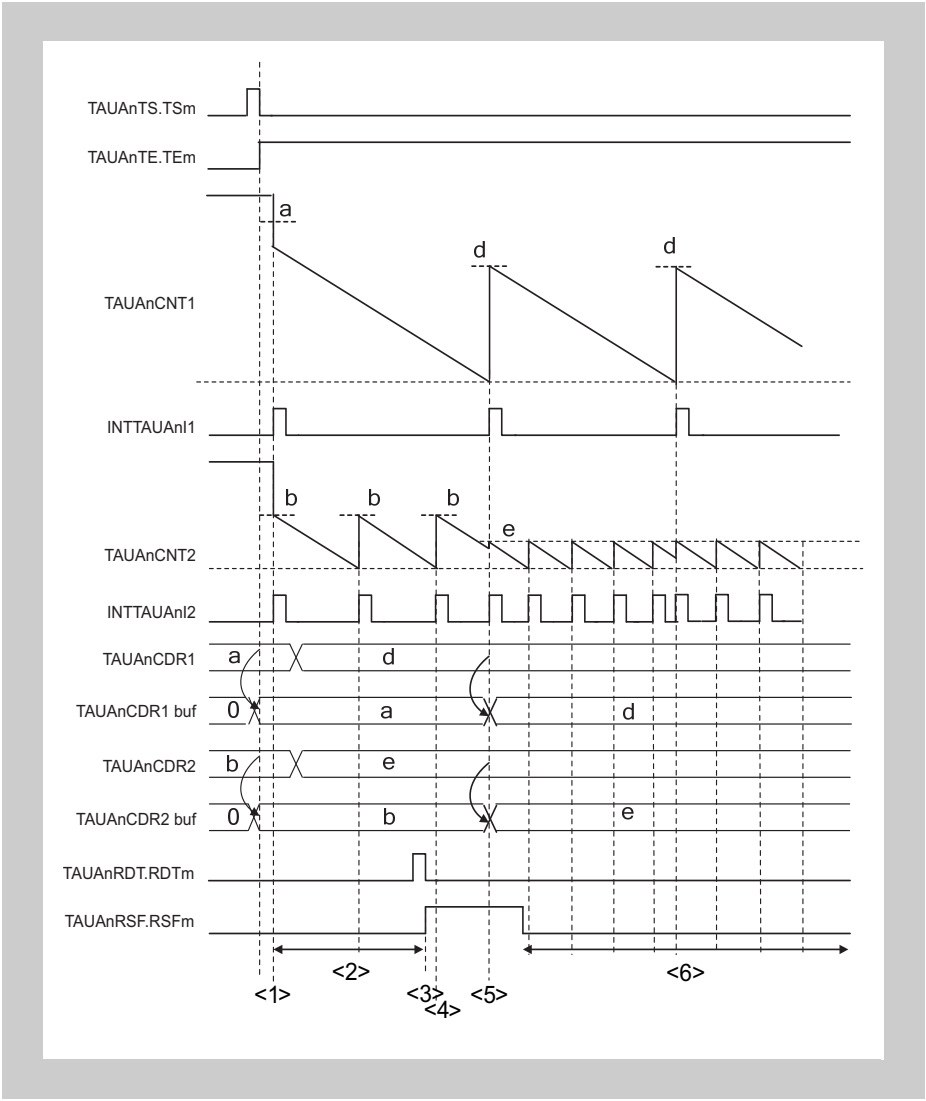


Figure 15-72 General timing diagram for Simultaneous Rewrite Trigger Generation Function Type 1

(4) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-78 TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-79 TAUAnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the upper channel

Table 15-80 Simultaneous rewrite settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	1: Channel is monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger

(5) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the lower channel(s)

Table 15-81 Simultaneous rewrite settings for the lower channel in Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous-rewrite trigger

(6) Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

Table 15-82 Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-78 "TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 837* and *Table 15-79 "TAUAnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 837* Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *5 "Register settings for the lower channel(s)"* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.MD0 = 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	TAUAnRDT.RDTm can be changed. TAUAnRSF.RSFm can be read at all times.	TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation • INTTAUAnIm is generated • TAUAnTTOUTm toggles. Simultaneous rewrite is controlled when INTTAUAnIm is generated from the channel where TAUAnRDC.RDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

15.19.2 Simultaneous Rewrite Trigger Generation Function Type 2

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is triggered by a valid TAUAnTTINm input edge or the function starting.

Prerequisites

- Two (or more) channels, each with simultaneous rewrite enabled (TAUAnRDE.RDEm = 1)
- The operation mode of the upper channel must be set to Capture Mode, refer to *Table 15-83 “TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 2” on page 843*
- The operation mode of the lower channel(s) can be set as desired
- The channel output mode of the upper channel must be set to Direct Channel Output Mode, refer to *15.9 “Channel Output Modes” on page 731*
- The channel output mode of the lower channel(s) can be set as desired

Description The counters are started by setting the channel trigger bits (TAUAnTS.TSm and TAUAnTS.TSm_lower) to 1. This in turn sets TAUAnTE.TEm and TAUAnTE.TEm_lower = 1, enabling count operation. The counter of the upper channel starts to count up, and the counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a TAUAnTTINm input pulse occurs on the upper channel, an interrupt is generated, and the counter restarts. The value of the counter is copied to the data register (TAUAnCDRm) but it is meaningless. The trigger is detected by the lower channel(s), which then also generate an interrupt.

TAUAnRDC.RDCm = 1 on the upper channel, therefore simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUAnRSF.RSFm = 1).

The values of the data registers are copied to the corresponding data register buffers.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel which is monitored for INTTAUAnIm is specified by setting TAUAnRDC.RDCm = 1 for the corresponding channel. The TAUAnRDC.RDCm bit must be 0 for all other channels.
- If the TAUAnCMORm.MD0 bit is set to 1, an interrupt is generated when the function starts. For details refer to *15.11 “TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts” on page 743* .

(2) Block diagram and general timing diagram

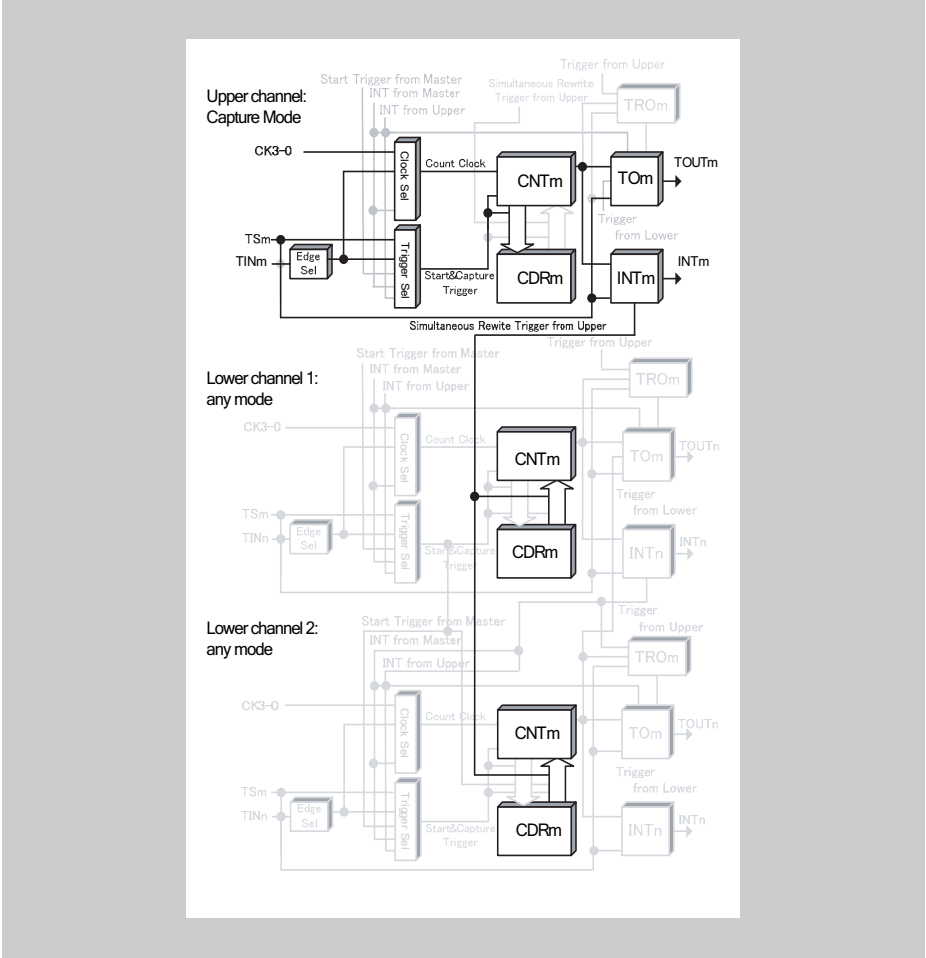


Figure 15-73 Block diagram for Simultaneous Rewrite Trigger Generation Function Type 2

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)
- Upper channel (channel 1) generates simultaneous rewrite trigger

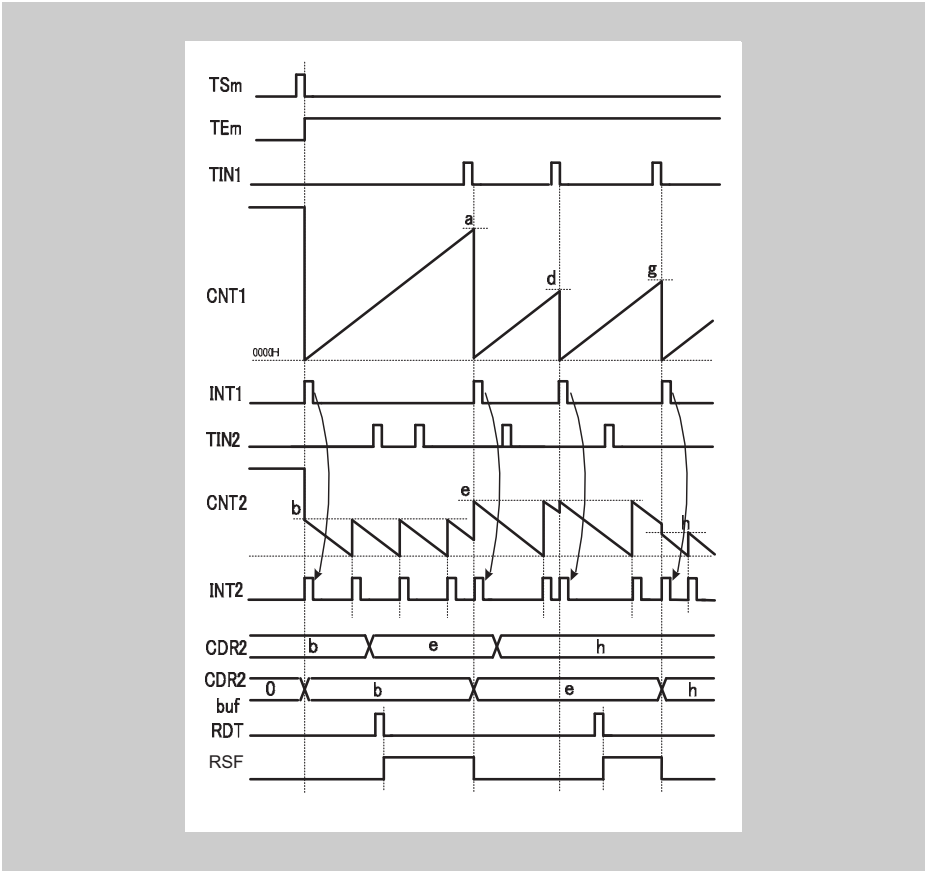


Figure 15-74 General timing diagram for Simultaneous Rewrite Trigger Generation Function Type 2

(3) Register settings for the upper channel

(a) TAUAnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-83 TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external capture trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0010: Capture Mode
MD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-84 TAUAnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 2

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the upper channel

Table 15-85 Simultaneous rewrite settings for Simultaneous Rewrite Trigger Generation Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	1: Channel is monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger

(4) Register settings for the lower channel(s)

(a) TAUAnCMORm for the lower channel(s)

The TAUAnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower channel(s)

The TAUAnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the lower channel(s)

Table 15-86 Simultaneous rewrite settings for the lower channel in Simultaneous Rewrite Trigger Generation Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous-rewrite trigger

(5) Operating procedure for simultaneous rewrite trigger generation function type 2

Table 15-87 Operating procedure for simultaneous rewrite trigger generation function type 2

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *Table 15-83 "TAUAnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 2" on page 843* and *Table 15-84 "TAUAnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 2" on page 843* Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *4 "Register settings for the lower channel(s)"* Set the value of the TAUAnCDRm register.	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm is cleared to 0000_H. INTTAUAnIm is generated when TAUAnCMORm.MD0 is set to 1.
During operation	TAUAnRDT.RDTm can be set at any time. TAUAnRSF.RSFm can be read at any time..	TAUAnCNTm counts up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm and returns to 0000_H • INTTAUAnIm is generated. Simultaneous rewrite is controlled when INTTAUAnIm is generated from the channel where TAUAnRDC.RDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.OVF retain their current values.

Restart

15.20 Independent Channel One-Phase PWM Function

This chapter describes the One-Phase PWM Function:

- *15.20.1 “One-Phase PWM Output Function”*

15.20.1 One-Phase PWM Output Function

(1) Overview

Summary This function adds dead time to a TAUAnTTINm input signal. The resulting PWM signal is output via TAUAnTTOUTm of the channel and TAUAnTTOUTm of (an) upper channel(s).

Prerequisites

- Two (or more) channels, each with dead time control enabled (TAUAnTDE.TDEm = 1)
- The operation mode of the lower channel must be set to One Count Mode, refer to *Table 15-89 "TAUAnCMORm settings for One-Phase PWM Output Function" on page 850*
- The operation mode of the upper channel(s) can be set as desired
- The channel output mode of the upper and lower channels must be set to Synchronous Channel Output Mode 2 with One-Phase PWM Output, refer to *15.9 "Channel Output Modes" on page 731*

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUAnTTINm input start edge is detected. The value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from the TAUAnCDRm value.

When the counter reaches 0000_H an interrupt is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input start edge.

TAUAnTTOUTm corresponds to the TAUAnTTINm input of the lower channel with dead time added as described in the following table:

Table 15-88 TAUAnTTOUTm to which dead time is added and state of TAUAnTTINm

TAUAnCMUR.TISm	TAUAnTOL.TOLm	TAUAnTTOUTm to which dead time is added	TAUAnTDL.TDLm	Added when TAUAnTTINm_lower is
10	0	TAUAnTTOUTm low	0	High
			1	Low
	1	TAUAnTTOUTm high	0	High
			1	Low
11	0	TAUAnTTOUTm low	0	Low
			1	High
	1	TAUAnTTOUTm high	0	Low
			1	High

Conditions

- The TAUAnCMURm.TIS[1:0] bits specify the type of width measurement:
  - For high width measurement, the start edge is a rising TAUAnTTINm edge and the stop edge is a falling TAUAnTTINm edge.
  - For low width measurement, the start edge is a falling TAUAnTTINm edge and the stop edge is a rising TAUAnTTINm edge.
- The TAUAnTDL.TDLm bit specifies the behavior of TAUAnTTOUTm for each channel when an interrupt or valid TAUAnTTINm edge is detected on the lower channel:
  - If TAUAnTDL.TDLm = 0, an interrupt is the TAUAnTTOUTm toggle trigger and a valid TAUAnTTINm edge is the TAUAnTTOUTm toggle trigger
  - If TAUAnTDL.TDLm = 1, a valid TAUAnTTINm edge is the TAUAnTTOUTm toggle trigger and an interrupt is the TAUAnTTOUTm toggle trigger
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

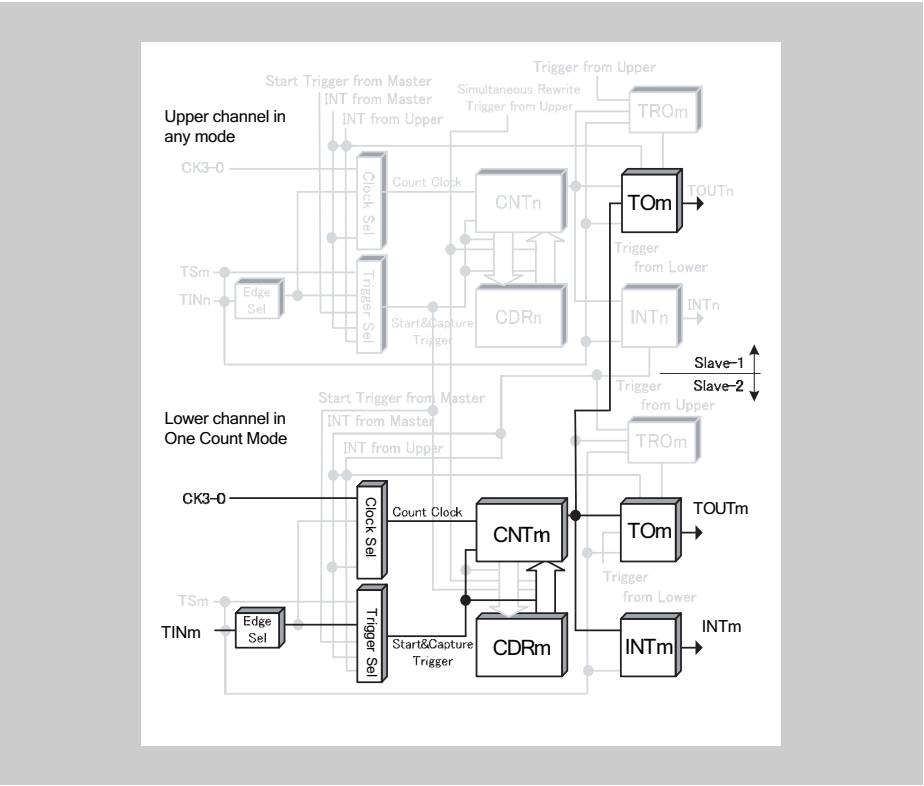


Figure 15-75 Block diagram for One-Phase PWM Output Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUAnCMURm.TIS[1:0] = 11_B)

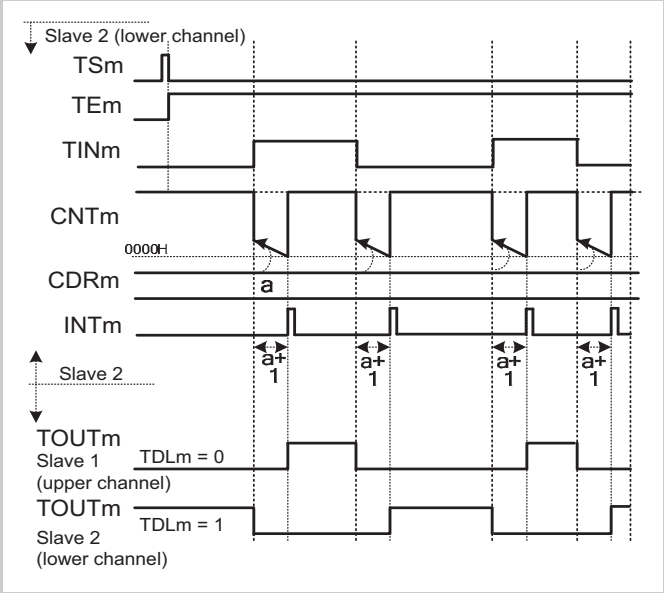


Figure 15-76 General timing diagram for One-Phase PWM Output Function

(3) Register settings for the lower channel

(a) TAUAnCMORm for the lower channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-89 TAUAnCMORm settings for One-Phase PWM Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUAnTTINm input signal is the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for the lower channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-90 TAUAnCMURm settings for One-Phase PWM Output Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode for the lower channel

Table 15-91 Control bit settings for Synchronous Channel Output Mode 2 with One-Phase PWM Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	1: Enables dead time operation
TDM.TDMm	1: Dead time is added upon detection of a TAUAnTTINm input edge from a lower odd channel
TDL.TDLm	0: An interrupt is the TAUAnTTOUTm start trigger and a valid TAUAnTTINm edge is the TAUAnTTOUTm stop trigger 1: A valid TAUAnTTINm edge is the TAUAnTTOUTm start trigger and an interrupt is the TAUAnTTOUTm stop trigger
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

(d) Simultaneous rewrite for the lower channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the One-Phase PWM Output Function. Therefore, these registers must be set to 0.

Table 15-92 Simultaneous rewrite settings for One-Phase PWM Output Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(4) Register settings for the upper channel(s)

(a) TAUAnCMORm for the upper channel(s)

The TAUAnCMORm register of the upper channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the upper channel(s)

The TAUAnCMURm register of the upper channel(s) can be set arbitrarily.

(c) Channel output mode for the upper channel(s)

Table 15-93 Control bit settings for upper channel(s) for Synchronous Channel Output Mode 2 with One-Phase PWM Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	1: Enables dead time operation
TDM.TDMm	1: Dead time is added upon detection of a TAUAnTTINm input edge from a lower odd channel
TDL.TDLm	0: An interrupt on the lower channel is the TAUAnTTOUTm start trigger and a valid TAUAnTTINm edge on the lower channel is the TAUAnTTOUTm stop trigger 1: A valid TAUAnTTINm edge on the lower channel is the TAUAnTTOUTm start trigger and an interrupt on the lower channel is the TAUAnTTOUTm stop trigger
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

(d) Simultaneous rewrite for the upper channel(s)

The simultaneous rewrite registers of the upper channel(s) can be set arbitrarily.

(5) Operating procedure for One-phase PWM Output Function

Table 15-94 Operating procedure for One-phase PWM Output Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers for the lower channel as described in *Table 15-89 "TAUAnCMORm settings for One-Phase PWM Output Function" on page 850* and *Table 15-90 "TAUAnCMURm settings for One-Phase PWM Output Function" on page 850* Set the TAUAnCMORm register and TAUAnCMURm registers for the upper channel as described in *4 "Register settings for the upper channel(s)"* Set the value of the TAUAnCDRm register Set the channel output mode for the upper and lower channel by setting the control bits as described in *Table 15-91 "Control bit settings for Synchronous Channel Output Mode 2 with One-Phase PWM Output" on page 851*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTOE.TOEm (slave channels 1 and 2) to 1 (at operation restart only). Set TAUAnTS.TSm = for slave channel 2. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUAnTTINm start edge	TAUAnTE.TEm is set to 1 (slave channel 2) and TAUAnCNTm waits for TAUAnTTINm start edge detection. TAUAnCNTm loads the TAUAnCDRm value.
During operation	The value of the TAUAnCDRm register can be changed at any time. The TAUAnCNTm register can be read at any time.	TAUAnCNTm of slave channel 2 counts down. When it reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm stops counting TAUAnTTOUTm toggles low when a valid TAUAnTTINm edge is detected and toggles high when an INTTAUAnIm is generated. The phase to which the dead time is added is specified by the value of the TAUAnTDL.TDLm bit.
Stop operation	Set TAUAnTT.TTm = 1 for slave channel 2. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

15.21 Other Independent Channel Functions

This chapter describes a function that generates an interrupt when a certain number of TAUAnTTINm pulses has occurred, a function that divides the frequency of TAUAnTTINm, and a function that measures the duration between the function start and a TAUAnTTINm input signal:

- *15.21.1 “External Event Count Function”*
- *15.21.2 “Clock Divide Function”*
- *15.21.3 “TAUAnTTINm Input Position Detection Function”*

15.21.1 External Event Count Function

(1) Overview

Summary This function is used as an event timer. It generates an interrupt (INTTAUAnIm) when a specific number of TAUAnTTINm input pulses has occurred.

Prerequisites
- The operation mode must be set to Event Count Mode, refer to *Table 15-95 "TAUAnCMORm settings for External Event Count Function" on page 857*
- TAUAnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. When the counter starts, the current value of TAUAnCDRm is written to TAUAnCNTm.

When a valid TAUAnTTINm input edge is detected, the value of TAUAnCNTm reduces by 1. TAUAnCNTm retains this value until a valid TAUAnTTINm input edge is detected or the counter is restarted.

When the valid edge is detected for the (TAUAnCDRm + 1) times, INTTAUAnIm is generated. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm stops and retains its value. The counter can be restarted by setting TAUAnTS.TSm to 1. The counter can also be restarted without stopping it first (forced restart) by setting TAUAnTS.TSm to 1 during operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUAnCMURm.TIS[1:0] bits:
- If TAUAnCMURm.TIS[1:0] = 00_B, the falling edges are counted.
- If TAUAnCMURm.TIS[1:0] = 01_B, the rising edges are counted.
- If TAUAnCMURm.TIS[1:0] = 10_B, the rising and falling edges are counted.

(2) Equations

Number of valid edges,
detected before INTTAUAnIm is generated = TAUAnCDRm + 1

(3) Block diagram and general timing diagram

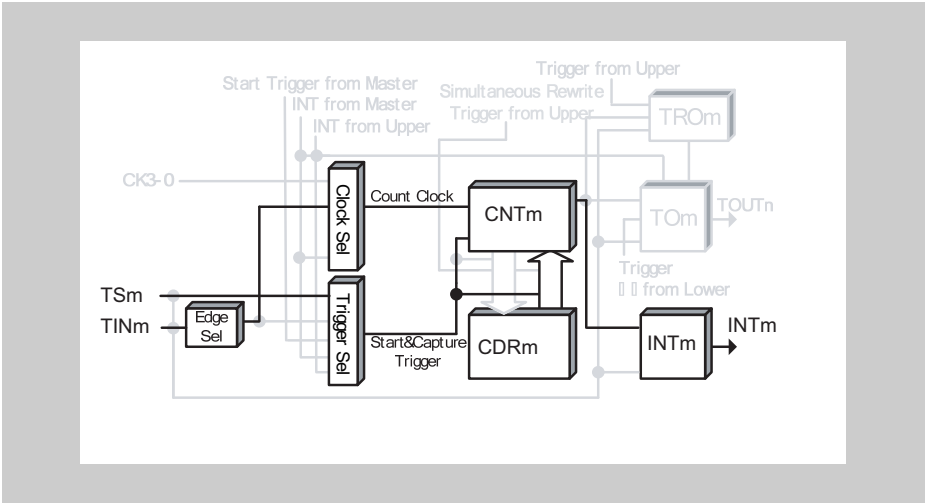


Figure 15-77 Block diagram for External Event Count Function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUAnCMURm.TIS[1:0] = 01_B)

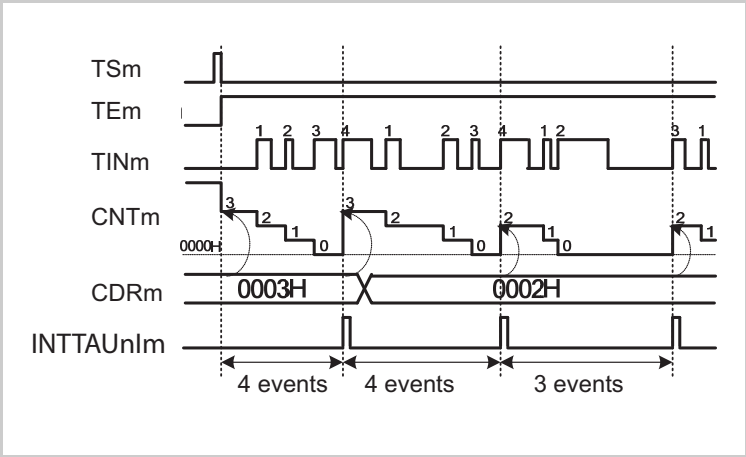


Figure 15-78 General timing diagram for External Event Count Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-95 TAUAnCMORm settings for External Event Count Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	01: Valid TAUAnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-96 TAUAnCMURm settings for External Event Count Function

Bit name	Setting
TIS[1:0]	00: Falling edge 01: Rising edge 10: Rising and falling edge

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the External Event Count Function. Therefore, these registers must be set to 0.

Table 15-97 Simultaneous rewrite settings for External Event Count Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for External Event Count Function

Table 15-98 Operating procedure for External Event Count Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-95 "TAUAnCMORm settings for External Event Count Function" on page 857* and *Table 15-96 "TAUAnCMURm settings for External Event Count Function" on page 857* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value and waits for detection of the TAUAnTTINm input edge.
During operation	Detection of TAUAnTTINm edges. The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at any tme.	TAUAnCNTm performs count-down operation each time a TAUAnTTINm input edge is detected. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation • INTTAUAnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and retains its current value.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H

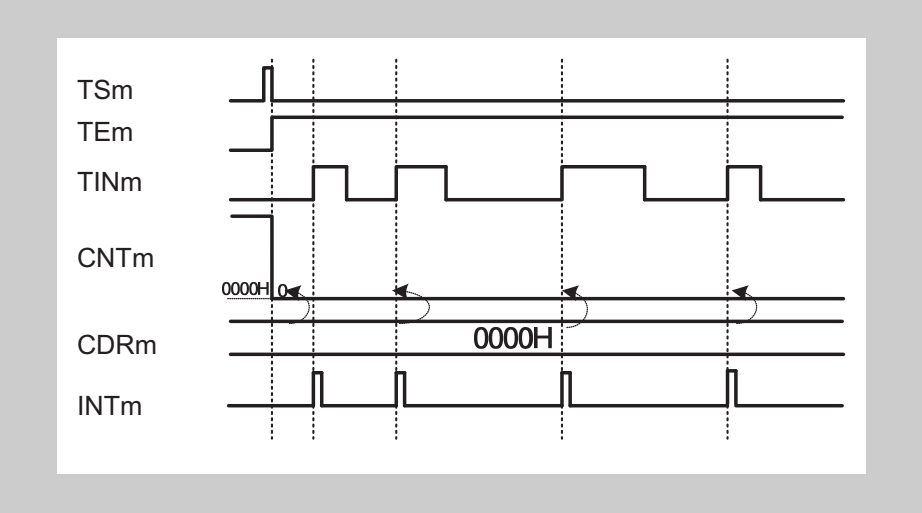


Figure 15-79 TAUAnCDRm = 0000H, TAUAnCMURm.TIS[1:0] = 01

- If 0000_H = TAUAnCDRm, 0000_H is written to TAUAnCNTm every time a valid TAUAnTTINm input edge is detected.

  This means, INTTAUAnIm is generated every time a valid TAUAnTTINm input edge is detected.

(b) Operation stop and restart

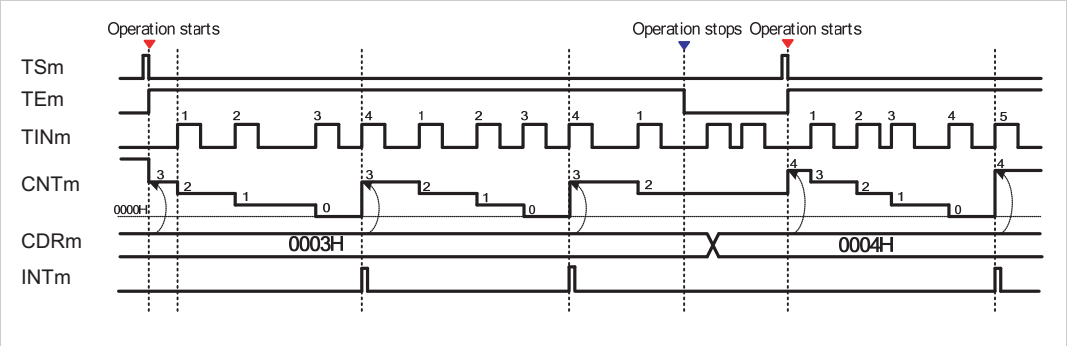


Figure 15-80 Operation stop and restart, TAUAnCMURm.TIS[1:0] = 01

- The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0.
- TAUAnCNTm stops and the current value is retained. TAUAnTTINm continues and TAUAnCNTm ignores the valid edge.
- The counter can be restarted by setting TAUAnTS.TSm to 1. TAUAnCNTm loads the TAUAnCDRm value and restarts count operation.

(c) Forced restart

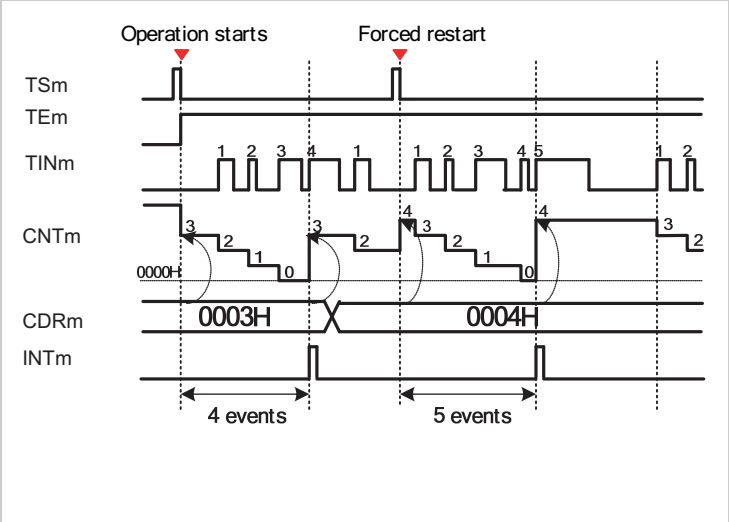


Figure 15-81 Forced restart, TAUAnCMURm.TIS[1:0] = 01

A forced restart applies a change to TAUAnCDRm immediately.

- The counter can be restarted (without stopping it first), by setting TAUAnTS.TSm to 1 during operation.
- The value of TAUAnCDRm is written to TAUAnCNTm and the counter awaits the next valid TAUAnTTINm input edge.

15.21.2 Clock Divide Function

(1) Overview

Summary This function is used as a frequency divider. The frequency of the input signal TAUAnTTINm is divided by a factor related to TAUAnCDRm, and the resulting signal is output to TAUAnTTOUTm.

Prerequisites

- TAUAnTTINm must have a fixed frequency
- The operation mode must be set to Interval Timer Mode, refer to *Table 15-99 "TAUAnCMORm settings for Clock Divide Function" on page 864*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *15.9 "Channel Output Modes" on page 731*

Description The counter is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value, using TAUAnTTINm as the count clock.

When the counter value reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the function starts to count down.

The counter can be stopped by setting TAUAnTT.TTm = 1, which in turn sets TAUAnTE.TEm = 0. TAUAnCNTm and TAUAnTTOUTm stop but retain their values. The function can be restarted by setting TAUAnTS.TSm = 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUAnTS.TSm = 1 during operation.

Conditions If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUAnTTOUTm does not toggle. This results in an inverted TAUAnTTOUTm signal compared to when TAUAnCMORm.MD0 is set to 1. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

Note The input TAUAnTTINm is sampled at the frequency of the operation clock, specified by TAUAnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUAnTTOUTm has an error of ± 1 operation clock cycle.

(2) Equations

- When rising edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / [(TAUAnCDRm + 1) x 2]
- When falling edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / [(TAUAnCDRm + 1) x 2]
- When rising and falling edge detection is selected:
  TAUAnTTOUTm frequency = TAUAnTTINm frequency / (TAUAnCDRm + 1)

(3) Block diagram and general timing diagram

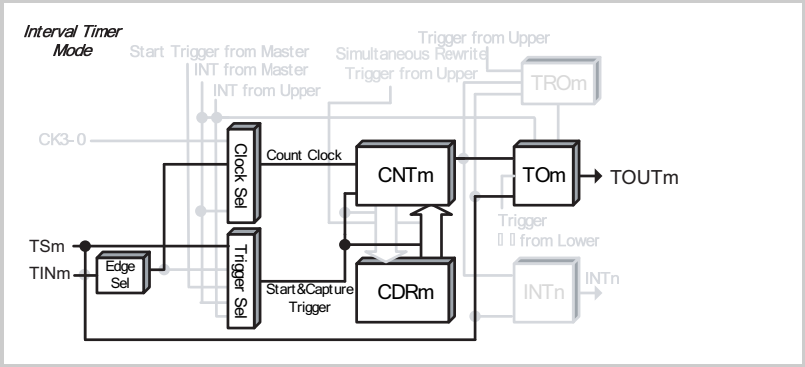


Figure 15-82 Block diagram for Clock Divide Function

The following settings apply to the general timing diagram:

- INTTAUAnIm generated at operation start (TAUAnCMORm.MD0 = 1)
- Rising edge detection (TAUAnCMURm.TIS[1:0] = 01_B)

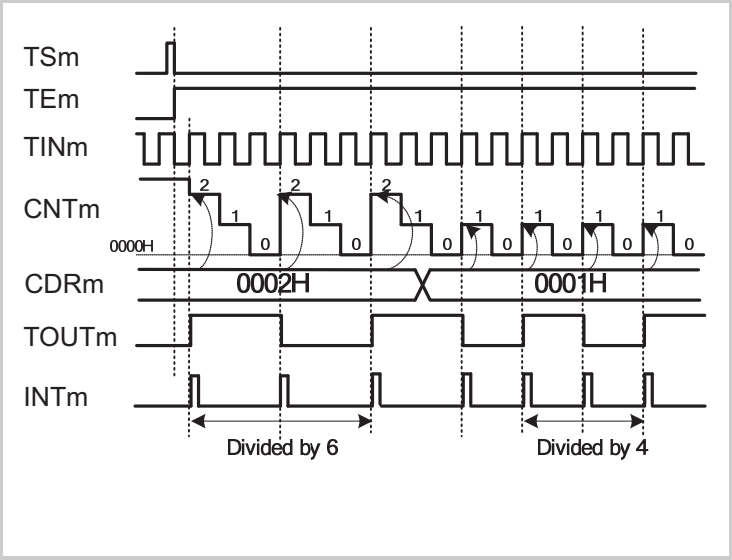


Figure 15-83 General timing diagram for Clock Divide Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-99 TAUAnCMORm settings for Clock Divide Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	01: Valid TAUAnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-100 TAUAnCMURm settings for Clock Divide Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 15-101 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 15-15 “Channel output modes” on page 732* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Clock Divide Function. Therefore, these registers must be set to 0.

Table 15-102 Simultaneous rewrite settings for Clock Divide function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for Clock Divide Function

Table 15-103 Operating procedure for Clock Divide Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-99 "TAUAnCMORm settings for Clock Divide Function" on page 864* and *Table 15-100 "TAUAnCMURm settings for Clock Divide Function" on page 864* Set the value of the TAUAnCDRm register Set the channel output mode by setting the control bits as described in *Table 15-101 "Control bit settings for Independent Channel Output Mode 1" on page 865*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. TAUAnCNTm loads the TAUAnCDRm value. When TAUAnCMORm.MD0 is set to 1, INTTAUAnIm is generated and TAUAnTTOUTm toggles.
During operation	The value of TAUAnCDRm can be changed at any time. The TAUAnCNTm register can be read at all times.	When a TAUAnTTINm input edge is detected, TAUAnCNTm counts down. When the counter reaches 0000_H: • TAUAnCNTm reloads the TAUAnCDRm value and continues count operation • INTTAUAnIm is generated • TAUAnTTOUTm toggles. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnTTOUTm retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm = 0000_H

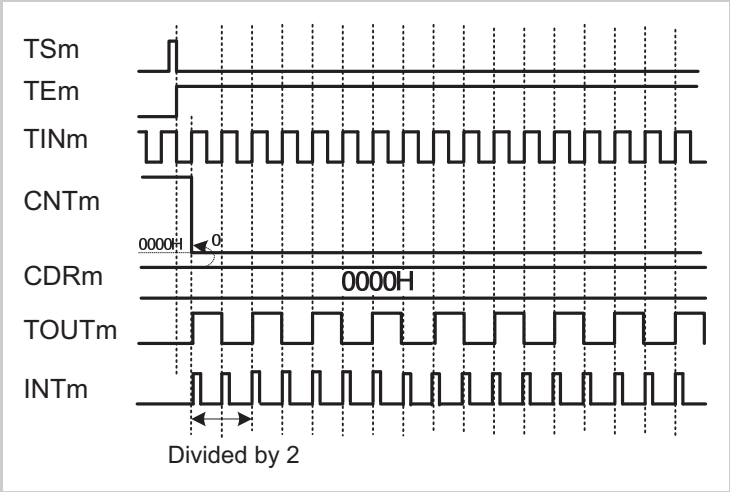


Figure 15-84 TAUAnCDRm = 0000_H, TAUAnCMORm.MD0 = 1, TAUAnCMURm.TIS[1:0] = 01

- If TAUAnCDRm is 0000_H, TAUAnCNTm is also always 0000_H.
- INTTAUAnIm is generated every count clock, resulting in TAUAnTTOUTm toggling every count clock.

(b) Restart

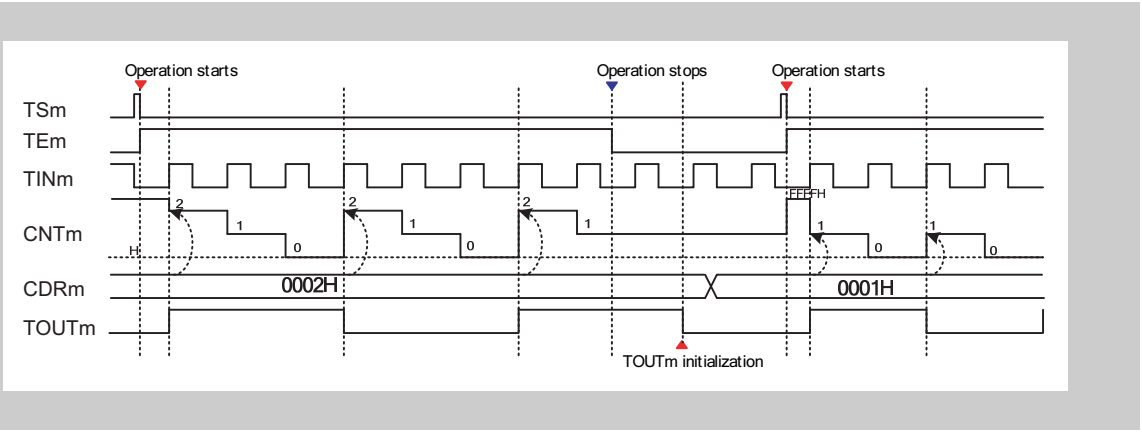


Figure 15-85 Restart, TAUAnCMORm.MD0 = 1, TAUAnCMURm.TIS[1:0] = 01

To reset the value of TAUAnTTOUTm:

- Set TAUAnTOE.TOEm = 0 when the counter is stopped (TAUAnTE.TEm = 0)
- Then write the either 0 or 1 to TAUAnTO.TOm to set the new start value of TAUAnTTOUTm

(c) Forced restart

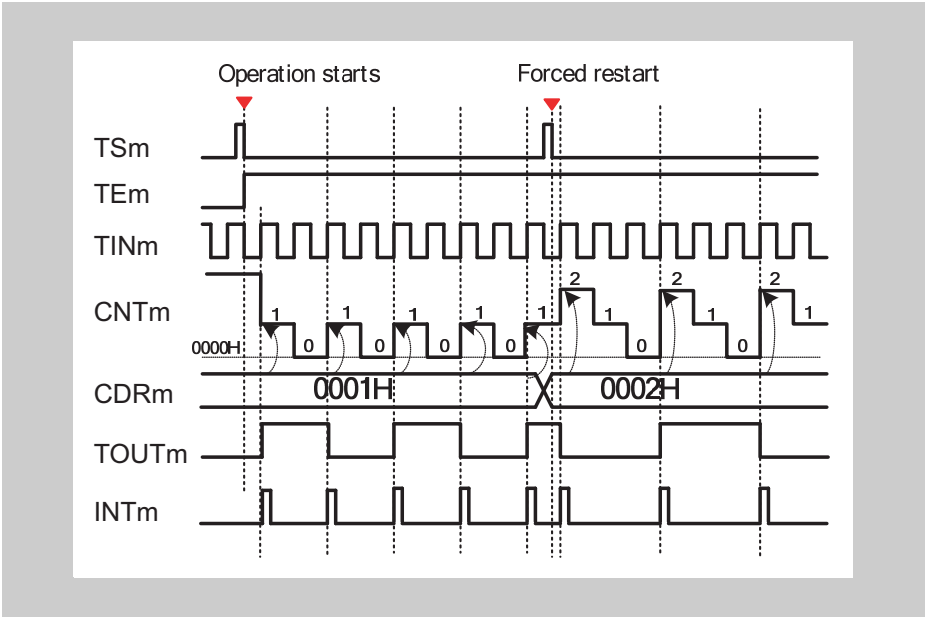


Figure 15-86 Forced restart, TAUAnCMORm.MD0 = 1, TAUAnCMURm.TIS[1:0] = 01

- The counter can be forcibly restarted (without stopping it first) by setting TAUAnTS.TSm = 1 during operation.
- The value of TAUAnCDRm is written to TAUAnCNTm and the count operation restarts.
- TAUAnTTOUTm restarts at the same level as before the forced restart.

15.21.3 TAUAnTTINm Input Position Detection Function

(1) Overview

Summary This function measures the duration between the function start and a TAUAnTTINm input signal.

Prerequisites
- The operation mode must be set to Count Capture Mode, refer to *Table 15-104 "TAUAnCMORm settings for TAUAnTTINm Input Position Detection Function" on page 871*
- TAUAnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The counter starts to count from 0000_H. When a valid TAUAnTTINm input stop edge is detected, the current TAUAnCNTm value is written to TAUAnCDRm and an interrupt (INTTAUAnIm) is generated. The counter continues to count from the current value until the next valid TAUAnTTINm input edge is detected.

When the counter reaches $FFFF_H$, the bit TAUAnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUAnCSRm.OVF is reset by the CPU by TAUAnCSCm.CLOV = 1.

Note The input TAUAnTTINm is sampled at the frequency of the operation clock, specified by the TAUAnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUAnTTOUTm has an error of ± 1 operation clock cycle.

Conditions If the TAUAnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *15.11 "TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts" on page 743* .

(2) Equations

Function duration at a TAUAnTTINm input pulse =

count clock cycle × [($FFFF_H$+1 × TAUAnCSRm.OVF) + (TAUAnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

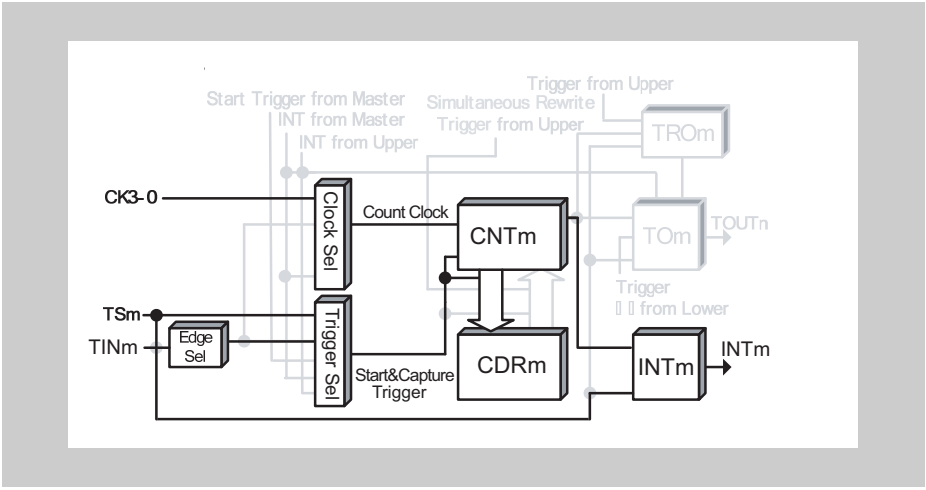


Figure 15-87 Block diagram for TAUAnTTINm Input Position Detection Function

The following settings apply to the general timing diagram:

- INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

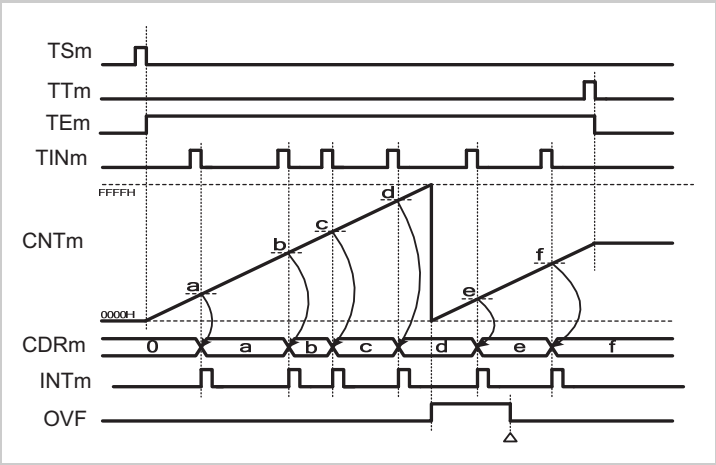


Figure 15-88 General timing diagram for TAUAnTTINm Input Position Detection Function

(4) Register settings

(a) TAUAnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-104 TAUAnCMORm settings for TAUAnTTINm Input Position Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the external capture trigger
COS[1:0]	01: Overflow (TAUAnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1011: Count Capture Mode
MD0	0: INTTAUAnIm not generated at operation start 1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-105 TAUAnCMURm settings for TAUAnTTINm Input Position Detection Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the TAUAnTTINm Input Position Detection Function. Therefore, these registers must be set to 0.

Table 15-106 Simultaneous rewrite settings for TAUAnTTINm Input Position Detection Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUAnTTINm Input Position Detection Function

Table 15-107 Operating procedure for TAUAnTTINm Input Position Detection Function

	Operation	Status of TAUAn
Initial channel setting	Set the TAUAnCMORm register and TAUAnCMURm registers as described in *Table 15-104 "TAUAnCMORm settings for TAUAnTTINm Input Position Detection Function" on page 871* and *Table 15-105 "TAUAnCMURm settings for TAUAnTTINm Input Position Detection Function" on page 871* Set the value of the TAUAnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm to 1. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is set to 1 and the counter starts. INTTAUAnIm is generated when TAUAnCMORm.MD0 is set to 1.
During operation	The TAUAnCMURm.TIS[1:0] bits can be changed at any time. The TAUAnCDRm and TAUAnCSRm registers can be read at any time.	TAUAnCNTm starts to count up from 0000_H. When a TAUAnTTINm valid edge is detected: • TAUAnCNTm transfers (captures) its value to TAUAnCDRm • TAUAnTTINm is output. • The counter value is not cleared to 0000_H and TAUAnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUAnTT.TTm to 1. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm stops and both it and TAUAnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

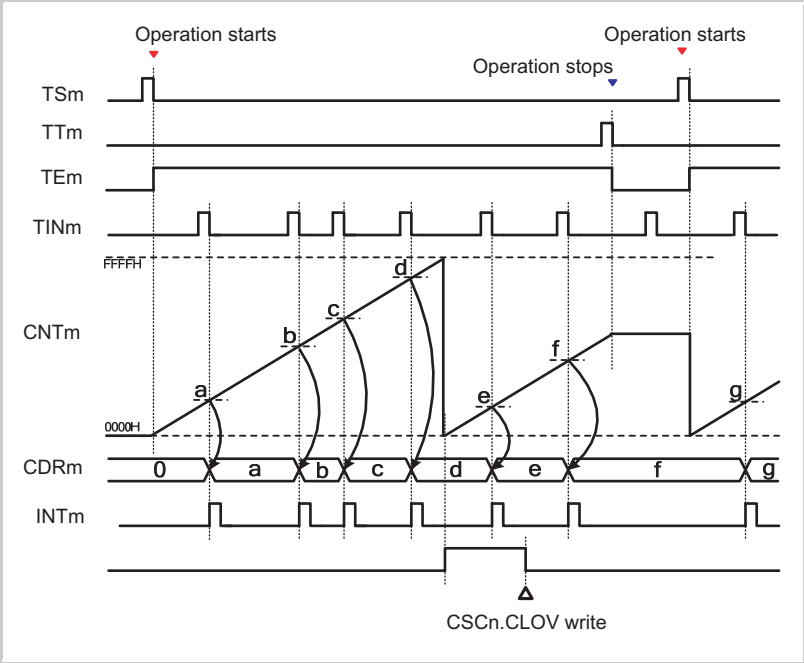


Figure 15-89 Operation stop and restart, TAUAnCMORm.MD0 = 0, TAUAnCMURm.TIS[1:0] = 00

- The counter can be stopped by setting TAUAnTT.TTm to 1, which in turn sets TAUAnTE.TEm to 0.
- TAUAnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUAnTTINm input edges are ignored.
- The counter can be restarted by setting TAUAnTS.TSm to 1. TAUAnCNTm restarts to count from 0000_H.

15.22 Synchronous Channel Operation Functions

This section lists all the synchronous channel operation functions provided by the Timer Array Unit A. For a general overview of synchronous channel operation, see *15.4 “Functional Description” on page 712* .

15.23 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes functions that generate PWM signals at regular intervals, that can be reset by a TAUAnTTINm input, with and without dead time.

- *15.23.1 “PWM Output Function”*
- *15.23.2 “Trigger Start PWM Output Function”*
- *15.23.3 “Delay Pulse Output Function”*
- *15.23.4 “AD Conversion Trigger Output Function Type 1”*

15.23.1 PWM Output Function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUAnTTOUTm to be set. The pulse cycle is set in the master channel. The pulse width is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-108 "TAUAnCMORm settings for the master channel of the PWM Output Function" on page 879*
- The operation mode of the slave channel(s) must be set to One Count Mode, refer to *Table 15-111 "TAUAnCMORm settings for the slave channel of the PWM Output Function" on page 881*
- TAUAnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 1 (*15.9 "Channel Output Modes" on page 731*)

Description The counters are started by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm is written to TAUAnCNTm and the counters start to count down from these values. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (slave) toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUAnIm is generated. The counter reloads the TAUAnCDRm value and counts down.

- Slave channel(s)

  The INTTAUAnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUAnCDRm (slave) is written to TAUAnCNTm (slave) and the counter starts to count down from this value. The TAUAnTTOUTm signal is set.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Note If a forced restart is executed during operation, TAUAnTTOUTm is not output as a PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUAnCDRm (slave) / (TAUAnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

  TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

  TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

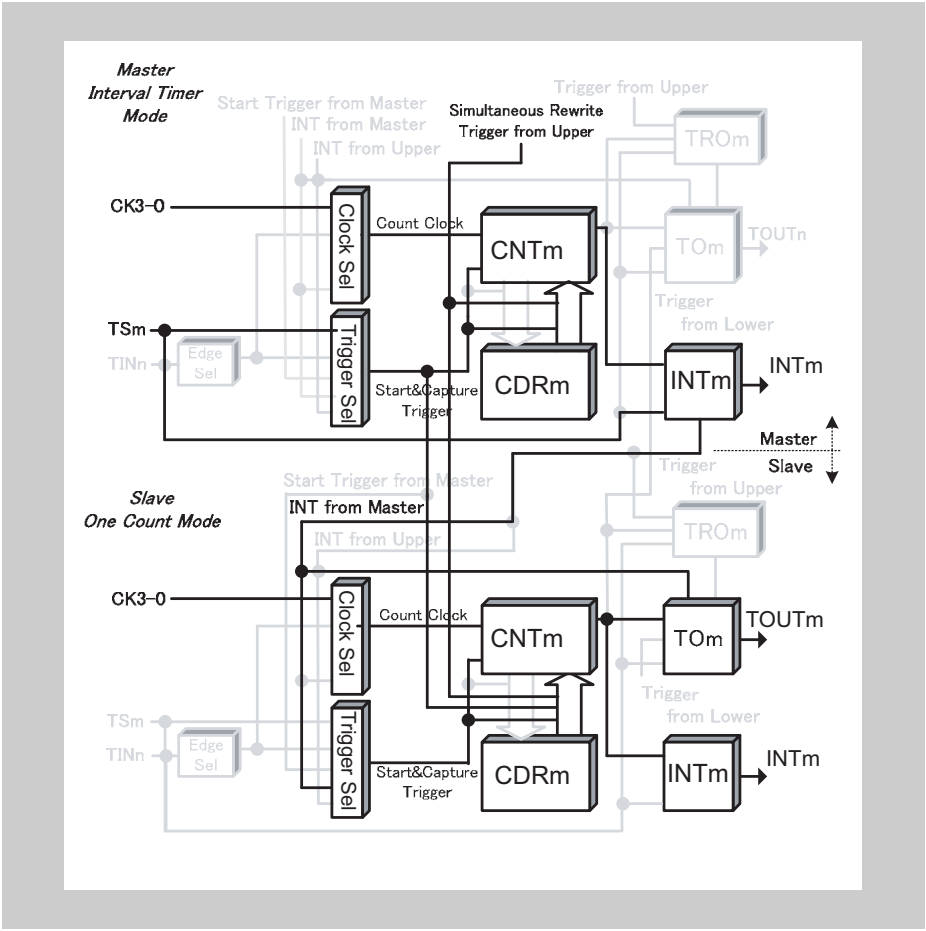


Figure 15-90 Block diagram for PWM Output Function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUAnTOL.TOLm = 0)

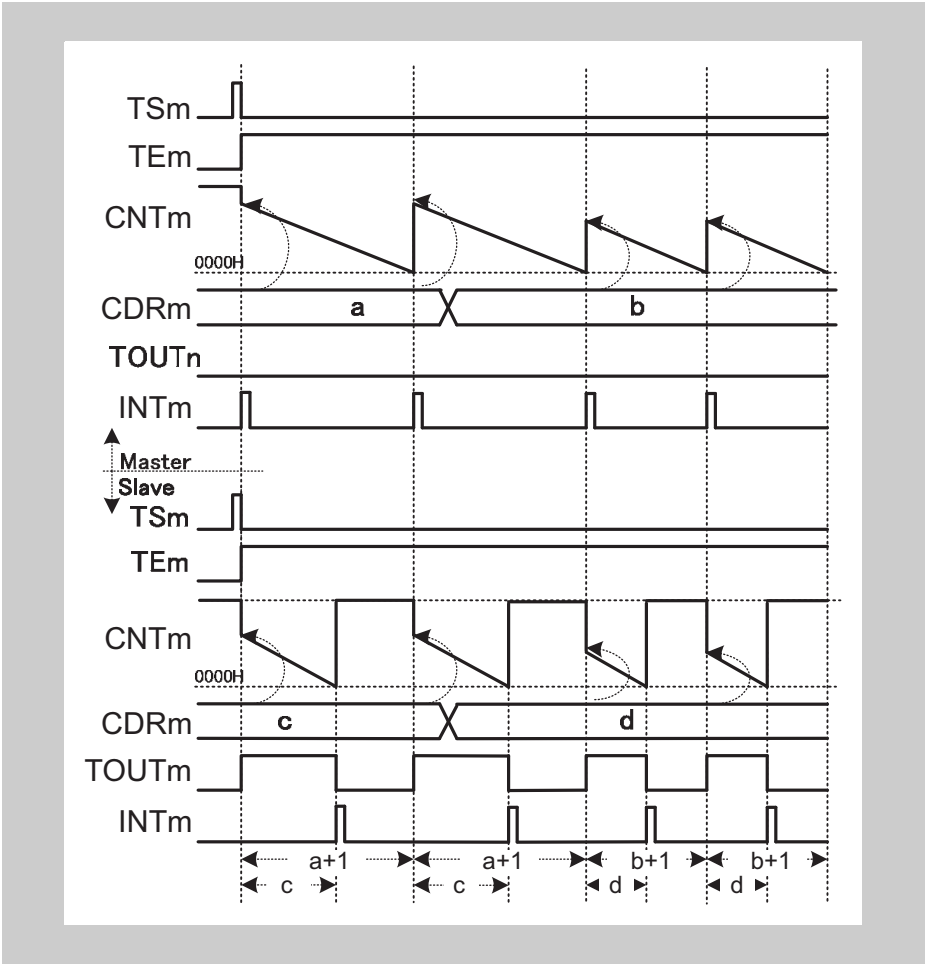


Figure 15-91 General timing diagram for PWM Output Function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUAnCDRm, whereas for the master channel the interval is the corresponding TAUAnCDRm + 1.

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-108 TAUAnCMORm settings for the master channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-109 TAUAnCMURm settings for the master channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-110 Simultaneous rewrite settings for the master channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUAnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-111 TAUAnCMORm settings for the slave channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-112 TAUAnCMURm settings for the slave channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 15-113 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-114 Simultaneous rewrite settings for the slave channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for PWM Output Function

Table 15-115 Operating procedure for PWM Output Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 879* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 881* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (slave) is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation • TAUAnCNTm (slave) reloads the TAUAnCDRm value and counts down • TAUAnTTOUTm (slave) is set When TAUAnCNTm (slave) reaches 0000_H: • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is reset
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

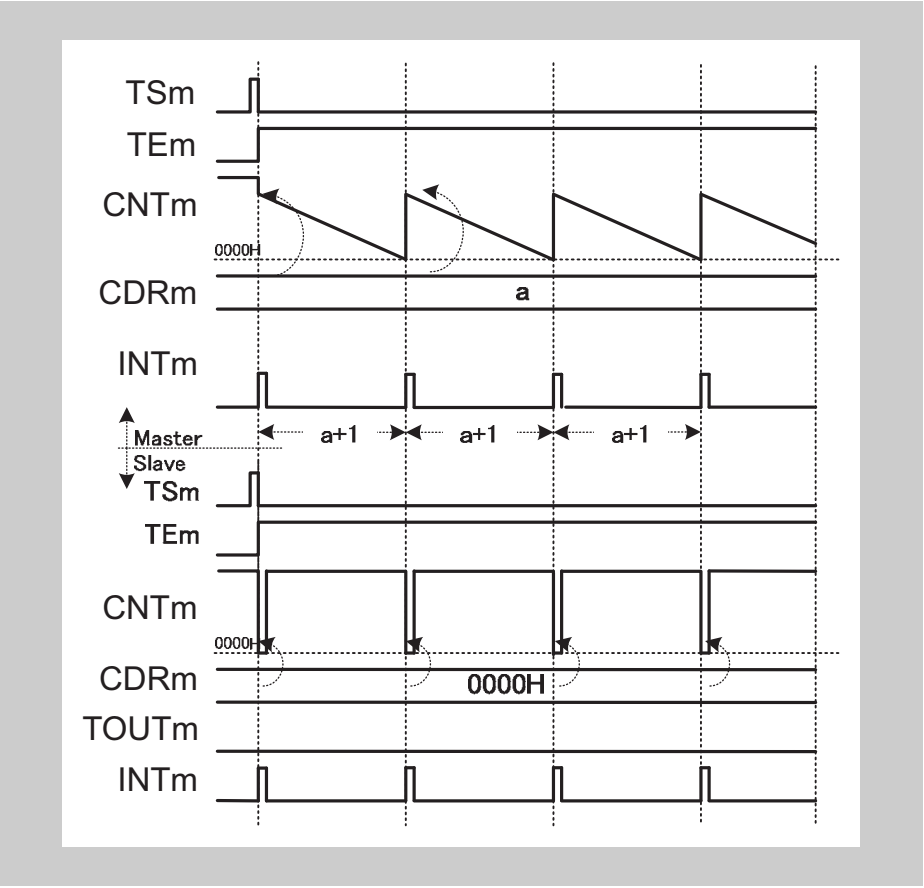


Figure 15-92 TAUAnCDRm (slave) = 0000_H, positive logic (TAUAnTOL.TOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUAnIm), 0000_H is written to TAUAnCNTm (slave). Therefore, TAUAnCNTm (slave) cannot start to count and TAUAnTTOUTm remains at not active state.
- TAUAnCNTm (slave) generates an interrupt every time the value of TAUAnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

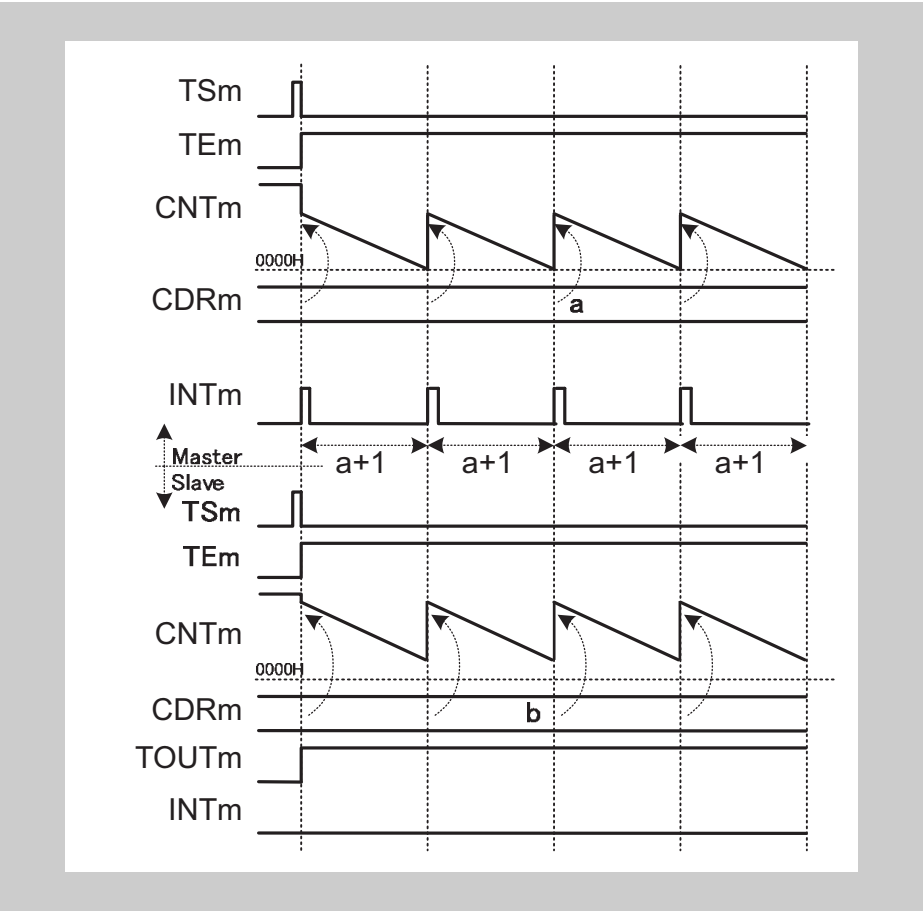


Figure 15-93 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1, positive logic (TAUAnTOL.TOLm (slave) = 0)

- If the value TAUAnCDRm (slave) is higher than the value TAUAnCDRm (master), the counter of the slave channel cannot reach 0000_H and cannot generate interrupts. The TAUAnTTOUTm remains at active state.

(c) Stop and restart operation

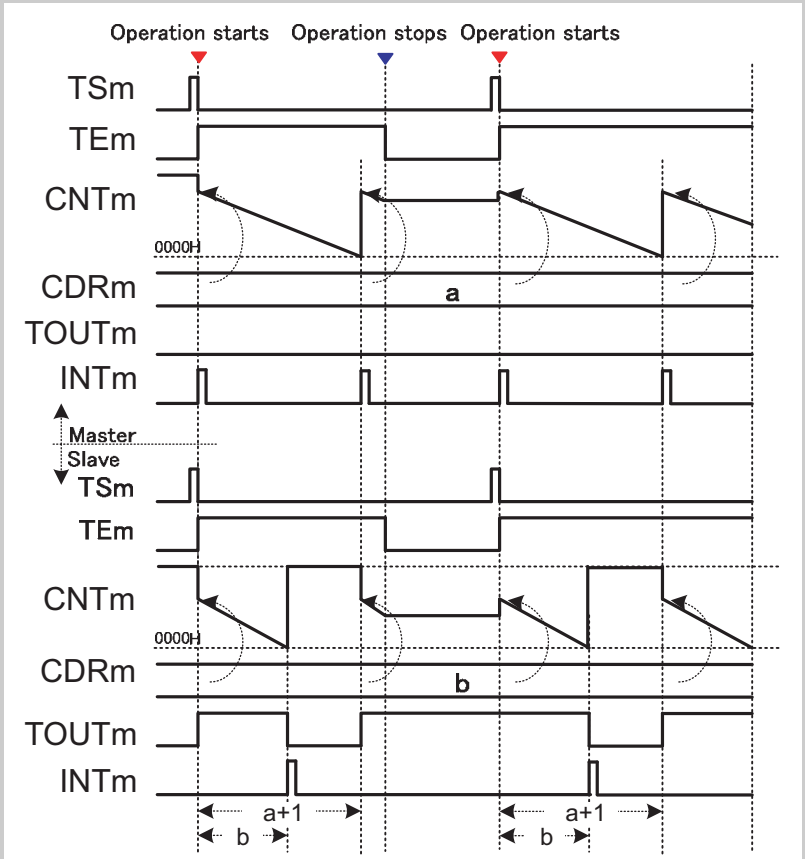


Figure 15-94 Stop and restart operation, positive logic (TAUAnTOL.TOLm (slave) = 0)

- The counter can be stopped by setting TAUAnTT.TTm of the master and slave channel(s) to 1, which in turn sets TAUAnTE.TEm to 0.
- TAUAnCNTm and TAUAnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUAnTS.TSm of master and slave channel(s) to 1. TAUAnCNTm of master and slave channel reload the current values of TAUAnCDRm and start to count down from these values.

15.23.2 Trigger Start PWM Output Function

(1) Overview

Summary This function generates a PWM output using a master and a slave channel. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUAnTTOUTm to be set. The pulse cycle is specified using the master channel. The pulse width is specified using the slave channel. The Trigger Start PWM Output Function is identical to PWM Output Function except that the master channel of this function can be reset by a valid TAUAnTTINm input edge.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-116 “TAUAnCMORm settings for the master channel of the Trigger Start PWM Output Function” on page 890*
- The operation mode of the slave channel must be set to One Count Mode, refer to *Table 15-119 “TAUAnCMORm settings for the slave channel of the Trigger Start PWM Output Function” on page 892*
- The channel output mode of the slave channel must be set to Synchronous Channel Output Mode 1, refer to *15.9 “Channel Output Modes” on page 731*
- TAUAnTTOUTm is not used for the master channel of this function.

Description The counters (master and slave) are started by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation.

- Master channel:

  The current value of TAUAnCDRm is written to the counter (TAUAnCNTm), INTTAUAnIm is generated and the counter starts to count down from this value.

  When the counter reaches 0000_H, pulse cycle time has elapsed, INTTAUAnIm is generated and the counters (master and slave) reload the current TAUAnCDRm values.

  If a valid TAUAnTTINm input edge is detected, the counter of the master channel reloads the current TAUAnCDRm value, restarts counting down and generates an interrupt.

- Slave channel:

  When the slave detects an interrupt from the master channel, it starts to count down from the current value of TAUAnCDRm. The TAUAnTTOUTm signal is set.

  When the counter reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Note If a forced restart is executed during operation, TAUAnTTOUTm is not output as a PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 “Simultaneous Rewrite” on page 719*

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = [TAUAnCDRm (slave) / (TAUAnCDRm (master) + 1)] x 100

– Duty cycle = 0 %

TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

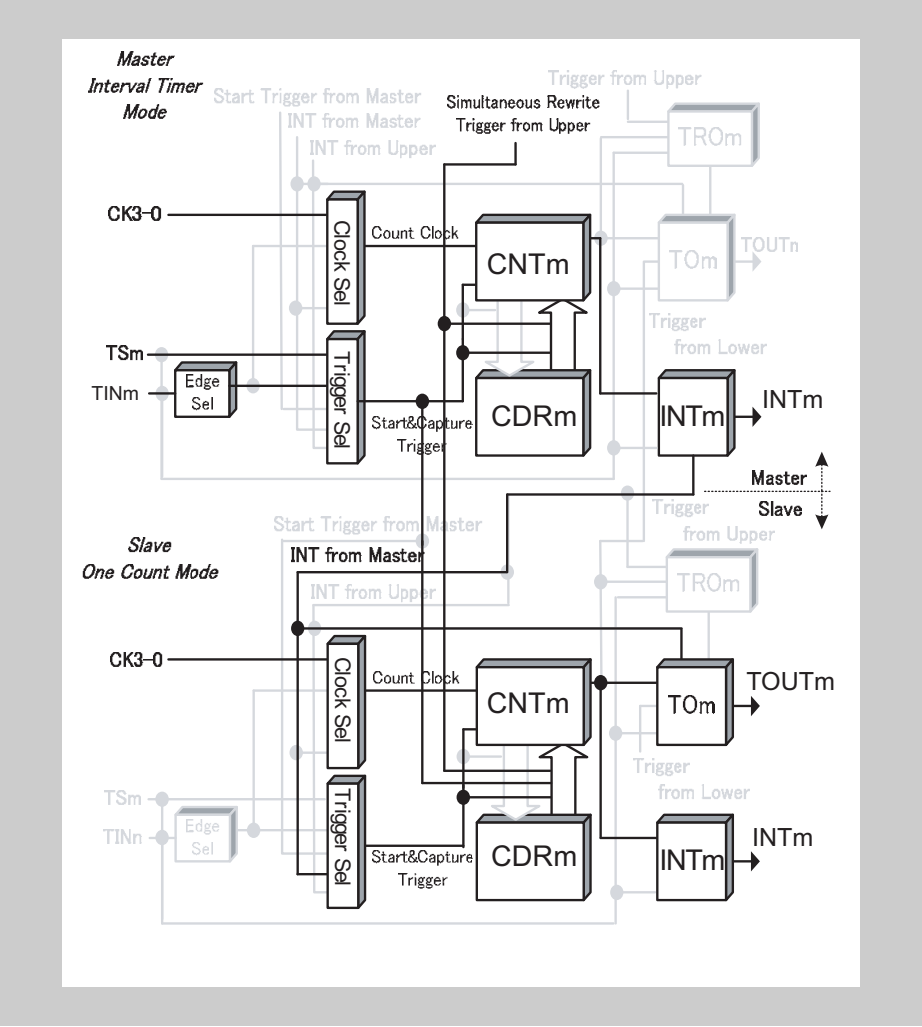


Figure 15-95 Block diagram for Trigger Start PWM Output Function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUAnCMURm.TIS[1:0] = 01_B)
- Positive logic (TAUAnTOL.TOLm (slave) = 0)

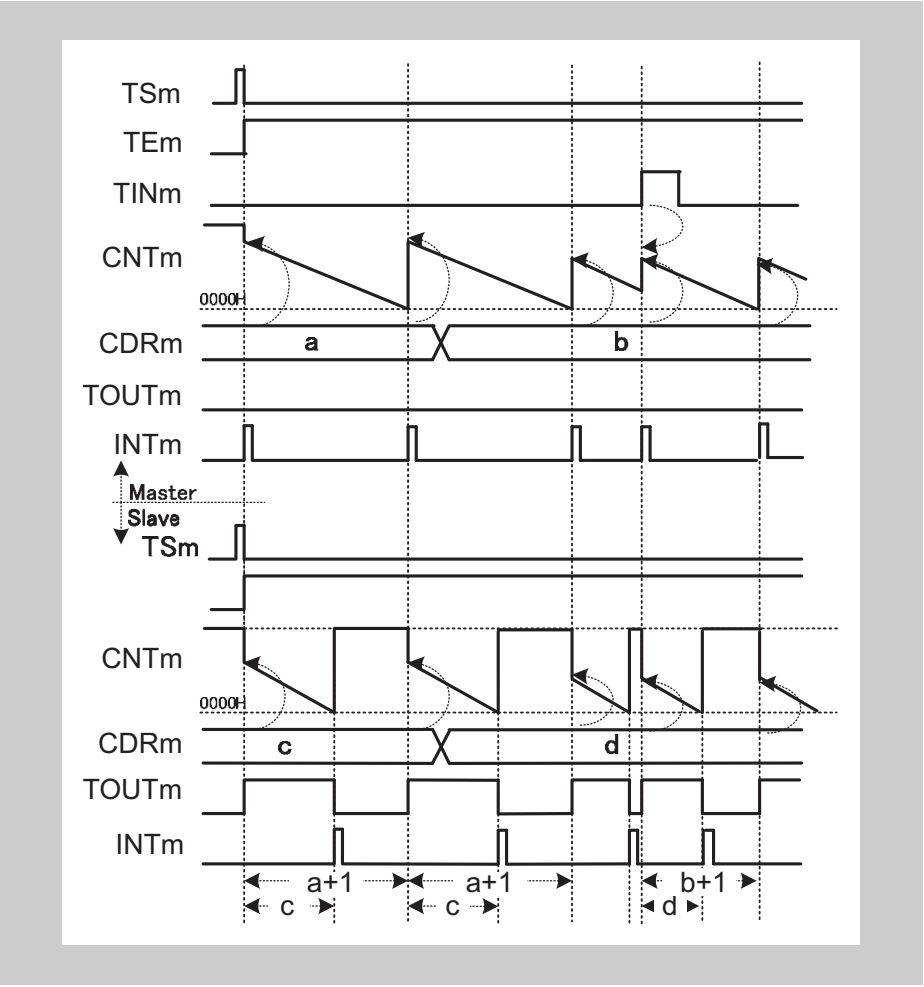


Figure 15-96 General timing diagram for Trigger Start PWM Output Function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-116 TAUAnCMORm settings for the master channel of the Trigger Start PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm at operation start The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-117 TAUAnCMURm settings for the master channel of the Trigger Start PWM Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-118 Simultaneous rewrite settings for the master channel of the Trigger Start PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-119 TAUAnCMORm settings for the slave channel of the Trigger Start PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-120 TAUAnCMURm settings for the slave channel of the Trigger Start PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

Table 15-121 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-122 Simultaneous rewrite settings for the slave channel of the Trigger Start PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for Trigger Start PWM Output Function

Table 15-123 Operating procedure for Trigger Start PWM Output Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 890* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel" on page 892* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation • TAUAnCNTm (slave) reloads the TAUAnCDRm value and starts to count down • TAUAnTTOUTm (slave) is set When TAUAnCNTm of the slave = 0000_H: • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is reset If a TAUAnTTINm input is detected on the master channel while the counter is counting down: • TAUAnCNTm (master and slave) reloads the TAUAnCDRm value and counts down • INTTAUAnIm (master) is generated • TAUAnTTOUTm (slave) is set
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart →

(7) Specific timing diagrams

(a) Duty cycle = 0 %

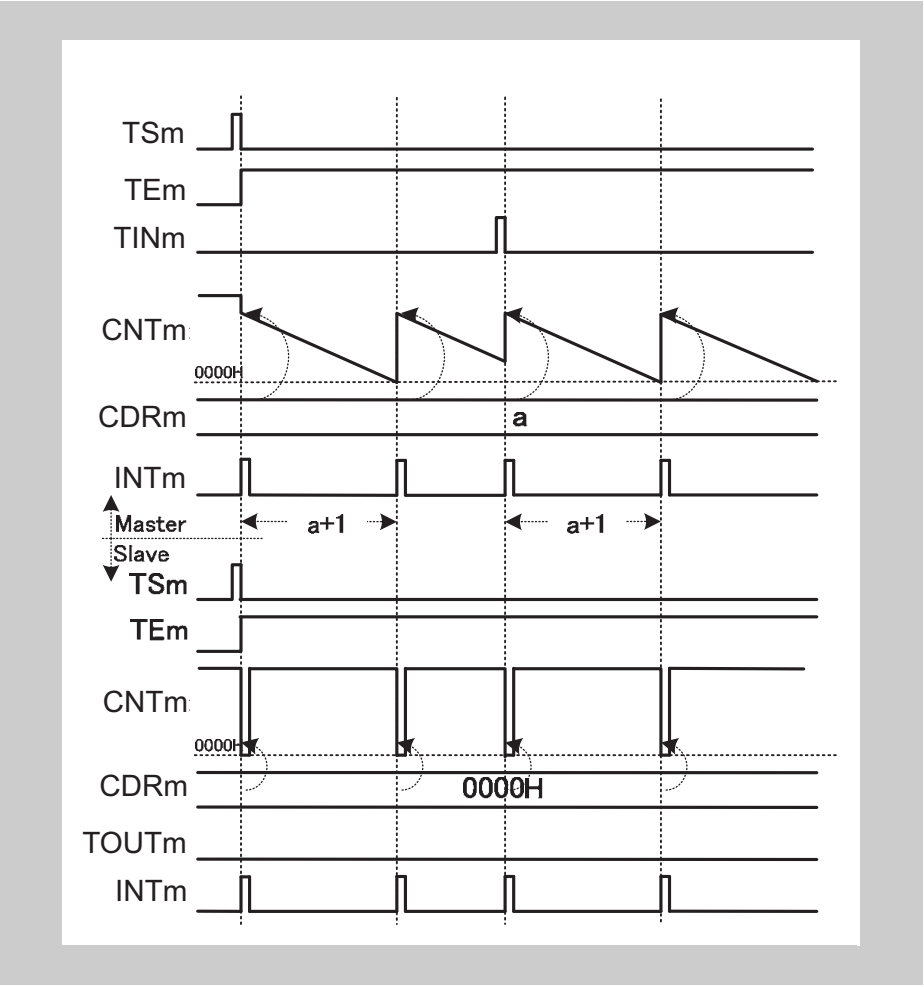


Figure 15-97 TAUAnCDRm (slave) = 0000_H, positive logic (TAUAnTOL.TOLm (slave) = 0) falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

- Every time the master channel generates an interrupt (INTTAUAnIm), 0000_H is written to TAUAnCNTm (slave). Therefore, TAUAnCNTm (slave) cannot start to count and TAUAnTTOUTm remains at not active state.
- TAUAnCNTm (slave) generates an interrupt every time the value of TAUAnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

  The detection of a valid TAUAnTTINm input edge has no effect on TAUAnTTOUTm (slave).

(b) Duty cycle = 100 %

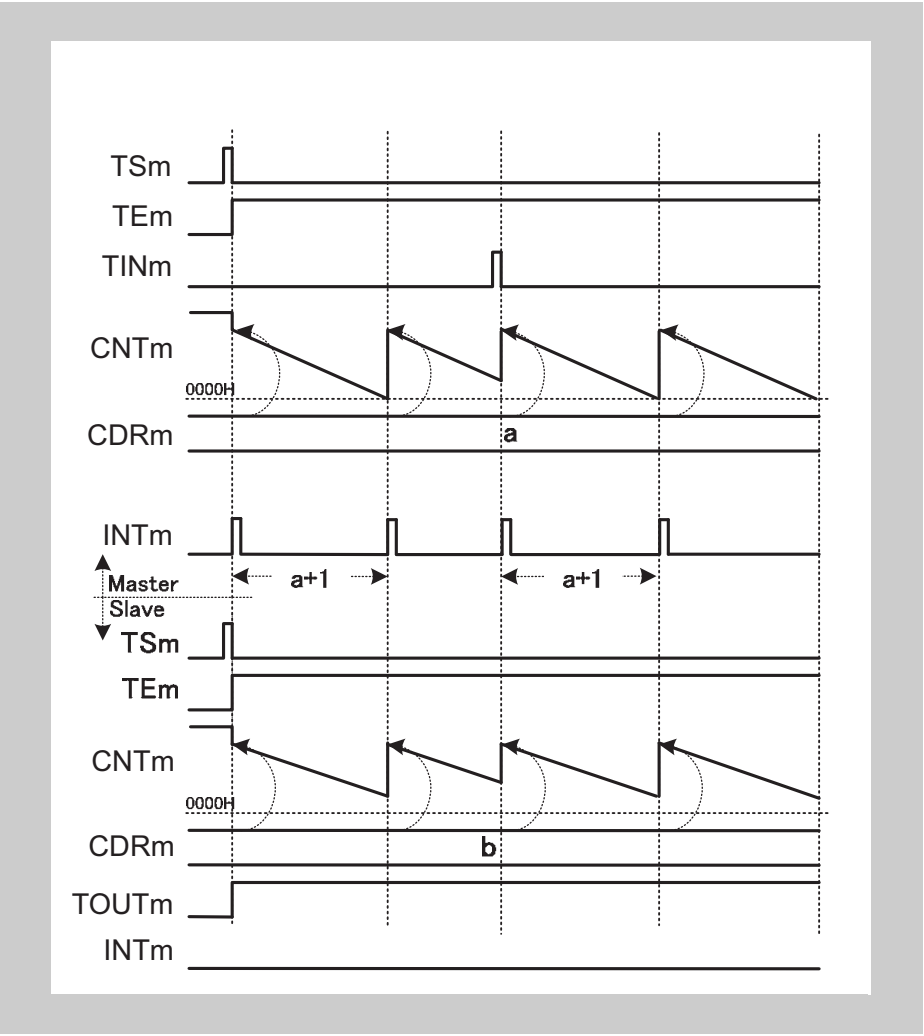


Figure 15-98 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1, positive logic (TAUAnTOL.TOLm (slave) = 0) falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

- If the value TAUAnCDRm (slave) is higher than the value TAUAnCDRm (master), the counter of the slave channel cannot reach 0000_H and cannot generate interrupts.

  The TAUAnTTOUTm remains at active state.

  The detection of a valid TAUAnTTINm input edge has no effect on TAUAnTTOUTm (slave).

(c) TAUAnTTINm detection and active slave counter

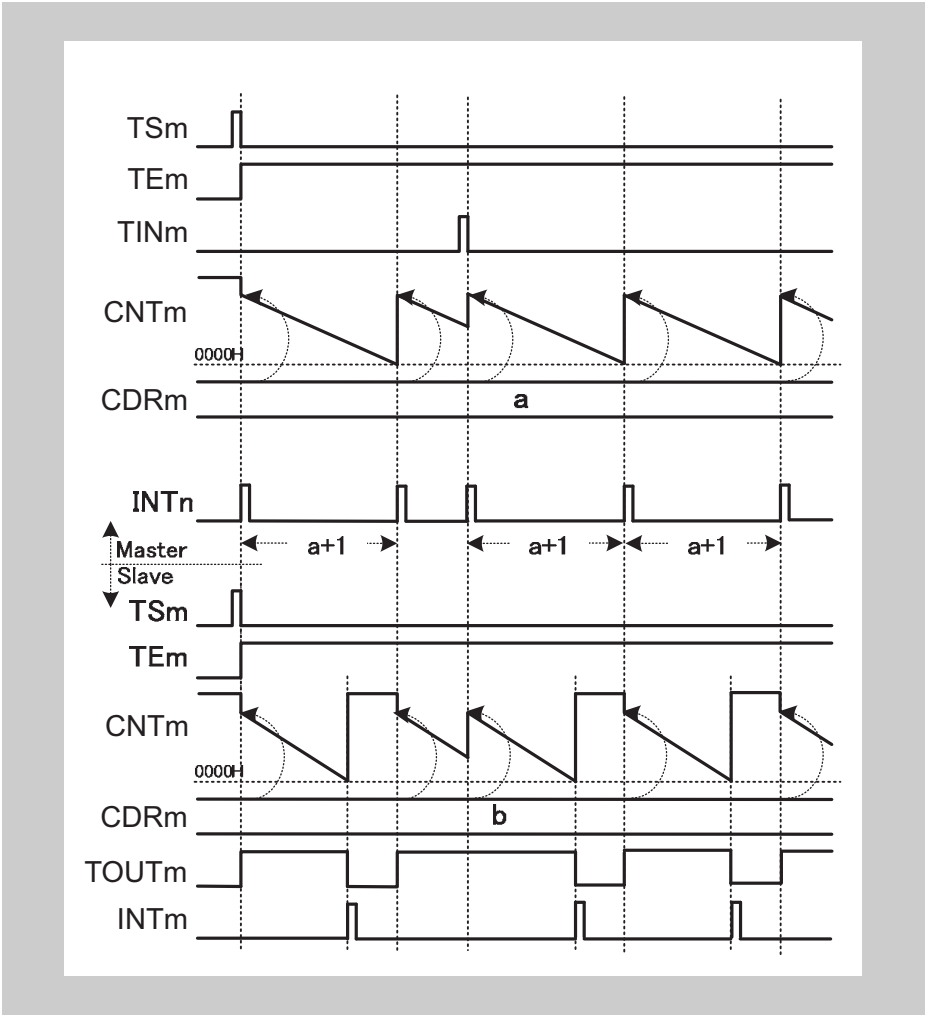


Figure 15-99 Positive logic (TAUAnTOL.TOLm (slave) = 0), falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

- If TAUAnCNTm (slave) reloads the value TAUAnCDRm (slave) while it is still counting down, TAUAnTTOUTm cannot toggle and extends the pulse width.

  The pulse width does not correspond to the value of the slave's data register.

15.23.3 Delay Pulse Output Function

(1) Overview

Summary This function outputs two signals. The reference signal has a defined pulse width and pulse cycle specified using the master channel and slave channel 1. Slave channels 2 and 3 output the reference signal with a specified delay. The delay signal is identical to the reference signal, but delayed by amount specified in slave channel 2.

The signal values are specified in the following way:

- The pulse cycle is specified using the master channel.
- The duty cycle of the reference signal is specified using slave channel 1. The duty cycle of the delay signal is specified using slave channel 3.
- The delay is specified in slave channel 2.

Prerequisites

- Four channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-124 “TAUAnCMORm settings for the master channel of the Delay Pulse Output Function” on page 902*
- The operation mode of slave channel 1 and 2 must be set to One Count Mode, refer to *Table 15-127 “TAUAnCMORm settings for slave channel 1 of the Delay Pulse Output Function” on page 904*
- The operation mode of slave channel 3 must be set to Pulse One Count Mode, refer to *Table 15-131 “TAUAnCMORm settings for slave channel 2 of the Delay Pulse Output Function” on page 906*
- TAUAnTTOUTm is not used for the master channel and slave channel 2
- The channel output mode of slave channel 1 must be set to Synchronous Channel Output Mode 1 (refer to *15.9 “Channel Output Modes” on page 731*)
- The channel output mode of slave channel 3 must be set to Independent Channel Output Mode 2 (refer to *15.9 “Channel Output Modes” on page 731*)

Description The counters of the channel group are started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation.

- Master channel:

  The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value. INTTAUAnIm is generated on the master channel.

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUAnIm is generated. The counter reloads the TAUAnCDRm value and counts down.

- Slave channels 1 and 2:

  When the slave channels 1 and 2 detect an interrupt from the master channel, they start to count down from the current value of TAUAnCDRm. The TAUAnTTOUTm signal (slave 1) is set.

– Slave channel 1:

When the counter of slave channel 1 reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

– Slave channel 2:

When the counter of slave channel 2 reaches 0000_H, delay time has elapsed and INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

INTTAUAnIm (slave 2) triggers the counter of slave channel 3

- Slave channel 3:

When slave channel 3 detects an interrupt from slave channel 2, it starts to count down from the current value of TAUAnCDRm. INTTAUAnIm is generated and the TAUAnTTOUTm signal (slave 3) toggles.

When the counter of slave channel 3 reaches 0000_H, duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles again.

The output from slave channel 3 is the delayed PWM pulse

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 “Simultaneous Rewrite” on page 719*

Equations Pulse cycle = (TAUAnCDRm (master) + 1) × count clock cycle

Duty cycle = (TAUAnCDRm (slave 1 and 3)) × count clock cycle

Delay = (TAUAnCDRm (slave 2) + 1) × count clock cycle

(2) Block diagram and general timing diagram

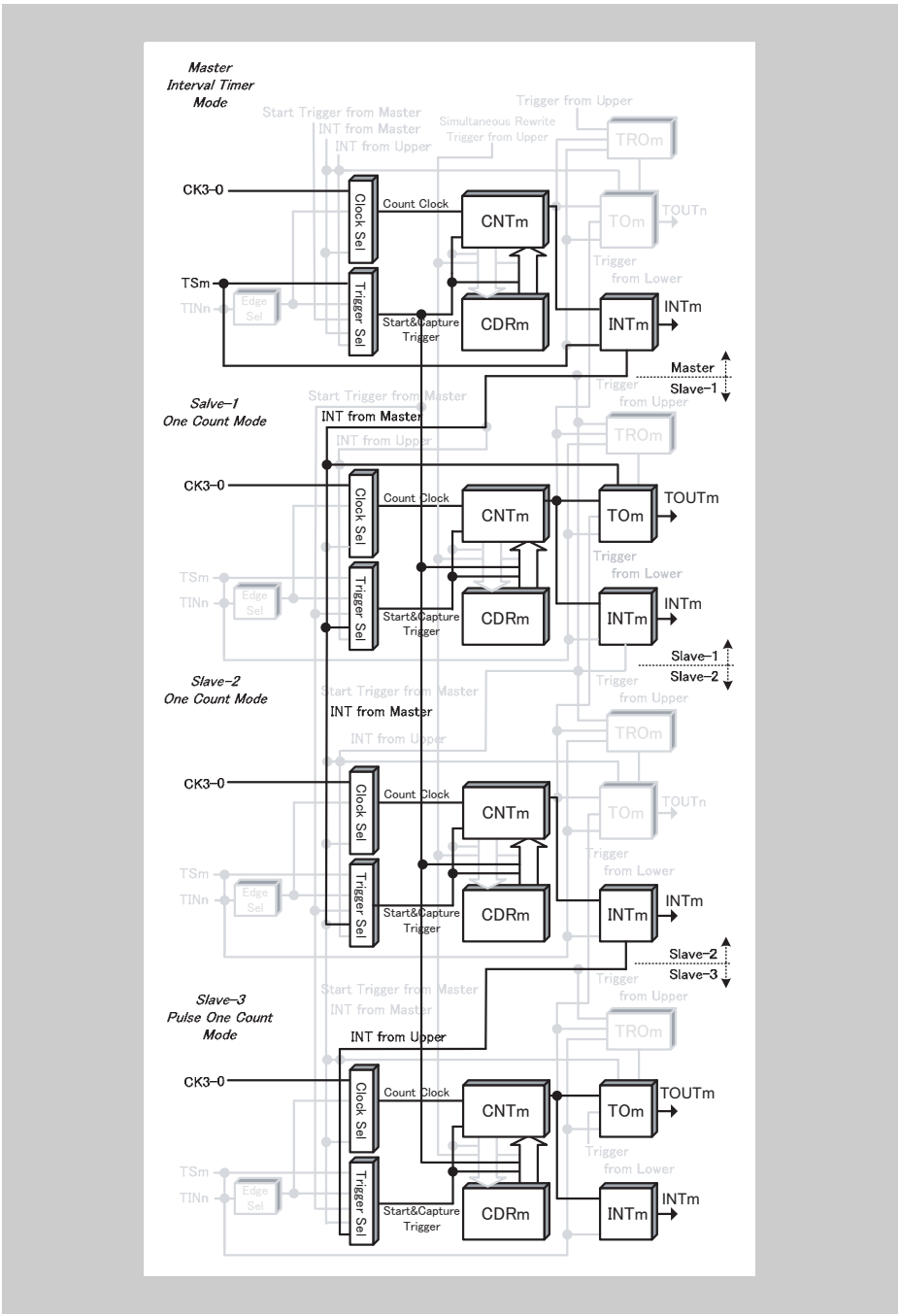


Figure 15-100 Block diagram for Delay Pulse Output Function

The following settings apply to the general timing diagram:

- All channels
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)

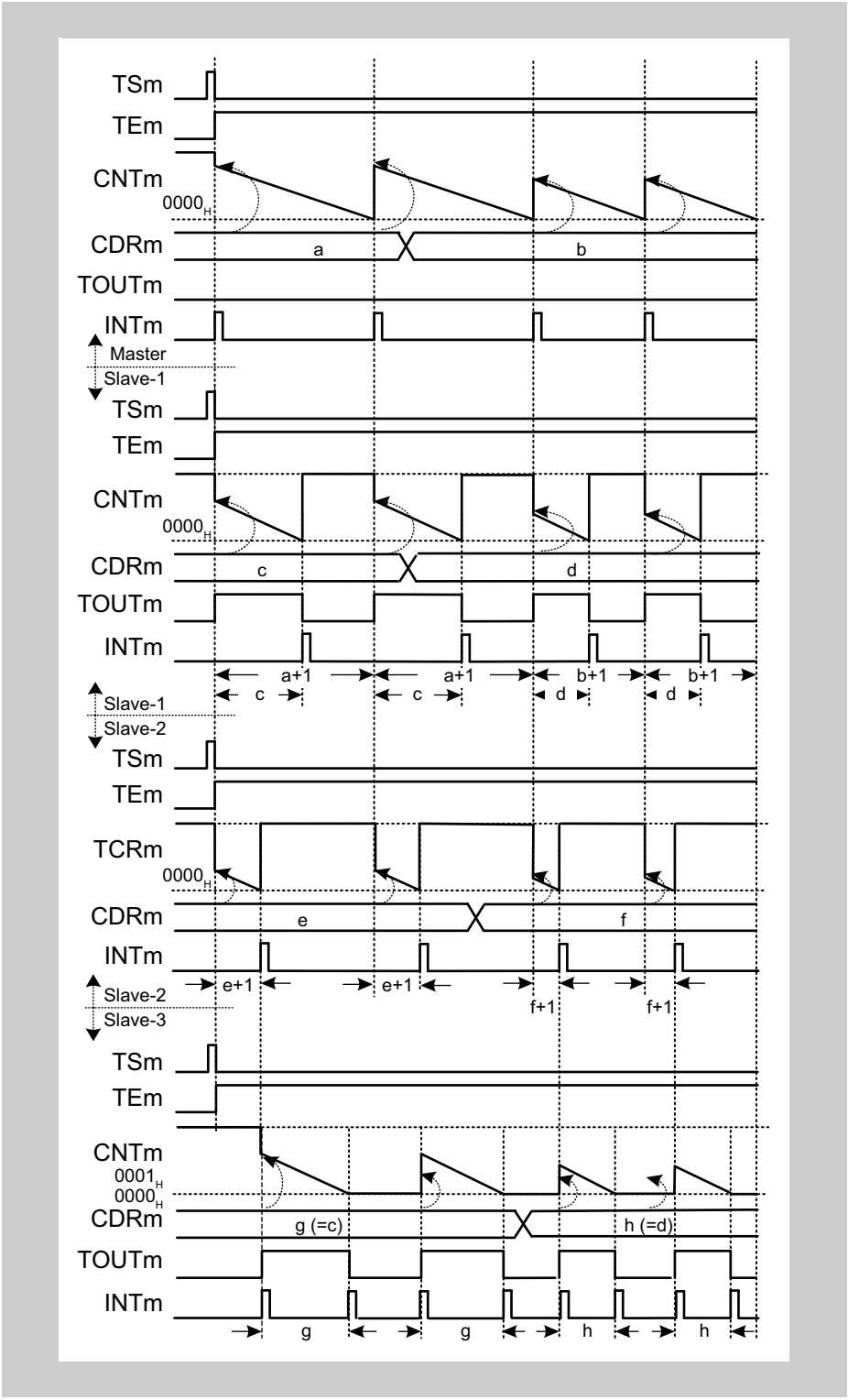


Figure 15-101 General timing diagram for Delay Pulse Output Function

(3) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-124 TAUAnCMORm settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-125 TAUAnCMURm settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by the master channel of this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-126 Simultaneous rewrite settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-127 TAUAnCMORm settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUAnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-128 TAUAnCMURm settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 1

Table 15-129 Control bit settings for slave channel 1 of the Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-130 Simultaneous rewrite settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUAnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-131 TAUAnCMORm settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUAnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-132 TAUAnCMURm settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-133 Simultaneous rewrite settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUAnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-134 TAUAnCMORm settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	101: INTTAUAnIm of the upper channel (m-1) is the start trigger, regardless of the master setting
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-135 TAUAnCMURm settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 15-136 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-137 Simultaneous rewrite settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Delay Pulse Output Function

Table 15-138 Operating procedure for Delay Pulse Output Function (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *3 "Register settings for the master channel" on page 902* Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for slave channel 1" on page 904* Slave channel 2: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 2" on page 906* Slave channel 3: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for slave channel 3" on page 908* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 15-138 Operating procedure for Delay Pulse Output Function (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master channel and slave channels 1 and 2 start. INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (slave 1) is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel and slave channels 1 and 2 load TAUAnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation • TAUAnCNTm (slave 1 and slave 2) reload the TAUAnCDRm value and start counting down • TAUAnTTOUTm (slave 1) is set When TAUAnCNTm (slave 1) reaches 0000_H: • INTTAUAnIm (slave 1) is generated The output from slave channel 1 acts as the reference pulse • TAUAnTTOUTm (slave 1) is reset When TAUAnCNTm (slave 2) reaches 0000_H: • INTTAUAnIm (slave 2) is generated • TAUAnTTOUTm (slave 3) toggles • TAUAnCNTm (slave 3) reloads the TAUAnCDRm value and starts counting down When TAUAnCNTm (slave 3) reaches 0000_H: • INTTAUAnIm (slave 3) is generated • TAUAnTTOUTm (slave 3) toggles The output from slave channel 3 is the delayed PWM pulse
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle (slave 3) = 100 %

The following values apply to the figure below:

- TAUAnCDRm (master) = $000A_H$
- TAUAnCDRm (slave 1) = $000B_H$
- TAUAnCDRm (slave 2) = 0000_H
- TAUAnCDRm (slave 3) = $000B_H$

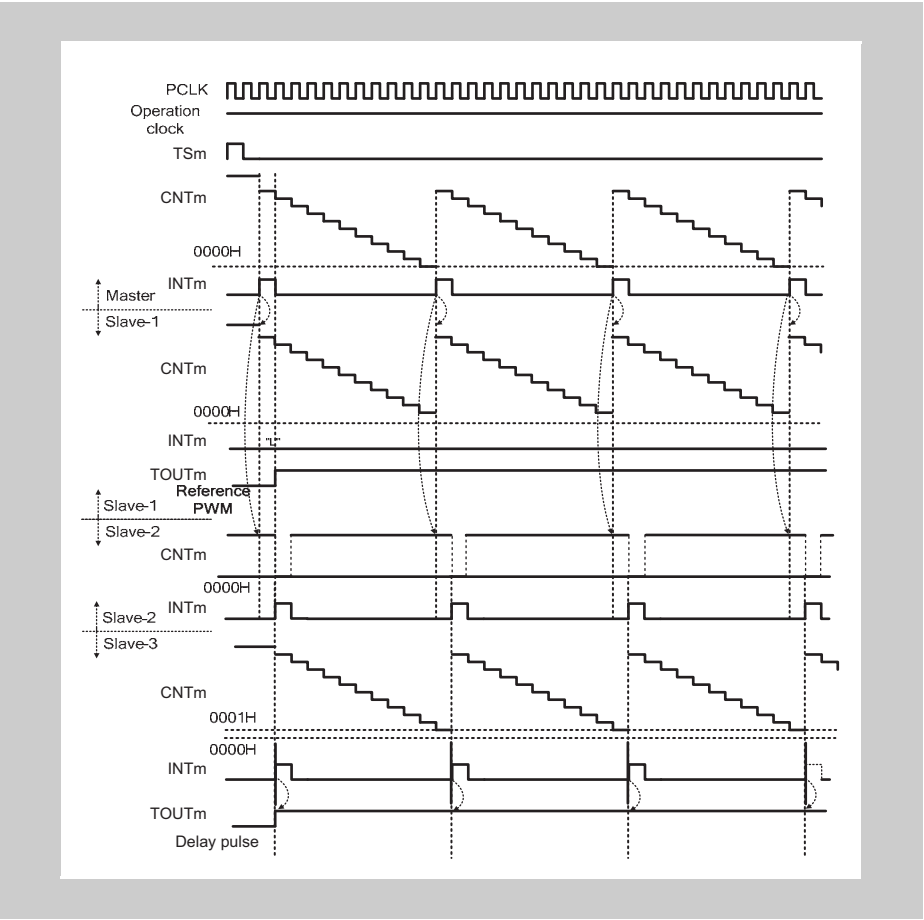


Figure 15-102 Duty cycle (slave 3) = 100 %

- If the value of TAUAnCDRm (slave 1 and 3) is higher than the value of TAUAnCDRm (master), the counter of the slave channels cannot reach 0000_H and cannot generate interrupts. TAUAnTTOUTm of channels 1 and 3 remain in the active state.

(b) TAUAnTTOUTm (slave 1) = TAUAnTTOUTm (slave 3)

The following values apply to the figure below:

- TAUAnCDRm (master) = $000A_H$
- TAUAnCDRm (slave 1) = 0005_H
- TAUAnCDRm (slave 2) = 0000_H
- TAUAnCDRm (slave 3) = 0005_H

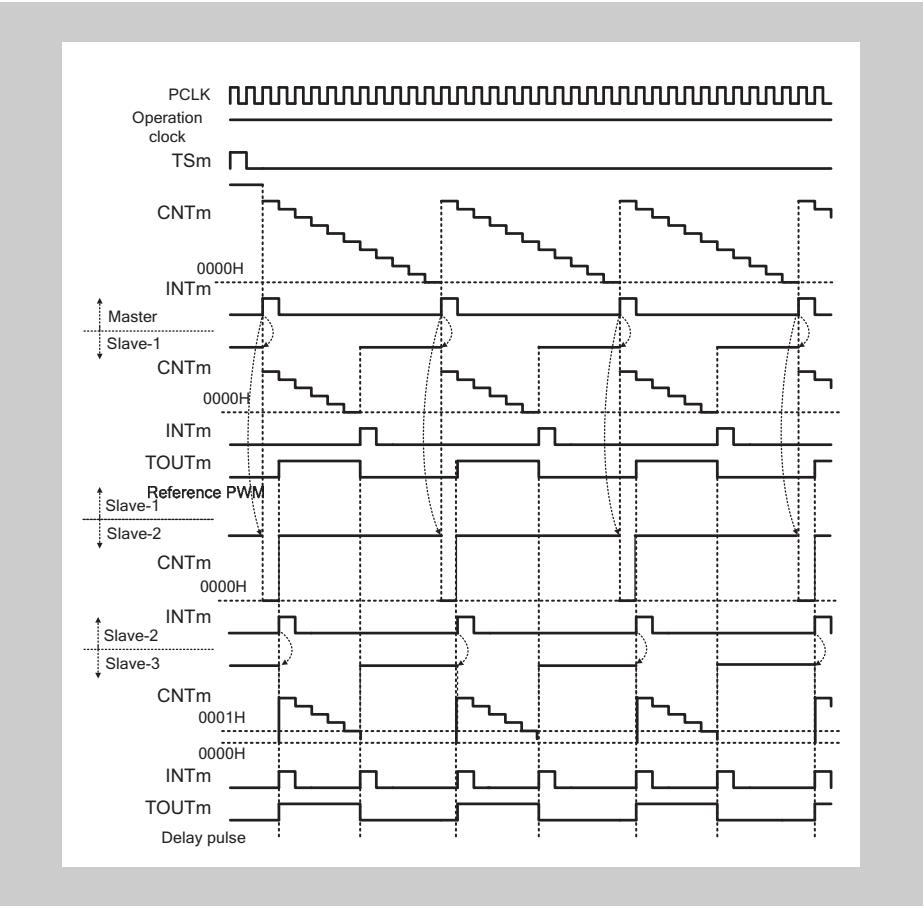


Figure 15-103 TAUAnTTOUTm (slave 1) = TAUAnTTOUTm (slave 3)

- If TAUAnCDRm (slave 2) = 0000_H, the counter of slave channel 3 starts counting one count clock later than the counter of slave channel 1. The reference pulse and the delay pulse are output with a delay of one clock count.

15.23.4 AD Conversion Trigger Output Function Type 1

(1) Overview

Summary This function is identical to *15.23.1 "PWM Output Function" on page 876* except that TAUAnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to Direct Channel Output Mode.

(2) Block diagram and general timing diagram

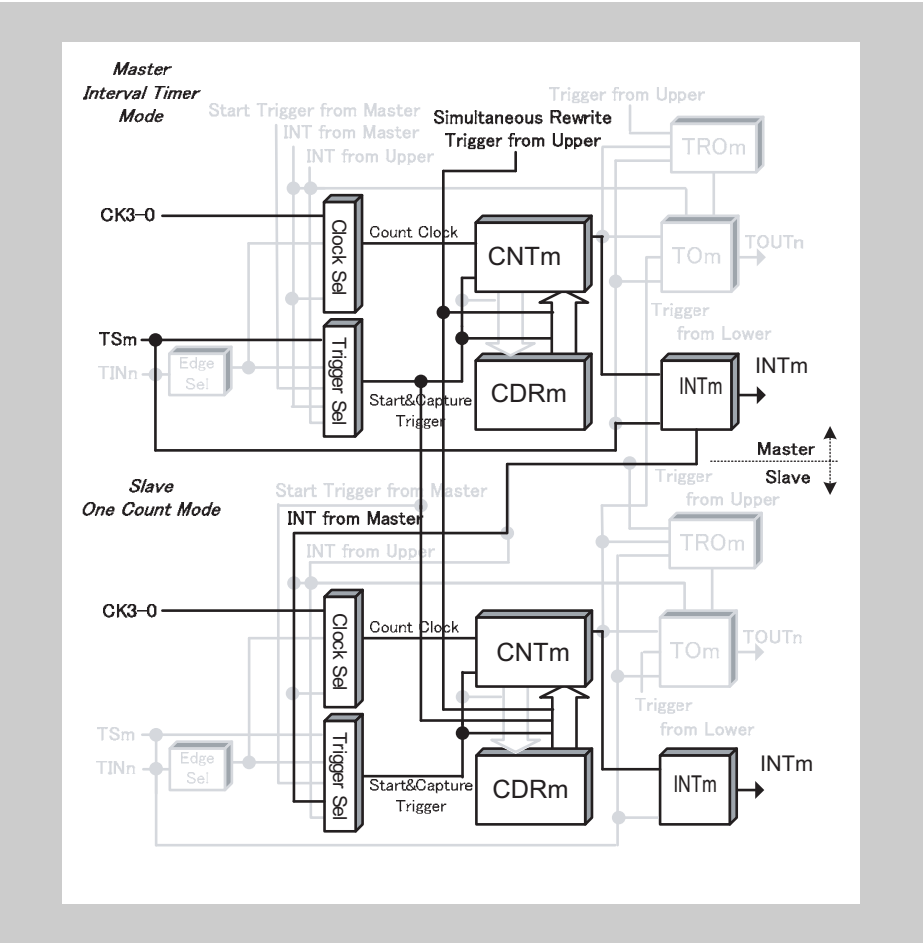


Figure 15-104 Block diagram for AD Conversion Trigger Output Function Type 1

(3) General timing diagram

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUAnTOL.TOLm = 0)

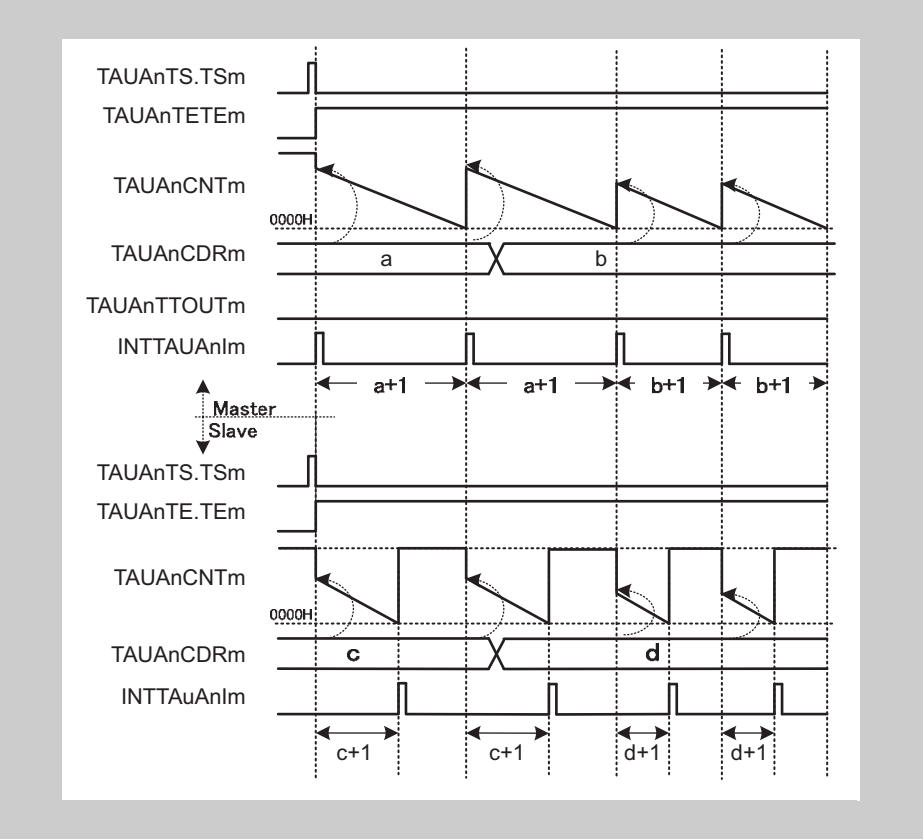


Figure 15-105 General timing diagram for AD Conversion Trigger Output Function Type 1

15.24 Synchronous PWM Signal Functions Triggered by an External Signal

This chapter describes functions that generate PWM signals and which are triggered by an external signal.

- *15.24.1 “One-Shot Pulse Output Function”*
- *15.24.2 “Offset Trigger Output Function”*

15.24.1 One-Shot Pulse Output Function

(1) Overview

Summary This function outputs a signal pulse with a defined pulse width and a specific delay time compared to an external input signal pulse by using a master and a slave channel. The delay time is specified using the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to One Count Mode, refer to *Table 15-139 “TAUAnCMORm settings for the master channel of the One-Shot Pulse Output Function” on page 920*
- The operation mode of the slave channel must be set to Pulse One Count Mode, refer to *Table 15-142 “TAUAnCMORm settings for the slave channel of the One-Shot Pulse Output Function” on page 922*
- TAUAnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel must be set to Synchronous Channel Output Mode 2 (refer to *15.9 “Channel Output Modes” on page 731*)
- TAUAnTTINm (master) has to be detected while TAUAnCNTm (master) and TAUAnCNTm (slave) await a trigger. Furthermore, the slave is only triggered by an interrupt from the master channel and not by TAUAnTTINm.

Description The counters are enabled by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation.

- Master channel:

  When the next valid TAUAnTTINm input edge is detected, the current value of TAUAnCDRm is written to TAUAnCNTm. The counter starts to count down from this value. If TAUAnCMORm.MD0 = 0, a trigger (TAUAnTTINm) which is detected within the delay time is ignored.
  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUAnTTINm input edge.

- Slave channel

  The INTTAUAnIm of the master channel triggers the counter of the slave channel. The current value of TAUAnCDRm (slave) is written to TAUAnCNTm (slave) and the counter starts to count down from this value. An interrupt is generated and the TAUAnTTOUTm signal is set.
  When the counter reaches 0001_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter remains at 0001_H and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

The counter of the master channel can be restarted without stopping it first (forced restart) by setting TAUAnTS.TSm to 1 during operation.

Notes

1. If a forced restart of the slave channel is executed during operation, the width of the output signal does not correspond to the value of TAUAnCDRm (slave).
2. The input TAUAnTTINm is sampled at the frequency of the operating clock, specified by TAUAnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUAnTTOUTm has an error of ± 1 operation clock cycle.

Conditions

- If TAUAnCMORn.MD0 of the master channel is set to 0, during counting detected TAUAnTTINm input edges are ignored.
- Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*

Equations

Delay to input pulse = (TAUAnCDRm (master) + 1) × count clock cycle

Pulse width = (TAUAnCDRm (slave)) × count clock cycle

(2) Block diagram and general timing diagram

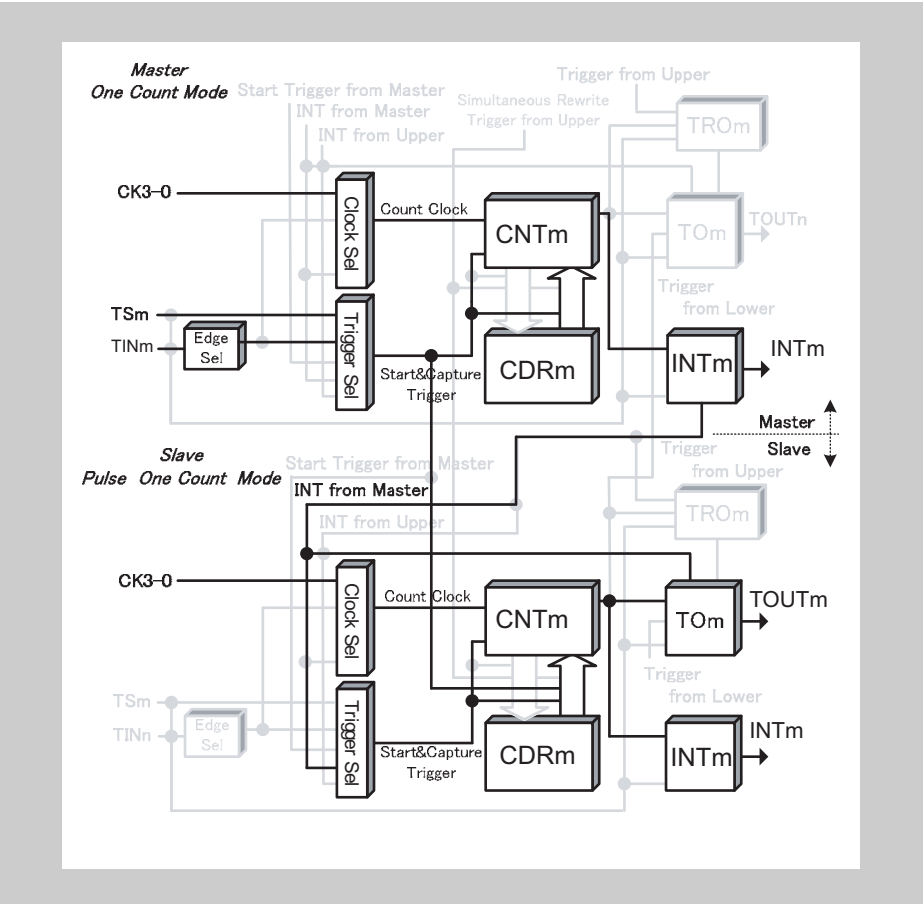


Figure 15-106 Block diagram for One-Shot Pulse Output Function

The following settings apply to the general timing diagram:

- Start trigger detection disabled during counting (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

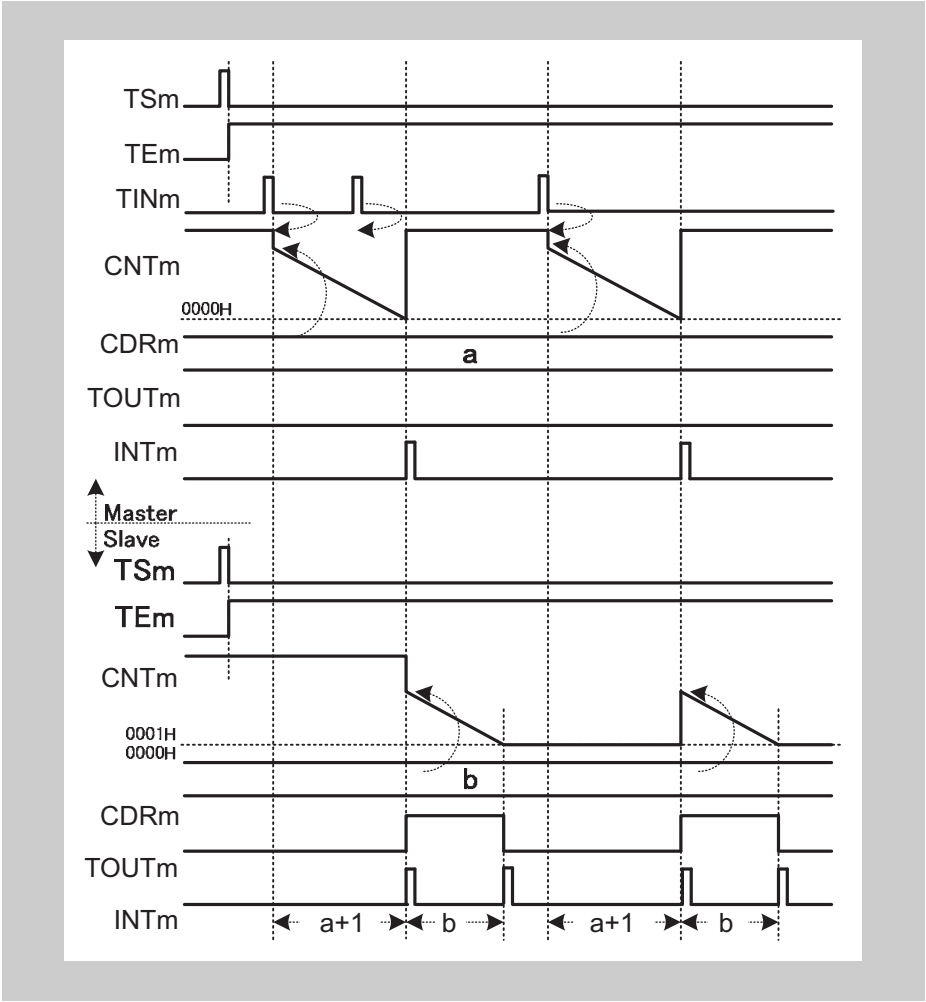


Figure 15-107 General timing diagram for One-Shot Pulse Output Function

(3) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-139 TAUAnCMORm settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-140 TAUAnCMURm settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-141 Simultaneous rewrite settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-142 TAUAnCMORm settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUAnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-143 TAUAnCMURm settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

Table 15-144 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-145 Simultaneous rewrite settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Operating procedure for One-Shot Pulse Output Function

Table 15-146 Operating procedure for One-Shot Pulse Output Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *3 "Register settings for the master channel" on page 920* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the slave channel" on page 922* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the master channel awaits a TAUAnTTINm input. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	When a valid TAUAnTTINm input edge is detected, TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation • TAUAnCNTm (slave) reloads the TAUAnCDRm value and starts to count down • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is set When TAUAnCNTm (slave) reaches 0001_H: • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is reset If a TAUAnTTINm input is detected on the master channel while the counter is counting, the input is ignored when TAUAnCMORm.MD0 = 0.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUAnCDRm (master) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

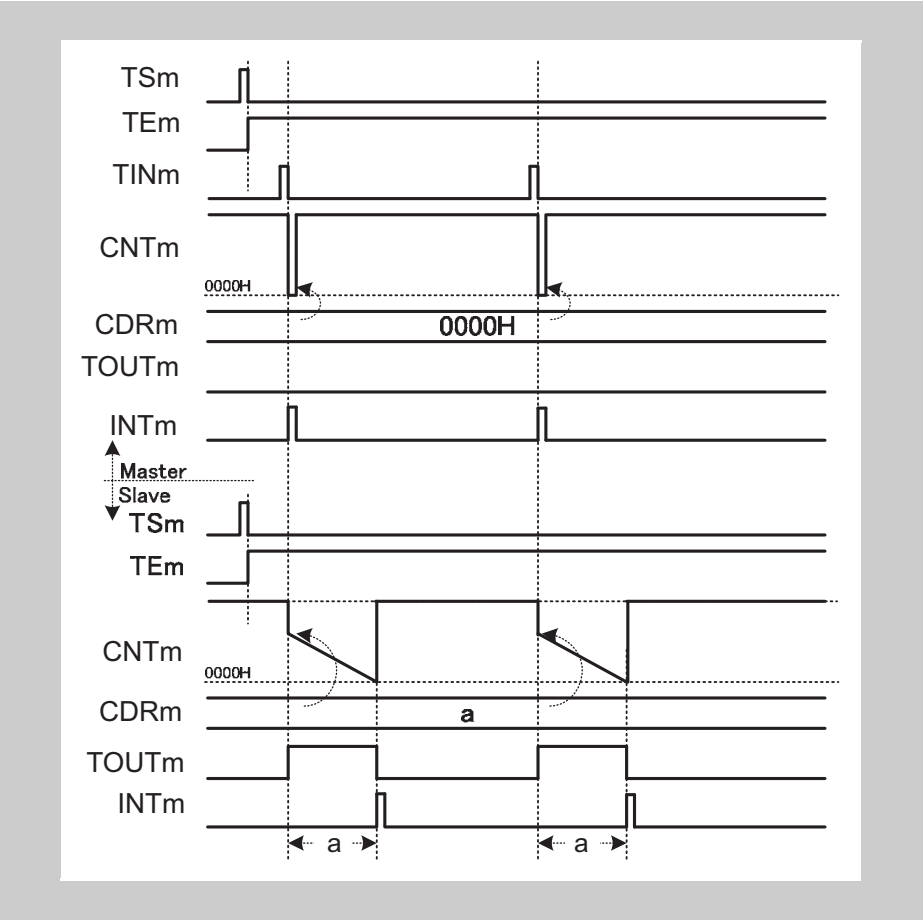


Figure 15-108 TAUAnCDRm (master) = 0000_H

- When a valid TAUAnTTINm input edge is detected, the value 0000_H is written to TAUAnCNTm (master). The counter is set to 0000_H for one count and returns to $FFFF_H$.

  Thus the slave channel starts to count down one count clock later to TAUAnTTINm (master).

(b) TAUAnCDRm (slave) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

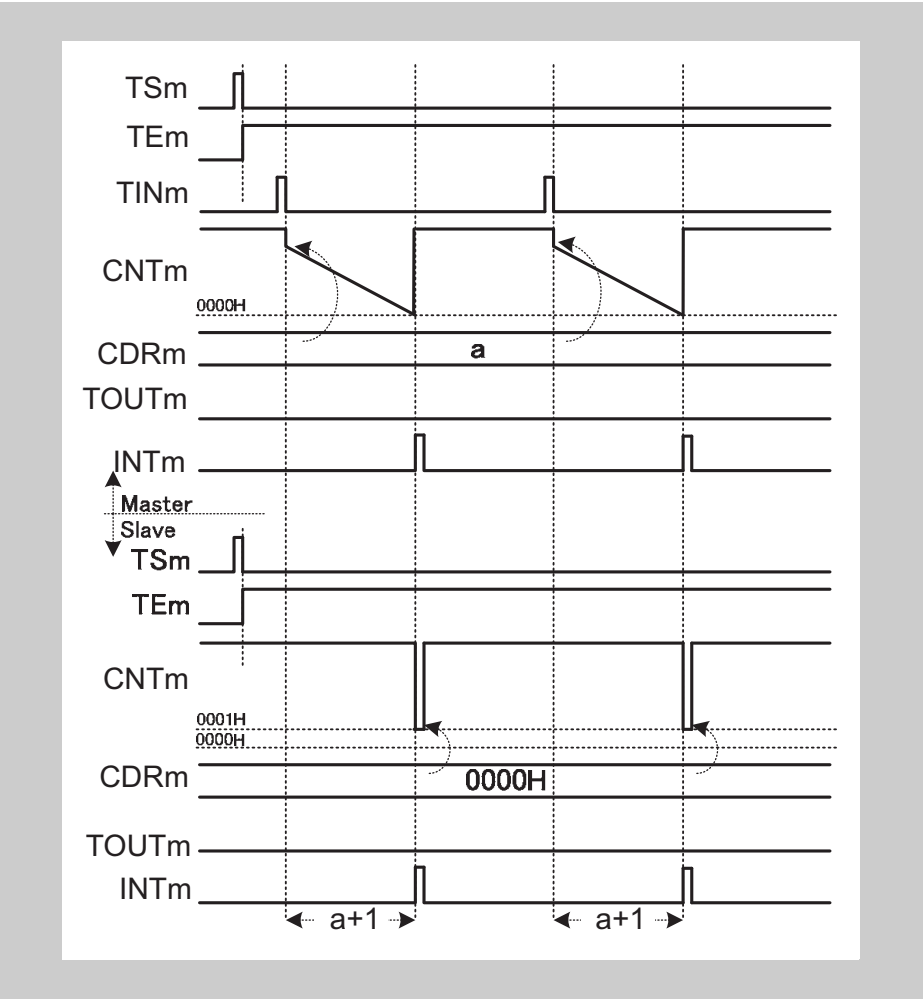


Figure 15-109 TAUAnCDRm (slave) = 0000_H

- The counter of the slave channel reloads the value 0000_H and returns to returns to $FFFF_H$ one clock count later.

  TAUAnTTOUTm remains at not active state, because the pulse width is zero.

(c) TAUAnCMORm.MD0 = 1

The following settings apply to this diagram:

- Start trigger detection enabled during counting (TAUAnCMORm.MD0 = 1)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

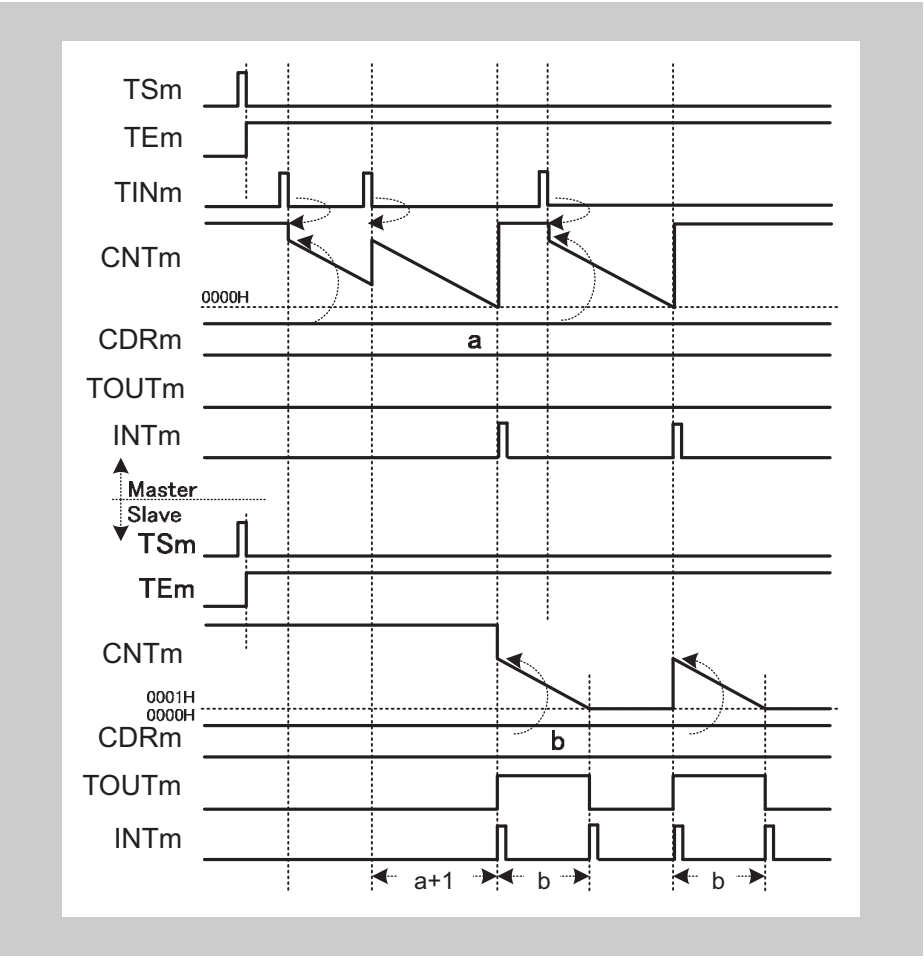


Figure 15-110 TAUAnCMORm.MD0 = 1

- If a valid TAUAnTTINm input edge is detected while the counter of the master channel counts down, TAUAnCNTm reloads the value of TAUAnCDRm. The counter restarts to count down.

  This means the delay is extended by the value of TAUAnCNTm at the time a valid TAUAnTTINm input edge is detected.

(d) Restarting the master channel while the slave channel is counting

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUAnCMORm.MD0 = 0)
- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

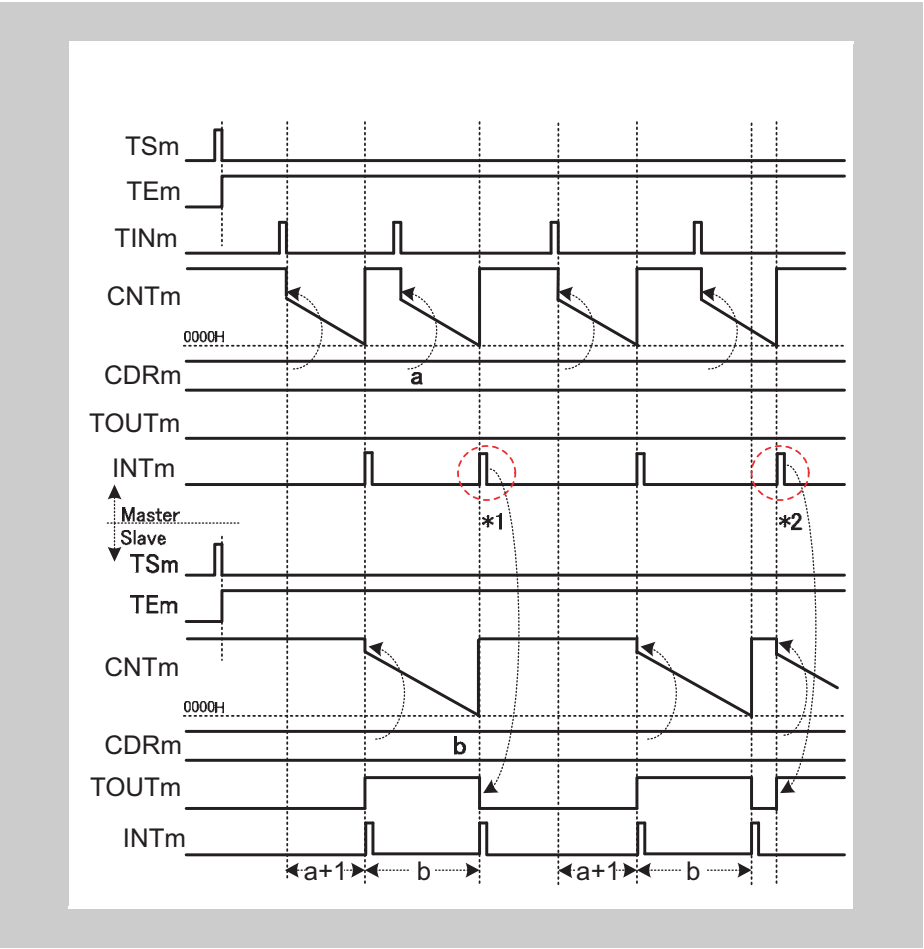


Figure 15-111 Interval of TAUAnTTINm ≤ delay time + pulse width +1

- If the master channel generates an interrupt before the counter of the slave channel has reached 0001_H or exactly when 0001_H is reached (*1), the interrupt (master) is ignored.
- If an interrupt of the master channel occurs when the counter of the slave channel awaits the next trigger, the value of TAUAnCDRm (slave) is reloaded. An interrupt is generated and TAUAnTTOUTm toggles. If TAUAnCNTm (master) has started to count down while the TAUAnCNTm (slave) is still counting (*2), TAUAnTTOUTm is not output with the expected delay time.
- To generate the correct one-shot pulse, the start trigger for the master channel must be detected while the master and slave channels are waiting for the start trigger, and not while they are counting.

15.24.2 Offset Trigger Output Function

(1) Overview

Summary This function generates a PWM output using a master and a slave channel. It enables the pulse width (duration) of the TAUAnTTOUTm to be set. The pulse cycle is set by the detection of a valid input edge of the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Capture Mode, refer to *Table 15-147 “TAUAnCMORm settings for the master channel of the Offset Trigger Output Function” on page 932*
- The operation mode of the slave channel must be set to One Count Mode, refer to *Table 15-150 “TAUAnCMORm settings for the slave channel of the Offset Trigger Output Function” on page 934*
- The channel output mode of the slave channel must be set to Synchronous Channel Output Mode 1 (refer to *15.9 “Channel Output Modes” on page 731*)
- TAUAnTTOUTm is not used for the master channel of this function.

Description The counters are started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The counter of the master channel (TAUAnCNTm) starts to count up from 0000_H.

- Master channel:

  When a valid TAUAnTTINm input edge is detected, the current value of the counter (TAUAnCNTm) is written to the data register of the master channel (TAUAnCDRm). INTTAUAnIm is generated and the counter restarts counting up from 0000_H.

- Slave channel:

  The INTTAUAnIm of the master channel sets the TAUAnTTOUTm (slave) signal and triggers the counter of the slave channel. The current value of TAUAnCDRm (slave) is written to TAUAnCNTm (slave) and the counter starts to count down from this value.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of the master channel.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 “Simultaneous Rewrite” on page 719*

(2) Equations

Pulse width = (TAUAnCDRm (slave) + 1) x count clock cycle

Duty cycle [%] = [TAUAnCDRm (slave) / TAUAnTTINm cycle + 1)] x 100

– Duty cycle = 0 %

TAUAnCDRm (slave) = 0000_H

– Duty cycle = 100 %

TAUAnCDRm (slave) ≥ TAUAnTTINm cycle + 1

(3) Block diagram and general timing diagram

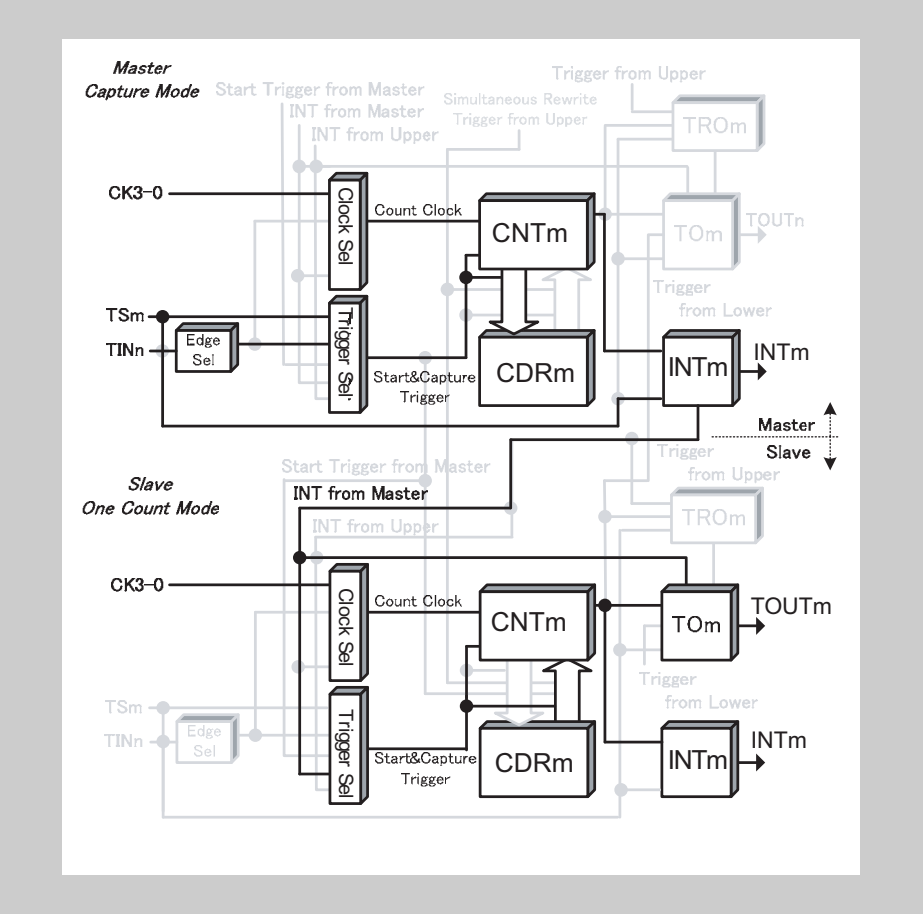


Figure 15-112 Block diagram for Offset Trigger Output Function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUAnCMURm.TIS[1:0] = 00_B)

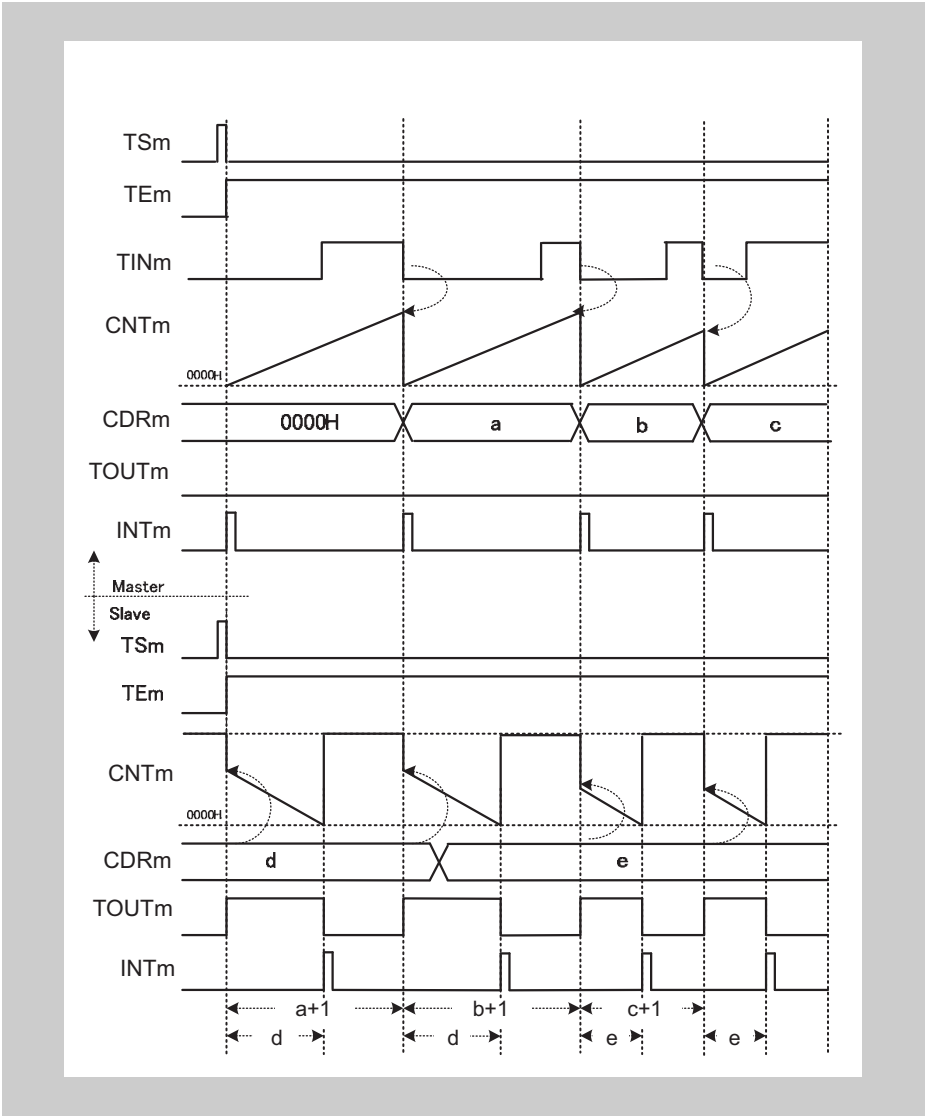


Figure 15-113 General timing diagram for Offset Trigger Output Function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-147 TAUAnCMORm settings for the master channel of the Offset Trigger Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	001: Valid TAUAnTTINm input edge signal is used as the start trigger
COS[1:0]	11: Capture register updated upon detection of a TAUAnTTINm input valid edge and upon counter overflow: - • TAUAnTTINm input valid edge: Counter value is written to TAUAnCDRm - • Overflow: $FFFF_H$ is written to TAUAnCDRm. The next TAUAnTTINm input valid edge detection is ignored. TAUAnCSRm.OVF set by counter overflow and cleared by a CPU instruction.
MD[4:1]	0010: Capture Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-148 TAUAnCMURm settings for the master channel of the Offset Trigger Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Offset Trigger Output Function. Therefore, these registers must be set to 0.

Table 15-149 Simultaneous rewrite settings for the master channel of the Offset Trigger Output Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-150 TAUAnCMORm settings for the slave channel of the Offset Trigger Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-151 TAUAnCMURm settings for the slave channel of the Offset Trigger Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

Table 15-152 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with the Offset Trigger Output Function. Therefore, these registers must be set to 0.

Table 15-153 Simultaneous rewrite settings for the slave channel of the Offset Trigger Output Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(6) Operating procedure for Offset Trigger Output Function

Table 15-154 Operating procedure for Offset Trigger Output Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 932* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel" on page 934* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start: • TAUAnCNTm (master) counts up • TAUAnCNTm (slave) loads TAUAnCDRm and counts down INTTAUAnIm is generated on the master channel and TAUAnTTOUTm (master) is set.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	When TAUAnCNTm of the slave reaches 0000_H: • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is reset When a TAUAnTTINm input is detected on the master channel: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) is reset to 0000_H and continues count operation • TAUAnCNTm (slave) reloads the TAUAnCDRm value and counts down • TAUAnTTOUTm (slave) is set
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart →

(7) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to this diagram:

- Falling edge detection: (TAUAnCMURm.TIS[1:0] = 00_B)

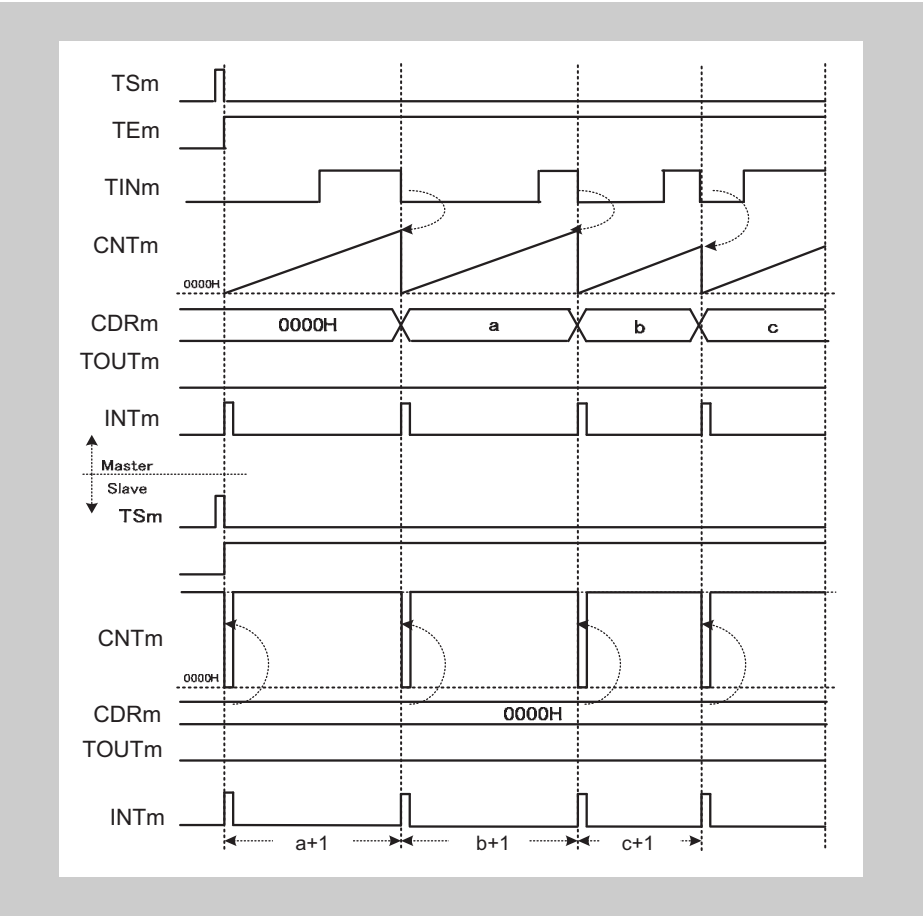


Figure 15-114 TAUAnCDRm (slave) = 0000_H

- When TAUAnCDRm (slave) = 0000_H, 0000_H is written to TAUAnCNTm every time the master channel generates an interrupt (INTTAUAnIm), TAUAnCNTm cannot start to count. The TAUAnTTOUTm remains at not active state.
- TAUAnCNTm (slave) generates an interrupt every time the value of TAUAnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

The following settings apply to this diagram:

- Falling edge detection: (TAUAnCMURm.TIS[1:0] = 00_B)

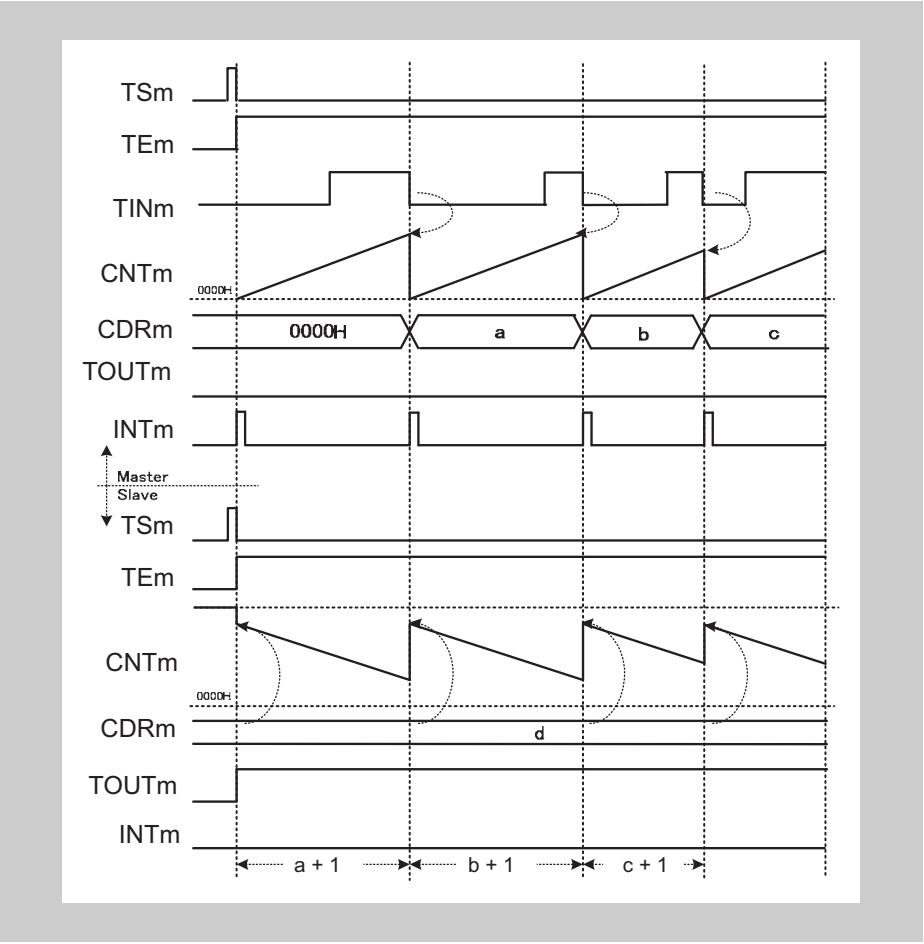


Figure 15-115 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

- If the value TAUAnCDRm (slave) is higher than the interval of valid input edges, the counter of the slave channel cannot reach 0000_H and cannot generate interrupts. The TAUAnTTOUTm remains at active state.

15.25 Synchronous Triangle PWM Functions

This chapter describes functions that generate a triangle PWM output.

- *15.25.1 “Triangle PWM Output Function”*
- *15.25.2 “Triangle PWM Output Function with Dead Time”*
- *15.25.3 “AD Conversion Trigger Output Function Type 2”*

15.25.1 Triangle PWM Output Function

(1) Overview

Summary This function generates multiple triangle PWM outputs by using a master and one or more slave channels. It enables the pulse cycle (frequency) and the duty cycle of TAUAnTTOUTm to be set using the master and slave channel(s) respectively.

The slave channel generates a carrier cycle from two pulse cycles. The first pulse of the master channel controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUAnCNTm (slave) means that signal duration of TAUAnTTOUTm (slave) is double that of the difference between TAUAnCDRm (master) +1 and TAUAnCDRm (slave).

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-155 “TAUAnCMORm settings for the master channel of the Triangle PWM Output Function” on page 944*
- The operation mode of the slave channel(s) must be set to Up Down Count Mode, refer to *Table 15-159 “TAUAnCMORm settings for the slave channel of the Triangle PWM Output Function” on page 946*
- The channel output mode of the master channel must be set to Independent Channel Output Mode 1 (refer to *15.9 “Channel Output Modes” on page 731*)
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 2 (refer to *15.9 “Channel Output Modes” on page 731*)
- The following settings establish TAUAnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUAnCMORm.MD0 (master) bit is set to 0,TAUAnTO.TOm must be set to 1 while TAUAnTOE.TOEm is 0.
  - If the TAUAnCMORm.MD0 (master) bit is set to 1, TAUAnTO.TOm must be set to 0 while TAUAnTOE.TOEm is 0.

Description The counters are started by setting the channel trigger bit (TAUAnTS.TSm) to 1 for every channel. This in turn sets TAUAnTE.TEm, enabling count operation. The current values of TAUAnCDRm (master and slave) are written to TAUAnCNTm (master and slave) and the counters start to count down from these values. Depending on the setting of the master channel TAUAnCMORm.MD0 bit an interrupt is generated and TAUAnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. TAUAnCNTm then reloads the TAUAnCDRm value and counts down.

- Slave channel:

  The INTTAUAnIm of the master channel triggers the counter of the slave channel:

  – If the slave counter currently counts down, it changes count direction .

  – If the slave counter currently counts up, the value of TAUAnCDRm is reloaded and the counter counts down.

  When the counter of the slave channel reaches 0001_H while counting up or down, INTTAUAnIm is generated and the TAUAnTTOUTm (slave) signal toggles:

  – It is set in the count-down status

  – It is reset in the count-up status

  The counter continues to count down or up and awaits the next INTTAUAnIm of the master channel.

TAUAnTTOUTm can be switched between positive and negative phase setting TAUAnTOL.TOLm during operation.

The counters can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values.

Note If a forced restart is executed during operation, TAUAnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 “Simultaneous Rewrite” on page 719*

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUAnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUAnCDRm (master) + 1 - TAUAnCDRm (slave))/ (TAUAnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUAnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

(3) Block diagram and general timing diagram

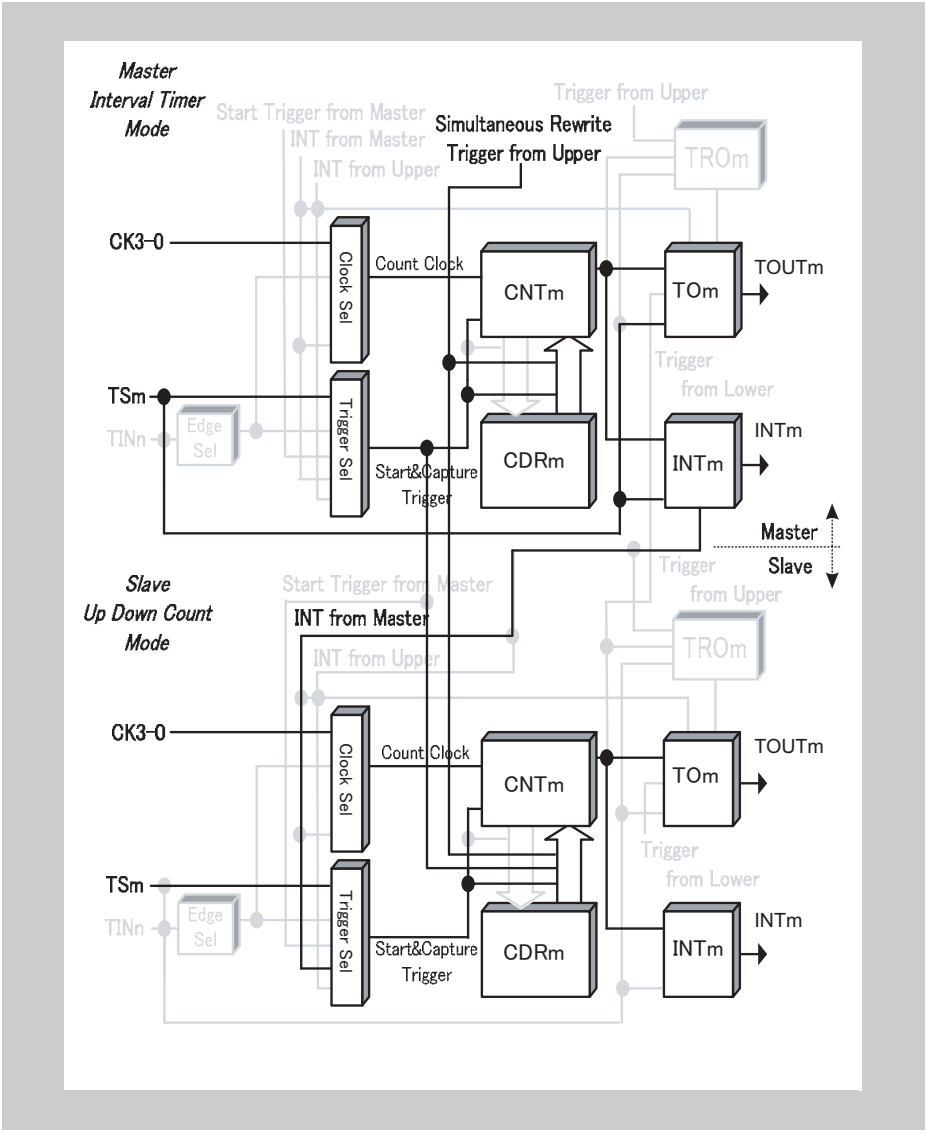


Figure 15-116 Block diagram for Triangle PWM Output Function

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)

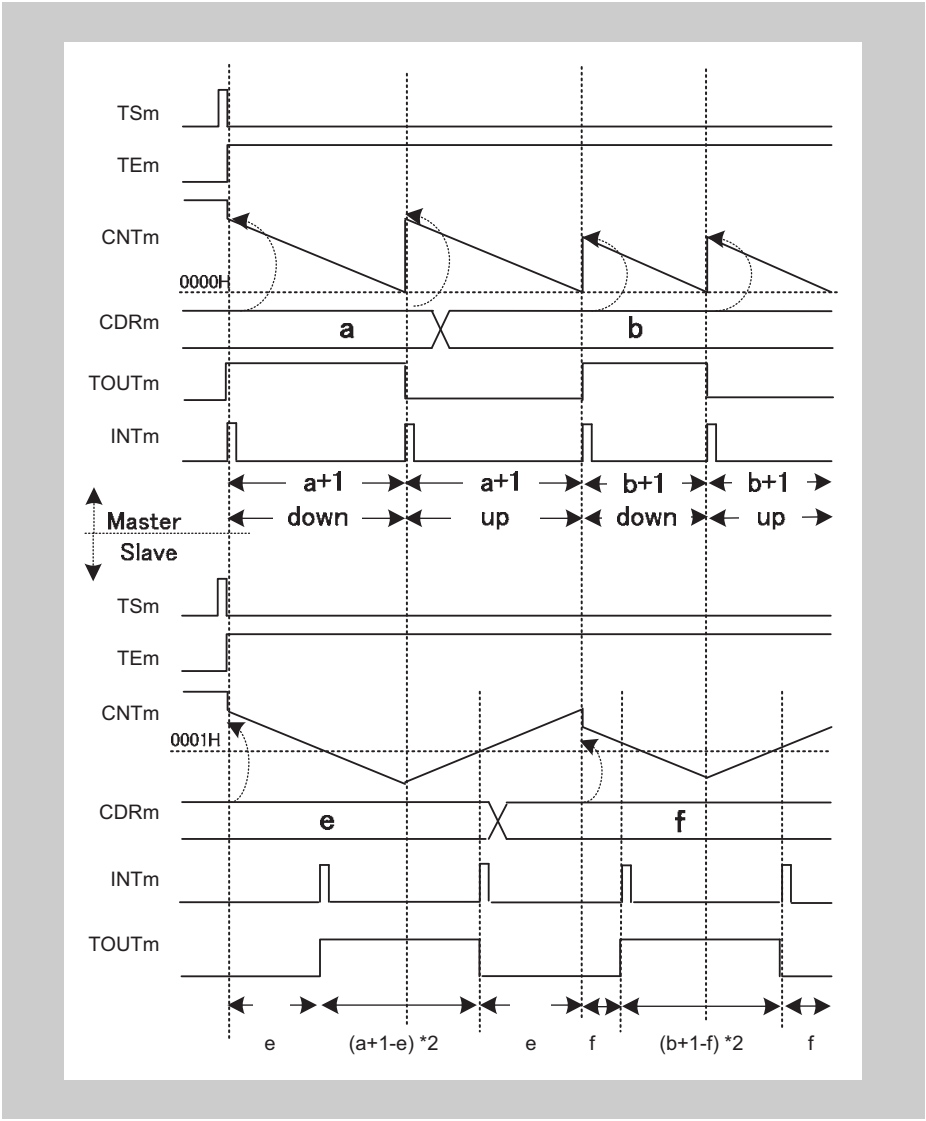


Figure 15-117 General timing diagram for Triangle PWM Output Function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-155 TAUAnCMORm settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-156 TAUAnCMURm settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 15-157 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-158 Simultaneous rewrite settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUAnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-159 TAUAnCMORm settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-160 TAUAnCMURm settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 15-161 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-162 Simultaneous rewrite settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for Triangle PWM Output Function

Table 15-163 Operating procedure for Triangle PWM Output Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 944* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 946* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel, depending on TAUAnCMORm.MD0.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master and slave channel loads TAUAnCDRm and counts down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnTTOUTm (master) toggles • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation. • TAUAnCNTm (slave) reloads the TAUAnCDRm value or counts in the reverse direction. When TAUAnCNTm of the slave = 0001_H: • INTTAUAnIm (slave) is generated • TAUAnTTOUTm (slave) is set (in count-down status) or reset (in count-up status)
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm is initialized to the value set by TAUAnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)
  - TAUAnCDRm = a = 5_H
- Slave channel:
  - TAUAnCDRm = 6_H

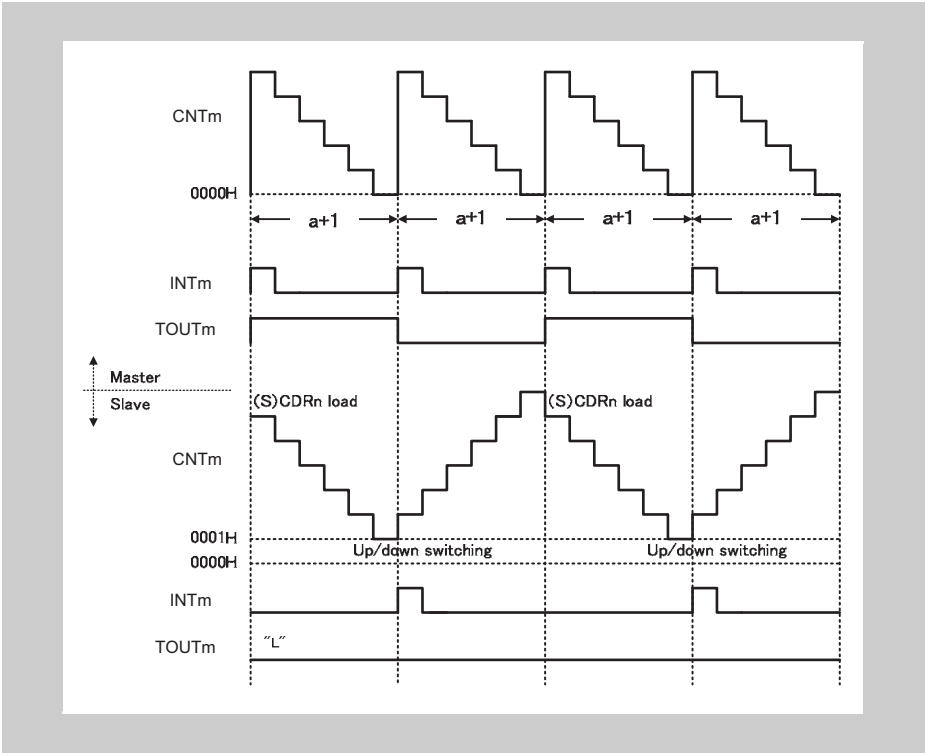


Figure 15-118 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

- If TAUAnCDRm (slave) ≥ TAUAnCDRm (master) the counter of slave channel cannot reach 0001_H during *counting down*. The set signal is never detected, so TAUAnTTOUTm remains at low state.

(b) Duty cycle = 100 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)
  - TAUAnCDRm = a = 5_H
- Slave channel:
  - TAUAnCDRm = 0_H

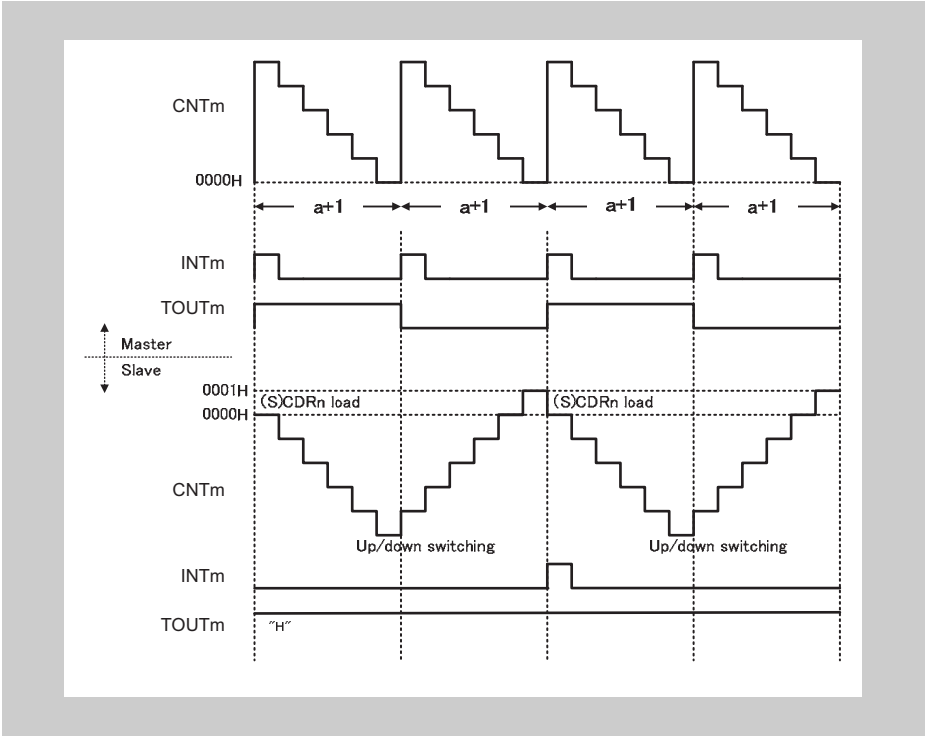


Figure 15-119 TAUAnCDRm (slave) = 0000_H

- If TAUAnCDRm (slave) =0000_H the counter of slave channel cannot reach 0001_H during *counting up*. The reset signal is never detected, so TAUAnTTOUTm remains at high state.

15.25.2 Triangle PWM Output Function with Dead Time

(1) Overview

Summary This function generates multiple triangle PWM outputs with a defined dead time by using a master and two or more slave channels. The resulting PWM signals are output via TAUAnTTOUTm of the slave channels 2 and 3. It enables the pulse cycle (frequency) and the duty cycle of TAUAnTTOUTm to be set using the master and slave channel(s) respectively.

Slave channel 2 generates a carrier cycle from two pulse cycles from the master channel. The first pulse controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUAnCNTm (slave 2) means that signal duration of TAUAnTTOUTm (slave) is double that of the difference between TAUAnCDRm (master) +1 and TAUAnCDRm (slave 2).

An interrupt on slave 2 causes TAUAnTTOUTm of the slave channels to toggle. Depending on the settings of TAUAnTDL.TDLm, delay time is added to positive or negative logic side of the signal (i.e. whether TAUAnTTOUTm toggles immediately or after dead time has elapsed). The duration of the dead time is specified by slave channel 3.

Prerequisites

- Three channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-165 “TAUAnCMORm settings for the master channel of the Triangle PWM Output Function with Dead Time” on page 956*
- Slave channel 1 is not used for this function. This ensures that slave channel 2 is an odd channel, and slave channel 3 is an even channel.
- The operation mode of slave channel 2 must be set to Up Down Mode, refer to *Table 15-169 “TAUAnCMORm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time” on page 958*
  Furthermore, slave channel 2 must be an even channel
- The operation mode of slave channel 3 must be set to One Count Mode, refer to *Table 15-173 “TAUAnCMORm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time” on page 960*
  Furthermore, slave channel 3 must be an odd channel
- The channel output mode of the master channel must be set to Independent Channel Output Mode 1 (refer to *15.9 “Channel Output Modes” on page 731*)
- The channel output mode of the slave channels 2 and 3 must be set to Synchronous Channel Output Mode 2 with Dead Time Output (refer to *15.9 “Channel Output Modes” on page 731*)
- The following settings establish TAUAnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUAnCMORm.MD0 (master) bit is set to 0,TAUAnTO.TOm must be set to 1 while TAUAnTOE.TOEm is 0.
  - If the TAUAnCMORm.MD0 (master) bit is set to 1, TAUAnTO.TOm must be set to 0 while TAUAnTOE.TOEm is 0.

Note Slave channel 1 is not used for Triangle PWM Output Function with Dead Time.

Description The counters are started by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The current values of TAUAnCDRm is written to TAUAnCNTm and the counters start to

count down from these values. Depending on the setting of the master channel TAUAnCMORm.MD0 bit an interrupt is generated and TAUAnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated and the TAUAnTTOUTm signal toggles. The counter reloads the TAUAnCDRm value and counts down.

- Slave channel 2:

  The INTTAUAnIm of the master channel triggers the counter of the slave channel 2:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUAnCDRm is reloaded and the counter counts down.

  The counter continues to count down or up and awaits the next INTTAUAnIm of the master channel.

- Slave channel 3:

  INTTAUAnIm of slave channel 2 triggers the counter of slave channel 3. The current value of TAUAnCDRm (slave 3) is written to TAUAnCNTm (slave 3) and the counter starts to count down from this value.

  When the counter reaches 0000_H, dead time has elapsed, INTTAUAnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUAnIm of slave channel 2.

An interrupt on slave channel 2 causes TAUAnTTOUTm to toggle as follows:

- It is set by the interrupt when slave channel 2 is counting down
- It is reset by the interrupt when slave channel 2 is counting up.

However, the TAUAnTDL.TDLm settings of the corresponding channel specify whether it is set/reset immediately, or after dead time has elapsed, as shown in *Table 15-164 “Behavior of TAUAnTTOUTm when an interrupt occurs on slave channel 2” on page 953* .

The TAUAnTOL.TOLm settings specify whether set corresponds to a high signal (TAUAnTOL.TOLm = 0) or a low signal (TAUAnTOL.TOLm = 1).

In Triangle PWM Output Function with Dead Time, TAUAnTTOUTm can be switched between positive and negative phase by setting TAUAnTOL.TOLm during operation.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel(s) stop but retain their values.

TAUAnCDRm value of slave channel 2 can be set to 0000_H to output 100 % TAUAnTTOUTm.

Note If a forced restart is executed during operation, TAUAnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*

TAUAnTOL.TOLm bits should be set before the counter starts, and slave channels 2 and 3 should have opposite TAUAnTOL.TOLm settings or opposite TAUAnTDL.TDLm settings.

Table 15-164 Behavior of TAUAnTTOUTm when an interrupt occurs on slave channel 2

TAUAnTDL.TDLm	Count direction of slave channel 2 when interrupt is generated	TAUAnTTOUTm toggles
0	Down	Set after dead time has elapsed
	Up	Reset immediately
1	Down	Set immediately
	Up	Reset after dead time has elapsed

Notes

1. When the set and reset condition for TAUAnTTOUTm occur simultaneously:
   - If TAUAnTOL.TOLm = 0, the reset condition has priority
   - If TAUAnTOL.TOLm = 1, the set condition has priority .
2. To generate a two-phase PWM output with added dead time, the settings of TAUAnTOL.TOLm bit of slave channels 2 and 3 must not be changed.

(2) Equations

Pulse cycle = (TAUAnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUAnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUAnCDRm (master) + 1 - TAUAnCDRm (slave))/ (TAUAnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUAnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

PWM signal width (positive phase) = [((TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2) × 2) - (TAUAnCDRm (slave 3) + 1)] × count clock cycle

PWM signal width (negative phase) = [((TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2) × 2) + (TAUAnCDRm (slave 3) + 1)] × count clock cycle

(3) Block diagram and general timing diagram

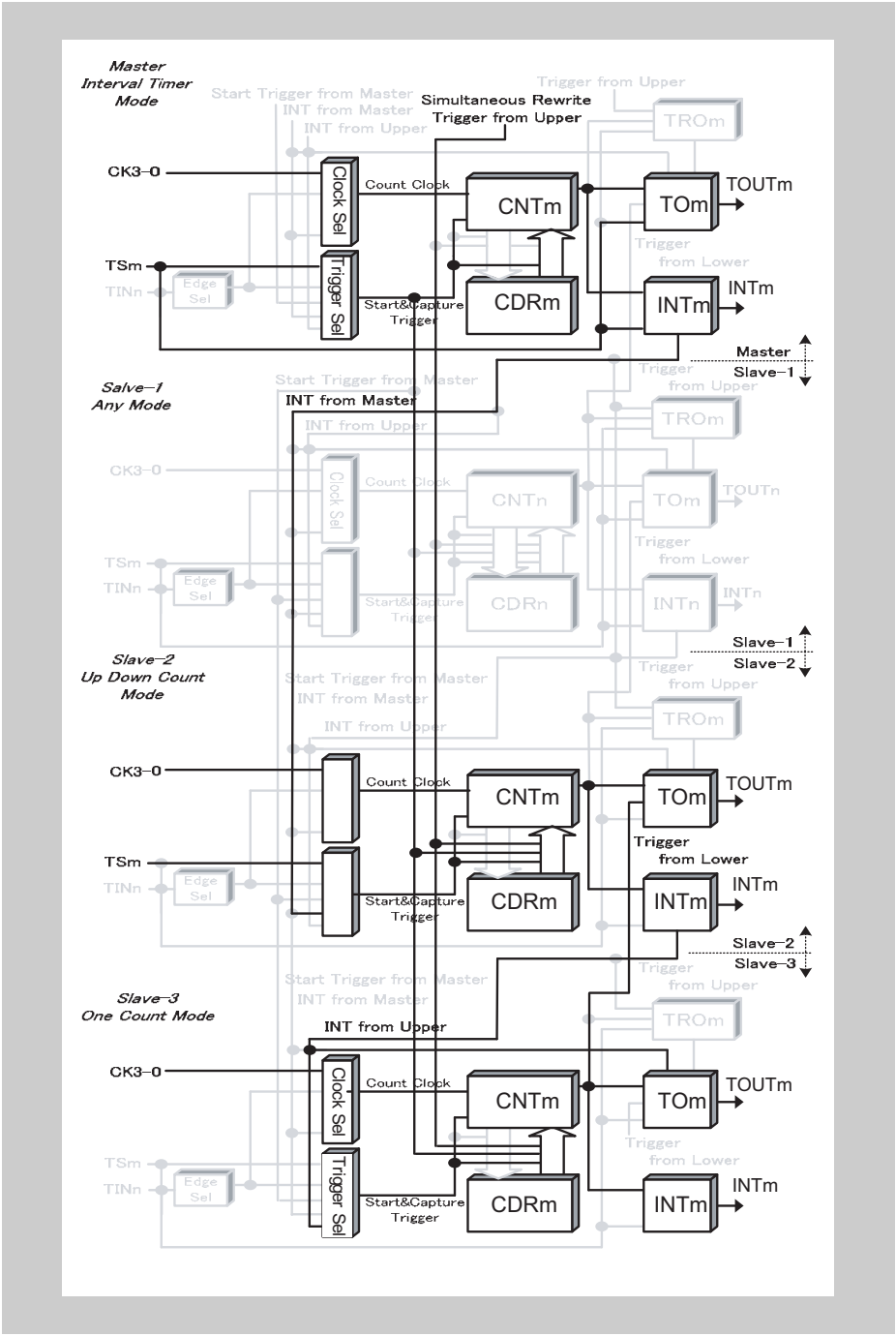


Figure 15-120 Block diagram for Triangle PWM Output Function with Dead Time

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)
- Slave channel 2:
  - INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
  - TAUAnTDL.TDLm = 0
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)
  - TAUAnTDL.TDLm = 1
  - Negative logic (TAUAnTOL.TOLm = 1)

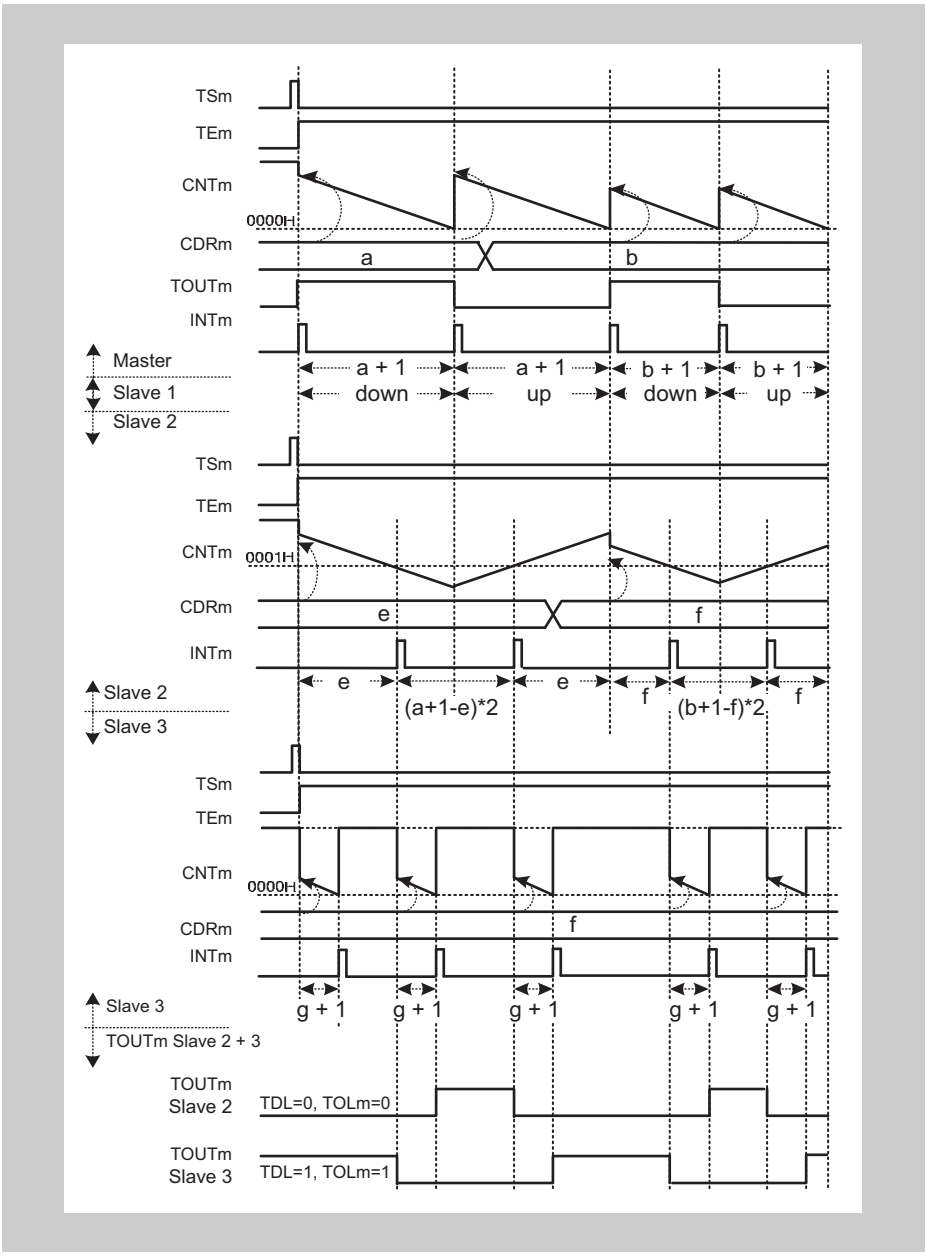


Figure 15-121 General timing diagram for Triangle PWM Output Function with Dead Time

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-165 TAUAnCMORm settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-166 TAUAnCMURm settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 15-167 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDLm	
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-168 Simultaneous rewrite settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUAnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-169 TAUAnCMORm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUAnIm not generated at operation start

(b) TAUAnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-170 TAUAnCMURm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

Table 15-171 Control bit settings for Synchronous Channel Output Mode 2 with Dead Time Output

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDMm	0: Dead time is added upon detection of an interrupt on the upper even channel, if the TAUAnTDL.TDLm condition is also fulfilled.
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-172 Simultaneous rewrite settings for slave channel 2 of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUAnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-173 TAUAnCMORm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	110: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-174 TAUAnCMURm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 15-175 Control bit settings for Synchronous Channel Output Mode 2 with Dead Time Output

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDMm	0: Dead time is added upon detection of an interrupt on the upper even channel, if the TAUAnTDL.TDLm condition is also fulfilled.
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TREm	0: Disables real-time output
TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRCm	
TMEm	0: Disables modulation

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-176 Simultaneous rewrite settings for slave channel 3 of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Triangle PWM Output Function with Dead Time

Table 15-177 Operating procedure for Triangle PWM Output Function with Dead Time (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 956* Slave channel 2: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 2" on page 958* Slave channel 3: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for slave channel 3" on page 960* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 15-177 Operating procedure for Triangle PWM Output Function with Dead Time (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel., depending on the setting of TAUAnCMORm.MD0.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel and slave channel 2 load TAUAnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation • TAUAnCNTm (slave 2) reloads the TAUAnCDRm value or counts in the reverse direction When TAUAnCNTm (slave 2) reaches 0001_H: • INTTAUAnIm (slave 2) is generated • TAUAnTTOUTm is reset (TAUAnTOL.TOLm=0) • TAUAnCNTm of slave channel 3 reloads the TAUAnCDRm value and counts down When TAUAnCNTm of slave channel 3 = 0000_H: • INTTAUAnIm is generated • TAUAnTTOUTm is set (TAUAnTOL.TOLm=0) Whether TAUAnTTOUTm (slave 2 and 3) are set/reset with INTTAUAnIm (slave 2) simultaneously or after dead time has elapsed depends on the settings in TAUAnTDL.TDLm and TAUAnTOL.TOLm (refer to *Table 15-164 "Behavior of TAUAnTTOUTm when an interrupt occurs on slave channel 2" on page 953*).
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm of slave channels 2 and 3 is initialized to the value set by TAUAnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

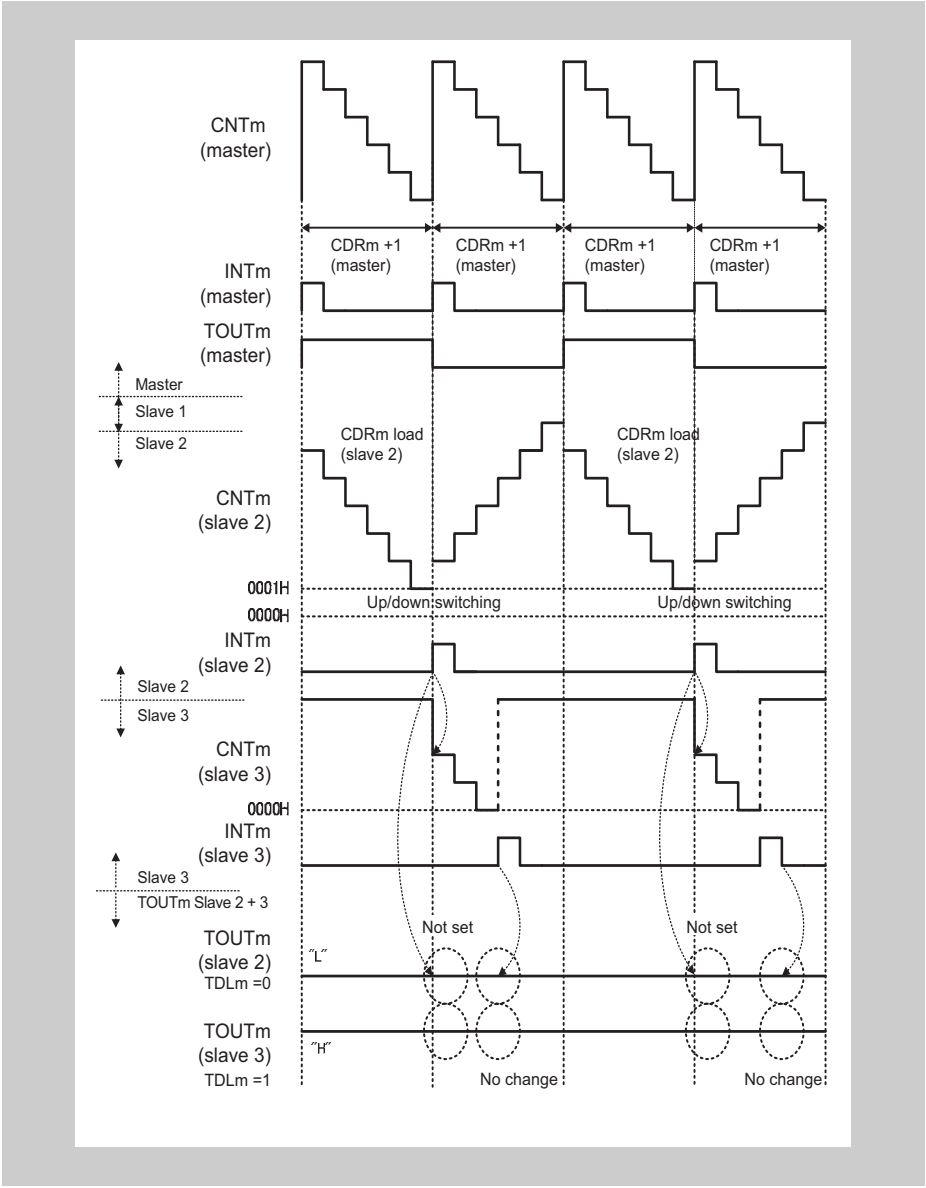


Figure 15-122 TAUAnCDRm (slave) ≥ TAUAnCDRm (master) + 1

- If TAUAnCDRm (slave 2) ≥ TAUAnCDRm (master) the counter of slave channel cannot reach 0000_H during counting down. Therefore TAUAnTTOUTm cannot toggle, i.e. it remains at its initial state. The interrupt from slave channel 2 occurs during count up, therefore it is a reset signal.

(b) Duty cycle = 100 %

The following settings apply to the diagram below:

- Slave channels 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channels 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

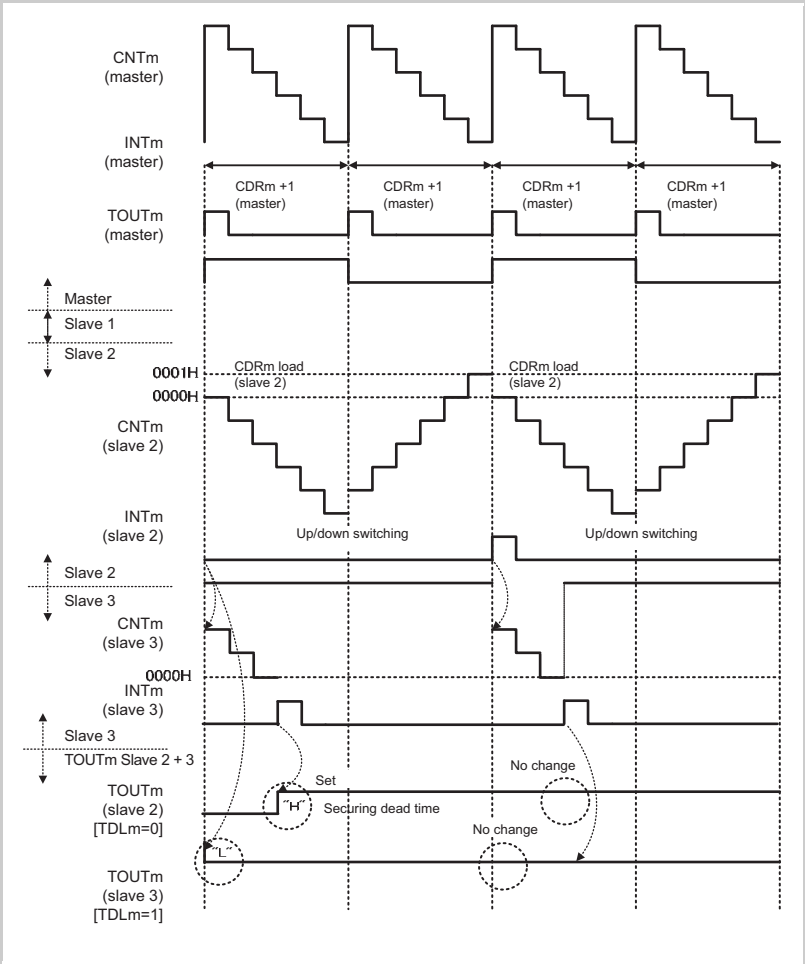


Figure 15-123 TAUAnCDRm (slave) = 0000_H

- If TAUAnCDRm (slave 2) =0000_H the counter of slave channel cannot reach 0001_H while counting up and therefore cannot generate an INTTAUAnIm while counting up.
  - The set conditions for a channel in which TAUAnTDL.TDLm = 0 are met after dead time has elapsed. TAUAnTTOUTm toggles but remains in the new state because the reset conditions never occur for such a channel.
  - Slave channel 3 in the diagram above is set when the counter starts. However, the reset conditions for a channel in which TAUAnTDL.TDLm = 1 never occur so TAUAnTTOUTm remains in its initial state for such a slave channel. In slave 3 in the diagram above, the initial state is high because TAUAnTOL.TOLm = 1.

(c) TAUAnTTOUTm (slave 2) = 0 % and TAUAnTTOUTm (slave 3) > 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

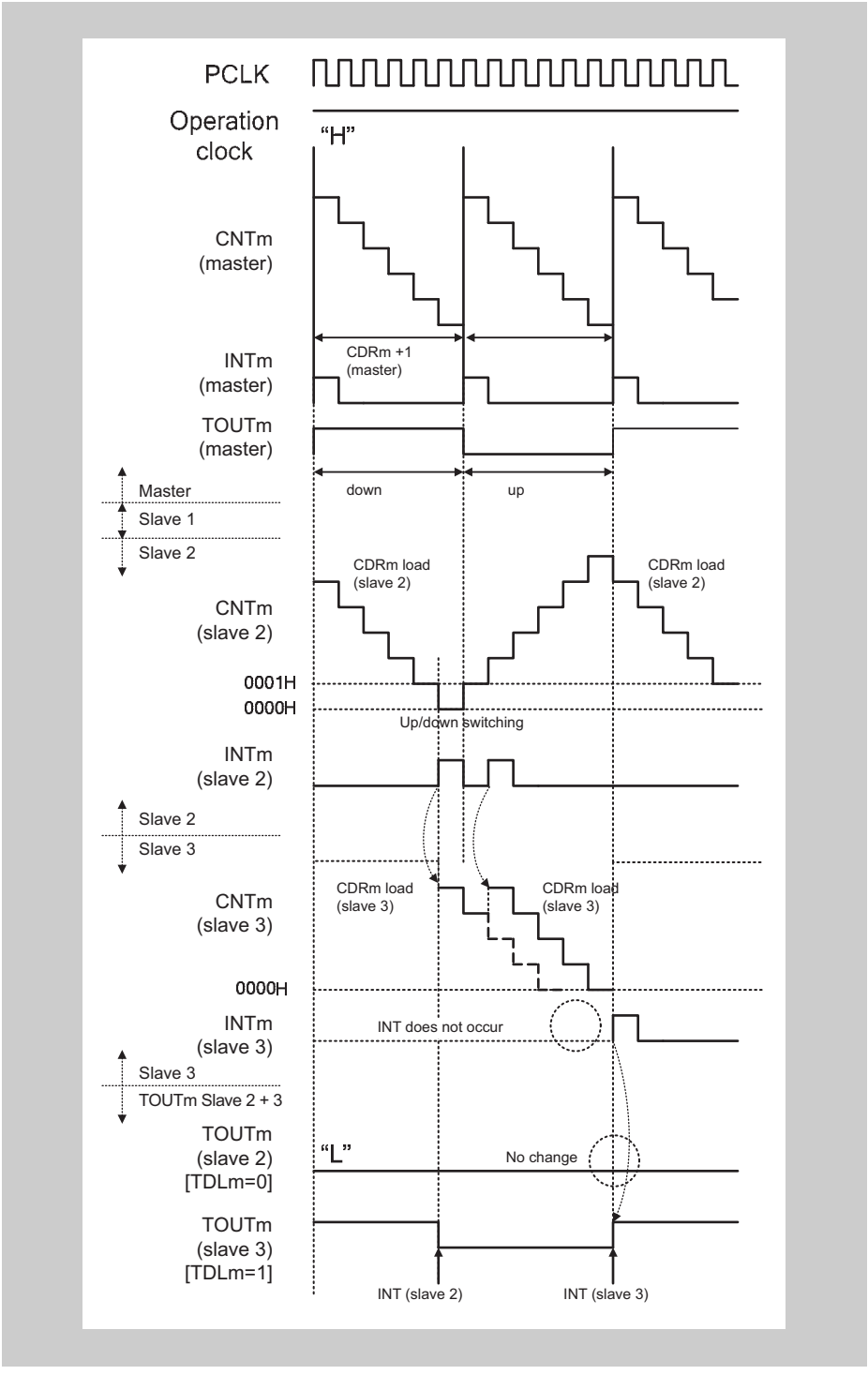


Figure 15-124 TAUAnCDRm (master) = 0005_H, TAUAnCDRm (slave 2) = 0005_H, TAUAnCDRm (slave 3) = 0004_H

- When the counter of slave channel 2 reaches 0000_H, INTTAUAnIm (slave 2) is generated. The counter of slave channel 3 starts to count down.
- If another INTTAUAnIm (slave 2) is generated while the counter of slave channel 3 is still counting down, the value of TAUAnCDRm (slave 3) is reloaded and the counter restarts counting down from this value.
- In the diagram above, the first interrupt on channel 2 occurs while the counter is counting down, and the second whilst it is counting up.
- After the first interrupt, a slave for which TAUAnTDL.TDLm = 0 waits for dead time to elapse before setting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a reset signal, meaning that a channel for which TAUAnTDL.TDLm = 0 always remains inactive.
- TAUAnTTOUTm of a slave channel for which TAUAnTDL.TDLm = 1 is set and reset as normal when the corresponding INTTAUAnIm is generated.

(d) TAUAnTTOUTm (slave 2) > 0 % and TAUAnTTOUTm (slave 3) = 100 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

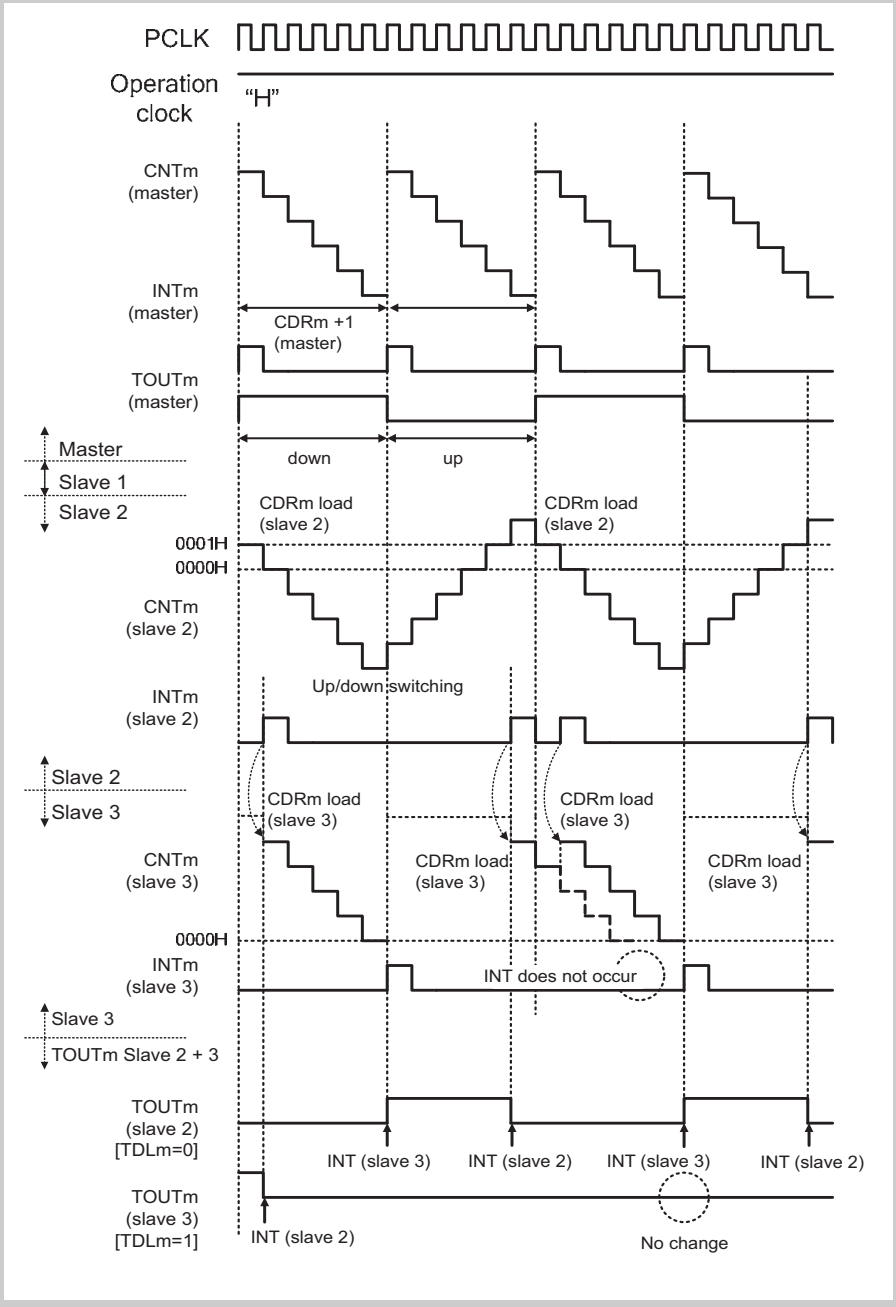


Figure 15-125 TAUAnCDRm (master) = 0005_H, TAUAnCDRm (slave 2) = 0002_H, TAUAnCDRm (slave 3) = 0004_H PWM signal width (negative phase) ≥ Carrier cycle

- After the second interrupt, a slave for which TAUAnTDL.TDLm = 1 waits for dead time to elapse before resetting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a set signal, meaning that a channel for which TAUAnTDL.TDLm = 1 always remains active.
- TAUAnTTOUTm of a slave channel for which TAUAnTDL.TDLm = 0 is set and reset as normal when the corresponding INTTAUAnIm is generated.

(e) Inhibited INTTAUAnIm to set TAUAnTTOUTm positive phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

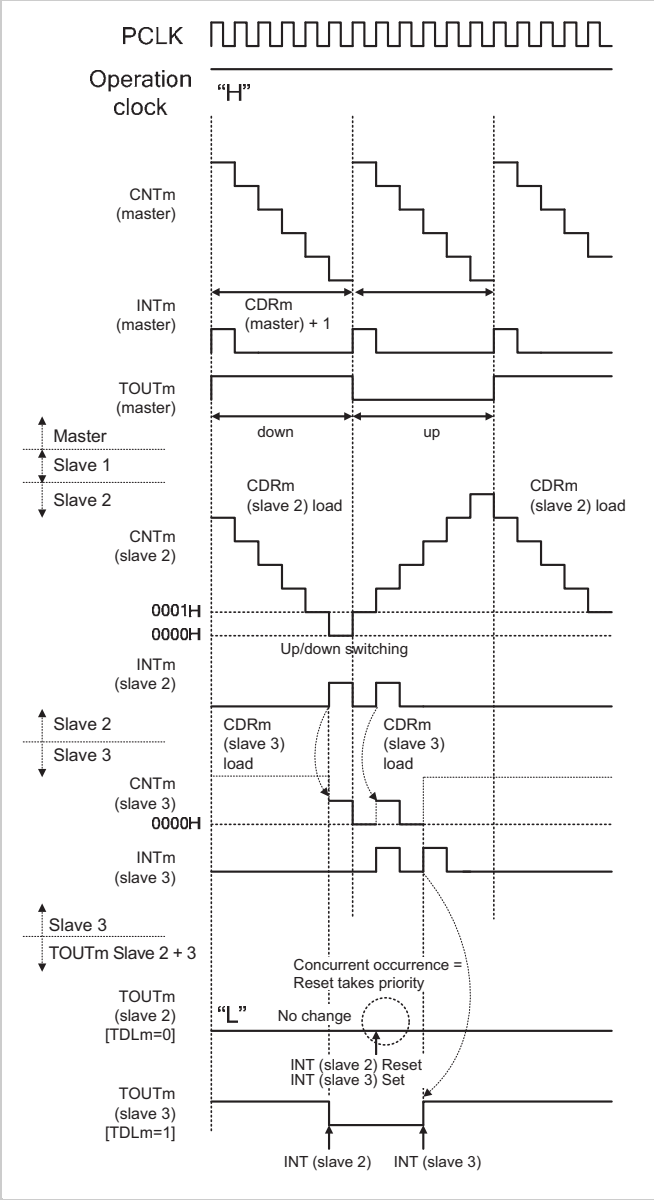


Figure 15-126 TAUAnCDRm (master) = 0005_H, TAUAnCDRm (slave 2) = 0005_H, TAUAnCDRm (slave 3) = 0001_H PWM signal width (positive phase) = 0

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUAnIm to set the TAUAnTTOUTm of slave channel for which TAUAnTDL.TDLm = 0 (slave channel 2 in this example).
- If channel 2 generates an INTTAUAnIm to reset TAUAnTTOUTm simultaneously, this reset signal has priority (assuming TAUAnTOL.TOLm = 0, otherwise the set signal has priority).
- Therefore, TAUAnTTOUTm of a slave channel for which TAUAnTDL.TDLm = 0 remains in its initial state.

(f) Inhibited INTTAUAnIm to set TAUAnTTOUTm negative phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUAnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUAnTOL.TOLm = 1)

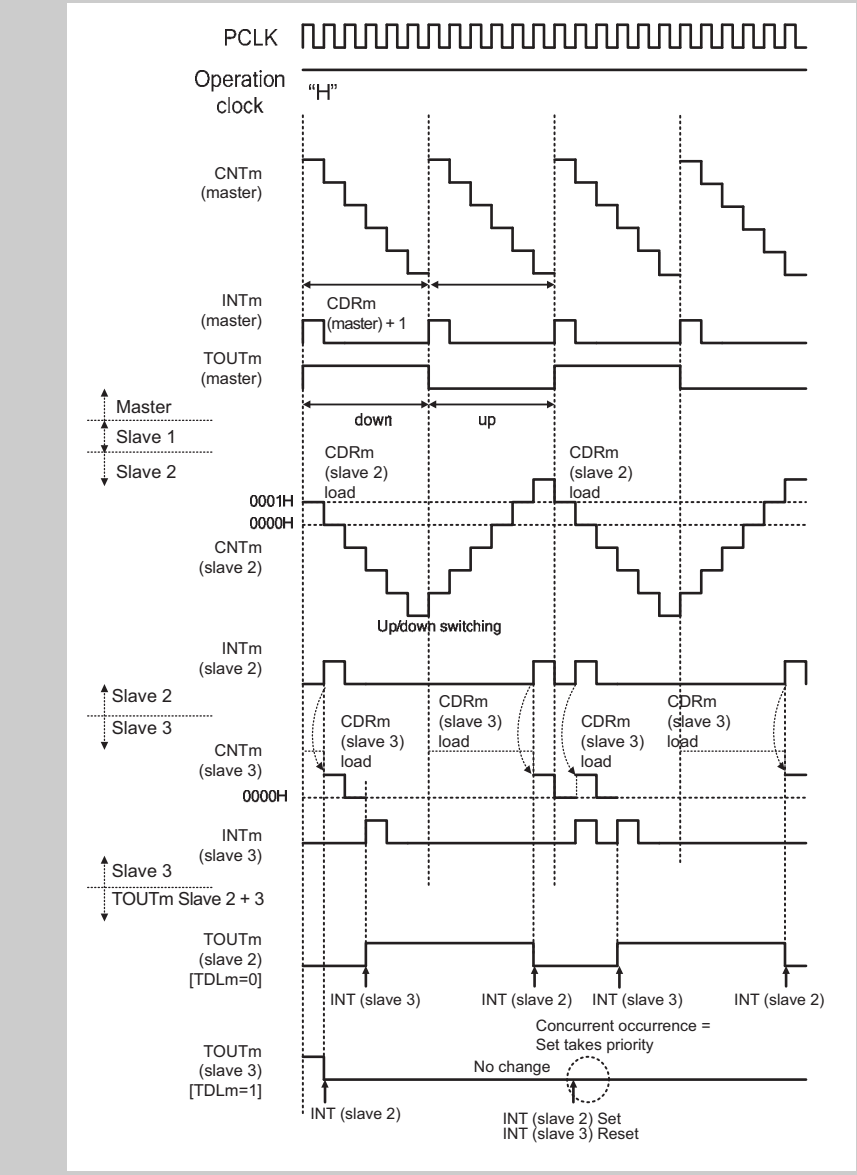


Figure 15-127 TAUAnCDRm (master) = 0005_H, TAUAnCDRm (slave 2) = 0001_H, TAUAnCDRm (slave 3) = 0001_H
PWM signal width (negative phase) = carrier cycle

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUAnIm to set the TAUAnTTOUTm of slave channel for which TAUAnTDL.TDLm = 1 (slave 3 in this example).
- If channel 2 generates an INTTAUAnIm to reset TAUAnTTOUTm simultaneously, the set signal has priority (assuming TAUAnTOL.TOLm = 1, otherwise the reset signal has priority).
- Therefore, TAUAnTTOUTm of slave channel for which TAUAnTDL.TDLm = 1 remains in its initial state.

15.25.3 AD Conversion Trigger Output Function Type 2

(1) Overview

Summary This function is identical to *15.25.1 "Triangle PWM Output Function" on page 940* except that TAUAnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to Direct Channel Output Mode.

(2) Block diagram and general timing diagram

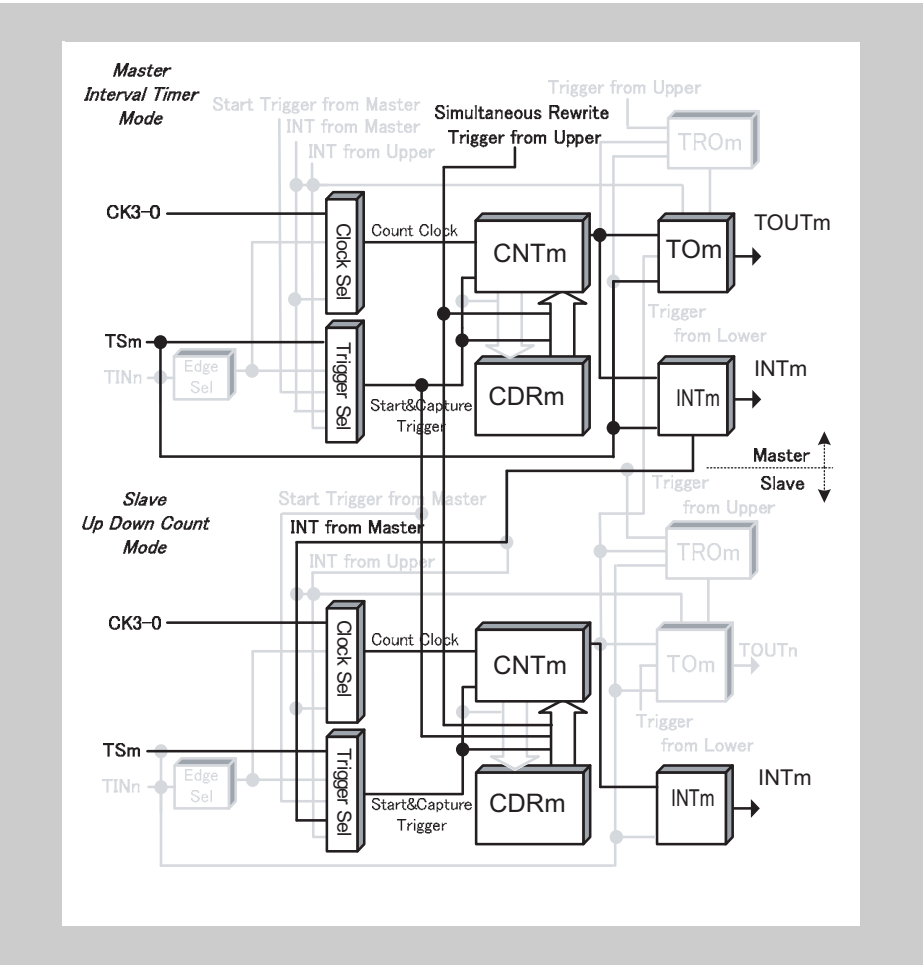


Figure 15-128 Block diagram for AD Conversion Trigger Output Function Type 2

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)

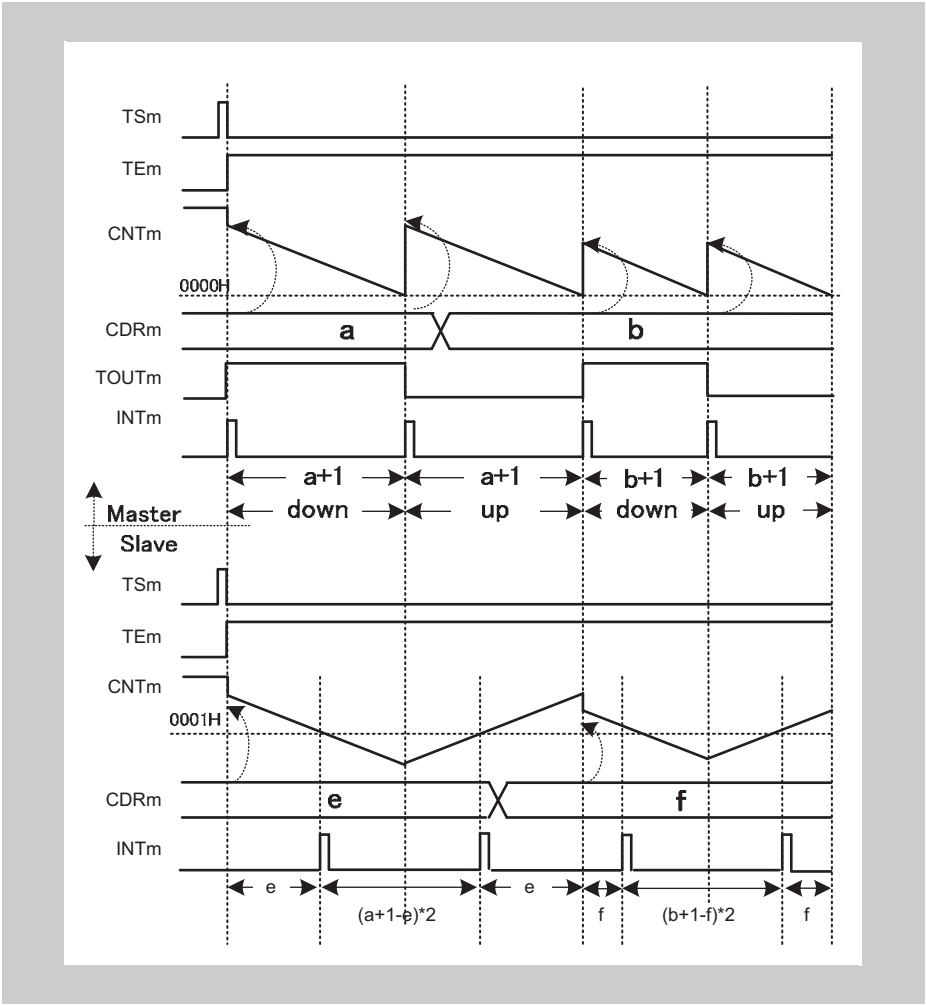


Figure 15-129 General timing diagram for AD Conversion Trigger Output Function Type 2

15.26 Synchronous Real-Time Output Functions

This chapter describes functions that output the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt. The interrupt is generated at certain specified intervals with dead time added, after a specified number of interrupts on the master channel have occurred, or a combination thereof.

These functions differ from the Independent Real-Time Output Functions, in that the real-time output trigger is generated by combining the timers of more than one channel. In the Independent Real-Time Output Functions, the trigger is generated using a single channel, even though this trigger can be detected by multiple channels.

- *15.16.1 “Interval Timer Function”*
- *15.26.2 “Synchronous Real-Time Output Function Type 2”*
- *15.26.3 “Synchronous Real-Time Output Function Type 3”*

15.26.1 Synchronous Real-Time Output Function Type 1

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). In this function, the interrupt is generated at certain specified intervals, but with dead time added. The dead time is specified by the upper slave channel.

Prerequisites

- Three (or more) channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-17 “TAUAnCMORm settings for Interval Timer Function” on page 754*
- The operation mode of the slave channels must be set to One Count Mode, refer to *Table 15-181 “TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 1” on page 983*
- TAUAnTTOUTm is not used for the master channel of this function
- The channel output mode of the upper slave must be set to Independent Channel Output Mode 1 with Real-Time Output, and TAUAnTRC.TRCm must be set to 1 (refer to *15.9 “Channel Output Modes” on page 731*)
- The channel output mode of the lower slave channels must be set to Synchronous Channel Output Mode 1 with Non-Complementary Modulation Output (refer to *15.9 “Channel Output Modes” on page 731*)

Description The counters (master and slave) are started by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm (master and slave) is written to TAUAnCNTm and the counters start to count down from these values.

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

- Upper slave channel:

  When the upper slave detects an interrupt from the master channel, it starts to count down from the current value of TAUAnCDRm. When the counter reaches 0000_H, it generates an INTTAUAnIm. TAUAnTTOUTm (slave) outputs the current value of the real-time output bit (TAUAnTRO.TROm). TAUAnCNTm then reloads the TAUAnCDRm value and awaits the next INTTAUAnIm from the master channel.

- Lower slave channel:

  This has the same start trigger as the upper slave channel, but TAUAnTRC.TRCm = 0.

- Upper and lower slave channel:

  The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel for an interrupt. The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.

The value of TAUAnCDRm (master and slave) can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*

(2) Equations

Amount INTTAUAnIm of the channel that generates the real-time output trigger is delayed by compared to the INTTAUAnIm of the master channel

= [TAUAnCDRm (upper slave) + 1] x count clock cycle

(3) Block diagram and general timing diagram

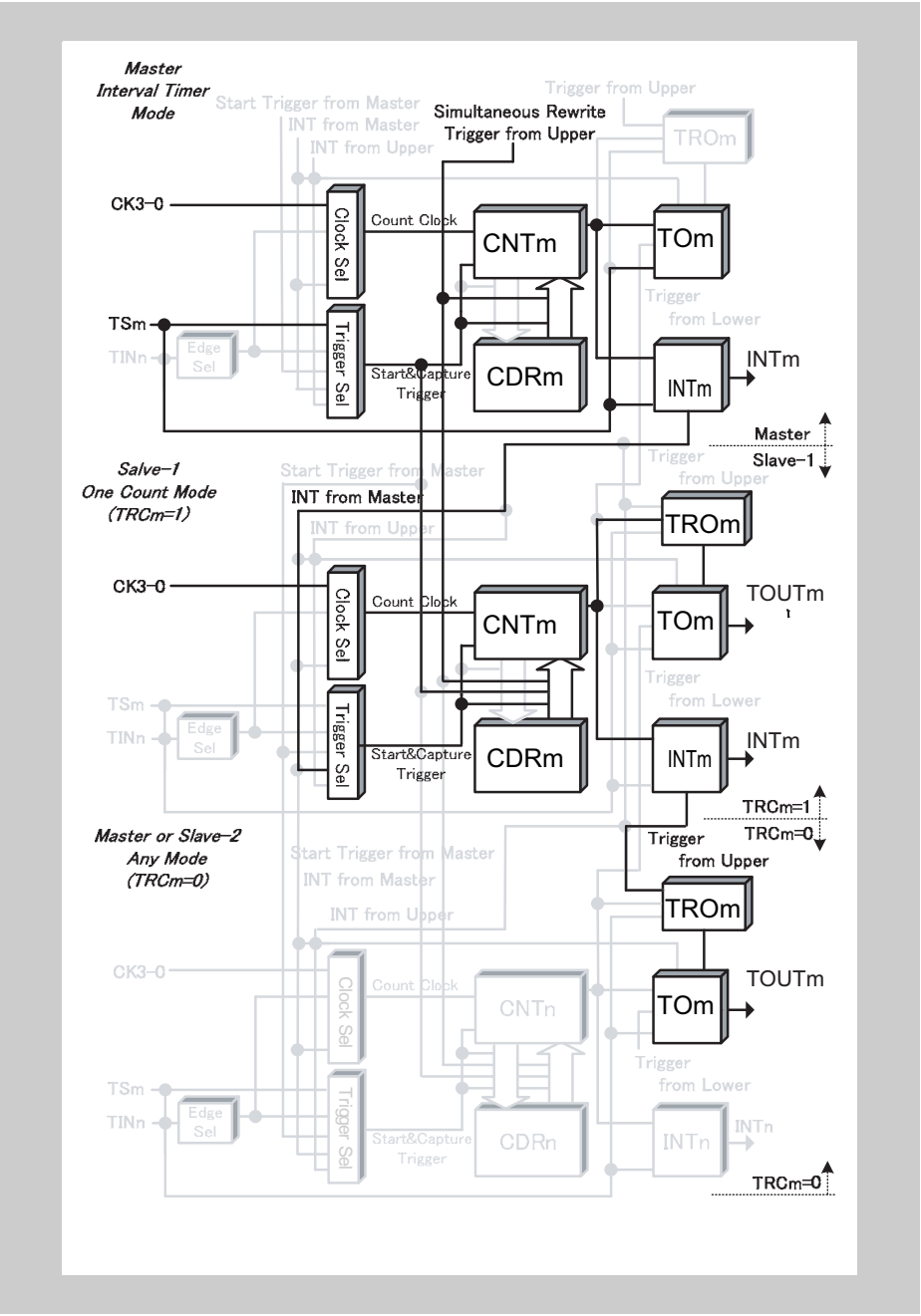


Figure 15-130 Block diagram for Real-Time Output Function Type 1

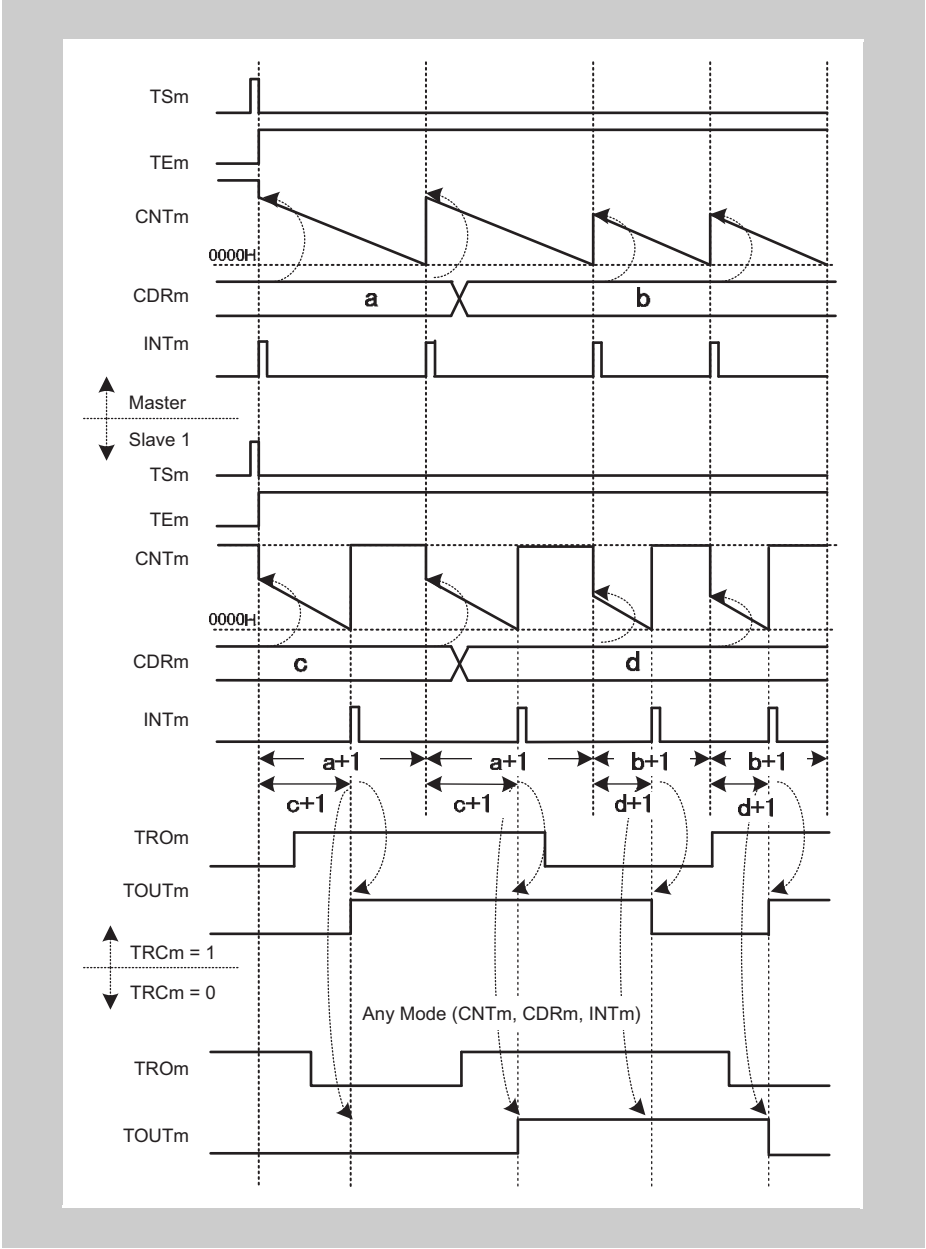


Figure 15-131 General timing diagram for Real-Time Output Function Type 1

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-178 TAUAnCMORm settings for the master channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-179 TAUAnCMURm settings for the master channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite of the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-180 Simultaneous rewrite settings for the master channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the upper slave channel

(a) TAUAnCMORm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-181 TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm on the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-182 TAUAnCMURm settings for the upper slave channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the upper slave channel

Table 15-183 Control bit settings for Independent Channel Output Mode 1 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite of the upper slave channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-184 Simultaneous rewrite settings for the upper slave channel of Synchronous Real-Time Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for the lower slave channel

(a) TAUAnCMORm for the lower slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-185 TAUAnCMORm settings for Synchronous Real-Time Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for the lower slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-186 TAUAnCMURm settings for Synchronous Real-Time Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the lower slave channel

Table 15-187 Control bit settings for Synchronous Channel Output Mode 1 with Non-Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	0: Operation mode 1
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 "Channel Output Modes" on page 731* .

(d) Simultaneous rewrite for the lower slave channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-188 Simultaneous rewrite settings for Synchronous Real-Time Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Synchronous Real-Time Output Function Type 1

Table 15-189 Operating procedure for Synchronous Real-Time Output Function Type 1

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 981* Upper slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the upper slave channel" on page 983* Lower slave channels: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for the lower slave channel" on page 985* Set the values of the TAUAnCDRm registers of all channels Set TAUAnTRC.TRCm = 1 for the upper slave channel	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counter of the master channel starts. INTTAUAnIm is generated on the master channel, which causes the counter of the upper slave channel to start.
During operation	TAUAnCDRm and TAUAnTRO.TROm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues counting down • TAUAnCNTm of the upper slave channel loads TAUAnCDRm and counts down • When the counter of the upper slave channel reaches 0000_H: - INTTAUAnIm (upper slave) is generated - The counter reloads the TAUAnCDRm value and awaits the next interrupt from the master channel. - The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel for an interrupt.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

15.26.2 Synchronous Real-Time Output Function Type 2

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). It is the same as Synchronous Real-Time Output Function Type 1 except that dead-time is not applied. Instead, an interrupt is only generated after a certain number of intervals, specified by the upper slave channel, have elapsed on the master channel.

Prerequisites

- Three (or more) channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-190 “TAUAnCMORm settings for the master channel of Synchronous Real-Time Output Function Type 2” on page 992*
- The operation mode of the upper slave channel must be set to Event Count Mode, refer to *Table 15-193 “TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 2” on page 994*
- The operation mode of the lower slave channels can be set arbitrarily
- TAUAnTTOUTm is not used for the master channel of this function
- The channel output mode of the upper slave channel must be set ti Independent Channel Output Mode 1 with Real-Time Output, and TAUAnTRC.TRCm must be set to 1, refer to *15.9 “Channel Output Modes” on page 731*
- The channel output mode of the lower slave channels must be set to Synchronous Channel Output Mode 2 with Non-Complementary Modulation Output, refer to *15.9 “Channel Output Modes” on page 731*

Description The counters (master and slave) are started by setting the channel trigger bits (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation.

- Master channel:

  The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value. When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

- Upper slave:

  When the upper slave detects an interrupt from the master channel, the value of TAUAnCNTm reduces by 1. When the counter reaches 0000_H, it generates an INTTAUAnIm. TAUAnTTOUTm (upper slave) outputs the current value of the real-time output bit (TAUAnTRO.TROm). TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

- Lower slave channel:

  This has the same start trigger as the upper slave channel, but TAUAnTRC.TRCm = 0.

- Upper and lower slave channel:

  The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel for an interrupt. The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.

The value of TAUAnCDRm (master and upper slave) can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 “Simultaneous Rewrite” on page 719* .

(2) Equations

Number of interrupts generated on the master channel before the real-time output trigger is generated

= TAUAnCDRm (upper slave)

(3) Block diagram and general timing diagram

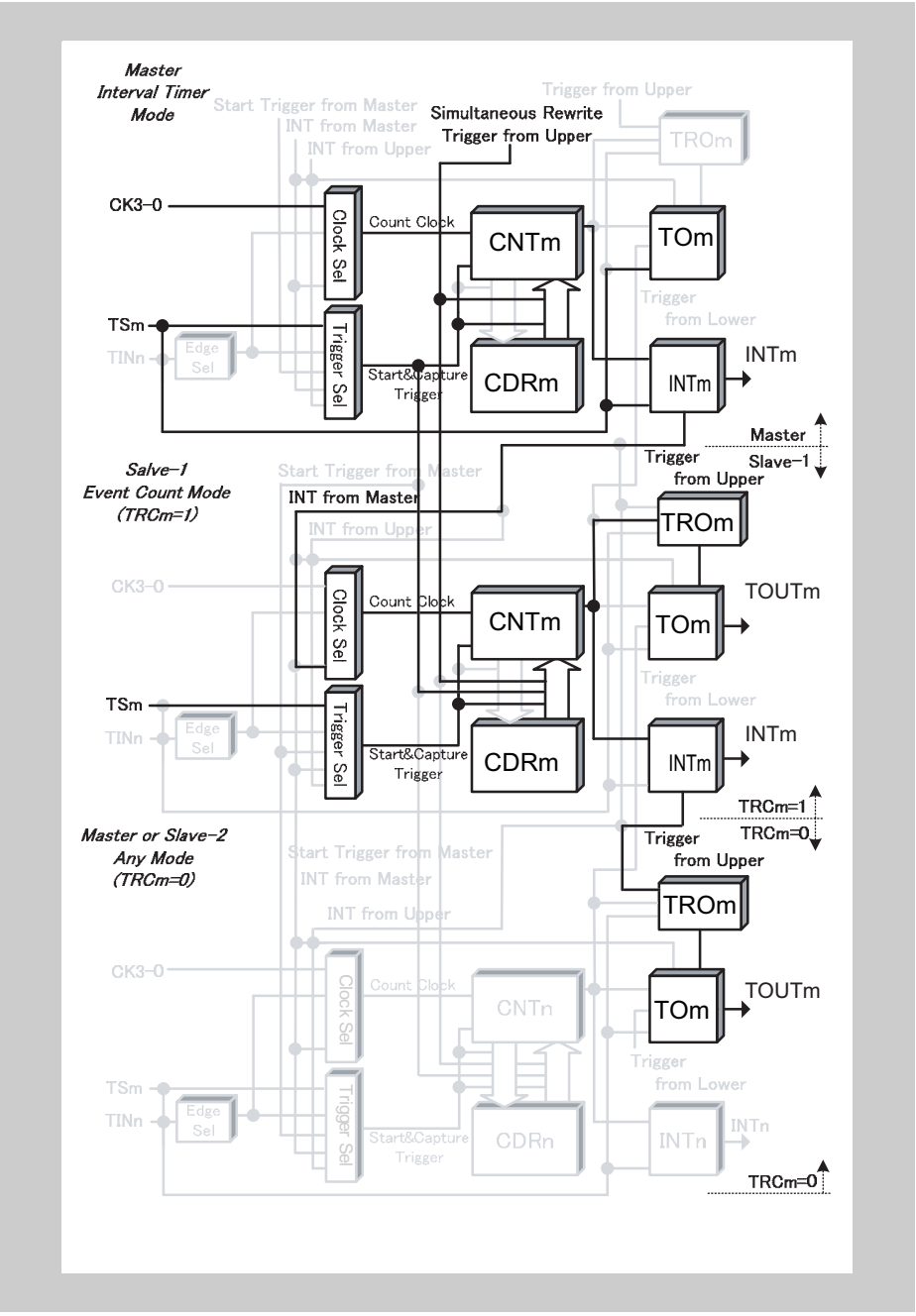


Figure 15-132 Block diagram for Real-Time Output Function Type 2

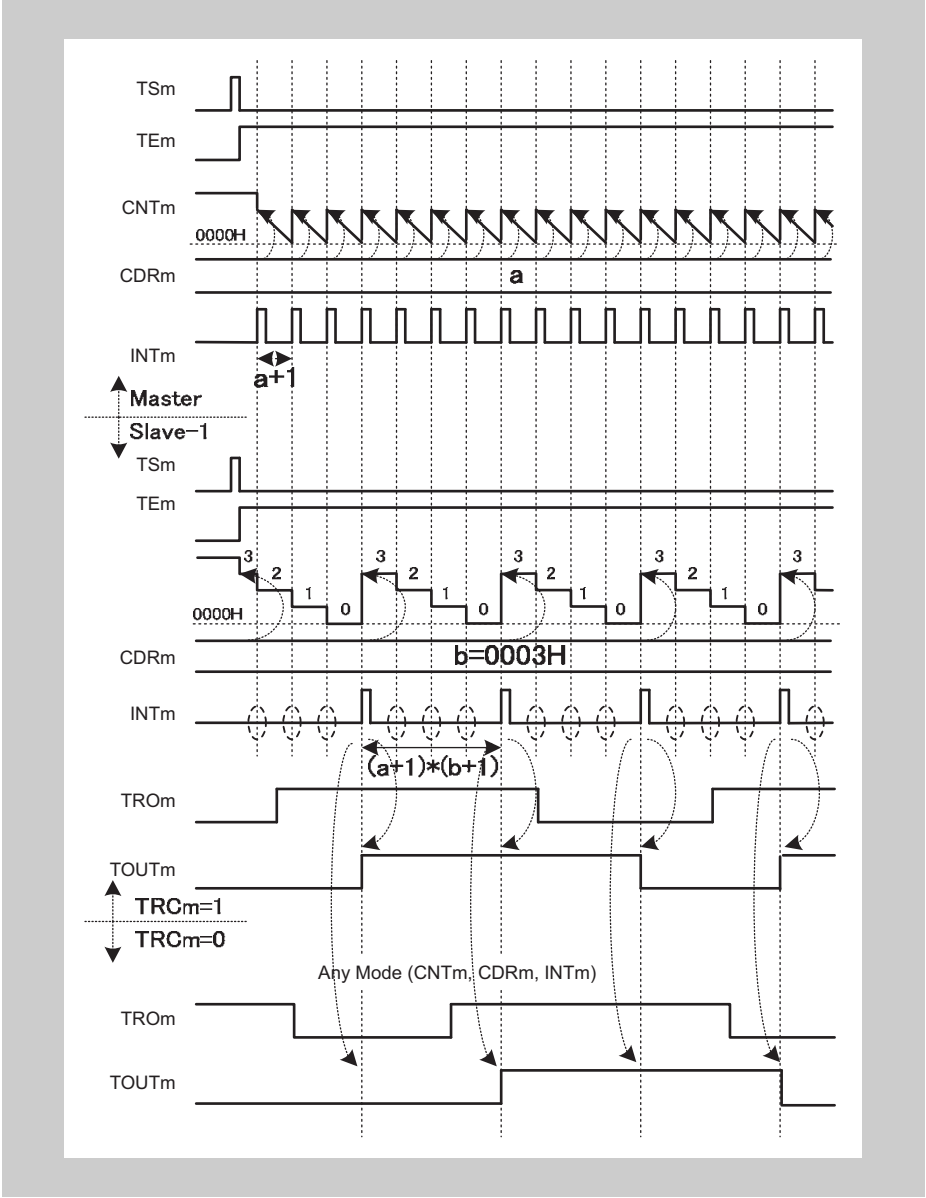


Figure 15-133 General timing diagram for Real-Time Output Function Type 2

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-190 TAUAnCMORm settings for the master channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm at operation start

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-191 TAUAnCMURm settings for the master channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite of the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-192 Simultaneous rewrite settings for the master channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the upper slave channel

(a) TAUAnCMORm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-193 TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	000: Software start
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUAnIm not output and TAUAnTTOUTm does not toggle at operation start or restart

(b) TAUAnCMURm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-194 TAUAnCMURm settings for the upper slave channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the upper slave channel

Table 15-195 Control bit settings for Independent Channel Output Mode 1 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite of the upper slave channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-196 Simultaneous rewrite settings for the upper slave channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for the lower slave channel

(a) TAUAnCMORm for the lower slave channel

The operation mode of the lower slave channel can be set arbitrarily.

(b) TAUAnCMURm for the lower slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-197 TAUAnCMURm settings for the lower slave channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the lower slave channels

Table 15-198 Control bit settings for Synchronous Channel Output Mode 2 with Non-Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 "Channel Output Modes" on page 731* .

(d) Simultaneous rewrite for the lower slave channels

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-199 Simultaneous rewrite settings for the lower slave channel of Synchronous Real-Time Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Synchronous Real-Time Output Function Type 2

Table 15-200 Operating procedure for Synchronous Real-Time Output Function Type 2

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 992* Upper slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the upper slave channel" on page 994* Lower slave channels: the operation mode can be set arbitrarily Set the values of the TAUAnCDRm registers of all channels Set TAUAnTRC.TRCm = 1 for the upper slave channel	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counter of the master channel starts. INTTAUAnIm is generated on the master channel, which causes the counter of the upper slave channel to reduce by 1.
During operation	TAUAnCDRm and TAUAnTRO.TROm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues counting down • TAUAnCNTm of the upper slave channel reduces by 1 • When the counter of upper slave channel reaches 0000_H: - INTTAUAnIm (upper slave) is generated - The counter reloads the TAUAnCDRm value and awaits the next interrupt from the master channel. - The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel for an interrupt.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

15.26.3 Synchronous Real-Time Output Function Type 3

(1) Overview

Summary This function outputs the value of the TAUAnTRO.TROm bit from TAUAnTTOUTm when a specified channel generates an interrupt (INTTAUAnIm). It is the same as Synchronous Real-Time Output Function Type 2 except that dead-time is applied by waiting for a TAUAnTS.TSm signal on the upper slave before counting a specified number of intervals of the master channel.

Prerequisites

- Three (or more) channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-201 “TAUAnCMORm settings for the master channel of Synchronous Real-Time Output Function Type 3” on page 1003*
- The operation mode of the upper slave channel must be set to One Count Mode, refer to *Table 15-204 “TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 3” on page 1005*
- The operation mode of the lower slave channels can be set arbitrarily
- TAUAnTTOUTm is not used for the master channel of this function
- The channel output mode of the upper slave channel must be set to Independent Channel Output Mode 1 with Real-Time Output and TAUAnTRC.TRCm must be set to 1, refer to *15.9 “Channel Output Modes” on page 731*
- The channel output mode of the lower slave channels must be set to Simultaneous Channel Output Mode 2 with Complementary Modulation Output, refer to *15.9 “Channel Output Modes” on page 731*

Description

- Master channel:

  The counter of the master channel is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm = 1, enabling count operation. The current value of TAUAnCDRm is written to TAUAnCNTm and the counter starts to count down from this value.
  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

- Upper slave:

  When the upper slave detects another TAUAnTS.TSm pulse, it awaits the next interrupt from the master channel and then counts down from the current value of TAUAnCDRm each time it detects an interrupt on the master channel. If another TAUAnTS.TSm pulse occurs before the upper slave reaches 0000_H, this pulse is ignored. When the counter reaches 0000_H, it generates an INTTAUAnIm. TAUAnTTOUTm of the slave channels outputs the current value of the real-time output bit (TAUAnTRO.TROm). TAUAnCNTm then reloads the TAUAnCDRm value and subsequently continues operation.

- Lower slave channel:

  This has the same start trigger as the upper slave channel, but TAUAnTRC.TRCm = 0.

- Upper and lower slave channel:

  The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel for an interrupt. The TAUAnTTOUTm signal only changes when an interrupt is generated, and then only when its value is different to current value of TAUAnTRO.TROm at the moment that the interrupt is generated.

The value of TAUAnCDRm can be rewritten at any time, and the changed value of TAUAnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions

- The master channel cannot be used to trigger real-time output, even if TAUAnTRE.TREm of the master channel is set to 1.
- Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*

(2) Equations

Time between TAUAnTS.TSm pulse and INTTAUAnIm of the upper slave
= [TAUAnCDR (master) + 1] x [TAUAnCDR (upper slave) + 1]

(3) Block diagram and general timing diagram

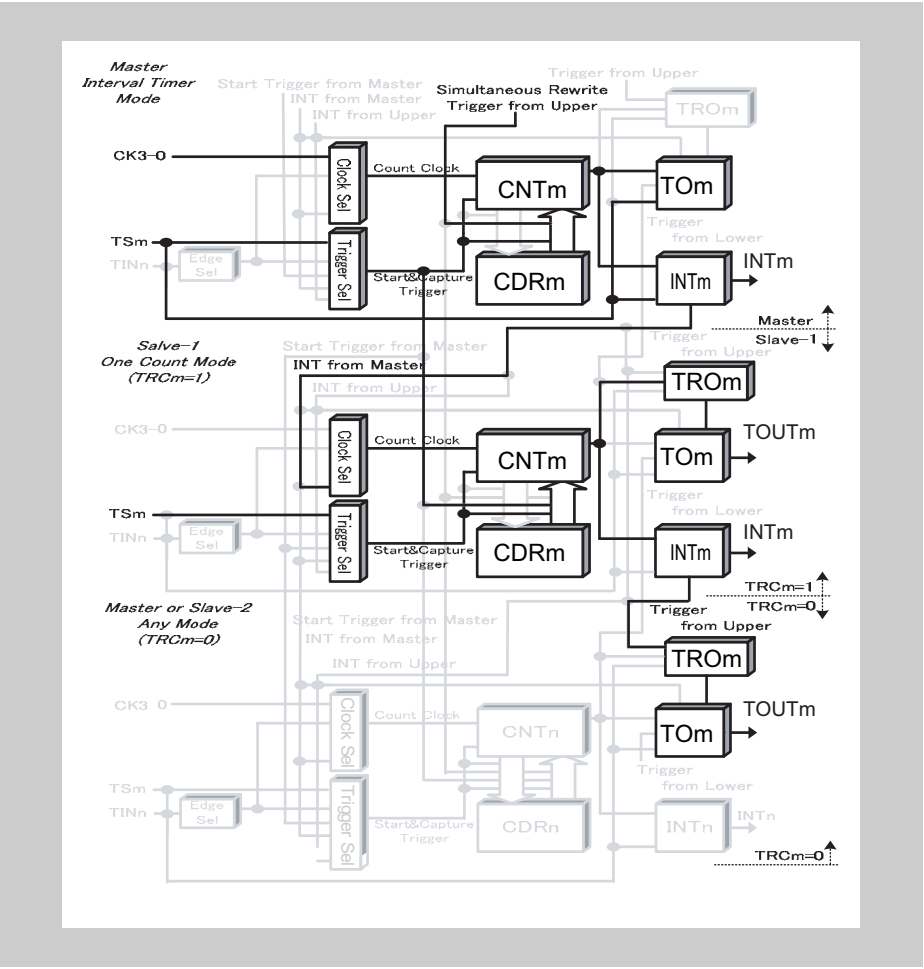


Figure 15-134 Block diagram for Real-Time Output Function Type 3

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm generated at operation start (TAUAnCMORm.MD0 = 1)
- Lower slave channel: dead time is added to the positive phase (TAUAnTDL.TDLm = 0)

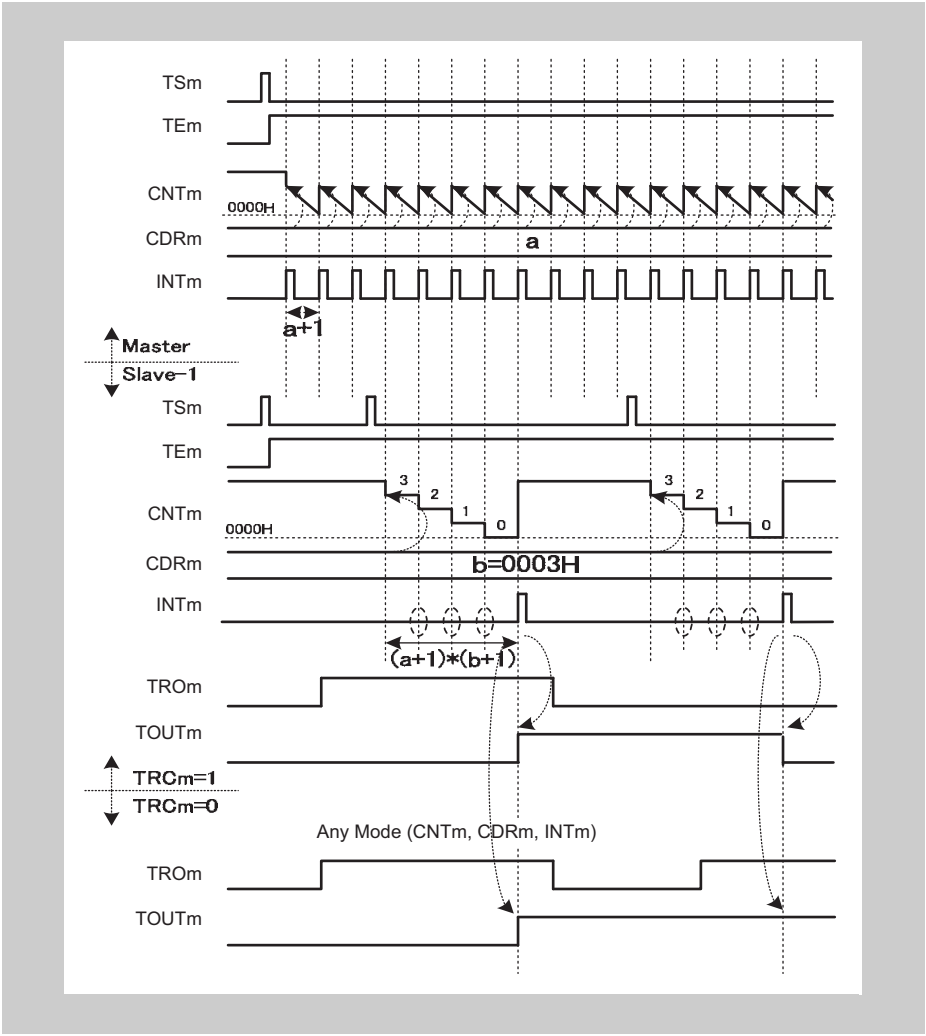


Figure 15-135 General timing diagram for Real-Time Output Function Type 3

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-201 TAUAnCMORm settings for the master channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not output at operation start or restart 1: Outputs INTTAUAnIm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-202 TAUAnCMURm settings for the master channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite of the master channel

The simultaneous rewrite settings of the master and lower slave channel must be identical.

Table 15-203 Simultaneous rewrite settings for the master channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the upper slave channel

(a) TAUAnCMORm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-204 TAUAnCMORm settings for the upper slave channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	000: Software start
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1100: One Count Mode
MD0	0: INTTAUAnIm not output and TAUAnTTOUTm does not toggle at operation start or restart

(b) TAUAnCMURm for the upper slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-205 TAUAnCMURm settings for the upper slave channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the upper slave channel

Table 15-206 Control bit settings for Independent Channel Output Mode 1 with Real-Time Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	1: Channel m generates its own real-time trigger
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite of the upper slave channel

The simultaneous rewrite registers (TAUAnRDE, TAUAnRDS, TAUAnRDM, and TAUAnRDC) cannot be used with Synchronous Real-Time Output Function Type 3. Therefore, these registers must be set to 0.

Table 15-207 Simultaneous rewrite settings for Synchronous Real-Time Output Function Type 3

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(6) Register settings for the lower slave channel

(a) TAUAnCMORm for the lower slave channel

The operation mode of the lower slave channel(s) can be set arbitrarily.

(b) TAUAnCMURm for the lower slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-208 TAUAnCMURm settings for the lower slave channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the lower slave channel

Table 15-209 Control bit settings for Synchronous Channel Output Mode 2 with Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic
TDE.TDEm	1: Enables dead time operation
TDM.TDMm	0: Dead time is added upon detection of a duty cycle at the upper even channel
TDL.TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite for the lower slave channel

The simultaneous rewrite settings of the master and lower slave channel must be identical.

Table 15-210 Simultaneous rewrite settings for the lower slave channel of Synchronous Real-Time Output Function Type 3

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Synchronous Real-Time Output Function Type 3

Table 15-211 Operating procedure for Synchronous Real-Time Output Function Type 3

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1003* Upper slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the upper slave channel" on page 1005* Lower slave channels: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for the lower slave channel" on page 1007* Set the values of the TAUAnCDRm registers of all channels Set TAUAnTRC.TRCm = 1 for the upper slave channel	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counter of the master channel starts. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm and TAUAnTRO.TROm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues counting down When TAUAnTS.TSm is set to 1 again on the upper slave channel: • TAUAnCNTm of the upper slave channel reduces by 1 every time it detects an interrupt on the master channel • When the counter of the upper slave channel reaches 0000_H: - INTTAUAnIm is generated - The counter reloads the TAUAnCDRm value and awaits the next TAUAnTS.TSm = 1 pulse on the upper slave channel. - The value of TAUAnTRO.TROm of the real-time output target channel is output on TAUAnTTOUTm on all channels that are monitoring the upper slave channel.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

15.27 Synchronous Non-Complementary and Complementary Functions

This chapter describes functions that generate 6-phase triangle PWM using a master channel and 7 slaves.

The trigger signal can be provided externally, e.g. from a Hall Sensor.

- *15.27.1 “Non-Complementary Modulation Output Function Type 1”*
- *15.27.2 “Non-Complementary Modulation Output Function Type 2”*
- *15.27.3 “Complementary Modulation Output Function”*

15.27.1 Non-Complementary Modulation Output Function Type 1

(1) Overview

Summary This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TROm) and the modulation output enable bits (TAUAnTME.TMEm) of a pair of slave channels. Three pairs of channels are typically used.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-213 "TAUAnCMORm settings for the master channel of Non-Complementary Modulation Output Function Type 1" on page 1015*
- The operation mode of slave channels 1 to 7 must be set to One Count Mode, refer to *Table 15-216 "TAUAnCMORm settings for slave channel 1 of Non-Complementary Modulation Output Function Type 1" on page 1017*
- TAUAnTTOUTm is not used for the master channel of this function.
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TRCm must be set to 1, refer to *15.9 "Channel Output Modes" on page 731*
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 1 with Non-Complementary Modulation Output, refer to *15.9 "Channel Output Modes" on page 731*
- TAUAnCDRm of slave channel 1 must be set to 0000_H

Description The counters of the master and slave channels are started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The current values of the data registers (TAUAnCDRm) are written to the counters (TAUAnCNTm) and the counters start to count down. When the counters reach 0000_H, INTTAUAnIm is generated.

- Slave channel 1:

  Slave channel 1 is set as the channel that triggers real-time output (TAUAnTRC.TRCm = 1), therefore the interrupt on slave channel 1 causes the value of real-time output bit (TAUAnTRO.TROm) of channels monitoring slave channel 1 to change. After generating an interrupt, the counter returns to $FFFF_H$ and awaits the next interrupt from the master channel.

- Slave channel 2:

  Slave channel 2 generates a PWM output. The master channel specifies the period of the PWM output, the slave channel 2 the duty cycle. After generating an interrupt, the counter returns to $FFFF_H$ and awaits the next interrupt from the master channel.

Slave channels 3 to 7 behave analogously to slave channel 2.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bit (TAUAnTRO.TROm) and the modulation output bit (TAUAnTME.TMEm) of the slave channel, as shown in *Table 15-212 "TAUAnTTOUTm output of a pair of slave channels in Non-Complementary Modulation Output Function Type 1" on page 1012* .

Forced restart is not possible with this function. The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and

slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions

- If TAUAnTME.TMEm = 0 for slave channels 2 to 7:
  - If TAUAnTRO.TROm of the channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTME.TMEm = 1 for slave channels 2 to 7:
  - If TAUAnTRO.TROm of the channel is 1, TAUAnTTOUTm outputs the corresponding PWM of the channel
  - If TAUAnTRO.TROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TOLm = 1 the high and low signals output from TAUAnTTOUTm are inverted. The PWM signals remain unaffected.

Table 15-212 TAUAnTTOUTm output of a pair of slave channels in Non-Complementary Modulation Output Function Type 1

TAUAnTME.TMEm	TAUAnTRO.TROm	TAUAnTTOUTm outputs
0	0	Low
	1	High
1	0	Low
	1	PWMm

- Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*
- The value of TAUAnCDRm of slave channel 1 must be set to 0000_H so that the real-time output trigger occurs at the same time as the PWM is generated by slave channels 2 to 7.
- If TAUAnTOL.TOLm = 0 for a slave channel 2 to 7, TAUAnTO.TOm must be set low before the function is enabled.
- If TAUAnTOL.TOLm = 1 for a slave channel 2 to 7, TAUAnTO.TOm must be set high before the function is enabled.

(2) Equations

For slave channels 2 to 7:

PWM output cycle time = [TAUAnCDRm (master) + 1] x count clock

PWM output duty time = [TAUAnCDRm (slave)] x count clock

(3) Block diagram and general timing diagram

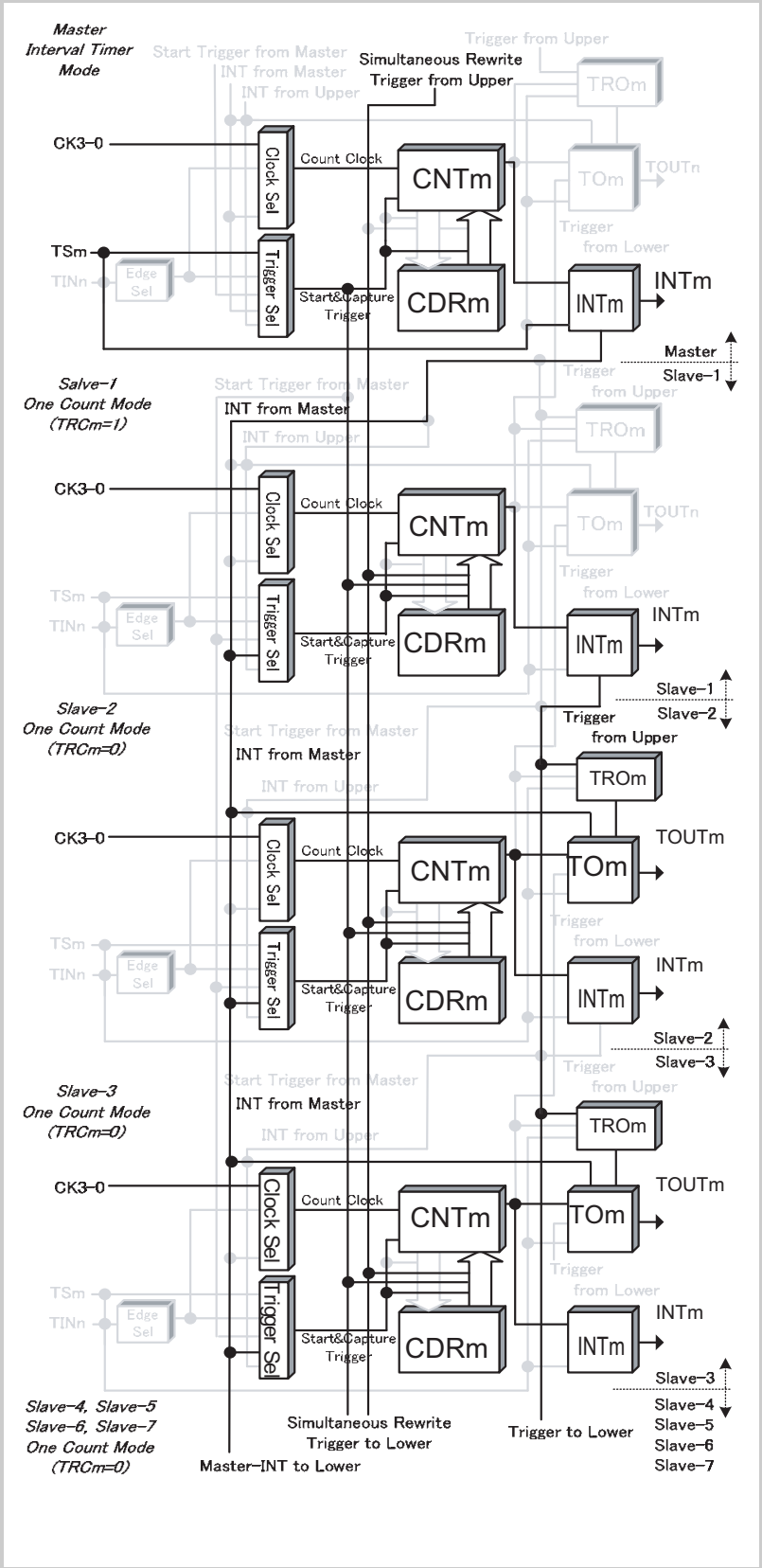


Figure 15-136 Block diagram for Non-Complementary Modulation Output Function Type 1

The following settings apply to the general timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)

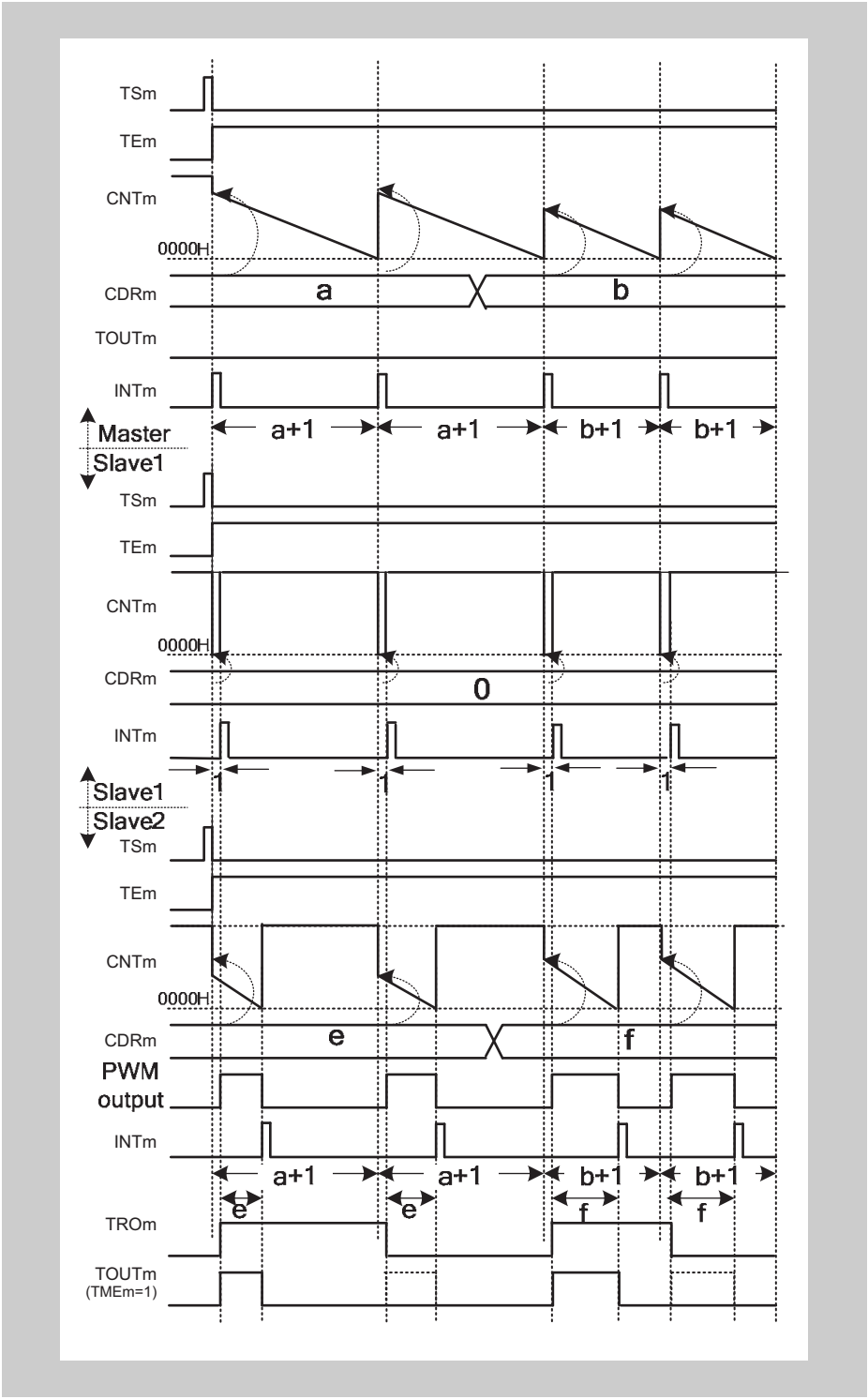


Figure 15-137 General timing diagram for Non-Complementary Modulation Output Function Type 1

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-213 TAUAnCMORm settings for the master channel of Non-Complementary Modulation Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-214 TAUAnCMURm settings for the master channel of Non-Complementary Modulation Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-215 Simultaneous rewrite settings for the master channel of Non-Complementary Modulation Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-216 TAUAnCMORm settings for slave channel 1 of Non-Complementary Modulation Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-217 TAUAnCMURm settings for slave channel 1 Non-Complementary Modulation Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

The channel output mode is not used by slave channel 1 of this function. However, it can be used in Direct Channel Output Mode. Furthermore, TAUAnTRC.TRCm must be set to 1, to enable slave 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-218 Simultaneous rewrite settings for slave channel 1 Non-Complementary Modulation Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2 to 7

(a) TAUAnCMORm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-219 TAUAnCMORm settings for slave channels 2 to 7 of the Non-Complementary Modulation Output Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUAnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-220 TAUAnCMURm settings for slave channels 2 to 7 of the Non-Complementary Modulation Output Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2 to 7

Table 15-221 Control bit settings for slave channels 2 to 7 of the Synchronous Channel Output Mode 1 with Non-Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	0: Operation mode 1
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 "Channel Output Modes" on page 731* .

(d) Simultaneous rewrite for slave channels 2 to 7

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-222 Simultaneous rewrite settings for slave channels 2 to 7 of the Non-Complementary Modulation Output Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating Procedure for Non-Complementary Modulation Output Function Type 1

Table 15-223 Operating procedure for Non-Complementary Modulation Output Function Type 1 (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1015* Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 1" on page 1017* Slave channel 2 to 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for slave channels 2 to 7" on page 1019* Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, set TAUAnCDRm of slave channel 1 to 0000_H, and set the duty width in TAUAnCDRm of slave channels 2 to 7 Set TAUAnTRC.TRCm = 1 for slave channel 1	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 15-223 Operating procedure for Non-Complementary Modulation Output Function Type 1 (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counters start to count down.
During operation	TAUAnCDRm, TAUAnTRO.TROm, and TAUAnTME.TMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel, slave channel 1, and slave channels 2 to 7 load TAUAnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm reloads the TAUAnCDRm value and continues counting down • The PWM output of slave channels 2 to 7 toggles • TAUAnCNTm of slave channel 1 reloads the TAUAnCDRm value and counts down with 1 clock cycle delay • TAUAnCNTm of slave channels 2 to 7 reload the TAUAnCDRm value and counts down • When the counter of slave channel 1 or slave channels 2 to 7 reaches 0000_H: - INTTAUAnIm is generated - The counter reloads the TAUAnCDRm value and awaits the next interrupt on the master channel. • When the counter of slave channels 2 to 7 reaches 0000_H: - The PWM output of slave channels 2 to 7 toggles TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TROm) and the modulation output bits (TAUAnTME.TMEm) of a pair of slave channels.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm of slave channels 2 to 7 is initialized to the value set by TAUAnTO.TOm.

Restart

(8) Specific timing diagrams

The following settings apply to the specific timing diagram:

- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)

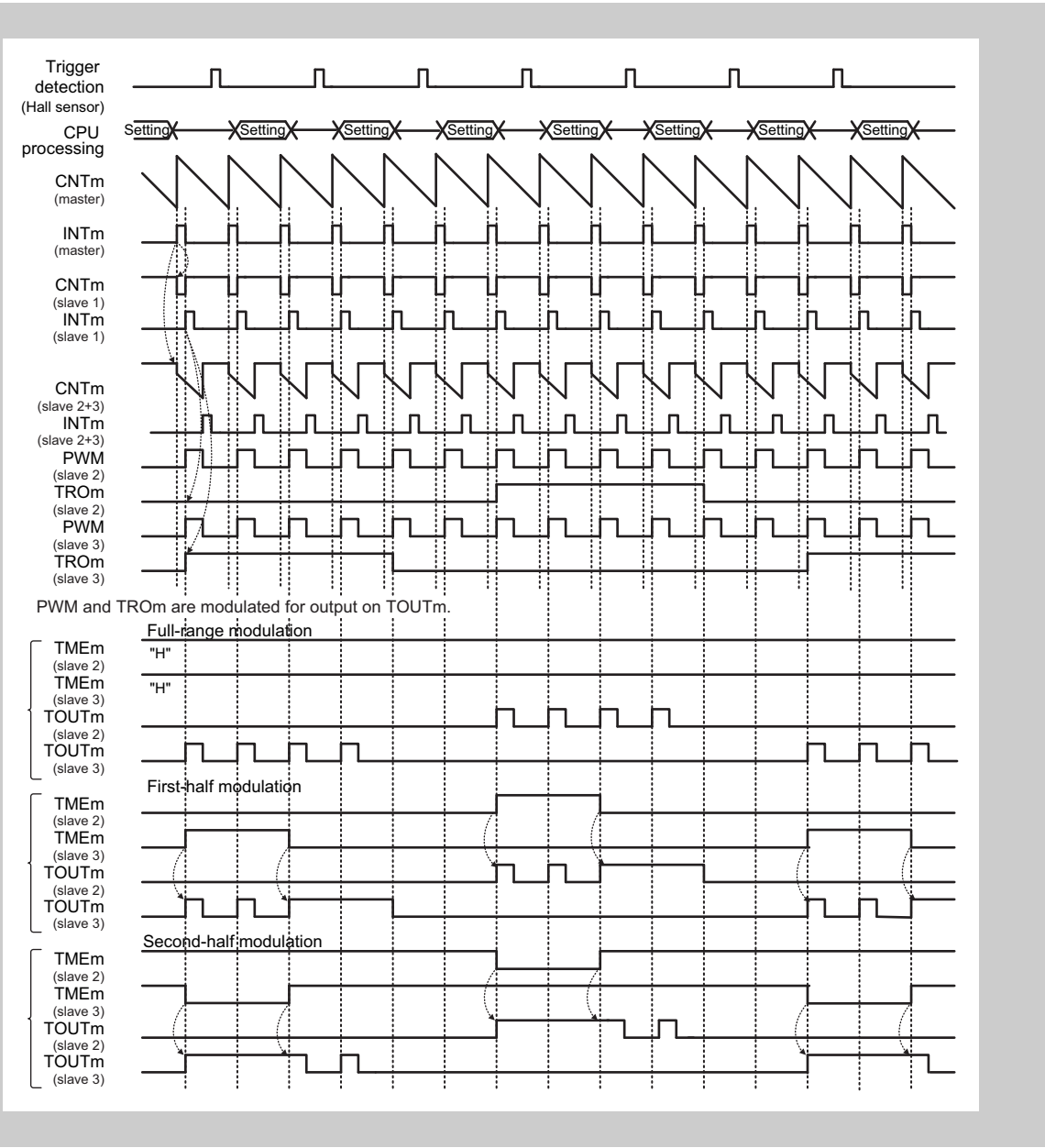


Figure 15-138 Specific timing diagram for Non-Complementary Modulation Output Function Type 1

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TMEm bits of the lower slave channels during operation.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TMEm and TAUAnTRO.TROm can be changed.

The values of the TAUAnTRO.TROm bits are specified by software, but the new values are only applied when an interrupt occurs on slave channel 1.

15.27.2 Non-Complementary Modulation Output Function Type 2

(1) Overview

Summary

This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TROm) and the modulation output enable bits (TAUAnTME.TMEm) of a pair of slave channels. Slave channel 1 specifies how many interrupts of the master channel are ignored before the value of TAUAnTRO.TROm changes. Three pairs of channels are typically used.

The main difference to Non-Complementary Modulation Output Function Type 1 is that in this function PWM output is symmetrical about the point at which the master counter reaches zero.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-225 “TAUAnCMORm settings for the master channel of Non-Complementary Modulation Output Function Type 2” on page 1029*
- The operation mode of slave channel 1 must be set to Event Count Mode, refer to *Table 15-228 “TAUAnCMORm settings for slave channel 1 of Non-Complementary Modulation Output Function Type 2” on page 1031*
- The operation mode of slave channels 2 to 7 must be set to Up Down Count Mode, refer to *Table 15-231 “TAUAnCMORm settings for slave channels 2 to 7 of Non-Complementary Modulation Output Function Type 2” on page 1033*
- TAUAnTTOUTm is not used for the master channel of this function.
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TRCm must be set to 1, refer to *15.9 “Channel Output Modes” on page 731*
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 2 with Non-Complementary Modulation Output, refer to *15.9 “Channel Output Modes” on page 731*

Description

The counters of the master and slave channels are started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The current values of the data registers (TAUAnCDRm) are written to the counters (TAUAnCNTm).

- Master channel:

  The counter of the master channel starts to count down. When the counter reaches 0000_H, INTTAUAnIm is generated.

- Slave channel 1:

  When slave channel 1 detects an interrupt on the master channel, counter reduces by 1. When the counter reaches 0000_H, it awaits the next interrupt from the master channel. This causes TAUAnCNTm (slave 1) to reload the value of TAUAnCDRm, and an INTTAUAnIm is generated.

  Slave channel 1 is set as the channel that triggers real-time output (TAUAnTRC.TRCm = 1), therefore the interrupt on slave channel 1 causes the value of real-time output bit (TAUAnTRO.TROm) of channels monitoring slave channel 1 to change.

- Slave channel 2:

  When an interrupt is detected from the master channel, the TAUAnCNTm counts in the reverse direction. If the interrupt is detected whilst it is counting up, it first reloads the value of TAUAnCDRm and then starts to count down. When TAUAnCNTm = 0001_H an interrupt is generated and the PWM output toggles.

The combination of the master channel and slave channel 2 generates a PWM output. The master channel generates the period of the PWM output, the slave channel the duty cycle.

Slave channels 3 to 7 behave analogously to slave channel 2.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bit (TAUAnTRO.TROm) and the modulation output bit (TAUAnTME.TMEm) of the slave channel, as shown in *Table 15-224 "TAUAnTTOUTm output of a pair of slave channels in Non-Complementary Modulation Output Function Type 2" on page 1025* .

Forced restart is not possible with this function. The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions

- If TAUAnTME.TMEm of the slave channel is 1:
  - If TAUAnTRO.TROm of the channel is 1, TAUAnTTOUTm outputs the corresponding PWM of the channel
  - If TAUAnTRO.TROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTME.TMEm of the slave channel is 0:
  - If TAUAnTRO.TROm of the channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TROm of the channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TOLm = 1 the high and low signals output from TAUAnTTOUTm are inverted. The PWM signals remain unaffected.

Table 15-224 TAUAnTTOUTm output of a pair of slave channels in Non-Complementary Modulation Output Function Type 2

TAUAnTME.TMEm	TAUAnTRO.TROm	TAUAnTTOUTm outputs
0	0	Low
	1	High
1	0	Low
	1	PWMm

- Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719*
- If TAUAnTOL.TOLm = 0 for a slave channel 2 to 7, TAUAnTO.TOm must be set low before the function is enabled.
- If TAUAnTOL.TOLm = 1f or a slave channel 2 to 7, TAUAnTO.TOm must be set high before the function is enabled.

(2) Equations

For slave channels 2 to 7:

PWM output cycle time
= [TAUAnCDRm (master) + 1] x count clock

PWM output duty time
= [TAUAnCDRm (master) + 1 - TAUAnCDRm (slave)] x 2 x count clock

(3) Block diagram and general timing diagram

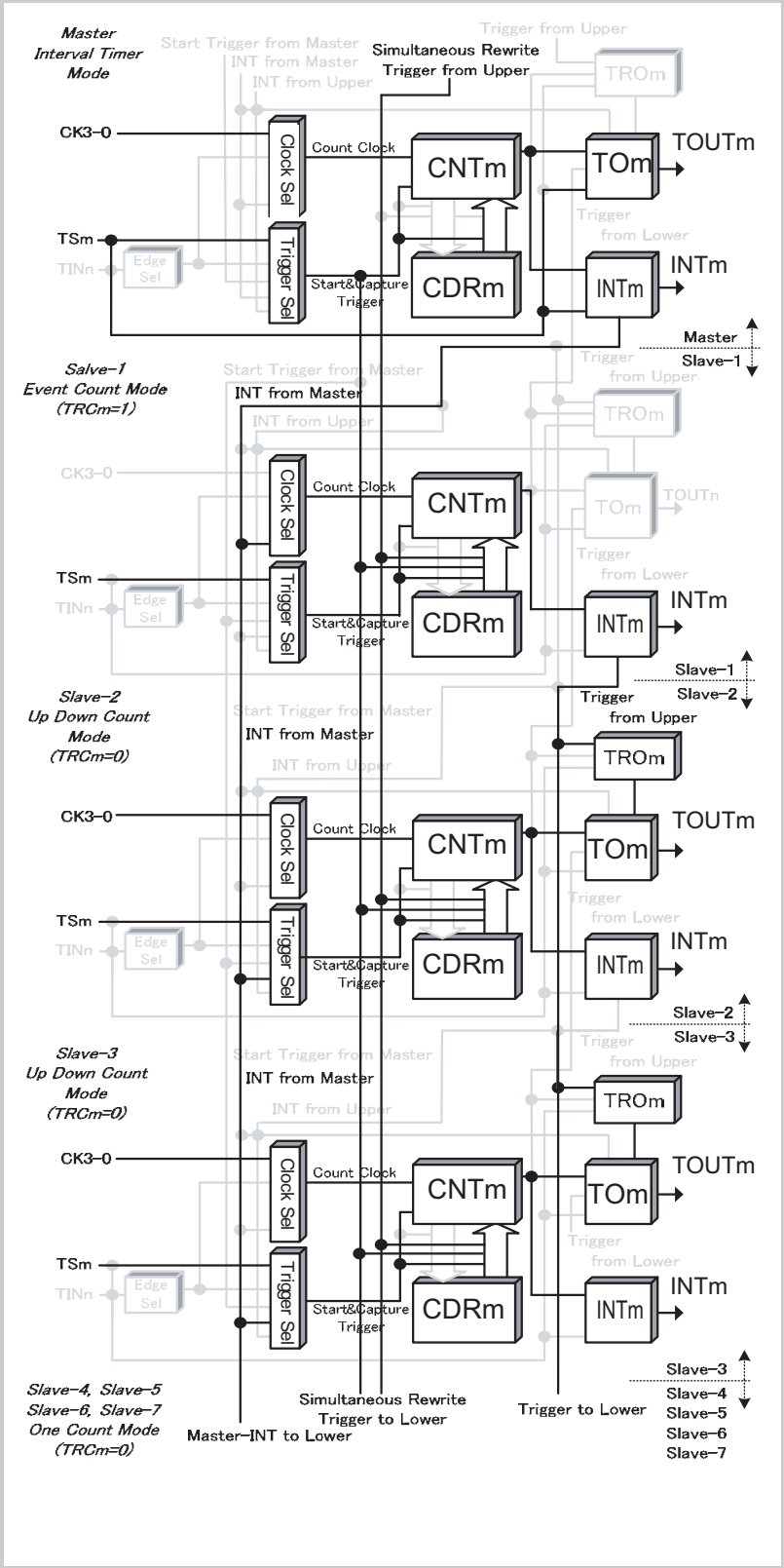


Figure 15-139 Block diagram for Non-Complementary Modulation Output Function Type 2

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)

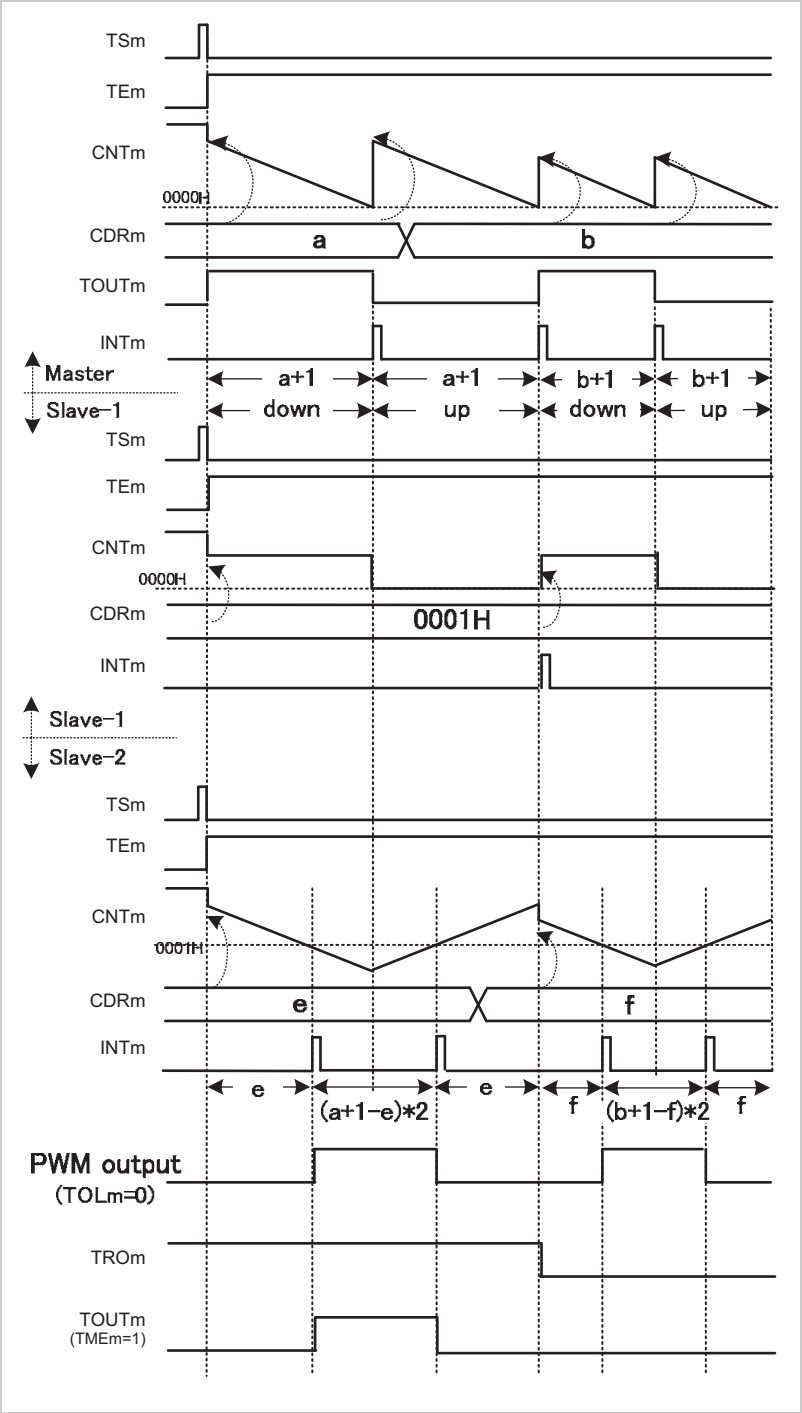


Figure 15-140 General timing diagram for Non-Complementary Modulation Output Function Type 2

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-225 TAUAnCMORm settings for the master channel of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated at operation start or restart 1: Generates INTTAUAnIm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-226 TAUAnCMURm settings for the master channel of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-227 Simultaneous rewrite settings for the master channel of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: When simultaneous rewrite is disabled (TAUAnRDE.RDEm = 0), set these bits to 0. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-228 TAUAnCMORm settings for slave channel 1 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-229 TAUAnCMURm settings for slave channel 1 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

The channel output mode is not used by slave channel 1 of this function. However, it can be used in Direct Channel Output Mode. Furthermore, TAUAnTRC.TRCm must be set to 1, to enable slave 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-230 Simultaneous rewrite settings for slave channel 1 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2 to 7

(a) TAUAnCMORm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-231 TAUAnCMORm settings for slave channels 2 to 7 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channels 2 to 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-232 TAUAnCMURm settings slave channels 2 to 7 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2 to 7

Table 15-233 Control bit settings slave channels 2 to 7 of Synchronous Channel Output Mode 2 with Non-Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite for slave channels 2 to 7

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-234 Simultaneous rewrite settings slave channels 2 to 7 of Non-Complementary Modulation Output Function Type 2

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating Procedure for Non-Complementary Modulation Output Function Type 2

Table 15-235 Operating procedure for Non-Complementary Modulation Output Function Type 2 (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1029* Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 1" on page 1031* Slave channel 2 to 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for slave channels 2 to 7" on page 1033* Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, using TAUAnCDRm of slave channel 1 set the number of interrupts on the master channel to be ignored before slave 1 generates an input, and set the duty width in TAUAnCDRm of slave channels 2 to 7 Set TAUAnTRC.TRCm = 1 for slave channel 1	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 15-235 Operating procedure for Non-Complementary Modulation Output Function Type 2 (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counters start to count down.
During operation	TAUAnCDRm, TAUAnTRO.TROm, and TAUAnTME.TMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel and slave channels 2 to 7 load TAUAnCDRm and count down. Slave channel 1 loads TAUAnCDRm and waits for an interrupt on the master channel. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm reloads the TAUAnCDRm value and continues counting down • TAUAnCNTm of slave channel 1 reduces by 1 and awaits the next interrupt on the master channel • TAUAnCNTm of slave channels 2 to 7 counts in the other direction • When the counter of slave channel 1 reaches 0000_H it awaits the next interrupt from the master channel. When it is detected: - TAUAnCNTm reloads the TAUAnCDRm value and awaits the next interrupt on the master channel - INTTAUAnIm is generated • When the counter of slave channels 2 to 7 reaches 0001_H: - INTTAUAnIm is generated - The PWM output of slave channels 2 to 7 toggles TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TROm) and the modulation output bits (TAUAnTME.TMEm) of a pair of slave channels.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm of slave channels 2 to 7 is initialized to the value set by TAUAnTO.TOm.

Restart

(8) Specific timing diagrams

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)

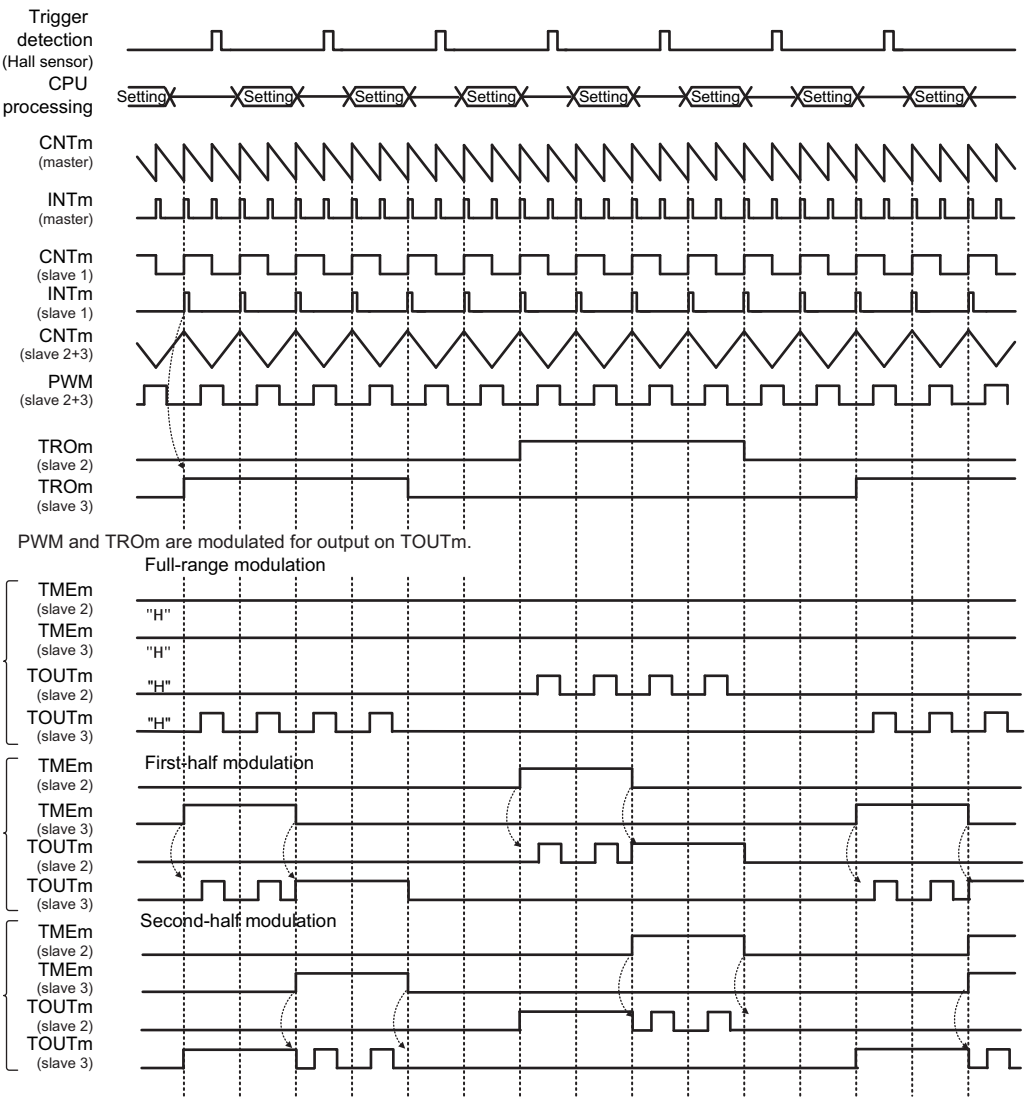


Figure 15-141 Specific timing diagram for Non-Complementary Modulation Output Function Type 2

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TMEm bits of the lower slave channels during operation.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TMEm and TAUAnTRO.TROm can be changed.

The values of the TAUAnTRO.TROm bits are specified by software, but the new values are only applied when an interrupt occurs on slave channel 1.

15.27.3 Complementary Modulation Output Function

(1) Overview

Summary This function outputs a PWM signal, a high signal, or a low signal from TAUAnTTOUTm depending on the value of the real-time output bits (TAUAnTRO.TROm), the modulation output bits (TAUAnTME.TMEm), and the output level bits (TAUAnTOL.TOLm) of a pair of slave channels. Three pairs of channels are typically used.

Prerequisites

- One master channel and seven slave channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-238 "TAUAnCMORm settings for the master channel of the Complementary Modulation Output Function" on page 1044*
- The operation mode of slave channel 1 must be set to Event Count Mode, refer to *Table 15-242 "TAUAnCMORm settings for slave channel 1 of the Complementary Modulation Output Function" on page 1046*
- The operation mode of slave channels 2, 4, and 6 must be set to Up Down Count Mode, refer to *Table 15-245 "TAUAnCMORm settings for slave channels 2, 4, and 6 of the Complementary Modulation Output Function" on page 1048*
- The operation mode of slave channels 3, 5, and 7 must be set to One Count Mode, refer to *Table 15-249 "TAUAnCMORm settings for slave channels 3, 5, and 7 of the Complementary Modulation Output Function" on page 1050*
- The channel output mode of the master channel must be set to Independent Channel Output Mode 1, refer to *15.9 "Channel Output Modes" on page 731*
- TAUAnTTOUTm of slave channel 1 is not used for this function, but TAUAnTRC.TRCm must be set to 1, refer to *15.9 "Channel Output Modes" on page 731*
- The channel output mode of slave channels 2 to 7 must be set to Simultaneous Channel Output Mode 2 with Complementary Modulation Output, refer to *15.9 "Channel Output Modes" on page 731*

Description

- Master channel:

  The counter of the master channel is started by setting the channel trigger bit (TAUAnTS.TSm) to 1. This in turn sets TAUAnTE.TEm, enabling count operation. The current value of the data register of the master channel (TAUAnCDRm) is written to the counter (TAUAnCNTm) and the counter starts to count down from this value.

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated. This triggers the counter of slave channel 1 to reduce by 1 and the counter of slave channel 2 to count in the reverse direction.

- Slave channel 1:

  When the counter reaches 0000_H, it awaits the next interrupt from the master channel. This causes TAUAnCNTm (slave 1) to reload the value of TAUAnCDRm, and an INTTAUAnIm is generated.

  Slave channel 1 is set as the channel that triggers real-time output (TAUAnTRC.TRCm = 1). The interrupt causes channels monitoring slave channel 1 to apply the current value of their real-time output bit (TAUAnTRO.TROm). The value of the real-time output bits can be changed at any time by the application software, but the new values are only applied when the interrupt occurs on slave channel 1.

- Slave channel 2:

  When the counter of slave channel 2 reaches 0001_H, the counter of slave channel 3 starts to count down. When the counter of slave channel 3 reaches 0000_H it generates an interrupt.

- Slave channels 2 and 3:

  The combination of the master channel and slave channels 2 and 3 generates a PWM output. The master channel generates the period of the PWM output, slave channel 2 the duty cycle, and slave channel 3 generates the dead time.

- Slave channels 4 to 7:

  Slave channels 4 and 6 behave analogously to slave channel 2. Slave channels 5 and 7 behave analogously to slave channel 3.

The signal that is output from TAUAnTTOUTm depends on the value of the real-time output bits (TAUAnTRO.TROm), the modulation output bits (TAUAnTME.TMEm), and the output level bits (TAUAnTOL.TOLm) of a pair of slave channels, as shown in *Table 15-212 "TAUAnTTOUTm output of a pair of slave channels in Non-Complementary Modulation Output Function Type 1" on page 1012* .

However, the output of slave channels 2 and 3 must not be high simultaneously (for example to prevent a motor driver from short circuiting). To prevent both channels being high at the instant that one channel switches to low and the other switches to high, a delay must be added to the channel that switches to high so that this transition occurs later. This is achieved by adding dead time to either the positive or negative-phased PWM signal as specified by the value of the channel dead time output level register (TAUAnTDL.TDLm).

The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channels, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUAnTS.TSm to 1.

Conditions

- If TAUAnTME.TMEm of both channels in a pair is 1:
  - If TAUAnTRO.TROm of either or both channel(s) is 1, TAUAnTTOUTm outputs the corresponding PWM of the channel
  - If TAUAnTRO.TROm of both channels is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTME.TMEm of both channels in a pair is 0:
  - If TAUAnTRO.TROm of a channel is 1, TAUAnTTOUTm outputs a high signal
  - If TAUAnTRO.TROm of a channel is 0, TAUAnTTOUTm outputs a low signal
- If TAUAnTOL.TOLm = 1, the high and low signals output from TAUAnTTOUTm are inverted. The PWM signals remain unaffected.

Table 15-236 TAUAnTTOUTm output of a pair of slave channels in Complementary Modulation Output Function for TAUAnTOL.TOLm = 0

TAUAnTME.TME2	TAUAnTME.TME3	TAUAnTRO.TRO2	TAUAnTRO.TRO3	TAUAnTTOUT2 outputs	TAUAnTTOUT3 outputs
0	0	0	0	LOW	LOW
		0	1	LOW	HIGH
		1	0	HIGH	LOW
		1	1	Forbidden	Forbidden
1	1	0	0	LOW	LOW
		0	1	~PWMm	PWMm
		1	0	PWMm	~PWMm
		1	1	Forbidden	Forbidden

Notes 1. ~PWM indicates an inverted PWM signal.

2. All settings that are not listed in this table are not permitted.

- If TAUAnTME.TMEm of both channels in a pair is 1 for the entire time TAUAnTRO.TROm is 1, this is referred to as “full-range modulation”.
- If TAUAnTME.TMEm of both channels in a pair is 1 for the first half of the period for which TAUAnTRO.TROm is 1, this is referred to as “first-half modulation”.
- If TAUAnTME.TMEm of both channels in a pair is 1 for the second half of the period for which TAUAnTRO.TROm is 1, this is referred to as “second-half modulation”.
- Whether dead time is added to the positive or negative PWM signal at the instant that two channels try to output a high signal simultaneously is specified by the TAUAnTDL.TDLm bit:
  - If TAUAnTDL.TDLm = 0, dead time is added to the positive PWM signal
  - If TAUAnTDL.TDLm = 1 dead time is added to the negative PWM signal
  - The values of the TAUAnTDL.TDLm bits must be manipulated by the application software during operation.

Table 15-237 TAUAnTDL.TDLm settings for a pair of slave channels in Complementary Modulation Output Function for TAUAnTOL.TOLm = 0

TAUAnTME.TME2	TAUAnTME.TME3	TAUAnTRO.TRO2	TAUAnTRO.TRO3	TAUAnTDL.TDL2	TAUAnTDL.TDL3
0	0	0	0	1	1
		0	1	1	0
		1	0	0	1
1	1	0	0	1	1
		0	1	1	0
		1	0	0	1

- The value of TAUAnCDRm of slave channel 1 must be set so that INTTAUAnIm is generated from slave 1 at the summit of a carrier cycle.
- Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719* .

(2) Equations

For slaves 2, 4, and 6

When TAUAnTOL.TOLm = 0 and TAUAnCDR (slave 1):

PWM output cycle time = 2 x [TAUAnCDRm (master) + 1] x count clock

PWM output duty time = { [TAUAnCDRm (master) + 1 - TAUAnCDRm (slave 2)] x 2 - [TAUAnCDRm (slave 3) + 1] } x count clock

(3) Block diagram and general timing diagram

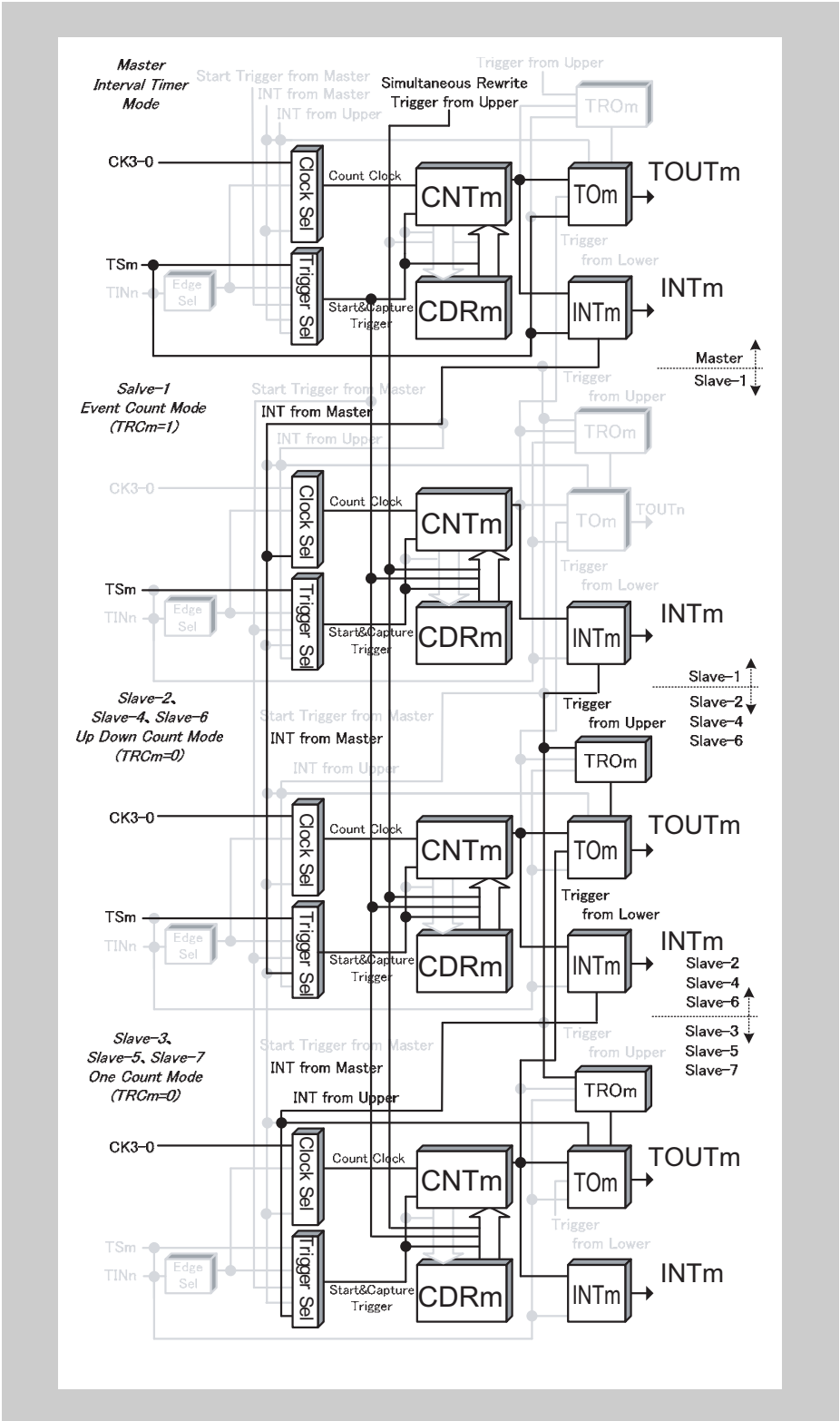


Figure 15-142 Block diagram for Complementary Modulation Output Function

The following settings apply to the general timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)

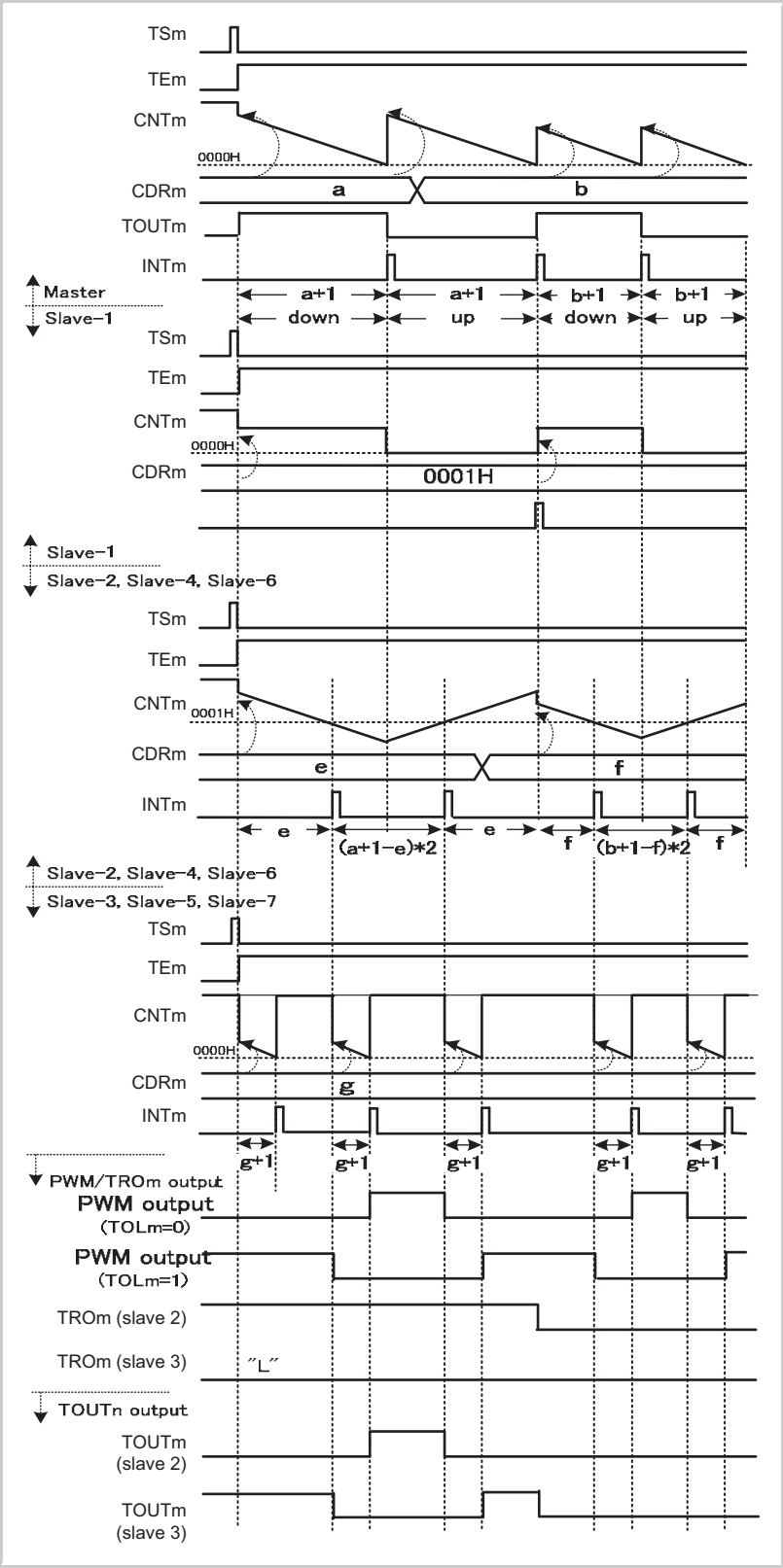


Figure 15-143 General timing diagram for Complementary Modulation Output Function

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-238 TAUAnCMORm settings for the master channel of the Complementary Modulation Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not generated and TAUAnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUAnIm and toggles TAUAnTTOUTm at operation start or restart

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-239 TAUAnCMURm settings for the master channel of the Complementary Modulation Output Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 15-240 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUAnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDM.TDMm	0: When dead time operation is disabled (TAUAnTDE.TDEm = 0), set these bits to 0
TDL.TDLm	
TRE.TREm	0: Disables real-time output
TRO.TROm	0: When real-time output is disabled (TAUAnTRE.TREm = 0), set these bits to 0
TRC.TRCm	
TME.TMEm	0: Disables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 "Channel Output Modes" on page 731* .

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-241 Simultaneous rewrite settings for the master channel of the Complementary Modulation Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 1

(a) TAUAnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-242 TAUAnCMORm settings for slave channel 1 of the Complementary Modulation Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-243 TAUAnCMURm settings for slave channel 1 of the Complementary Modulation Output Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

The channel output mode is not used by slave channel 1 of this function. However, it can be used in Direct Channel Output Mode. Furthermore, TAUAnTRC.TRCm must be set to 1, to enable slave 1 to be used as the real-time output trigger.

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 15-244 Simultaneous rewrite settings for slave channel 1 of the Complementary Modulation Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channels 2, 4, and 6

(a) TAUAnCMORm for slave channels 2, 4, and 6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-245 TAUAnCMORm settings for slave channels 2, 4, and 6 of the Complementary Modulation Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3 The value of the CKS[1:0] bit of the master and slave channels must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUAnIm not generated at operation start or restart

(b) TAUAnCMURm for slave channels 2, 4, and 6

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-246 TAUAnCMURm settings for slave channels 2, 4, and 6 of the Complementary Modulation Output Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 2, 4, and 6

Table 15-247 Control bit settings for Synchronous Channel Output Mode 2 with Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	1: Enables dead time operation
TDM.TDMm	0: Dead time is added upon detection of a duty cycle at the upper even channel
TDL.TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite for slave channels 2, 4, and 6

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-248 Simultaneous rewrite settings for slave channels 2, 4, and 6 of the Complementary Modulation Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Register settings for slave channels 3, 5, and 7

(a) TAUAnCMORm for slave channels 3, 5, and 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-249 TAUAnCMORm settings for slave channels 3, 5, and 7 of the Complementary Modulation Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUAnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	1: Enables start trigger detection during counting

(b) TAUAnCMURm for slave channels 3, 5, and 7

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-250 TAUAnCMURm settings for slave channels 3, 5, and 7 of the Complementary Modulation Output Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode of slave channels 3, 5, and 7

Table 15-251 Control bit settings for Synchronous Channel Output Mode 2 with Complementary Modulation Output

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	1: Synchronous channel output
TOC.TOCm	1: Operation mode 2
TOL.TOLm	0: Positive logic 1: Inverted logic
TDE.TDEm	1: Enables dead time operation
TDM.TDMm	0: Dead time is added upon detection of a duty cycle at the upper even channel
TDL.TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase
TRE.TREm	1: Enables real-time output
TRO.TROm	0: Real-time output is low 1: Real-time output is high
TRC.TRCm	0: The next upper channel generates the real-time trigger for channel m
TME.TMEm	0: Disables modulation 1: Enables modulation

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUAnTOE.TOEm = 0. TAUAnTTOUTm can then be controlled independently of the interrupts. For details refer to *15.9 “Channel Output Modes” on page 731* .

(d) Simultaneous rewrite for slave channels 3, 5, and 7

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-252 Simultaneous rewrite settings for slave channels 3, 5, and 7 of the Complementary Modulation Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(8) Operating Procedure for Complementary Modulation Output Function

Table 15-253 Operating procedure for Complementary Modulation Output Function (1/2)

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1044* Slave channel 1: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 1" on page 1046* Slave channels 2, 4, and 6: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *6 "Register settings for slave channels 2, 4, and 6" on page 1048* Slave channels 3, 5, and 7: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *7 "Register settings for slave channels 3, 5, and 7" on page 1050* Set the values of the TAUAnCDRm registers of all channels: set the pulse cycle in TAUAnCDRm of the master channel, set the number of interrupts on the master channel to be ignored using TAUAnCDRm of slave channel 1, and set the duty width in TAUAnCDRm of slave channels 2, 4, and 6, and set the dead time delay in slave channels 3, 5, and 7. Set TAUAnTRC.TRCm = 1 for slave channel 1	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 15-253 Operating procedure for Complementary Modulation Output Function (2/2)

	Operation	Status of TAUAn
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously (for channel restart, only slaves 2 to 7). TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm of the master and slave channels is set to 1 and the counters start to count down.
During operation	TAUAnCDRm, TAUAnTRO.TROm, and TAUAnTME.TMEm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel and slave channels 2 to 7 load TAUAnCDRm and count down. Slave channel 1 loads TAUAnCDRm and waits for an interrupt on the master channel. When the counter of the master channel reaches 0000_H: • INTTAUAnIm is generated • TAUAnCNTm reloads the TAUAnCDRm value and continues counting down • TAUAnCNTm of slave channel 1 reduces by 1 and awaits the next interrupt on the master channel • TAUAnCNTm of slave channels 2, 4 and 6 counts in the reverse direction • When the counter of slave channel 1 reaches 0000_H it awaits the next interrupt from the master channel. When it is detected: - TAUAnCNTm reloads the TAUAnCDRm value and awaits the next interrupt on the master channel - INTTAUAnIm is generated - TAUAnTRO.TROm can change • When the counter of slave channels 2, 4, and 6 reaches 0001_H: - INTTAUAnIm is generated - The PWM output of slave channel m resets - TAUAnCNTm of slave channels 3, 5, and 7 load the TAUAnCDRm value and count down • When the counter of slave channels 2, 4, and 6 reaches 0001_H: - INTTAUAnIm is generated TAUAnTTOUTm of slave channels 2 to 7 outputs a PWM signal, a high signal, or a low signal depending on the value of the real-time output bits (TAUAnTRO.TROm), the modulation output bits (TAUAnTME.TMEm), and the output level bits (TAUAnTOL.TOLm) of a pair of slave channels.
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm of slave channels 2 to 7 is initialized to the value set by TAUAnTO.TOm.

Restart

(9) Specific timing diagrams

The following settings apply to the timing diagram:

- Master channel: INTTAUAnIm not generated at operation start (TAUAnCMORm.MD0 = 0)
- Slave channels 2 to 7: positive logic (TAUAnTOL.TOLm = 0)
- In first-half modulation, TAUAnTDL.TDLm is set to 0 for slave channels 2 and 3 so that dead time is added to the positive PWM signal
- In second-half modulation, TAUAnTDL.TDLm is set to 1 for slave channels 2 and 3 so that dead time is added to the negative PWM signal

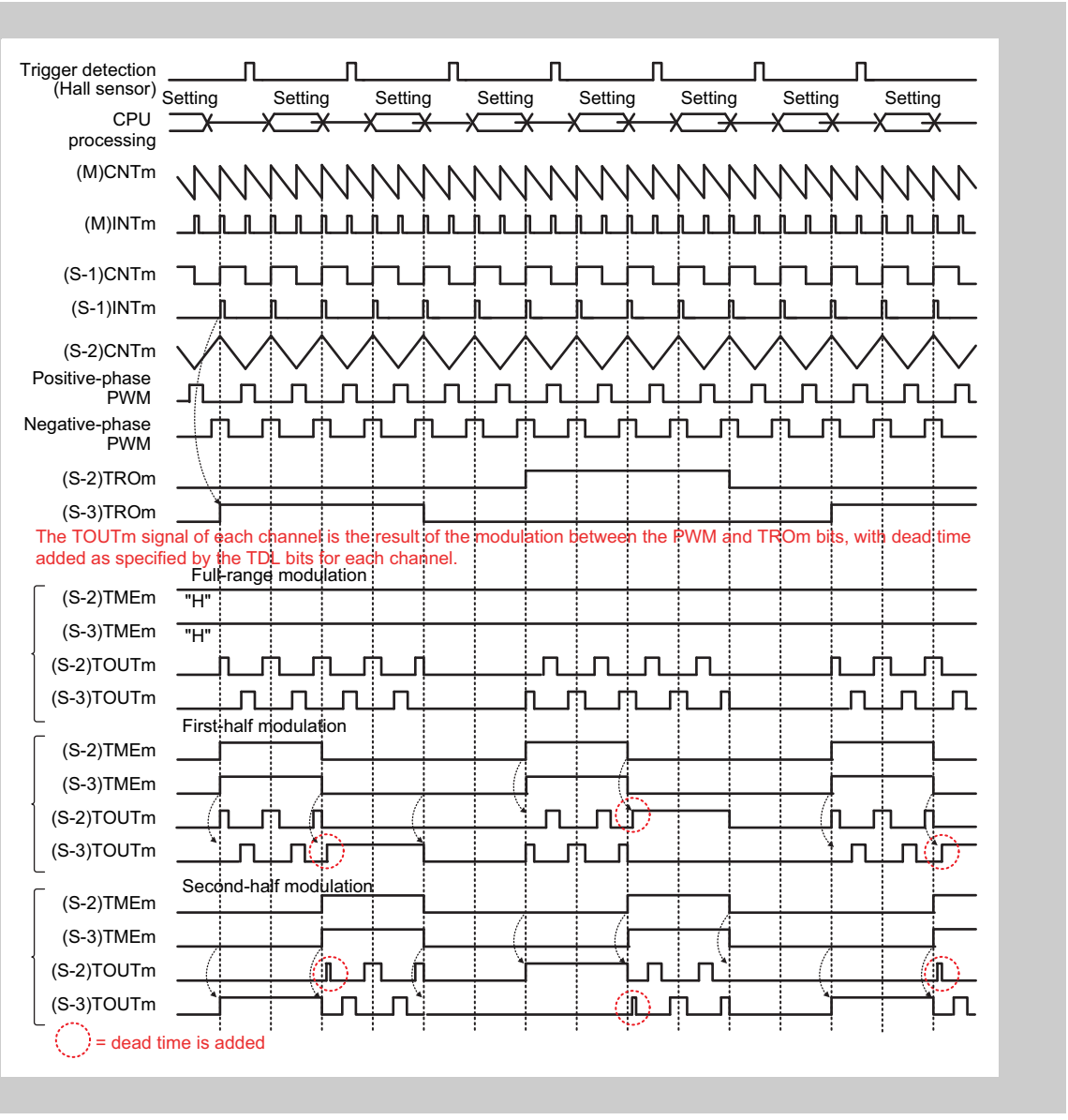


Figure 15-144 Specific timing diagram for Complementary Modulation Output Function

The above timing diagram shows how full modulation, first-half modulation, and second-half modulation can be achieved by modifying the TAUAnTME.TMEm bits of the lower slave channels during operation.

The output of the channels 2 and 3 is the modulation of the PWM output and the value of the TAUAnTRO.TROm bits. Thus the type of PWM signal (positive

or negative) output by an individual channel varies depending on the value of these bits.

The values of the TAUAnTRO.TROm bits are specified by software, but the new values are only applied when an interrupt occurs on slave channel 1.

Note Dead time is added where necessary to prevent both channels outputting a high signal simultaneously.

The “Setting” symbol indicates the time period during which the values of TAUAnCDRm, TAUAnTME.TMEm and TAUAnTRO.TROm can be changed.

15.28 Other Synchronous Channel Functions

This chapter describes a function that is used to ignore a specified number of interrupts on the master channel.

- *15.28.1 “Interrupt Culling Function” on page 1056*

15.28.1 Interrupt Culling Function

(1) Overview

Summary

This function divides the number of interrupts of the master channel by a specified value using a slave channel.

The Interrupt Culling Function is a sub function of the following functions:

- PWM Output Function

  (*15.23.1 “PWM Output Function” on page 876*)
- Triangle PWM Output Function

  (*15.25.1 “Triangle PWM Output Function” on page 940*)
- Triangle PWM Output Function with Dead Time

  (*15.25.2 “Triangle PWM Output Function with Dead Time” on page 951*)

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 15-254 “TAUAnCMORm settings for the master channel of the Interrupt Culling Function” on page 1059*
- The operation mode of the slave channel must be set to Event Count Mode, refer to *Table 15-257 “TAUAnCMORm settings for the slave channel of the Interrupt Culling Function” on page 1061*
- TAUAnTTOUTm is not used for the master or slave channel of this function

Description

The counters (master and slave) are started by setting the channel trigger bit (TAUAnTS.TSm) to 1 for both channels. This in turn sets TAUAnTE.TEm, enabling count operation. The current value of the data register of the master channel and slave channel (TAUAnCDRm) are written to the counter (TAUAnCNTm).

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUAnIm is generated and TAUAnCDRm value is reloaded.
- Slave channel:

  Every time the master channel generates an INTTAUAnIm, the counter of the slave channel reduces by one. When the counter reaches 0000_H, it awaits the next interrupt from the master channel. This causes TAUAnCNTm (slave) to reload the value of TAUAnCDRm, and an INTTAUAnIm is generated.

Forced restart is not possible for this function. The counter can be stopped by setting TAUAnTT.TTm to 1 for the master and slave channel, which in turn sets TAUAnTE.TEm to 0. TAUAnCNTm and TAUAnTTOUTm of master and slave channel stop but retain their values.

Conditions Simultaneous rewrite can be used with this function. Please refer to *15.8 "Simultaneous Rewrite" on page 719* .

(2) Equations

Interrupt divider = TAUAnCDRm (slave channel)

– One INTTAUAnIm is generated for the number of INTTAUAnIm of the master channel defined as TAUAnCDRm (slave channel) + 1.

(3) Block diagram and general timing diagram

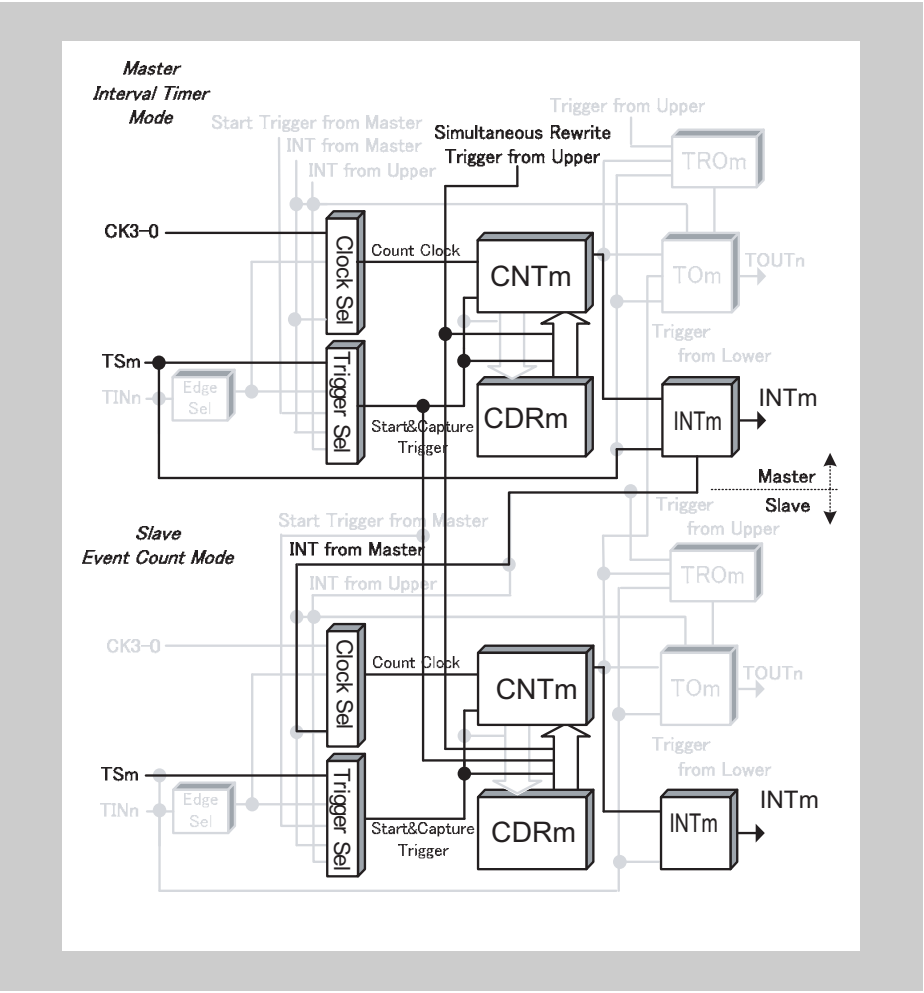


Figure 15-145 Block diagram for Interrupt Culling Function

The following settings apply to the general timing diagram:

Master channel:

- INTTAUAnIm is generated at operation start (TAUAnCMORm.MD0 = 1)

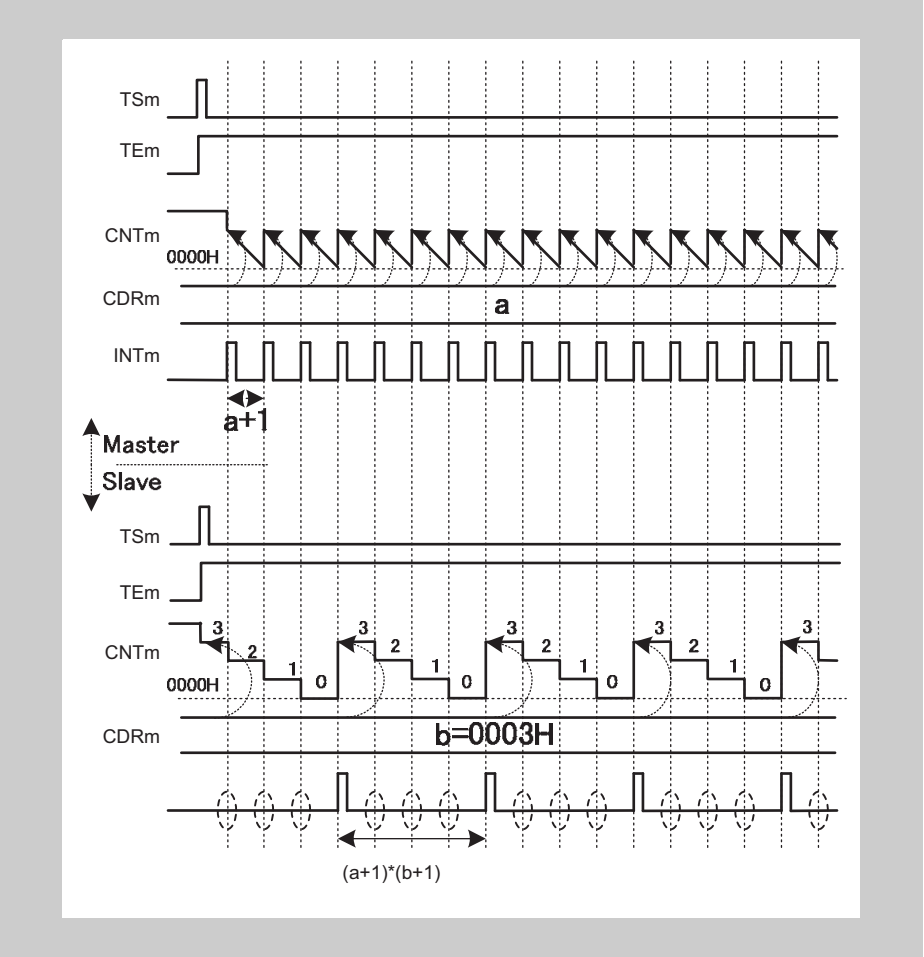


Figure 15-146 General timing diagram for Interrupt Culling Function

The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave

(4) Register settings for the master channel

(a) TAUAnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-254 TAUAnCMORm settings for the master channel of the Interrupt Culling Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUAnIm not output at operation startup 1: Generates INTTAUAnIm at operation startup

(b) TAUAnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-255 TAUAnCMURm settings for the master channel of the Interrupt Culling Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-256 Simultaneous rewrite settings for the master channel of the Interrupt Culling Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting 1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel

(a) TAUAnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 15-257 TAUAnCMORm settings for the slave channel of the Interrupt Culling Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS[1:0]	11: INTTAUAnIm of the master channel is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUAnIm not output at operation startup

(b) TAUAnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 15-258 TAUAnCMURm settings for the slave channel of the Interrupt Culling Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 15-259 Simultaneous rewrite settings for the slave channel of the Interrupt Culling Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting 1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger. If TAUAnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for Interrupt Culling Function

Table 15-260 Operating procedure for Interrupt Culling Function

	Operation	Status of TAUAn
Initial channel setting	Master channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1059* Slave channel: set the TAUAnCMORm and TAUAnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel" on page 1061* Set the values of the TAUAnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUAnTS.TSm of the master and slave channels to 1 simultaneously. TAUAnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUAnIm is generated on the master channel.
During operation	TAUAnCDRm can be changed at any time. TAUAnCNTm and TAUAnRSF.RSFm can be read at any time. TAUAnRDT.RDTm can be changed during operation.	TAUAnCNTm of the master channel loads TAUAnCDRm and counts down. When the counter reaches 0000_H: • INTTAUAnIm (master) is generated • TAUAnCNTm (master) reloads the TAUAnCDRm value and continues count operation. • TAUAnCNTm of the slave channel counts down by one unit. When TAUAnCNTm of the slave reaches 0000_H: • INTTAUAnIm (slave) is generated
Stop operation	Set TAUAnTT.TTm of the master and slave channels to 1 simultaneously. TAUAnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUAnTE.TEm is cleared to 0 and the counter stops. TAUAnCNTm and TAUAnTTOUTm stop and retain their current values. When TAUAnTOE.TOEm is 0, TAUAnTTOUTm output is initialized to the value set by TAUAnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Number of interrupts (master) = number of interrupts (slave)

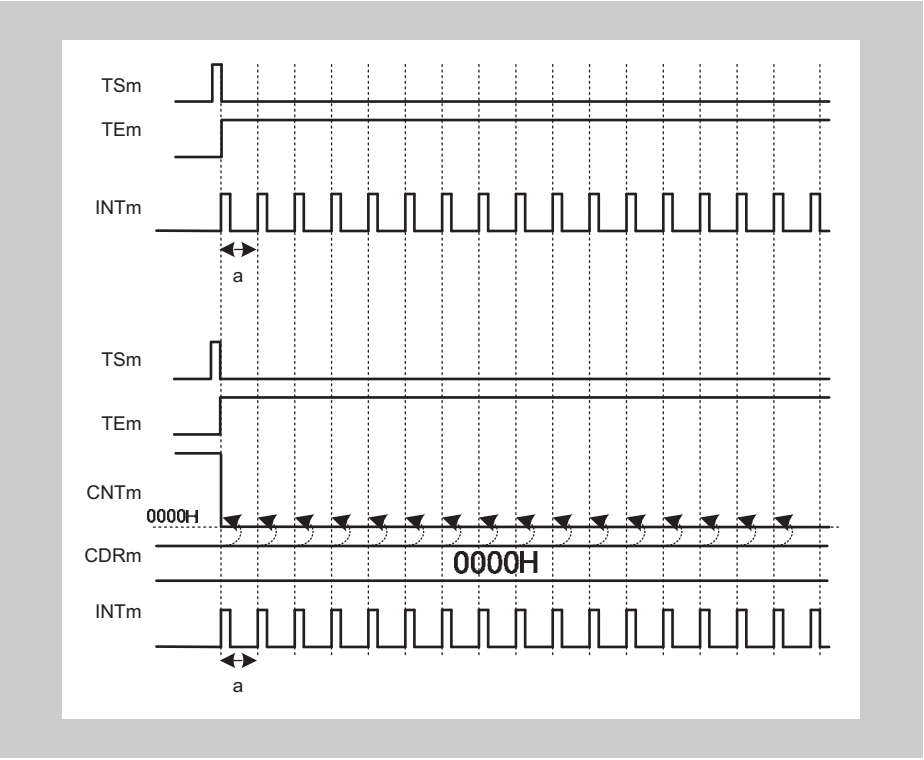


Figure 15-147 TAUAnCDRm (slave) = 0000_H

- Every time the master channel generates an INTTAUAnIm, the slave channel reloads 0000_H.
- Therefore, the slave channel generates an interrupt at the same time as an interrupt is generated on the master channel.

15.29 Registers

This section contains a description of all the registers of the 16-bit Timer Array Unit A.

15.29.1 TAUAn registers overview

The TAUAn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 15.

Table 15-261 TAUAn registers overview (1/2)

Register name	Shortcut	Address
TAUAn prescaler registers		
TAUAn prescaler clock select register	TAUAnTPS	<TAUAn_base> + 240_H
TAUAn prescaler baud rate setting register	TAUAnBRS	<TAUAn_base> + 244_H
TAUAn control registers		
TAUAn channel data register m	TAUAnCDRm	<TAUAn_base> + m x 4_H
TAUAn channel counter register m	TAUAnCNTm	<TAUAn_base> + 80_H + m x 4_H
TAUAn channel mode OS register m	TAUAnCMORm	<TAUAn_base> + 200_H + m x 4_H
TAUAn channel mode user register m	TAUAnCMURm	<TAUAn_base> + $C0_H$ + m x 4_H
TAUAn channel status register m	TAUAnCSRm	<TAUAn_base> + 140_H + m x 4_H
TAUAn channel status clear trigger register m	TAUAnCSCm	<TAUAn_base> + 180_H + m x 4_H
TAUAn channel start trigger register	TAUAnTS	<TAUAn_base> + $1C4_H$
TAUAn channel enable status register	TAUAnTE	<TAUAn_base> + $1C0_H$
TAUAn channel stop trigger register	TAUAnTT	<TAUAn_base> + $1C8_H$
TAUAn output registers		
TAUAn channel output enable register	TAUAnTOE	<TAUAn_base> + $5C_H$
TAUAn channel output register	TAUAnTO	<TAUAn_base> + 58_H
TAUAn channel output mode register	TAUAnTOM	<TAUAn_base> + 248_H
TAUAn channel output configuration register	TAUAnTOC	<TAUAn_base> + $24C_H$
TAUAn channel output active level register	TAUAnTOL	<TAUAn_base> + 040_H
TAUAn channel dead time output enable register	TAUAnTDE	<TAUAn_base> + 250_H
TAUAn channel dead time output mode register	TAUAnTDM	<TAUAn_base> + 254_H
TAUAn channel dead time output level register	TAUAnTDL	<TAUAn_base> + 54_H
TAUAn channel real-time output register	TAUAnTRO	<TAUAn_base> + $4C_H$
TAUAn channel real-time output enable register	TAUAnTRE	<TAUAn_base> + 258_H
TAUAn channel real-time output control register	TAUAnTRC	<TAUAn_base> + $25C_H$
TAUAn channel modulation output enable register	TAUAnTME	<TAUAn_base> + 50_H
TAUAn simultaneous rewrite registers		
TAUAn channel reload data enable register	TAUAnRDE	<TAUAn_base> + 260_H
TAUAn channel reload data mode register	TAUAnRDM	<TAUAn_base> + 264_H
TAUAn channel reload data control CH select register	TAUAnRDS	<TAUAn_base> + 268_H
TAUAn channel reload data control register	TAUAnRDC	<TAUAn_base> + $26C_H$

Table 15-261 TAUAn registers overview (2/2)

Register name	Shortcut	Address
TAUAn channel reload data trigger register	TAUAnRDT	<TAUAn_base> + 44_H
TAUAn channel reload status register	TAUAnRSF	<TAUAn_base> + 48_H
TAUAn DMA window registers		
TAUAn DMA window address setting register 0	TAUAnDAS0	<TAUAn_base> + 270_H
TAUAn DMA window address setting register 1	TAUAnDAS1	<TAUAn_base> + 274_H
TAUAn DMA window address setting register 2	TAUAnDAS2	<TAUAn_base> + 278_H
TAUAn DMA window address setting register 3	TAUAnDAS3	<TAUAn_base> + $27C_H$
TAUAn DMA window address setting register 4	TAUAnDAS4	<TAUAn_base> + 280_H
TAUAn DMA window address setting register 5	TAUAnDAS5	<TAUAn_base> + 284_H
TAUAn DMA window address setting register 6	TAUAnDAS6	<TAUAn_base> + 288_H
TAUAn DMA window address setting register 7	TAUAnDAS7	<TAUAn_base> + $28C_H$
TAUAn DMA window register m	TAUAnDWRm	<TAUAn_base> + 100_H + m x 4_H
TAUAn emulation register		
TAUAn emulation register	TAUAnEMU	<TAUAn_base> + 290_H

<TAUAn_base> The <TAUAn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

15.29.2 TAUAn prescaler registers details

(1) TAUAnTPS - TAUAn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3_PRE for all channels. CK3 is generated by dividing CK3_PRE by the factor specified in TAUAnBRS.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 240_H

Initial Value $FFFF_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS3[3:0]				PRS2[3:0]				PRS1[3:0]				PRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-262 TAUAnTPS register contents (1/2)

Bit position	Bit name	Function
15 to 12	PRS3[3:0]	Specifies the CK3_PRE clock. Clock CK3_PRE is the input clock of the BRG unit. The BRG unit supplies the CK3 operation clock for all channels. **PRS3[3:0] / CK3_PRE clock** 0000_B / PCLK/2^0 0001_B / PCLK/2^1 0010_B / PCLK/2^2 0011_B / PCLK/2^3 ... / ... 1110_B / PCLK/2^{14} 1111_B / PCLK/2^{15} These bits can only be rewritten when all counters using CK3 are stopped (TAUAnTE.TEm = 0).
11 to 8	PRS2[3:0]	Specifies the CK2 clock. **PRS2[3:0] / CK2 clock** 0000_B / PCLK/2^0 0001_B / PCLK/2^1 0010_B / PCLK/2^2 0011_B / PCLK/2^3 ... / ... 1110_B / PCLK/2^{14} 1111_B / PCLK/2^{15} These bits can only be rewritten when all counters using CK2 are stopped (TAUAnTE.TEm = 0).

Table 15-262 TAUAnTPS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUAnTE.TEm = 0).</td></tr>
<tr><td>3 to 0</td><td>PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUAnTE.TEm = 0).</td></tr>
</table>

Note The TAUAn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

(2) TAUAnBRS - TAUAn prescaler baud rate setting register

This register specifies the division factor of prescaler clock CK3.

CK3 is generated by dividing CK3_PRE by the factor specified in this register plus one. The PCLK prescaler for CK3_PRE is specified in TAUAnTPS.PRS3[3:0].

Access This register can be read/written in 8-bit units.

Address <TAUAn_base> + 244_H

Initial Value 00_H

7	6	5	4	3	2	1	0
BRS[07:00]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-263 TAUAnBRS register contents

Bit position	Bit name	Function
7 to 0	BRS[07:00]	Specifies the CK3_PRE clock division factor for generating CK3:

BRS[07:00]	CK3 clock
$0000\ 0000_B$	CK3_PRE / 1
$0000\ 0001_B$	CK3_PRE / 2
$0000\ 0010_B$	CK3_PRE / 3
$0000\ 0011_B$	CK3_PRE / 4
...	...
$1111\ 1110_B$	CK3_PRE / 255
$1111\ 1111_B$	CK3_PRE / 256

15.29.3 TAUAn control registers details

(1) TAUAnCDRm - TAUAn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUAnCMORm.MD[4:0].

Access This register can be read/written in 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUAn_base> + 0_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDR[15:0]															
R/W															

Table 15-264 TAUAnCDRm register contents

Bit position	Bit name	Function
15 to 0	CDR[15:0]	Data register for the capture/compare value.

(2) TAUAnCNTm - TAUAn channel counter register

This register is the channel m counter register.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + 80_H + m x 4_H

Initial Value 0000_H or $FFFF_H$ The initial value depends on the operation mode, see *Table 15-266 "TAUAnCNTm read values after the counter is re-enabled" on page 1071* .

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R															

Table 15-265 TAUAnCNTm register contents

Bit position	Bit name	Function
15 to 0	CNT[15:0]	16-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUAnTS.TSm and TAUAnTT.TTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUAnTT.TTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUAnTE.TEm= 0) and re-enabled (TAUAnTS.TSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUAnTS.TSm = 1) for modes where the counter waits for a start trigger.

Table 15-266 TAUAnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUAnCNTm value		
		After reset	After stop trigger	After one count
Interval Timer mode	Count down	$FFFF_H$	Stop value	-
Judge mode	Count down	$FFFF_H$	Stop value	-
Capture mode	Count up	0000_H	Stop value	-
Event Count mode	Count down	$FFFF_H$	Stop value	-
One Count mode	Count down	$FFFF_H$	Stop value	$FFFF_H$
Capture & One Count mode	Count up	0000_H	Stop value	Captured value + 1 (TAUAnCDRm)
Judge & One Count mode	Count down	$FFFF_H$	Stop value	TAUAnCNTm value - 1
Up Down Count mode	Count up/down	$FFFF_H$	Stop value	-
Pulse One Count mode	Count down	$FFFF_H$	Stop value	0000_H
Count Capture Mode	Count up	0000_H	Stop value	-
Gate Count Mode	Count down	$FFFF_H$	Stop value	Stop value
Capture & Gate Count Mode	Count up	0000_H	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUAnCMORm.MD[4:0].

(3) TAUAnCMORm - TAUAn channel mode OS register

This register controls channel m operation. It specifies, for example, the operation clock, count clock, and master/slave function.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUAnTE.TEm = 0).

Address <TAUAn_base> + 200_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:0]				
R/W		R/W		R/W	R/W			R/W		R	R/W				

Table 15-267 TAUAnCMORm register contents (1/3)

Bit position	Bit name	Function
15,14	CKS[1:0]	Selects the operation clock. The operation clock is used for the TAUAnTTINm input edge detection circuit. It can also be used as the count clock depending on bits TAUAnCMORm.CCS[1:0]. **CKS1 / CKS0 / Selected operation clock** 0 / 0 / CK0 0 / 1 / CK1 1 / 0 / CK2 1 / 1 / CK3
13,12	CCS[1:0]	Selects the count clock for TAUAnCNTm counter: **CCS1 / CCS0 / Selected count clock** 0 / 0 / Operation clock as specified by TAUAnCMORm.CKS[1:0]. 0 / 1 / Valid edge of TAUAnTTINm input signal 1 / 0 / Setting prohibited 1 / 1 / INTTAUAnIm signal of the master channel
11	MAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

Table 15-267 TAUAnCMORm register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>STS[2:0]</td><td>Selects the external start trigger:
<table>
<tr><th>STS2</th><th>STS1</th><th>STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUAnTTINm input signal. TAUAnCMURm.TIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUAnTTINm input signal is the start trigger and the reverse edge is the stop (capture) trigger</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INTTAUAnI of the master channel</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>INTTAUAnI of the upper channel (m-1), regardless of the master setting</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Dead-time output signal TAUAnTTDL of the TAUAnTTOUTm generation unit</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Up/down output trigger signal TAUAnTUDSm of the master channel.</td></tr>
</table></td></tr>
<tr><td>7, 6</td><td>COS[1:0]</td><td>Specifies when the capture register TAUAnCDRm and the overflow flag TAUAnCSRm.OVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>COS1</th><th>COS0</th><th>Capture register</th><th>TAUAnCSRm.OVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUAnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUAnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUAnCSRm.OVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUAnCSR.OVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUAnTTINm input valid edge and upon counter overflow:
• TAUAnTTINm input valid edge: Counter value is written to TAUAnCDRm
• Overflow: $FFFF_H$ is written to TAUAnCDRm. The next TAUAnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
</table></td></tr>
</table>

Table 15-267 TAUAnCMORm register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>4 to 0</td><td>MD[4:0]</td><td>Specifies the operation mode.
<table>
<tr><th>MD4</th><th>MD3</th><th>MD2</th><th>MD1</th><th>MD0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1/0</td><td>Interval Timer mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1/0</td><td>Judge mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Capture mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>Event Count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1/0</td><td>One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Capture & One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1/0</td><td>Judge & One Count mode</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Up Down Count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Pulse One Count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Count Capture mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>Gate Count mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Capture & Gate Count mode</td></tr>
</table>

Mode	Role of the MD0 bit
Interval Timer mode Capture mode Count Capture mode	Specifies whether an INTTAUAnIm is generated when the counter is triggered: 0: No INTTAUAnIm generated 1: INTTAUAnIm generated
Event Count mode Up Down Count mode	This bit must be set to 0: 0: No INTTAUAnIm generated when the counter is triggered 1: -
One Count mode Gate Count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUAnIm and TAUAnTTOUTm are not output when the counter is triggered.
Pulse One Count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUAnIm and TAUAnTTOUTm are output when the counter is triggered.
Capture & One Count mode Capture & Gate Count mode	This bit must be set to 0: 0: No INTTAUAnIm generated when the counter is triggered. Start trigger is disabled during counting. 1: -
Judge mode Judge One Count mode	Specifies when INTTAUAnIm is generated: 0: When TAUAnCNTm ≤ TAUAnCDRm 1: When TAUAnCNTm > TAUAnCDRm
</td></tr>
</table>

(4) TAUAnCMURm - TAUAn channel mode user register

This register specifies the type of valid edge detection used for the TAUAnTTINm input.

Access This register can be read/written in 8-bit units. It can only be written when the counter is enabled (TAUAnTE.TEm = 1).

Address <TAUAn_base> + $C0_H$ + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	TIS[1:0]	
R	R	R	R	R	R	R/W	R/W

Table 15-268 TAUAnCMURm register contents

Bit position	Bit name	Function
1, 0	TIS[1:0]	Specifies the valid edge of the TAUAnTTINm input: • To detect rising and falling edges when TAUAnCMORm.STS[2:0] is not set to 010_B, set TAUAnCMURm.TIS[1:0] = 10_B. • Edge detection for TAUAnTTINm input signals is performed based on the operation clock selected by TAUAnCMORm.CKS[1:0].

TIS1	TIS0	Description
0	0	Falling edge
0	1	Rising edge
1	0	Rising and falling edges (low-width measurement selection). Start trigger: falling edge Stop trigger (capture): rising edge
1	1	Rising and falling edges (high-width measurement selection). Start trigger: rising edge Stop trigger (capture): falling edge This setting is only valid if the start trigger selection is set to TAUAnCMORm.STS[2:0] = 010_B

(5) TAUAnCSRm - TAUAn channel status register

This register indicates the count direction and the overflow status of channel m's counter.

Access This register can be read in 8-bit units.

Address <TAUAn_base> + 140_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	CSF	OVF
R	R	R	R	R	R	R	R

Table 15-269 TAUAnCSRm register contents

Bit position	Bit name	Function
1	CSF	Indicates the count direction: 0: Counts up 1: Counts down The read value of this bit is only valid in the following mode: • Up Down Count mode For channel 0 this bit is fixed to 0.
0	OVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & One Count mode • Count Capture mode • Capture & Gate Count mode The function of this bit depends on the setting of control bits TAUAnCMORm.COS[1:0]. OVF is not be set when TAUAnCMORm.COS[1:0] = 10_B.

(6) TAUAnCSCm - TAUAn channel status clear register

This register is a trigger register for clearing the overflow flag TAUAnCSRm.OVF of a channel m.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUAn_base> + 180_H + m x 4_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	0	CLOV
R	R	R	R	R	R	R	W

Table 15-270 TAUAnCSCm register contents

Bit position	Bit name	Function
0	CLOV	0: No function 1: Clears the overflow flag TAUAnCSRm.OVF

(7) TAUAnTS - TAUAn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + $1C4_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS 15	TS 14	TS 13	TS 12	TS 11	TS 10	TS 09	TS 08	TS 07	TS 06	TS 05	TS 04	TS 03	TS 02	TS 01	TS 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-271 TAUAnTS register contents

Bit position	Bit name	Function
15 to 0	TSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUAnTE.TEm = 1. When the counter is enabled, this bit immediately returns to 0. TAUAnTE.TEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUAnTE - TAUAn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + $1C0_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE15	TE14	TE13	TE12	TE11	TE10	TE09	TE08	TE07	TE06	TE05	TE04	TE03	TE02	TE01	TE00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15-272 TAUAnTE register contents

Bit position	Bit name	Function
15 to 0	TEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting TAUAnTS.TSm to 1 or trigger input detection TAUAnTSSTm = 1 sets this bit to 1. Setting TAUAnTT.TTm to 1 resets this bit to 0.

(9) TAUAnTT - TAUAn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + $1C8_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TT15	TT14	TT13	TT12	TT11	TT10	TT09	TT08	TT07	TT06	TT05	TT04	TT03	TT02	TT01	TT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-273 TAUAnTT register contents

Bit position	Bit name	Function
15 to 0	TTm	Stops the counter of channel m: 0: No function 1: Stops the counter and sets TAUAnTE.TEm = 0. When the counter has stopped, this bit immediately returns to 0. TAUAnCNTm stops counting and TAUAnCNTm, TAUAnTO.TOm, and TAUAnTTOUTm all retain the values they had before the counter was stopped.

15.29.4 TAUAn output registers details

(1) TAUAnTOE - TAUAn channel output enable register

This register enables and disables Direct Channel Output Mode.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + $5C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOE 15	TOE 14	TOE 13	TOE 12	TOE 11	TOE 10	TOE 09	TOE 08	TOE 07	TOE 06	TOE 05	TOE 04	TOE 03	TOE 02	TOE 01	TOE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-274 TAUAnTOE register contents

Bit position	Bit name	Function
15 to 0	TOEm	Enables/disables Direct Channel Output Mode: 0: Enables Direct Channel Output Mode (TAUAnTTOUT m output is controlled by the application software) 1: Disables Direct Channel Output Mode (TAUAnTTOUT m output is controlled by the timer)

(2) TAUAnTOM - TAUAn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUAnTE.TEm = 0).

Address <TAUAn_base> + 248_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM 15	TOM 14	TOM 13	TOM 12	TOM 11	TOM 10	TOM 09	TOM 08	TOM 07	TOM 06	TOM 05	TOM 04	TOM 03	TOM 02	TOM 01	TOM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-275 TAUAnTOM register contents

Bit position	Bit name	Function
15 to 0	TOMm	Specifies the timer controlled channel output mode, if Direct Channel Output Mode is disabled (TAUAnTOE.TOEm = 1): 0: Independent Channel Output Mode 1: Synchronous Channel Output Mode The output mode depends on several channel output control bits, as can be seen in the table *"Channel output modes"* in the section *"Channel output modes"* above.

(3) TAUAnTOC - TAUAn channel output configuration register

This register specifies the output mode of each channel in combination with TAUAnTOMm.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUAnTE.TEm = 0).

Address <TAUAn_base> + $24C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC 15	TOC 14	TOC 13	TOC 12	TOC 11	TOC 10	TOC 09	TOC 08	TOC 07	TOC 06	TOC 05	TOC 04	TOC 03	TOC 02	TOC 01	TOC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-276 TAUAnTOC register contents

Bit position	Bit name	Function
15 to 0	TOCm	Specifies the output mode: 0: Operation mode 1 1: Operation mode 2 The output mode also depends on TAUAnTOM.TOMm, as can be seen in the following table.

TOMm	TOCm	Description
0	0	Toggle mode: TAUAnTTOUTm toggles when INTTAUAnIm occurs.
	1	Set/reset mode: TAUAnTTOUTm set when INTTAUAnIm occurs upon count start and reset when INTTAUAnIm occurs due to detection of a match between TAUAnCNTm and TAUAnCDRm, or, in One-Shot Pulse Output Function when the counter is 0.
1	0	Synchronous Channel Operation Mode 1: TAUAnTTOUTm set when INTTAUAnI occurs on the master channel and reset when INTTAUAnI occurs on the slave channel.
	1	Synchronous Channel Operation Mode 2: TAUAnTTOUTm set when INTTAUAnIm occurs while the slave channel is counting down and reset when INTTAUAnIm occurs while the slave channel is counting up.

(4) TAUAnTDE - TAUAn channel dead time output enable register

This register enables/disables dead time operation for each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUAnTE.TEm = 0).

Address <TAUAn_base> + 250_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDE 15	TDE 14	TDE 13	TDE 12	TDE 11	TDE 10	TDE 09	TDE 08	TDE 07	TDE 06	TDE 05	TDE 04	TDE 03	TDE 02	TDE 01	TDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-277 TAUAnTDE register contents

Bit position	Bit name	Function
15 to 0	TDEm	Enables/disables dead time control operation of channel m: 0: Disables dead time operation 1: Enables dead time operation The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUAnTOE.TOEm, TAUAnTOM.TOMm, and TAUAnTOC.TOCm = 1.

(5) TAUAnTDM - TAUAn channel dead time output mode register

This register specifies when dead time is added during dead time output.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUAnTE.TEm = 0).

Address <TAUAn_base> + 254_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDM 15	TDM 14	TDM 13	TDM 12	TDM 11	TDM 10	TDM 09	TDM 08	TDM 07	TDM 06	TDM 05	TDM 04	TDM 03	TDM 02	TDM 01	TDM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-278 TAUAnTDM register contents

Bit position	Bit name	Function
15 to 0	TDMm	Specifies when dead time is added during dead time output: 0: Upon detection of a duty cycle at the upper even channel (duty dead time output) 1: Upon detection of a TAUAnTTIN input edge at the lower odd channel (one-phase dead time output) The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUAnTOE.TOEm, TAUAnTOM.TOMm, TAUAnTOC.TOCm, and TAUAnTDE.TDEm = 1.

(6) TAUAnTDL - TAUAn channel dead time output level register

This register selects the phase period to which dead time is added.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 54_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDL 15	TDL 14	TDL 13	TDL 12	TDL 11	TDL 10	TDL 09	TDL 08	TDL 07	TDL 06	TDL 05	TDL 04	TDL 03	TDL 02	TDL 01	TDL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-279 TAUAnTDL register contents

Bit position	Bit name	Function
15 to 0	TDLm	Selects the phase period to which dead time is added: 0: Positive phase period 1: Negative phase period These bits only apply when: • TAUAnTOE.TOEm, TAUAnTOM.TOMm, TAUAnTOC.TOCm, and TAUAnTDE.TDEm = 1.

(7) TAUAnTRE - TAUAn channel real-time output enable register

This register enables or disables real-time output.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0.

Address <TAUAn_base> + 258_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRE 15	TRE 14	TRE 13	TRE 12	TRE 11	TRE 10	TRE 09	TRE 08	TRE 07	TRE 06	TRE 05	TRE 04	TRE 03	TRE 02	TRE 01	TRE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-280 TAUAnTRE register contents

Bit position	Bit name	Function
15 to 0	TREm	Enables or disables real-time output of channel m. 0: Disables real-time output 1: Enables real-time output These bits only apply when TAUTnTOE.TOEm = 1. When TAUAnTRE.TREm = 0, TAUAnTTOUTm is not affected by real-time output. When TAUAnTRE.TREm = 1, TAUAnTTOUTm outputs the value of the real-time output bit TAUAnTRO.TROm, dependent on the timer operation.

(8) TAUAnTRC - TAUAn channel real-time control register

This register controls the real-time output trigger for each channel.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0.

Address <TAUAn_base> + $25C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRC 15	TRC 14	TRC 13	TRC 12	TRC 11	TRC 10	TRC 09	TRC 08	TRC 07	TRC 06	TRC 05	TRC 04	TRC 03	TRC 02	TRC 01	TRC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-281 TAUAnTRC register contents

Bit position	Bit name	Function
15 to 0	TRCm	Specifies which channel generates the real-time output trigger for channel m: 0: The next upper channel where this bit is set to 1 1: Channel m These bits only apply when TAUAnTRE.TREm = 1.

(9) TAUAnTRO - TAUAn channel real-time output register

This register specifies and reads the real-time output.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + $04C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRO 15	TRO 14	TRO 13	TRO 12	TRO 11	TRO 10	TRO 09	TRO 08	TRO 07	TRO 06	TRO 05	TRO 04	TRO 03	TRO 02	TRO 01	TRO 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-282 TAUAnTRO register contents

Bit position	Bit name	Function
15 to 0	TROm	Specifies and reads the real-time output: 0: Real-time output “Low” 1: Real-time output “High” These bits only apply when TAUAnTRE.TRE = 1.

(10) TAUAnTME - TAUAn channel modulation output enable register

This register enables and disables modulation output for the timer output and real-time output.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 050_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TME 15	TME 14	TME 13	TME 12	TME 11	TME 10	TME 09	TME 08	TME 07	TME 06	TME 05	TME 04	TME 03	TME 02	TME 01	TME 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-283 TAUAnTME register contents

Bit position	Bit name	Function
15 to 0	TMEm	Enables/disables modulation output for timer output and real-time output of channel m: 0: Disables modulation 1: Enables modulation These bits only apply when TAUAnTOE.TOEm and TAUAnTRE.TREm = 1.

15.29.5 TAUAn channel output level registers details

(1) TAUAnTO - TAUAn channel output register

This register specifies and reads the level of TAUAnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 58_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO09	TO08	TO07	TO06	TO05	TO04	TO03	TO02	TO01	TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-284 TAUAnTO register contents

Bit position	Bit name	Function
15 to 0	TOm	Specifies/reads the level of TAUAnTTOUTm: 0: Low 1: High Only TOm bits for which Independent Channel Output function is disabled (TAUAnTOEm = 0) can be written.

(2) TAUAnTOL - TAUAn channel output level register

This register specifies the output logic of the channel output bit (TAUAnTO.TOm).

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 040_H

Initial Value 00_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOL 15	TOL 14	TOL 13	TOL 12	TOL 11	TOL 10	TOL 09	TOL 08	TOL 07	TOL 06	TOL 05	TOL 04	TOL 03	TOL 02	TOL 01	TOL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-285 TAUAnTOL register contents

Bit position	Bit name	Function
15 to 0	TOLm	Specifies the output logic of the channel m output bit (TAUAnTO.TOm): 0: Positive logic (active high) 1: Inverted logic (active low) These bits apply in all channel output modes except Direct Channel Output Mode and channel output modes with real-time output.

15.29.6 TAUAn simultaneous rewrite register details

(1) TAUAnRDE - TAUAn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUAnCDRm. It also enables simultaneous rewrite of the data register TAUAnTOLm for the PWM output function and the triangle PWM output function.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0.

Address <TAUAn_base> + 260_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDE 15	RDE 14	RDE 13	RDE 12	RDE 11	RDE 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-286 TAUAnRDE register contents

Bit position	Bit name	Function
15 to 0	RDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUAnRDM - TAUAn channel reload data mode register

This register selects when the signal that controls simultaneous rewrite is loaded.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0.

Address <TAUAn_base> + 264_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDM 15	RDM 14	RDM 13	RDM 12	RDM 11	RDM 10	RDM 09	RDM 08	RDM 07	RDM 06	RDM 05	RDM 04	RDM 03	RDM 02	RDM 01	RDM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-287 TAUAnRDM register contents

Bit position	Bit name	Function
15 to 0	RDMm	Selects when the signal that triggers simultaneous is generated: 0: When the master channel counter starts counting 1: At the top of a triangle wave cycle These bits only apply when TAUAnRDE.RDEm = 1 and TAUAnRDS.RDSm = 0.

(3) TAUAnRDS - TAUAn channel reload data control channel select register

This register selects the control channel for simultaneous rewrite.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0.

Address <TAUAn_base> + 268_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDS 15	RDS 14	RDS 13	RDS 12	RDS 11	RDS 10	RDS 09	RDS 08	RDS 07	RDS 06	RDS 05	RDS 04	RDS 03	RDS 02	RDS 01	RDS 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-288 TAUAnRDS register contents

Bit position	Bit name	Function
15 to 0	RDSm	Specifies which channel is monitored for the simultaneous rewrite trigger: 0: Master channel 1: Another upper channel

(4) TAUAnRDC - TAUAn channel reload data control register

This register specifies the channel that generates the INTTAUAnIm signal that triggers simultaneous rewrite.

Access This register can be read/written in 16-bit units. It can only be written when TAUAnTE.TEm = 0

Address <TAUAn_base> + $26C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDC 15	RDC 14	RDC 13	RDC 12	RDC 11	RDC 10	RDC 09	RDC 08	RDC 07	RDC 06	RDC 05	RDC 04	RDC 03	RDC 02	RDC 01	RDC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-289 TAUAnRDC register contents

Bit position	Bit name	Function
15 to 0	RDCm	Specifies whether the channel is monitored for an INTTAUAnIm signal that is used as the simultaneous rewrite trigger: 0: Channel is not monitored 1: Channel is monitored These bits only apply when TAUAnRDS.RDSm = 1.

(5) TAUAnRDT - TAUAn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUAn_base> + 044_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT 15	RDT 14	RDT 13	RDT 12	RDT 11	RDT 10	RDT 09	RDT 08	RDT 07	RDT 06	RDT 05	RDT 04	RDT 03	RDT 02	RDT 01	RDT 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 15-290 TAUAnRDT register contents

Bit position	Bit name	Function
15 to 0	RDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUAnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger. TAUAnRDT.RDTm immediately returns to 0.

(6) TAUAnRSF - TAUAn channel reload status register

This flag register indicates that simultaneous rewrite is possible.

Access This register can be read in 16-bit units.

Address <TAUAn_base> + 048_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSF 15	RSF 14	RSF 13	RSF 12	RSF 11	RSF 10	RSF 09	RSF 08	RSF 07	RSF 06	RSF 05	RSF 04	RSF 03	RSF 02	RSF 01	RSF 00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15-291 TAUAnRSF register contents

Bit position	Bit name	Function
15 to 0	RSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewrite disabled 1: Simultaneous rewrite enabled

15.29.7 TAUAn DMA window registers

(1) TAUAnDASi - TAUAn DMA window address setting register i (i = 0 to 7)

This register specifies addresses of the window registers used for DMA. Eight TAUAnDASi registers control sixteen TAUAnDWR registers, i.e. each TAUAnDASi register controls two TAUAnDWR registers independently.

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 0270_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DAS *odd7*	DAS *odd6*	DAS *odd5*	DAS *odd4*	DAS *odd3*	DAS *odd2*	0	0	DAS *even7*	DAS *even6*	DAS *even5*	DAS *even4*	DAS *even3*	DAS *even2*	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

odd = 01, 03, 05, 07, 09, 11, 13, 15

even = 00, 02, 04, 06, 08, 10, 12, 14

The following table shows the relationship between the TAUAnDAS control registers and the TAUAnDWRm registers:

Table 15-292 Relationship between TAUAnDAS registers and TAUAnDWRm registers

Control register	Control bits	Control target register TAUAnDWRm
TAUAnDAS0	Bits 7 to 0	TAUAnDWR0
TAUAnDAS0	Bits 15 to 8	TAUAnDWR1
TAUAnDAS1	Bits 7 to 0	TAUAnDWR2
TAUAnDAS1	Bits 15 to 8	TAUAnDWR3
TAUAnDAS2	Bits 7 to 0	TAUAnDWR4
TAUAnDAS2	Bits 15 to 8	TAUAnDWR5
TAUAnDAS3	Bits 7 to 0	TAUAnDWR6
TAUAnDAS3	Bits 15 to 8	TAUAnDWR7
TAUAnDAS4	Bits 7 to 0	TAUAnDWR8
TAUAnDAS4	Bits 15 to 8	TAUAnDWR9
TAUAnDAS5	Bits 7 to 0	TAUAnDWR10
TAUAnDAS5	Bits 15 to 8	TAUAnDWR11
TAUAnDAS6	Bits 7 to 0	TAUAnDWR12
TAUAnDAS6	Bits 15 to 8	TAUAnDWR13
TAUAnDAS7	Bits 7 to 0	TAUAnDWR14
TAUAnDAS7	Bits 15 to 8	TAUAnDWR15

Table 15-293 TAUAnDASm register contents

Bit position	Bit name	Function
15 to 8	DAS*odd*[15:8]	00_H to $3C_H$: Specifies the CDR0 to CDR15 registers. 40_H: Specifies the TOL register. 44_H: Specifies the RDT register. 48_H: Specifies the RSF register. $4C_H$: Specifies the TRO register. 50_H: Specifies the TME register. 54_H: Specifies the TDL register. 58_H: Specifies the TO register. $5C_H$: Specifies the TOE register. 60_H to $7C_H$: Setting prohibited 80_H to BC_H: Specifies the CNT0 to CNT15 registers. $C0_H$ to FC_H: Setting prohibited
7 to 0	DAS*even*[7:0]	00_H to $3C_H$: Specifies the CDR0 to CDR15 registers. 40_H: Specifies the TOL register. 44_H: Specifies the RDT register. 48_H: Specifies the RSF register. $4C_H$: Specifies the TRO register. 50_H: Specifies the TME register. 54_H: Specifies the TDL register. 58_H: Specifies the TO register. $5C_H$: Specifies the TOE register. 60_H to $7C_H$: Setting prohibited 80_H to BC_H: Specifies the CNT0 to CNT15 registers. $C0_H$ to FC_H: Setting prohibited

(2) TAUAnDWRm - TAUAn DMA window register m

This register is used for DMA (m = 0 to 15). TAUAnDWRm mirrors the addresses specified in the appropriate half of the corresponding TAUAnDASi (i = 0 to 7) register (see *1 "TAUAnDASi - TAUAn DMA window address setting register i (i = 0 to 7)"*).

Access This register can be read/written in 16-bit units.

Address <TAUAn_base> + 0100_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA window address as specified in the appropriate half of TAUAnDASm															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15.29.8 TAUAn emulation register

(1) TAUAnEMU - TAUA emulation register

This register controls whether the TAUAn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <TAUAn_base> + 290_H

Initial Value 00_H

7	6	5	4	3	2	1	0
TAUAn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 15-294 TAUAnEMU register contents

Bit position	Bit name	Function
7	TAUAn SVSDIS	Emulation control 0: TAUAn can be stopped during emulation 1: TAUAn continuous operating during emulation

Chapter 16 Timer Array Unit B (TAUB)

This chapter contains a generic description of the Timer Array Unit B (TAUB).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

16.1 V850E2/Fx4-H TAUB Features

Instances

This microcontroller has following number of instances of the Timer Array Unit B.

Table 16-1 Instances of TAUB

Timer Array Unit B	
Instance	2
Name	TAUB1, TAUB2

Instances index n

Throughout this chapter, the individual instances of a Timer Array Unit B is identified by the index "n" (n = 0, 1), for example, TAUBnTOM for the TAUBn channel output mode register.

Channel index m

The Timer Array Unit B has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7 , 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All TAUBn register addresses are given as address offsets to the individual base address <TAUBn_base>.
The base address <TAUBn_base> of each TAUBn is listed in the following table:

Table 16-2 Register base addresses <TAUBn_base>

TAUBn instance	<TAUBn_base> address
TAUB1	FF80 9000_H
TAUB2	FF80 $A000_H$

Clock supply All Timer Array Units B provide one clock input:

Table 16-3 TAUBn clock supply

TAUBn instance	TAUBn clock	Connected to
TAUB1	PCLK	Clock Controller CKSCLK_006
TAUB2	PCLK	Clock Controller CKSCLK_111

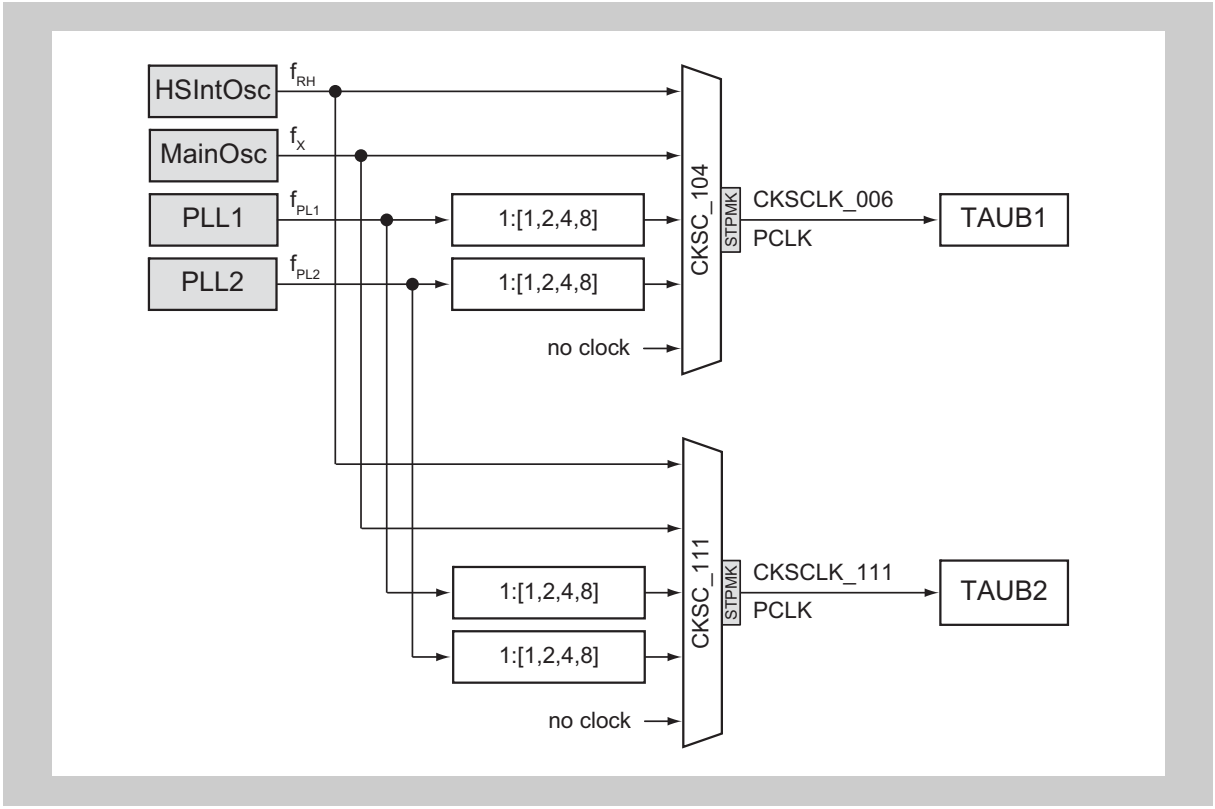


Figure 16-1 TAUB clock supply

Interrupts and DMA/DTS

The Timer Array Unit B can generate the following interrupt and DMA/DTS requests:

Table 16-4 TAUBn interrupt and DMA/DTS requests (1/2)

TAUBn signals	Function	Connected to
TAUB1:		
INTTAUB1I0[a]	Channel 0 interrupt	Interrupt Controller INTTAUB1I0 DMA Controller trigger 20 DTS Controller trigger 3
INTTAUB1I1[a]	Channel 1 interrupt	Interrupt Controller INTTAUB1I1 DMA Controller trigger 21 DTS Controller trigger 4
INTTAUB1I2[a] to INTTAUB1I3[a]	Channel 2 to 3 interrupt	Interrupt Controller INTTAUB1I2 to INTTAUB1I3
INTTAUB1I4[a]	Channel 4 interrupt	Interrupt Controller INTTAUB1I4 DMA Controller trigger 22
INTTAUB1I5[a]	Channel 5 interrupt	Interrupt Controller INTTAUB1I5 DMA Controller trigger 23
INTTAUB1I6[a] to INTTAUB1I7[a]	Channel 6 to 7 interrupt	Interrupt Controller INTTAUB1I6 to INTTAUB1I7
INTTAUB1I8[a]	Channel 8 interrupt	Interrupt Controller INTTAUB1I8 DMA Controller trigger 24 DTS Controller trigger 46
INTTAUB1I9[a]	Channel 9 interrupt	Interrupt Controller INTTAUB1I9 DMA Controller trigger 25 DTS Controller trigger 47
INTTAUB1I10[a] to INTTAUB1I11[a]	Channel 10 to 11 interrupt	Interrupt Controller INTTAUB1I10 to INTTAUB1I11
INTTAUB1I12[a]	Channel 12 interrupt	Interrupt Controller INTTAUB1I12 DMA Controller trigger 26 DTS Controller trigger 48
INTTAUB1I13[a]	Channel 13 interrupt	Interrupt Controller INTTAUB1I13 DMA Controller trigger 27 DTS Controller trigger 49
INTTAUB1I14[a] to INTTAUB1I15[a]	Channel 14 to 15 interrupt	Interrupt Controller INTTAUB1I14 to INTTAUB1I15
TAUB2:		
INTTAUB2I0	Channel 0 interrupt	Interrupt Controller INTTAUB2I0 DMA Controller trigger 28 DTS Controller trigger 5
INTTAUB2I1	Channel 1 interrupt	Interrupt Controller INTTAUB2I1 DMA Controller trigger 29 DTS Controller trigger 6
INTTAUB2I2 to INTTAUB2I3	Channel 2 to 3 interrupt	Interrupt Controller INTTAUB2I2 to INTTAUB2I3
INTTAUB2I4	Channel 4 interrupt	Interrupt Controller INTTAUB2I4 DMA Controller trigger 30 DTS Controller trigger 50
INTTAUB2I5	Channel 5 interrupt	Interrupt Controller INTTAUB2I5 DMA Controller trigger 31 DTS Controller trigger 51
INTTAUB2I6 to INTTAUB2I7	Channel 6 to 7 interrupt	Interrupt Controller INTTAUB2I6 to INTTAUB2I7

Table 16-4 TAUBn interrupt and DMA/DTS requests (2/2)

TAUBn signals	Function	Connected to
INTTAUB2I8	Channel 8 interrupt	Interrupt Controller INTTAUB2I8 DMA Controller trigger 32 DTS Controller trigger 52
INTTAUB2I9	Channel 9 interrupt	Interrupt Controller INTTAUB2I9 DMA Controller trigger 33 DTS Controller trigger 53
INTTAUB2I10 to INTTAUB2I11	Channel 10 to 11 interrupt	Interrupt Controller INTTAUB2I10 to INTTAUB2I11
INTTAUB2I12	Channel 12 interrupt	Interrupt Controller INTTAUB2I12 DMA Controller trigger 34 DTS Controller trigger 54
INTTAUB2I13	Channel 13 interrupt	Interrupt Controller INTTAUB2I13 DMA Controller trigger 35 DTS Controller trigger 55
INTTAUB2I14 to INTTAUB2I15[a]	Channel 14 to 15 interrupt	Interrupt Controller INTTAUB2I14 to INTTAUB2I15

a) These interrupts can be used to as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

TAUB H/W reset

The Time Array Units B and their registers are initialized by the following reset signal:

Table 16-5 TAUBn reset signal

TAUBn	Reset signal
TAUB1	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
TAUB2	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the Timer Array Unit B are used for various purposes, as listed in the following table.

Note √ / – : used / not used for the function

Table 16-6 TAUB1 I/O signals

TAUB1 signal	Connected to		
	Miscellaneous[a]	Port	PWM delay[b]
TAUB1 input channels: All TAUB1 port input signals are passed through a noise filter, refer to *2.5 "Port Filters" on page 143* for further details.			
TAUB1TTIN0	URTE0RX	–	–
TAUB1TTIN1	URTE1RX	TAUB1I1	–
TAUB1TTIN2	URTE2RX	TAUB1I2	–
TAUB1TTIN3	URTE3RX	TAUB1I3	–
TAUB1TTIN4	URTE4RX	–	–
TAUB1TTIN5	URTE5RX	TAUB1I5	–
TAUB1TTIN6	URTE6RX	–	–
TAUB1TTIN7	URTE7RX	TAUB1I7	–
TAUB1TTIN8	–	–	–
TAUB1TTIN9	–	TAUB1I9	–
TAUB1TTIN10	–	–	–
TAUB1TTIN11	–	TAUB1I11	–
TAUB1TTIN12	–	–	–
TAUB1TTIN13	–	TAUB1I13	–
TAUB1TTIN14	–	TAUB1I14	–
TAUB1TTIN15	–	TAUB1I15	–
TAUB1 output channels:			
TAUB1TTOUT0	–	–	–
TAUB1TTOUT1	–	TAUB1O1	√
TAUB1TTOUT2	–	TAUB1O2	√
TAUB1TTOUT3	–	TAUB1O3	–
TAUB1TTOUT4	–	–	–
TAUB1TTOUT5	–	TAUB1O5	√
TAUB1TTOUT6	–	TAUB1O6	√
TAUB1TTOUT7	–	TAUB1O7	–
TAUB1TTOUT8	–	–	–
TAUB1TTOUT9	–	TAUB1O9	√
TAUB1TTOUT10	–	TAUB1O10	√
TAUB1TTOUT11	–	TAUB1O11	–
TAUB1TTOUT12	–	–	–
TAUB1TTOUT13	–	TAUB1O13	√
TAUB1TTOUT14	–	TAUB1O14	√
TAUB1TTOUT15	–	TAUB1O13	–

a) Refer to section *15.30 "TAUB Input Selection" on page 1025* for further details.

b) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Table 16-7 TAUB2 I/O signals

TAUB2 signal	Connected to		
	Miscellaneous[a]	Port	PWM delay[b]
TAUB2 input channels: All TAUB2 port input signals are passed through a noise filter, refer to *2.5 “Port Filters” on page 143* for further details.			
TAUB2TTIN0	–	–	–
TAUB2TTIN1	–	TAUB2I1	–
TAUB2TTIN2	–	TAUB2I2	–
TAUB2TTIN3	–	TAUB2I3	–
TAUB2TTIN4	–	–	–
TAUB2TTIN5	–	TAUB2I5	–
TAUB2TTIN6	–	TAUB2I6	–
TAUB2TTIN7	–	TAUB2I7	–
TAUB2TTIN8	–	–	–
TAUB2TTIN9	–	TAUB2I9	–
TAUB2TTIN10	–	–TAUB2I10	–
TAUB2TTIN11	–	TAUB2I11	–
TAUB2TTIN12	–	–	–
TAUB2TTIN13	–	TAUB2I13	–
TAUB2TTIN14	–	TAUB2I14	–
TAUB2TTIN15	–	TAUB2I15	–
TAUB2 output channels:			
TAUB2TTOUT0	–	–	–
TAUB2TTOUT1	–	TAUB2O1	√
TAUB2TTOUT2	–	TAUB2O2	√
TAUB2TTOUT3	–	TAUB2O3	–
TAUB2TTOUT4	–	–	–
TAUB2TTOUT5	–	TAUB2O5	√
TAUB2TTOUT6	–	TAUB2O6	√
TAUB2TTOUT7	–	TAUB2O7	–
TAUB2TTOUT8	–	–	–
TAUB2TTOUT9	–	TAUB2O9	√
TAUB2TTOUT10	–	TAUB2O10	√
TAUB2TTOUT11	–	TAUB2O11	–
TAUB2TTOUT12	–	–	–
TAUB2TTOUT13	–	TAUB2O13	√
TAUB2TTOUT14	–	TAUB2O14	√
TAUB2TTOUT15	–	TAUB2O13	–

a) Refer to section *15.30 “TAUB Input Selection” on page 1025* for further details.
b) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Caution The initial state of the input signals noise filters block the input signals. Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

16.2 TAUB Input Selection

16.2.1 TAUB1TTIN[7:0] input selections

TheTAUB1TTIN[7:0] input signals have several options to be connect:

- URTE0 to URTE7 data receive signals URTEnRX for baud rate measurement

The following figure depicts the TAUB1 selection scheme for the TAUB1TTIN[7:0] inputs:

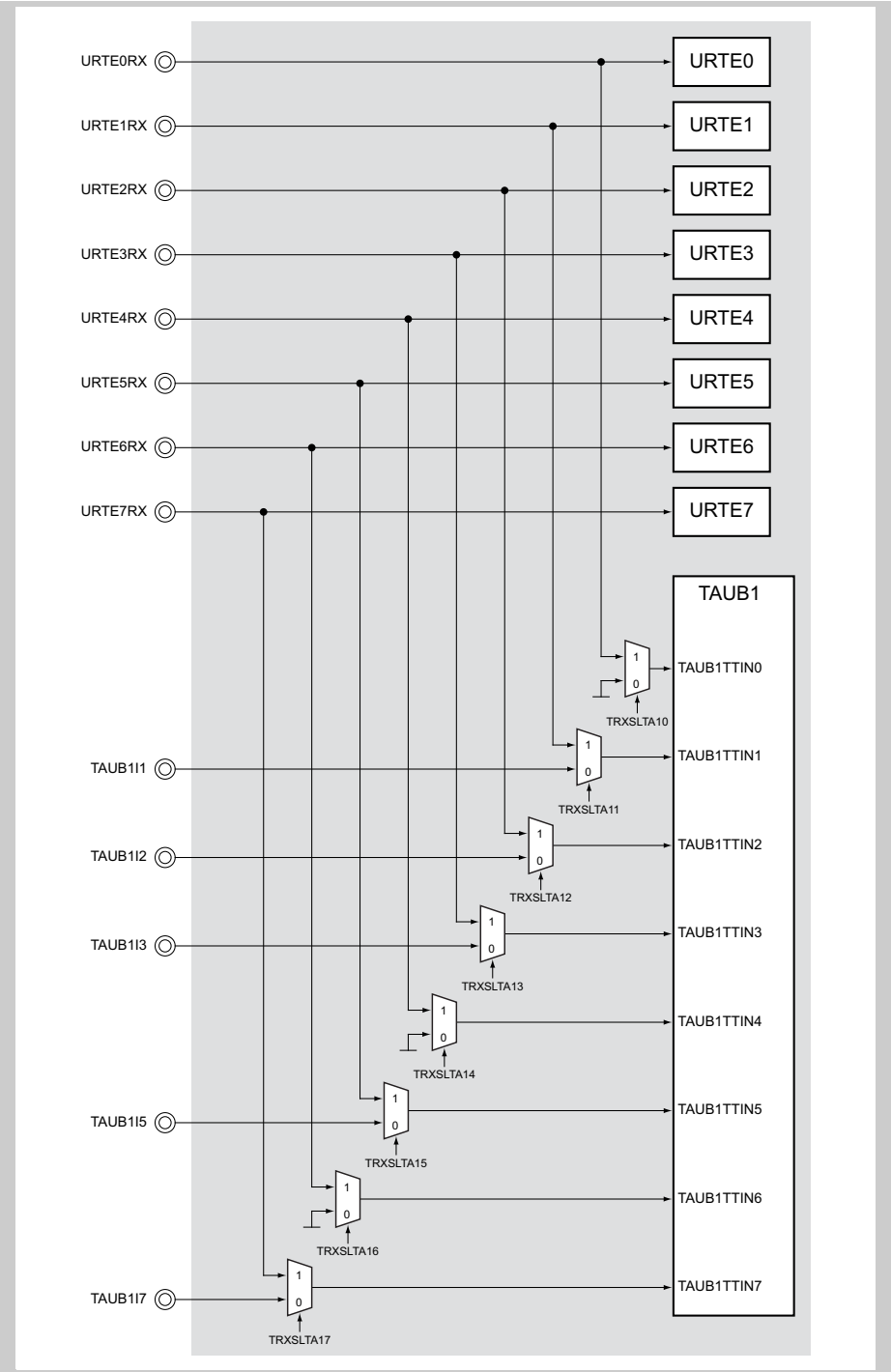


Figure 16-2 TAUB1TTIN[7:0] input selector scheme

(1) TRXSLTA1 - TAUB1 receive input selection register

This register selects the signals to several TAUB1 inputs.

Access This register can be read/written in 8-bit units.

Address FF77 2018$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
TRXSLTA 17	TRXSLTA 16	TRXSLTA 15	TRXSLTA 14	TRXSLTA 13	TRXSLTA 12	TRXSLTA 11	TRXSLTA 10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-8 TRXSLTA1 register contents

Bit position	Bit name	Function
7	TRXSLTA 17	Selection of TAUB1TTIN7 input: 0: TAUB1I7 1: URTE7RX
6	TRXSLTA 16	Selection of TAUB1TTIN6 input: 0: low level 1: URTE6RX
5	TRXSLTA 15	Selection of TAUB1TTIN5 input: 0: TAUB1I5 1: URTE5RX
4	TRXSLTA 14	Selection of TAUB1TTIN4input: 0: low level 1: URTE4RX
3	TRXSLTA 13	Selection of TAUB1TTIN3 input: 0: TAUB1I3 1: URTE3RX
2	TRXSLTA 12	Selection of TAUB1TTIN2 input: 0: TAUB1I2 1: URTE2RX
1	TRXSLTA 11	Selection of TAUB1TTIN1 input: 0: TAUB1I1 1: URTE1RX
0	TRXSLTA 10	Selection of TAUB1TTIN0 input: 0: low level 1: URTE0RX

16.3 Functional Overview

Features summary

The TAUB has the following functions:

- 16 channels
- 16-bit counter and 16-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUB:

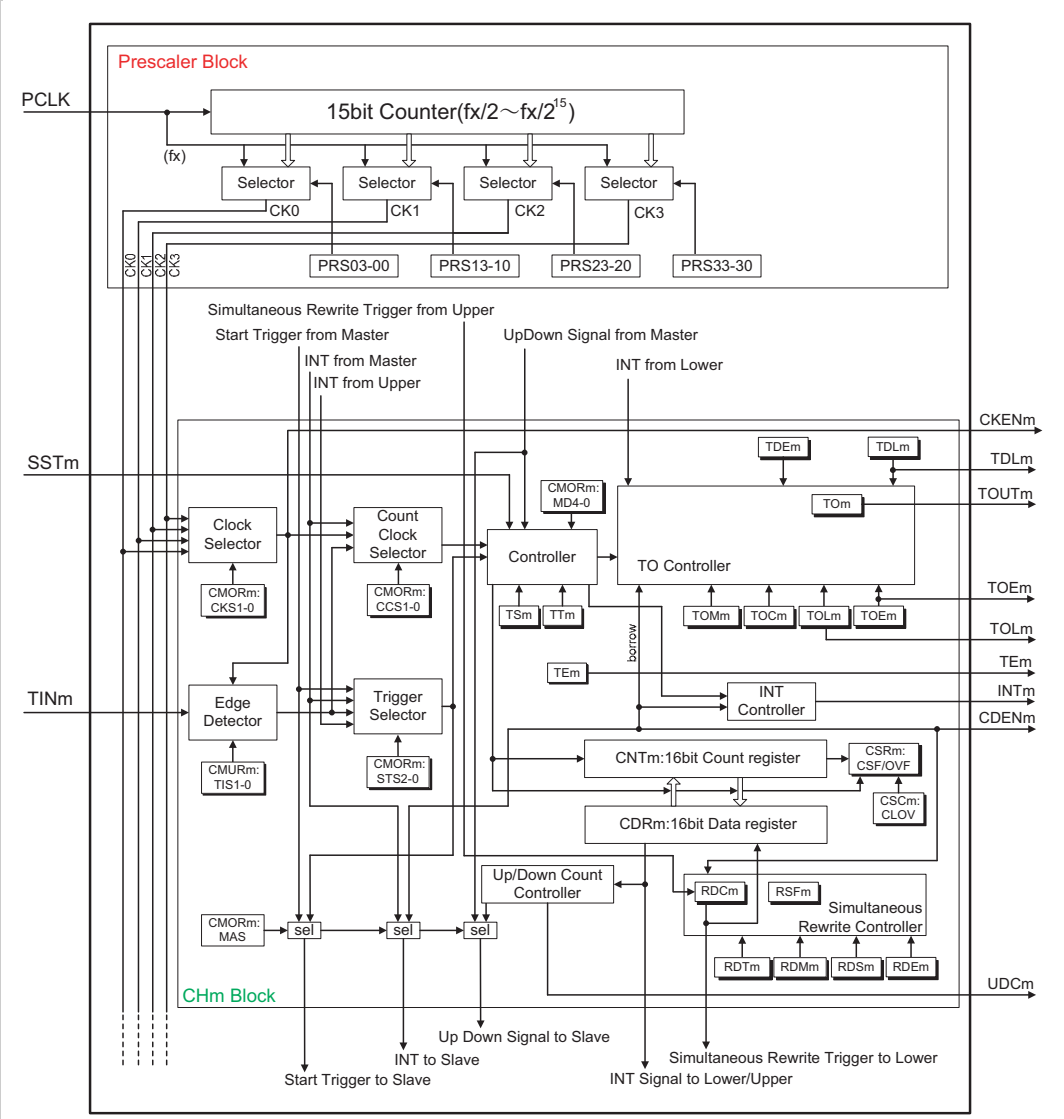


Figure 16-3 Block diagram of the TAUB

The prefix “TAUBn” has been omitted from the register names for the sake of clarity in the above figure.

16.3.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.

  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.

  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.

  Examples are “Capture Mode”, “Event Count Mode”, and “Interval Timer Mode”.

- **Channel output mode**

  The channel output mode defines the operation of TAUBnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Examples are “Independent Channel Output Mode” and “Synchronous Channel Output Mode with Dead Time”.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

  It defines the operation mode, start and capture trigger, count clock, the real-time trigger generation, and the channel output mode.

  Examples are “Divider Function” and “Triangle PWM Output Function”

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 5, channel 3 is an upper channel and channel 9 is a lower channel.

16.4 Functional Description

The Timer Array Unit B is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler block for count clock generation and 16 channels, each equipped with a 16 bit counter TAUBnCNTm and a 16-bit data register TAUBnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in different operation modes, either independently or in combination with other channels (synchronously), i.e. multiple channels depend on each other with one master and one or more slave channels.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler block

The prescaler block provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

Count clocks CK0 to CK3 are derived from PCLK by a configurable prescaler division factor of 2^0 to 2^{15}.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUBnIm from master channel
- Edge detected TAUBnTTINm input signal

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUBnCMORm.MD[4:0])
- Counter start enable (TAUBnTS.TSm) and counter stop (TAUBnTT.TTm)

  When counter start is enabled, status flag TAUBnTE.TEm is set.
- Count direction (can be controlled by master channel)

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUBnTE.TEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUBnTSSTm
- TAUBnTTINm input valid edge
- INTTAUBnIm from the master or any upper channel
- Up/down output trigger signal TAUBnTUDSm of the master channel
- Dead-time output signal TAUBnTDL.TDLm of the TAUBnTTOUTm generation unit.

Simultaneous rewrite controller

Simultaneous rewrite control is a special function that can be used in synchronous operation modes. The data registers of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller

ensures that new data register values of all channels become effective at the same time.

TAUBnTO Controller The output control of every channel enables the generation of various output signal forms such as PWM signals or triangular waves.

Signals The TAUB has various input and output signals. A full list can be found in the first section of this chapter under the keyword “I/O signals”.

16.5 General Operating Procedure

The following lists the general operation procedure for the TAUBn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. All circuits and registers of all channels are initialized. The control register of TAUBnTTOUTm is also initialized and outputs a low level.

1. Set the TAUBnTPS register to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUBn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUBnTS.TSm bit to 1.
   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
   The function is in operation.
4. If desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUBnTT.TTm bit to 0.

Note A detailed description of the required control bits and the operation of the individual functions is given in *16.13 “Independent Channel Operation Functions” on page 1134* and *16.18 “Synchronous Channel Operation Functions” on page 1225* .

16.6 Operation Modes

The TAUB contains 12 operation modes. These determine the basic behavior of a channel, for example whether a timer counts up or down, whether the data register TAUBnCDRm acts as a compare register or stores the initial value of the counter, etc.

One operation mode can be set for each channel. It is specified using the TAUBnCMOR.MD[4:0] bits.

When choosing a function, these settings cannot be specified individually, but are grouped under the term operation mode. If a function uses multiple channels, the operation mode of each channel must be set correctly for the function to work correctly.

For more information about the operation modes required by each function, refer to the required function in *16.13 “Independent Channel Operation Functions” on page 1134* and *16.18 “Synchronous Channel Operation Functions” on page 1225* .

16.7 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *16.7.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *16.7.2 "Simultaneous start and stop of synchronous channel counters" on page 1111*
- *16.8 "Simultaneous Rewrite" on page 1112*

16.7.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2, CH4, …) can be set as master channels. Any channel apart from CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 and the lower channels (CH4, CH5, …) can be set as slave channels.
- If multiple master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH4 are master channels, CH1 to CH3 can be set as slave channels for CH0, but CH5 to CH7 cannot.

Operation clock

- The same operation clock must be set for the slave channel and the master channel. This is achieved using the TAUBnCMORm.CKS[1:0] bits of the slave and master channel.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

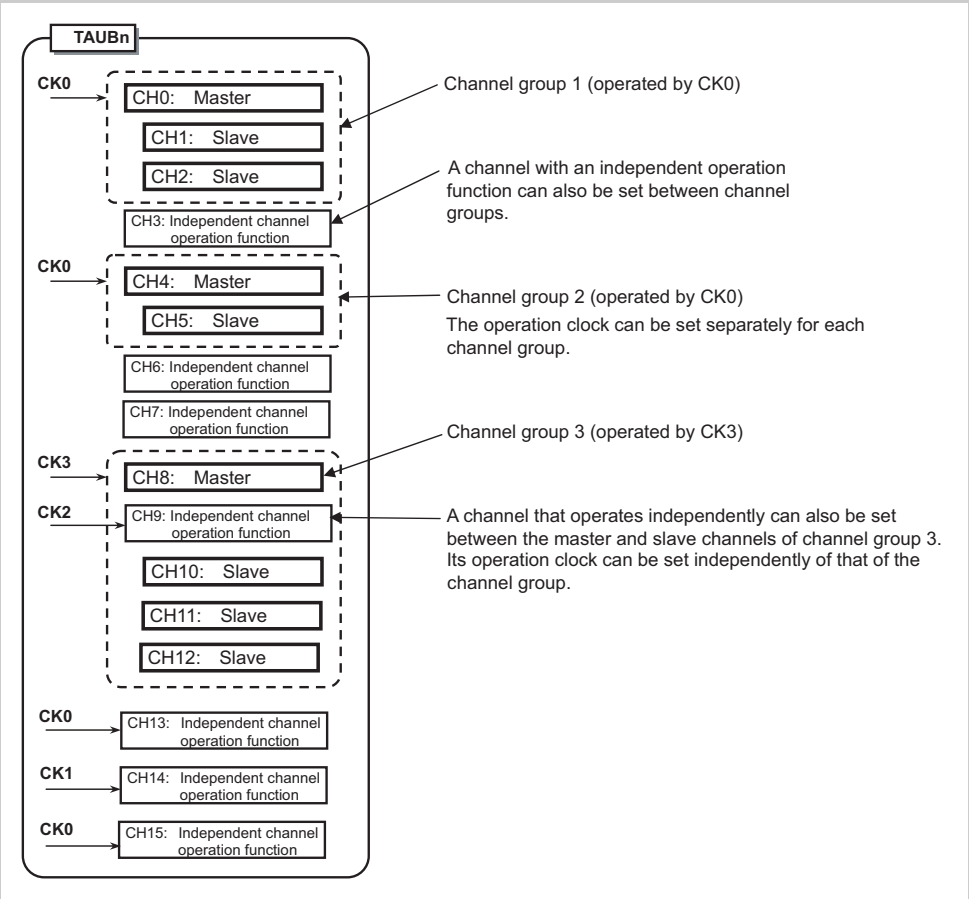


Figure 16-4 Grouping of the channels and assignment of operation clocks

INTTAUBnIm, start trigger, and count clock

- Master channels can transfer an interrupt request (INTTAUBnI), the start trigger, and the count clock to slave channels.
- Slave channels can use INTTAUBnI, the start trigger, and the count clock of the master channels but cannot transfer their INTTAUBnI, start trigger, or count clock to the lower channels.
- A master channel cannot use INTTAUBnI, the start trigger, or the count clock of the higher master channels.

16.7.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a TAUB unit, and between TAUB units.

(1) Simultaneous start and stop within a TAUB unit

- To simultaneously start synchronized channels, the TAUBnTS.TSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUBnTT.TTm bits of the channels must be set at the same time.

Writing to the TAUBnTS.TSm bits sets the corresponding TAUBnTE.TEm bits to 1, enabling counting. TAUBnTS.TSm = 1 only enables the corresponding counter to start; the exact time that it starts depends on the operation mode.

(2) Simultaneous start between TAUB units

Counters in different TAUB units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal. The simultaneous start trigger register is then sent to the TAUBnTSSTm input.

16.8 Simultaneous Rewrite

16.8.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUBnCDRm and TAUBnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by:

- The counter on the master channel or upper channel (depending on the selected operation mode) reaching a certain value
- INTTAUBnI being issued on the upper channel specified by TAUBnRDC.RDCm

There are four methods for simultaneous rewrite. These are listed in the following table, along with how to specify them and when they cause simultaneous rewrite to be triggered.

Table 16-9 Simultaneous rewrite methods and when they are triggered

Method	Simultaneous rewrite triggered when	TAUBn RDE. RDEm	TAUBn RDS. RDSm	TAUBn RDM. RDMm
-	No simultaneous rewrite	0	0	0
A	The master channel (re)starts counting	1	0	0
B	The slave channel starts counting down at the upper peak of a triangular cycle	1	0	1
C1	INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm	1	1	0
TOLm	For TOLm, the following table shows whether TOLm can be rewritten during operation. The TOLm rewrite method is the same as that of CDRn.			

The following table lists which of these four methods is available for each channel operation function. For more information about the individual channel operation functions, see the corresponding sections in *16.13 “Independent Channel Operation Functions” on page 1134* and *16.18 “Synchronous Channel Operation Functions” on page 1225* .

Table 16-10 Channel operation functions and methods they use (1/2)

Function	A	B	C1	TOLm
Simultaneous Rewrite Trigger Output Function Type 1			X	
PWM Output Function	X		X	X
One-Shot Pulse Output Function	X			
Delay Pulse Output Function	X			
Triangle PWM Output Function		X	X	X

Table 16-10 Channel operation functions and methods they use (2/2)

Function	A	B	C1	TOLm
Triangle PWM Output Function with Dead Time		X	X	
AD Conversion Trigger Output Function Type 1	X		X	
AD Conversion Trigger Output Function Type 2		X	X	

16.8.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

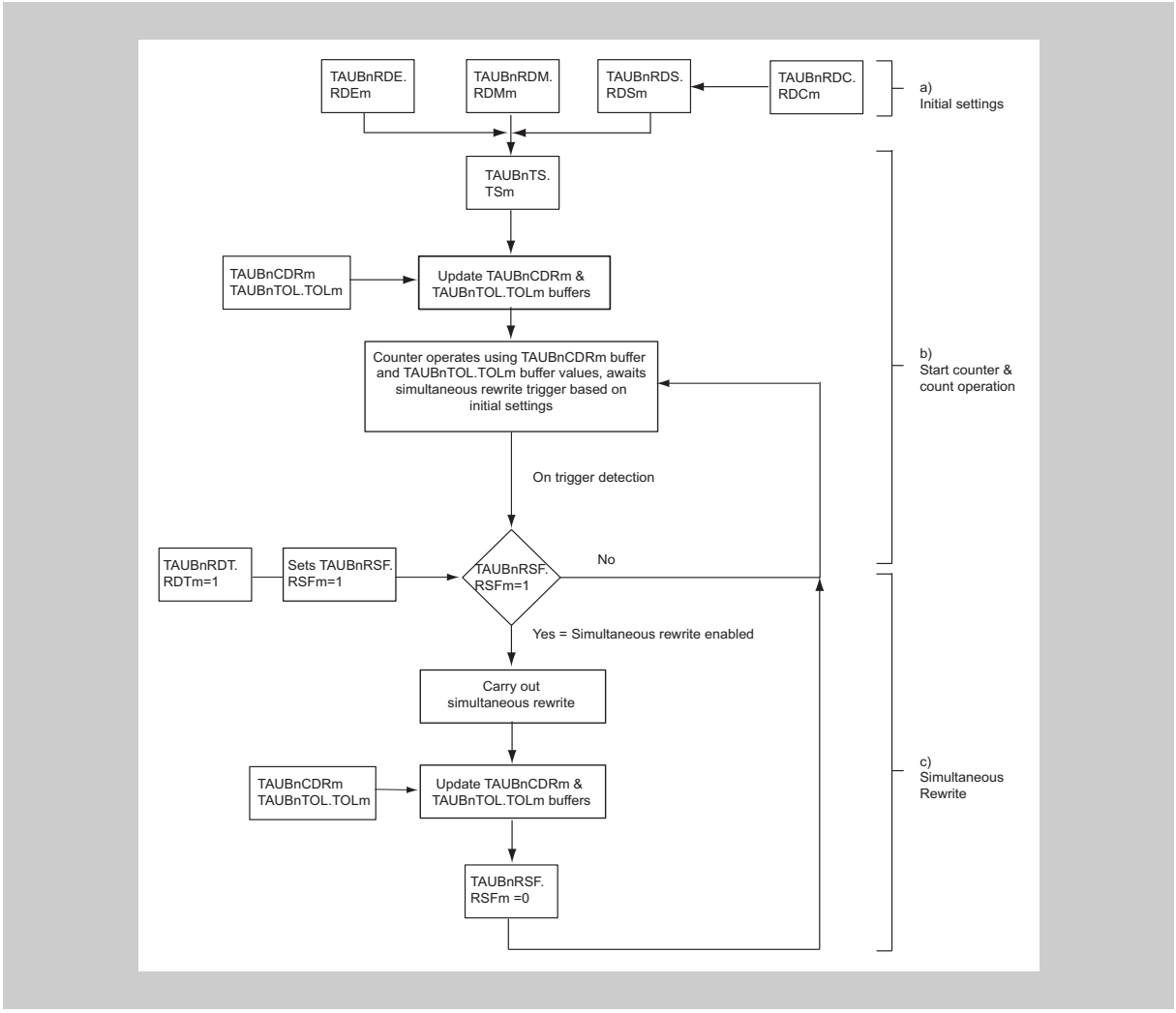


Figure 16-5 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUBnRDE.RDEm = 1
- To select the type of simultaneous rewrite, set TAUBnRDM.RDMm and TAUBnRDS.RDSm according to the values in *Table 16-9 “Simultaneous rewrite methods and when they are triggered” on page 1112*
- To select which upper channel is monitored for the simultaneous rewrite trigger use TAUBnRDC.RDCm (prerequisite: TAUBnRDS.RDSm is set to upper channel)

(2) Start counter and count operation

- To start all the TAUBnCNTm counters in the channel group, set the corresponding TAUBnTS.TSm bits to 1. TAUBnTOL.TOLm and the values in the data registers (TAUBnCDRm) are written to the corresponding TAUBnTOL.TOLm buffer (TAUBnTOL.TOLm buf) and data buffer registers (TAUBnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUBnRDT.RDTm) to 1 sets the reload flag (TAUBnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUBnRDT.RDTm then immediately returns to 0, but TAUBnRSF.RSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUBnRSF.RSFm bit is checked to see if simultaneous rewrite is enabled (TAUBnRSF.RSFm = 1). If it is, simultaneous rewrite is carried out. Otherwise the value of the TAUBnRSF.RSFm bit is re-evaluated the next time the trigger is detected.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUBnRSF.RSFm = 1), the current values of the data registers are copied to their buffers. These values are then written to the corresponding counters and the values are applied the next time the counter starts or restarts.
- The TAUBnRSF.RSFm bit is set to 0, and the system awaits the next simultaneous rewrite trigger.

16.8.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUBnRDE.RDEm, TAUBnRDS.RDSm, TAUBnRDM.RDMm, and TAUBnRDC.RDCm cannot be changed while the counter is in operation (TAUBnTE.TEm = 1).
- TAUBnTOL.TOLm can only be rewritten during operation when in PWM output function or triangle PWM output function. For all other output functions, TAUBnTOL.TOLm must be written before the counter starts. If it is rewritten in another function, TAUBnTTOUTm outputs an invalid wave.
- When an upper channel is used as the channel issuing the simultaneous rewrite trigger (TAUBnRDS.RDSm = 1), the TAUBnRDC.RDCm bit controls all the lower channels. This means that if the TAUBnRDC.RDCm bits of CH2 and CH7 are set to 1 and the TAUBnRDC.RDCm bits of other channels are set to 0, CH2 and CH7 serve as simultaneous rewrite trigger generation channels. CH2 controls the lower channels CH3 to CH6, and CH7 controls the lower channels CH8 to CH15.
- If simultaneous rewrite is enabled and an upper channel is selected for the simultaneous rewrite trigger (TAUBnRDE.RDEm and TAUBnRDS.RDSm = 1) but no upper channel is set (TAUBnRDC.RDC[15:0] = 0), simultaneous rewrite cannot take place.

16.8.4 Types of simultaneous rewrite

In the following section the four simultaneous rewrite methods are explained using timing diagrams.

(1) Simultaneous rewrite when the master channel (re)starts counting (method A)

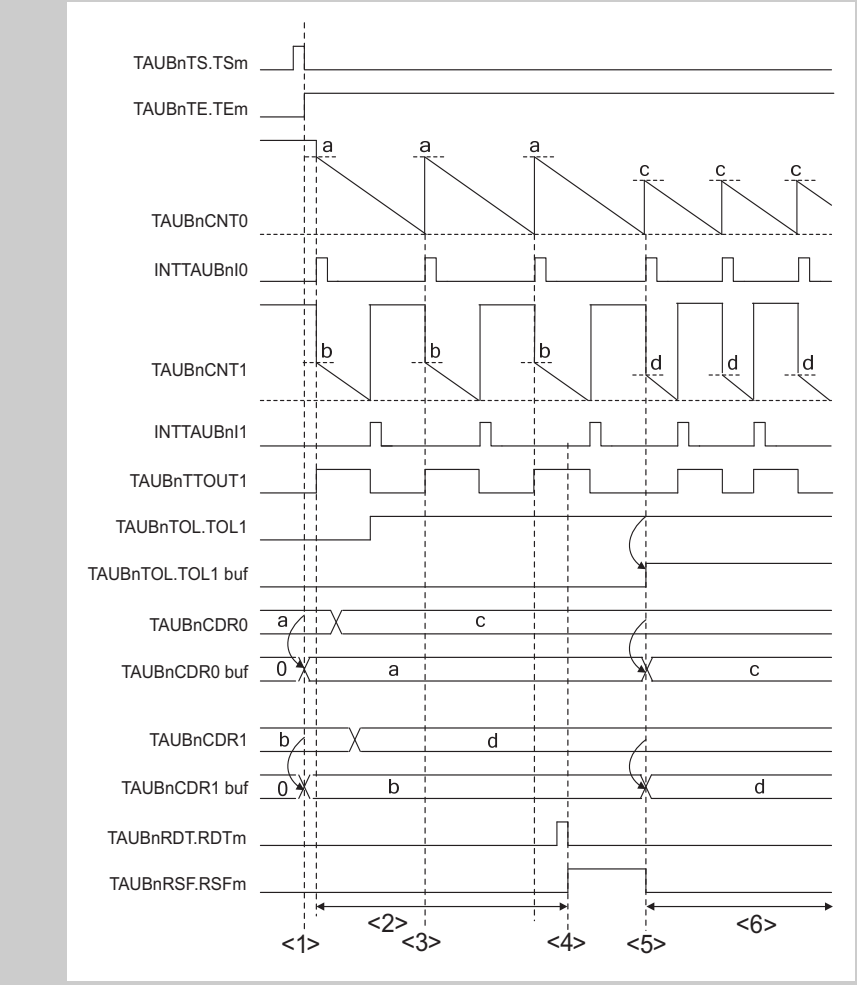


Figure 16-6 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method A is applied.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer and the value of TAUBnTOL.TOLm is copied to the TAUBnTOL.TOLm buffer. The TAUBnCDRm buffer value is written to the counter.
2. The TAUBnCDRm and TAUBnTOL.TOLm registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUBnRSF.RSFm = 0).
4. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewrite is triggered by counter TAUBnCNT0 starting to count down. The TAUBnCDRm value is written to the TAUBnCDRm buffer and the TAUBnTOL.TOLm value is written to the TAUBnTOL.TOLm buffer.
   – The counter starts to count down from the value in the TAUBnCDRm buffer and the TAUBnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUBnTOL.TOLm becomes effective.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUBnCDRm and TAUBnTOL.TOLm can be changed again.

(2) Simultaneous rewrite at the peak of a triangular cycle of the slave channel (method B)

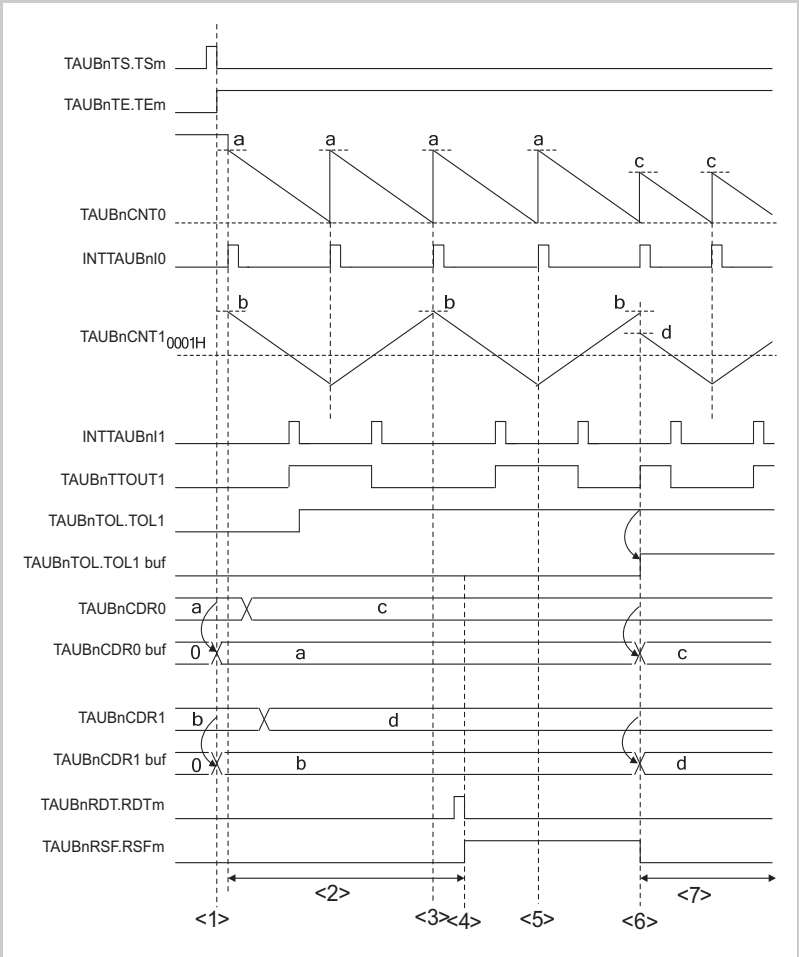


Figure 16-7 Simultaneous rewrite at the peak of a triangular cycle of the slave channel

Setup:

- CH0 is the master channel, counting up and down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method B is applied.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer. The buffer value is written to the counter.
2. The TAUBnCDRm and TAUBnTOL registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. Simultaneous rewrite does not occur because it is disabled (TAUBnRSF.RSFm = 0).
4. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by the slave channel starting to count down at an upper peak.
6. Simultaneous rewrite is triggered; the TAUBnCDRm value is written to the TAUBnCDRm buffer, the TAUBnTOL.TOLm value is written to the TAUBnTOL.TOLm buffer.
   – The counters start to count down from the value in the TAUBnCDRm buffer and the TAUBnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUBnTOL.TOLm becomes effective.
7. The counters count down and await the next simultaneous rewrite trigger. The values of TAUBnCDRm and TAUBnTOL.TOLm can be changed again.

(3) Simultaneous rewrite when INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm (method C1)

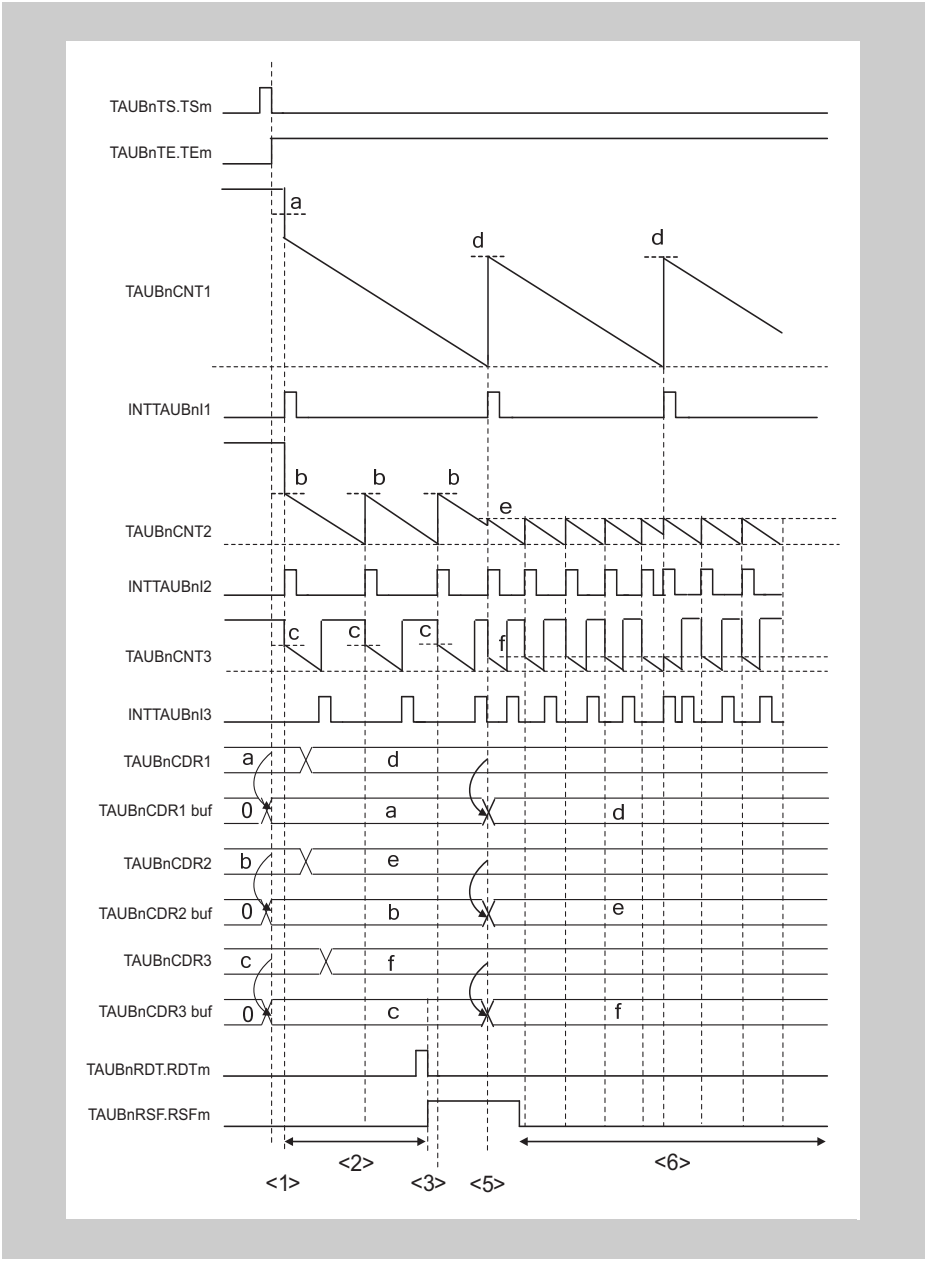


Figure 16-8 Simultaneous rewrite when INTTAUBnIm is generated on an upper channel specified by TAUBnRDC.RDCm

Setup:

- CH1 is an upper channel, counting down. CH2 is a master channel, CH3 is the slave channel, and simultaneous rewrite method C1 is applied. The TAUBnRDC register specifies which upper channel is monitored for an INTTAUBnI trigger.

Description:

1. When the counter starts, the value of TAUBnCDRm is copied to the TAUBnCDRm buffer. The buffer value is written to the counter.
2. The TAUBnCDRm register can be written at any time, but the value does not affect the counter as the counter reads the buffer value.
3. The reload data trigger bit (TAUBnRDT.RDTm) is set to 1 which sets the status flag (TAUBnRSF.RSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewrite is triggered by INT1 which is caused by counter 1 reaching 0000_H. The TAUBnCDRm values are written to the corresponding TAUBnCDRm buffers, the counters start to count down from the values in the TAUBnCDRm buffers, and the TAUBnRSF.RSFm bit is reset to 0.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUBnCDRm registers can be changed again.

16.9 Channel Output Modes

The output of the TAUBnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (Direct Channel Output Mode, TAUBnTOE.TOEm = 0)

  When controlled by software, the output register bit (TAUBnTO.TOm) can be written and the value of the bit is transferred to the output pin (TAUBnTTOUTm).

- By TAUB signals (TAUBnTOE.TOEm = 1)

  When operated by TAUB signals, the output level of TAUBnTTOUTm is set or reset or toggled by internal signals. The value of TAUBnTO.TOm is updated accordingly to reflect the value of TAUBnTTOUTm.

  – Independently (Independent Channel Output Mode, TAUBnTOM.TOMm = 0)

    When operated independently, the output of the TAUBnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUBnTOM.TOMm = 0).

  – Synchronously (Synchronous Channel Output Mode, TAUBnTOM.TOMm = 1)

    When operated synchronously, the output of the TAUBnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUBnTOM.TOMm = 1).

The TAUBnTO.TOm bit can always be read to determine the current value of TAUBnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 16-11 “Channel output modes” on page 1123* .

The channel output modes are described in detail in

- *16.9.2 “Channel output modes controlled independently by TAUBn signals” on page 1125*
- *16.9.3 “Channel output modes controlled synchronously by TAUBn signals” on page 1126* .

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUBnTOL.TOLm.

The value of the TAUBnTOL.TOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function or triangle PWM output function. Otherwise, changes to TAUBnTOL.TOLm result in an invalid TAUBnTTOUTm signal.

Refer to *16.8 “Simultaneous Rewrite” on page 1112* .

The various channel output modes and the channel output control bits are listed in the following table.

Multiple outputs

For a function on channel m that uses its output and the outputs of other channels (q):

- If channel m requires a certain operation mode for channel q, set the operation mode on channel q.
- If channel m does not require a certain operation mode for channel q, the counter on channel q must be disabled (TAUBnTE.TEq = 0), i.e. no operation mode can be used on channel q, even one that does not generate an output.

Table 16-11 Channel output modes

Channel output mode	TAUBn TOE. TOEm	TAUBn TOM. TOMm	TAUBn TOC. TOCm	TAUBn TDE. TDEm
By software				
Direct Channel Output Mode	0	x		
By timer signals, independently (Independent Channel Output Mode)				
Independent Channel Output Mode 1	1	0	0	0
Independent Channel Output Mode 2			1	
By timer signals, synchronously (Synchronous Channel Output Mode)				
Synchronous Channel Output Mode 1	1	1	0	0
Synchronous Channel Output Mode 2			1	
with Dead Time Output				1

- All combinations not listed in this table are forbidden.
- Bits marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUBnTE.TE = 1):

- TAUBnTOM.TOMm,
- TAUBnTOC.TOCm,
- TAUBnTDE.TDEm

16.9.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUBnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUBnTOE.TOEm = 0).

1. Set TAUBnTO.TOm to specify the initial level of the TAUBnTTOUTm output.
2. Set the channel output mode using *Table 16-11 “Channel output modes” on page 1123* and the output logic using the TAUBnTOL.TOLm bit.
3. Start the counter (TAUBnTS.TSm = 1).

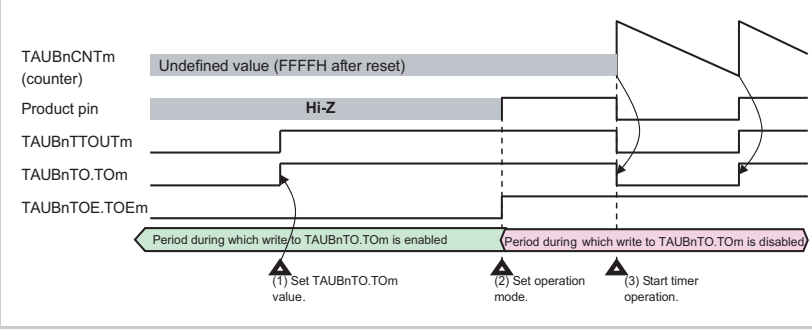


Figure 16-9 General procedure for specifying a TAUBnTTOUTm channel output mode

The following figure shows a general illustration of how the output changes when the counter is enabled:

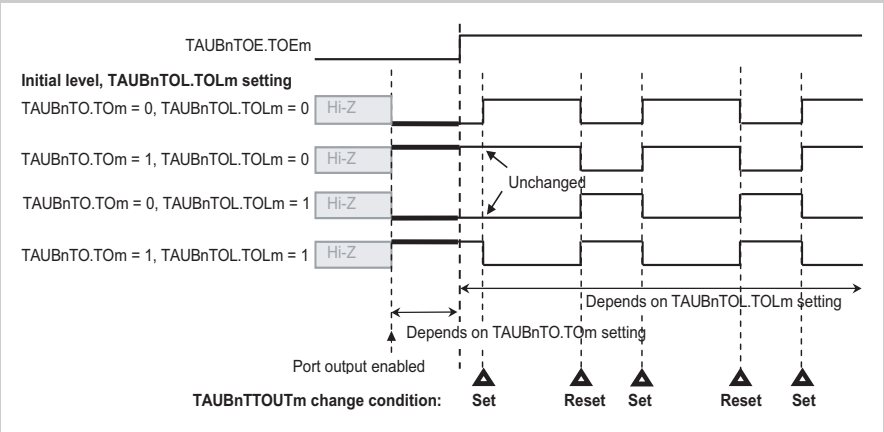


Figure 16-10 General change of the TAUBnTTOUTm output

- TAUBnTO.TOm sets the initial value of TAUBnTTOUTm and can be changed while TAUBnTOE.TOEm = 0.
- TAUBnTOL.TOLm specifies whether the set signal sets TAUBnTO.TOm to high (TAUBnTOL.TOLm = 0) or low (inverted logic, TAUBnTOL.TOLm = 1).

16.9.2 Channel output modes controlled independently by TAUBn signals

This chapter lists the channel output modes that are controlled independently by TAUBn signals. The control bits used to specify a mode are listed in *Table 16-11 “Channel output modes” on page 1123* .

(1) Independent Channel Output Mode 1

Set/reset conditions In this output mode, TAUBnTTOUTm toggles when INTTAUBnIm is detected. The value of TAUBnTOL.TOLm is ignored.

Prerequisites None, other than those in *Table 16-11 “Channel output modes” on page 1123* .

(2) Independent Channel Output Mode 2

Set/reset conditions In this output mode, TAUBnTTOUTm is set when INTTAUBnIm occurs upon count start and reset when INTTAUBnIm occurs due to a match between TAUBnCNTm and TAUBnCDRm.

Prerequisites None, other than those in *Table 16-11 “Channel output modes” on page 1123* .

16.9.3 Channel output modes controlled synchronously by TAUBn signals

This chapter lists the channel output modes that are controlled synchronously by TAUBn signals. The control bits used to specify a mode are listed in *Table 16-11 “Channel output modes” on page 1123* .

(1) Synchronous Channel Output Mode 1

Set/reset conditions

In this output mode, INTTAUBnIm of the master channel serves as the set signal and INTTAUBnIm of the slave channel as the reset signal. If INTTAUBnIm of the master channel and INTTAUBnIm of the slave channel are generated at the same time, INTTAUBnIm of the slave channel (reset signal) has priority over INTTAUBnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites

None, other than those in *Table 16-11 “Channel output modes” on page 1123* .

(2) Synchronous Channel Output Mode 2

In this output mode, the operation mode must be set to Up Down Count mode. The result is a triangle PWM wave at TAUBnTTOUTm. For details refer to *16.21.1 “Triangle PWM Output Function” on page 1269* .

Set/reset conditions TAUBnCNTm of the slave channel counts down and up alternatively. When it passes 0001_H it generates an interrupt, causing TAUBnTTOUTm to toggle.

Prerequisites A set of two channels is required to generate the triangle PWM output. TAUBnTTOUTm must be set to 0 before the function starts.

(3) Synchronous Channel Output Mode 2 with Dead Time Output

In this output mode, a dead time delay is added to TAUBnTTOUTm. The set/reset conditions are shown in the following figure.

Set/reset conditions

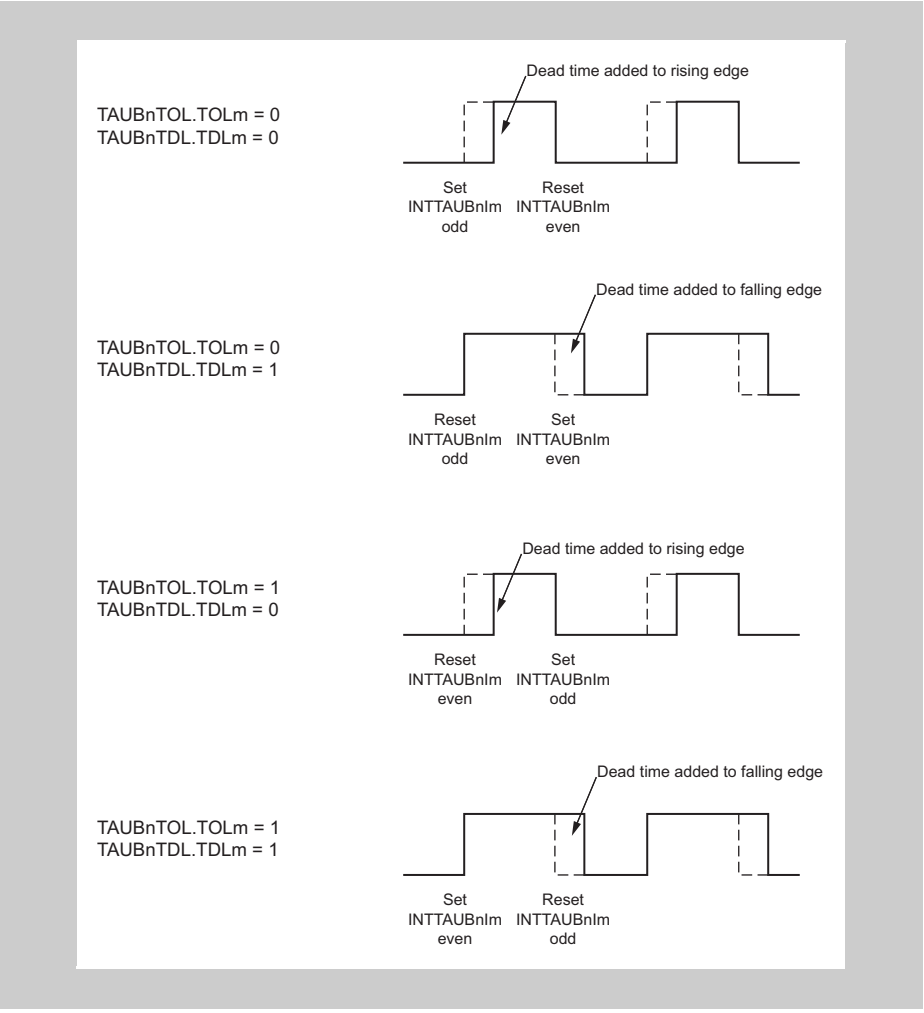


Figure 16-11 Set/reset conditions for Synchronous Channel Output Mode 2 with Dead Time Output

The edge to which the dead time is added is specified using the TAUBnTDL.TDLm bit; for rising edge set TAUBnTDL.TDLm = 0 and for falling edge set TAUBnTDL.TDLm = 1.

Prerequisites Dead time control requires a set of three channels, each operating in the following modes:

- One master channel

  The master channel must be set to Interval Timer Mode

- One even slave channel

  The even slave channel must be set to Up Down Count Mode

- One odd slave channel (even channel + 1)

  The odd slave channel must be set to One Count Mode

The values of the following bits must be the same for the odd channel and the even channel:

- TAUBnTOE.TOEm,
- TAUBnTOM.TOMm,
- TAUBnTOC.TOCm,
- TAUBnTDE.TDEm

16.10 Start Timing of Operating Modes

This chapter describes when the counters of the different operating modes start after the TAUBnTS.TSm bit is set to 1.

In all modes, the value of the data register and whether or not an interrupt is issued depends on the individual mode and corresponding register settings.

16.10.1 Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode

The counter starts at the start of the next count clock cycle after TAUBnTS.TSm is set to 1. The value of data register is also loaded at the point the counter starts.

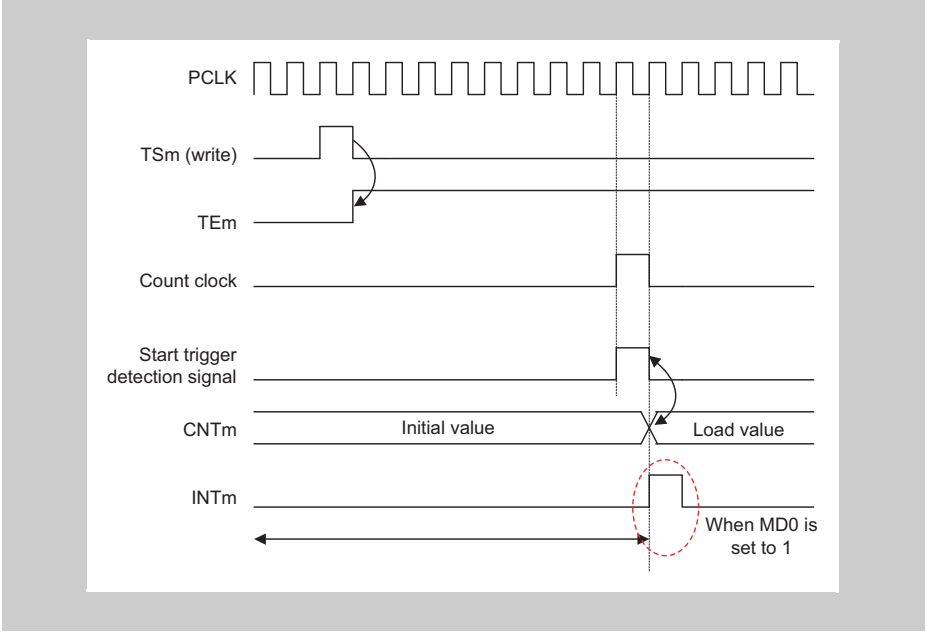


Figure 16-12 Start timing of Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode

Note In Up Down Count Mode, MD0 must be set to 0.

16.10.2 Event Mode

The value of the data register is loaded as soon as TAUBnTS.TSm is set to 1. The counter also starts immediately. The value of the data register changes with the subsequent count clock cycles.

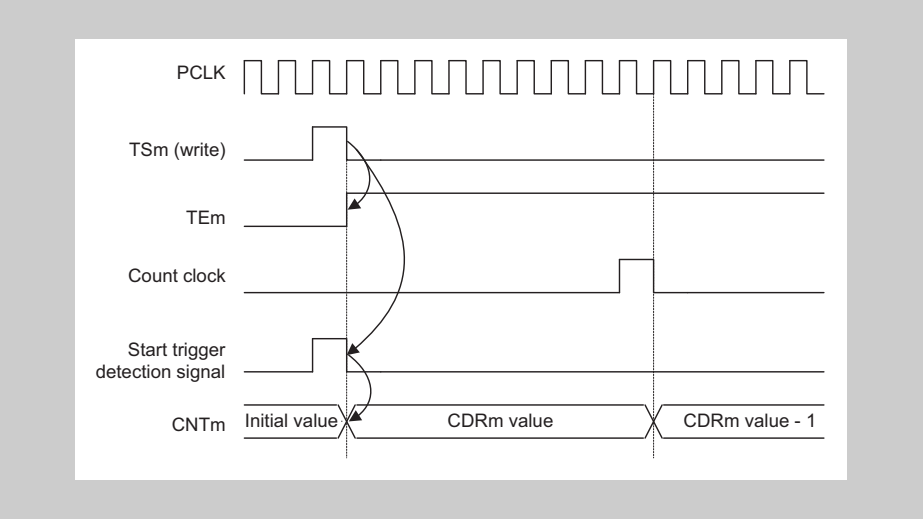


Figure 16-13 Start timing of Event Mode

16.10.3 All other operating modes

In all other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter is only triggered by detection of a valid TAUBnTTINm edge. The value of data register is also loaded at the point the counter starts. Nevertheless, the count clock cycles determine the frequency with which all operations take place.

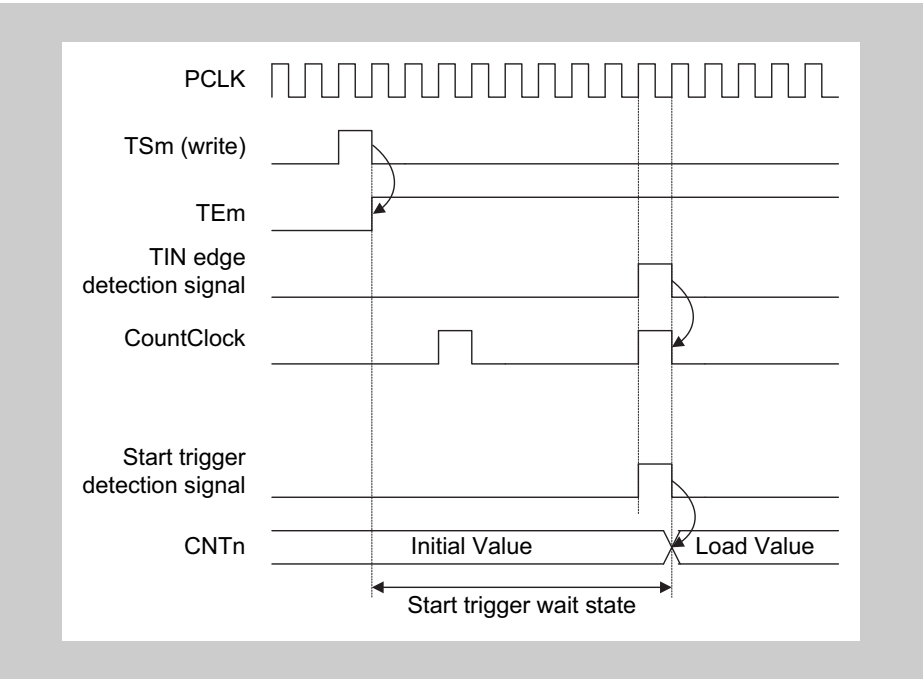


Figure 16-14 Start timing of all other operating modes

16.11 TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)

It is possible to specify whether an INTTAUBnIm is generated when the counter starts, gets restarted or is triggered by an external signal, using the TAUBnCMOR.MD0 bit. The effect of the bit depends on the selected mode, as shown in the following table. The effects of INTTAUBnIm on TAUBnTTOUTm depend on the selected channel operation function.

Table 16-12 Effect of CMOR.MD0 bit on generation of INTTAUBnIm when counter is triggered

Mode	TAUBnCMOR.MD0 bit	INTTAUBnIm generated when counter is (re)started or triggered by TINm input signal
Interval Timer Mode Capture Mode Count Capture Mode	0	No
	1	Yes
Capture & One Count Mode Capture & Gate Count Mode Event Count Mode Up Down Count Mode	0	No
One Count Mode Gate Count Mode	0/1	No, regardless of setting of TAUBnCMOR.MD0 bit.
Pulse One Count Mode		Yes, regardless of setting of TAUBnCMOR.MD0 bit.

Note As an example see figure *16-23 "Forced restart operation, TAUBnCMORm.MD0 = 1"* and figure *16-24 "Forced restart operation, TAUBnCMORm.MD0 = 0"*.
Refer to the description of *"Role of the MD0 bit"* also.

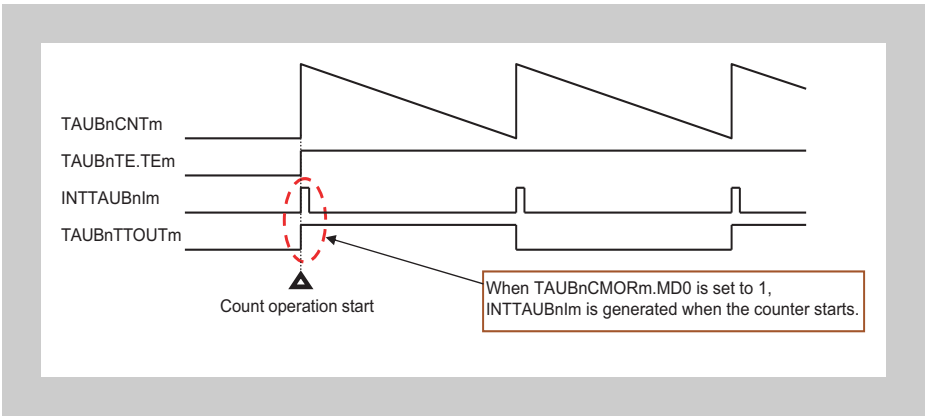


Figure 16-15 INTTAUBnIm generated when counter starts

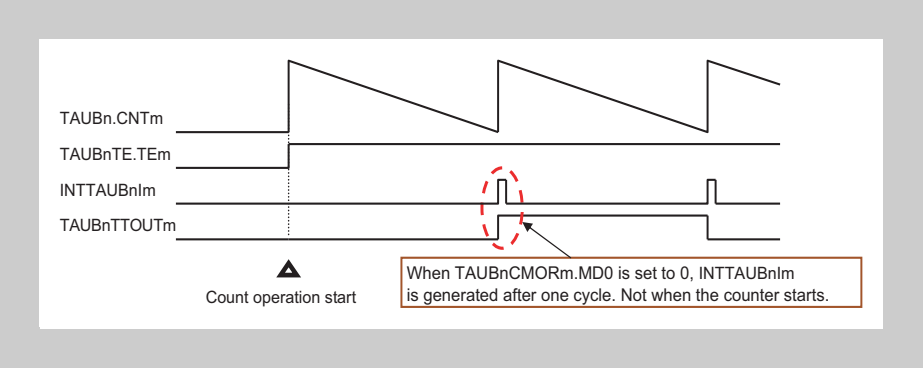


Figure 16-16 INTTAUBnIm not generated when counter starts

16.12 TAUBnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows when edge detection takes place.

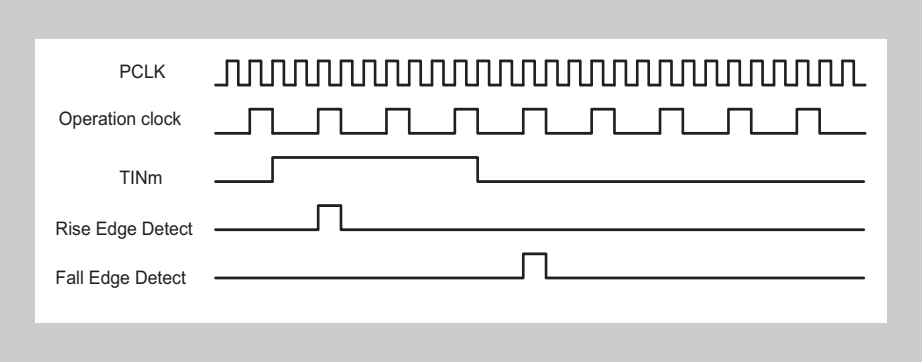


Figure 16-17 Basic edge detection timing

16.13 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by the Timer Array Unit B. For a general overview of independent channel operation, see *16.4 “Functional Description” on page 1105* .

16.14 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals or with a specified delay.

- *16.14.1 “Interval Timer Function”*
- *16.14.2 “TAUBnTTINm Input Interval Timer Function”*
- *16.14.3 “One-Pulse Output Function”*

16.14.1 Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUBnIm) at regular intervals. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 16-13 "TAUBnCMORm settings for Interval Timer Function" on page 1137*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *16.9 "Channel Output Modes" on page 1122*

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When the counter reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm stop but retain their values. The counter can be reset by setting TAUBnTS.TSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUBnTTOUTm does not toggle. This results in an inverted TAUBnTTOUTm signal compared to when TAUBnCMORm.MD0 is set to 1. For details refer to *16.11 "TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)" on page 1131* .

(2) Equations

INTTAUBnIm cycle = count clock cycle x (TAUBnCDRm + 1)

TAUBnTTOUTm square wave cycle = count clock cycle x (TAUBnCDRm + 1) x 2

(3) Block diagram and general timing diagram

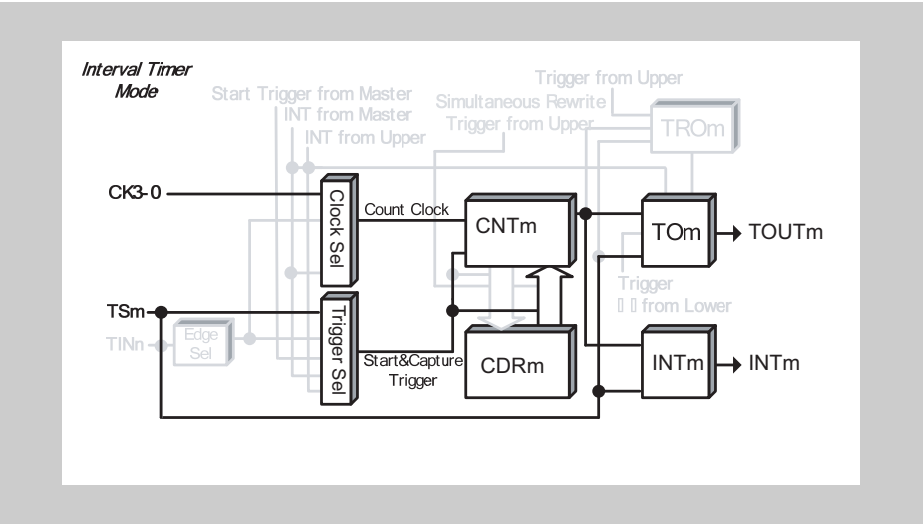


Figure 16-18 Block diagram for Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)

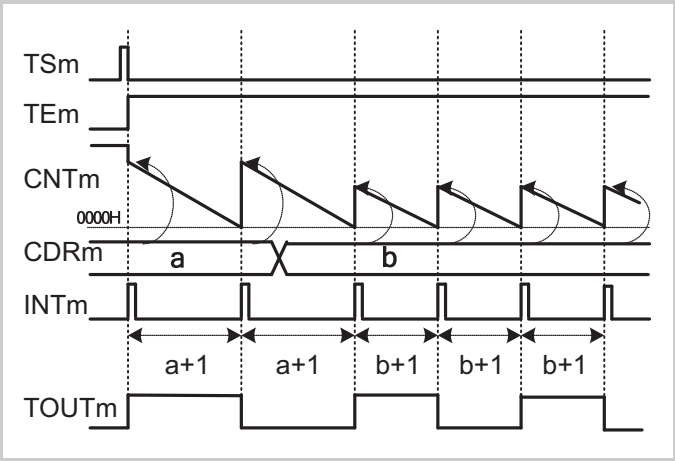


Figure 16-19 General timing diagram for Interval Timer Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-13 TAUBnCMORm settings for Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start or restart

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-14 TAUBnCMURm settings for Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 16-15 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details refer to *16-11 “Channel output modes” on page 1123* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the Interval Timer Function. Therefore, these registers must be set to 0.

Table 16-16 Simultaneous rewrite settings for Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for Interval Timer Function

Table 16-17 Operating procedure for Interval Timer Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-13 "TAUBnCMORm settings for Interval Timer Function" on page 1137* and *Table 16-14 "TAUBnCMURm settings for Interval Timer Function" on page 1137* Set the value of the TAUBnCDRm register Set the channel output mode by setting the control bits as described in *Table 16-15 "Control bit settings for Independent Channel Output Mode 1" on page 1138*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The TAUBnCDRm register value can be changed at any time. The TAUBnCNTm register can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated and TAUBnTTOUTm toggles.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H, count clock = PCLK/2

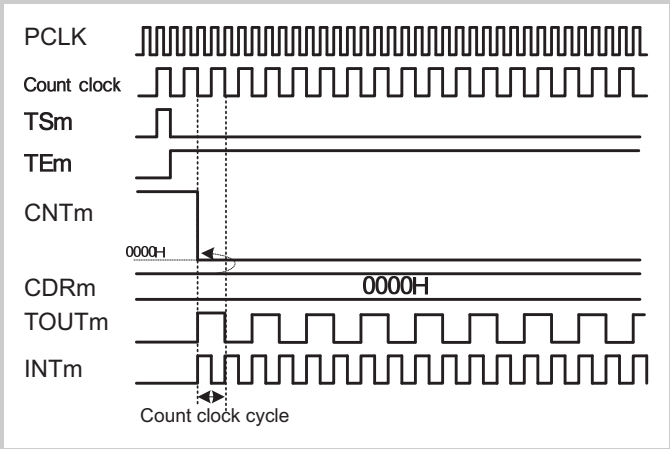


Figure 16-20 TAUBnCDRm = 0000_H, count clock = PCLK/2

- If TAUBnCDRm = 0000_H and the count clock = PCLK/2^1, the TAUBnCDRm value is written to TAUBnCNTm every count clock, meaning that TAUBnCNTm is always 0000_H.
- INTTAUBnIm is generated every count clock, resulting in TAUBnTTOUTm toggling every count clock.

(b) TAUBnCDRm = 0000_H, count clock = PCLK

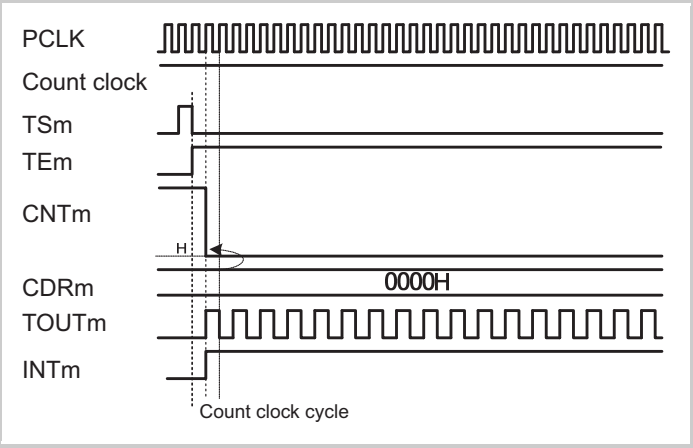


Figure 16-21 TAUBnCDRm = 0000_H, count clock = PCLK

- If TAUBnCDRm = 0000_H and the count clock = PCLK, the TAUBnCDRm value is written to TAUBnCNTm every PCLK clock, meaning that TAUBnCNTm is always 0000_H.
- INTTAUBnIm is generated continuously, resulting in TAUBnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

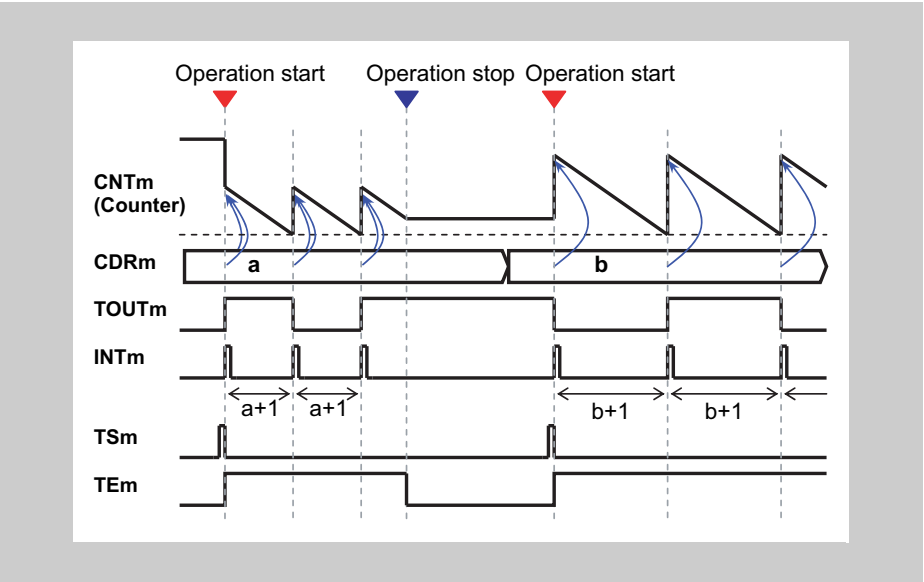


Figure 16-22 Operation stop and restart, TAUBnCMORm.MD0 = 1

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm and TAUBnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUBnTS.TSm to 1.

(d) Forced restart

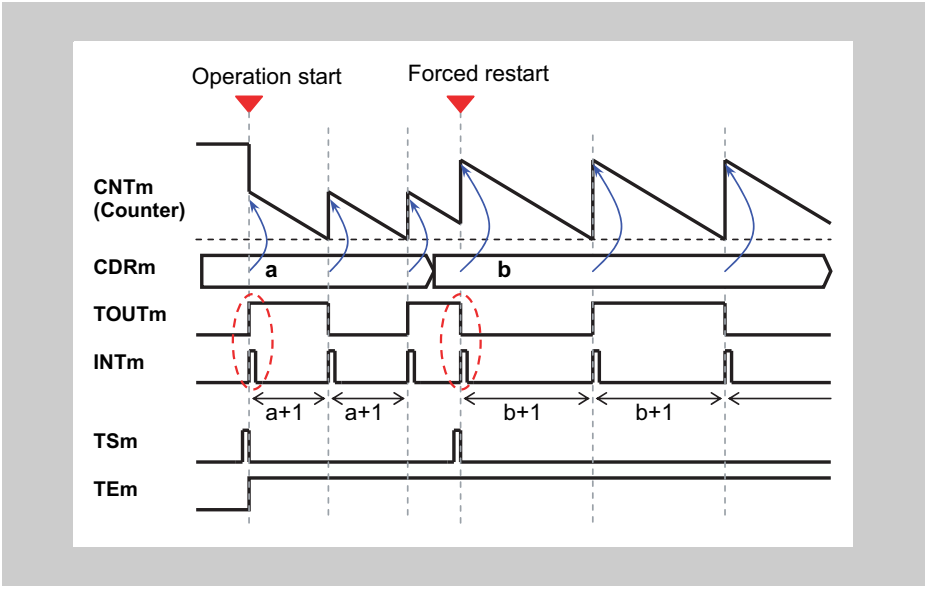


Figure 16-23 Forced restart operation, TAUBnCMORm.MD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.
- If the TAUBnCMORm.MD0 bit is set to 1, an interrupt at start or restart is generated and the output TOUTm toggles.

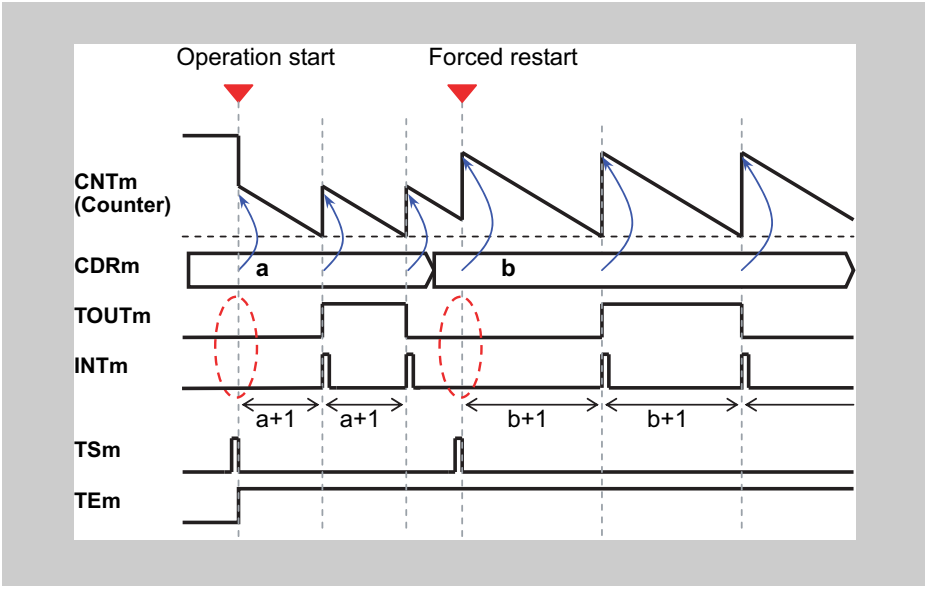


Figure 16-24 Forced restart operation, TAUBnCMORm.MD0 = 0

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm to 1 during operation.
- If the TAUBnCMORm.MD0 bit is set to 0, the interrupt at start or restart is not generated and TOUTm does not toggle..

16.14.2 TAUBnTTINm Input Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUBnIm) at regular intervals or when a valid TAUBnTTINm input edge is detected. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 16-18 "TAUBnCMORm settings for TAUBnTTINm Input Interval Timer Function" on page 1145*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *16.9 "Channel Output Modes" on page 1122*

Description This function operates in an identical manner to the Interval Timer Function (see *16.14.1 "Interval Timer Function" on page 1135*), except that this function is restarted by a valid TAUBnTTINm input edge. The type of edge used as the trigger is specified using the TAUBnCMURm.TIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUBnIm cycle = count clock cycle x (TAUBnCDRm + 1)

TAUBnTTOUTm square wave cycle = count clock cycle x (TAUBnCDRm + 1) x 2

(3) Block diagram and general timing diagram

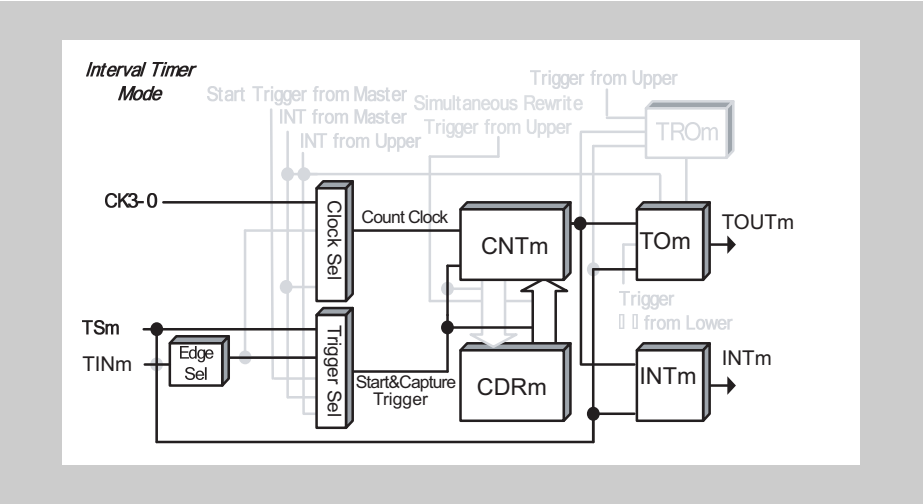


Figure 16-25 Block diagram for TAUBnTTINm Input Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1) and valid edge detection of input signal.
- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

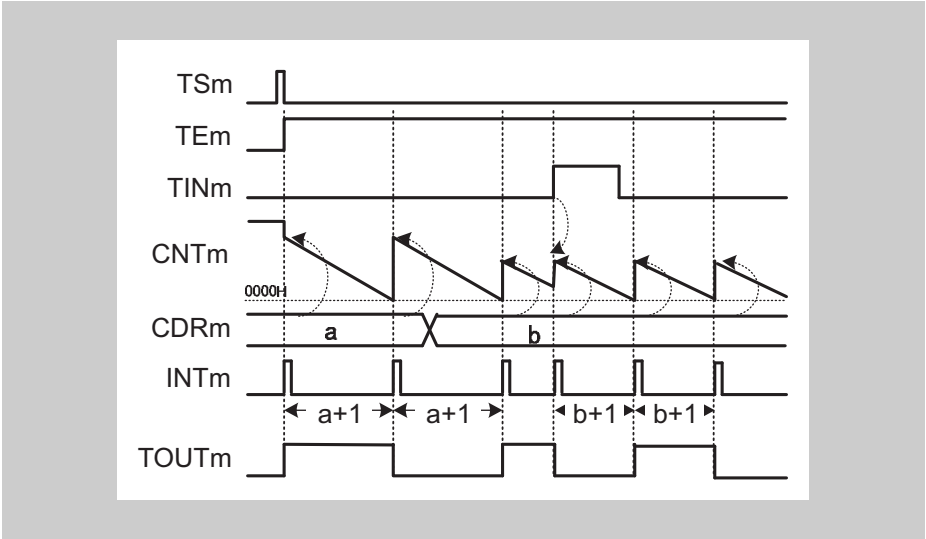


Figure 16-26 General timing diagram for TAUBnTTINm Input Interval Timer Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-18 TAUBnCMORm settings for TAUBnTTINm Input Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-19 TAUBnCMURm settings for TAUBnTTINm Input Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 16-20 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details refer to *16-11 “Channel output modes” on page 1123* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Interval Timer Function. Therefore, these registers must be set to 0.

Table 16-21 Simultaneous rewrite settings for TAUBnTTINm Input Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm Input Interval Timer Function

Table 16-22 Operating procedure for TAUBnTTINm Input Interval Timer Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-18 "TAUBnCMORm settings for TAUBnTTINm Input Interval Timer Function" on page 1145* and *Table 16-19 "TAUBnCMURm settings for TAUBnTTINm Input Interval Timer Function" on page 1145* Set the value of the TAUBnCDRm register Set the channel output mode by setting the control bits as described in *Table 16-20 "Control bit settings for Independent Channel Output Mode 1" on page 1146*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The values of the TAUBnCMURm.TIS[1:0] and TAUBnTO.TOm bits and the TAUBnCDRm register can be changed at any time. The TAUBnCNTm register can be read at all times. Detection of TAUBnTTINm edge	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated and TAUBnTTOUTm toggles When a TAUBnTTINm input valid edge is detected during count operation, TAUBnCNTm reloads the TAUBnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *16.14.1 "Interval Timer Function" on page 1135* also apply, except for this function the counter can also be restarted by a valid TAUBnTTINm input edge.

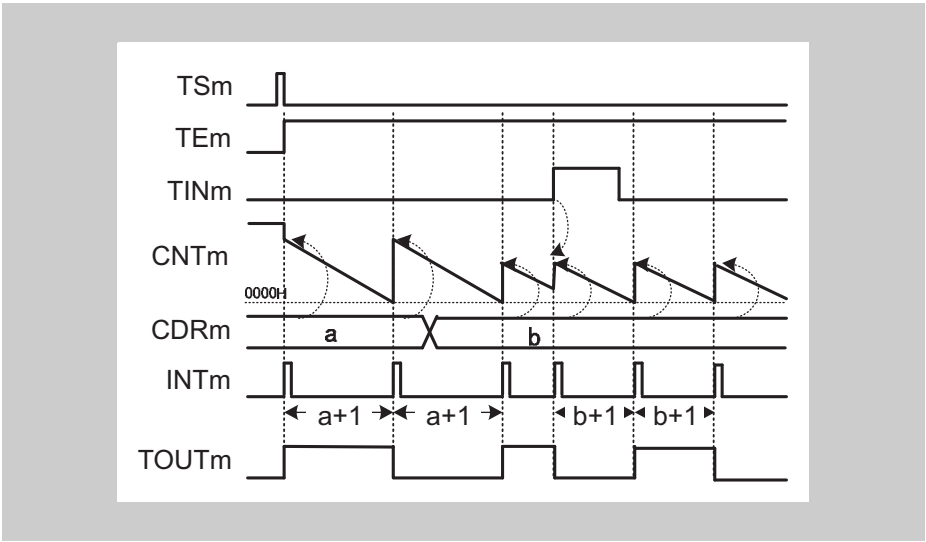


Figure 16-27 Counter triggered by rising TAUBnTTINm input edge (TAUBnCMURm.TIS[1:0] = 01_B), TAUBnCMORm.MD0 = 1

- If a valid TAUBnTTINm input edge is detected, an interrupt is generated which causes TAUBnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUBnCMURm.TIS[1:0] = 01_B).

16.14.3 One-Pulse Output Function

(1) Overview

Summary This function generates an interrupt (INTTAUBnIm) when a valid TAUBnTTINm input edge is detected and also a specific interval later. TAUBnTTINm input signal pulses that occur within the defined interval are ignored. When an interrupt is generated, the TAUBnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Pulse One Count Mode, refer to *Table 16-23 "TAUBnCMORm settings for One-Pulse Output Function" on page 1151*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *16.9 "Channel Output Modes" on page 1122*
- Trigger detection must be disabled during counting (TAUBnCMORn.MD0 = 0).

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input edge is detected. The value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from the TAUBnCDRm value. An interrupt is generated and TAUBnTTOUTm toggles.

When the counter reaches 0001_H an interrupt is generated and TAUBnTTOUTm toggles. The counter stops at 0000_H and awaits the next valid TAUBnTTINm input edge.

When the counter is counting down, further TAUBnTTINm input signals are ignored, i.e. the counter does not reset.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUBnCMURm.TIS[1:0] bits:

- If TAUBnCMURm.TIS[1:0] = 00_B, falling edges trigger the counter.
- If TAUBnCMURm.TIS[1:0] = 01_B, rising edges trigger the counter.
- If TAUBnCMURm.TIS[1:0] = 10_B, rising and falling edges trigger the counter.

(2) Equations

Interval between TAUBnTTINm and INTTAUBnIm = TAUBnTTOUTm (timer output) width = count clock cycle × TAUBnCDRm

(3) Block diagram and general timing diagram

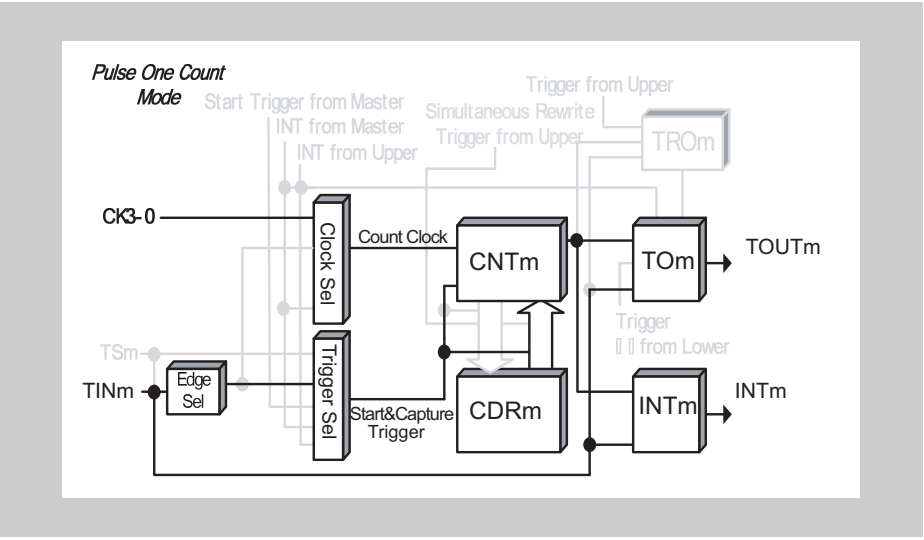


Figure 16-28 Block diagram for One-Pulse Output Function

The following settings apply to the general timing diagram:

- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

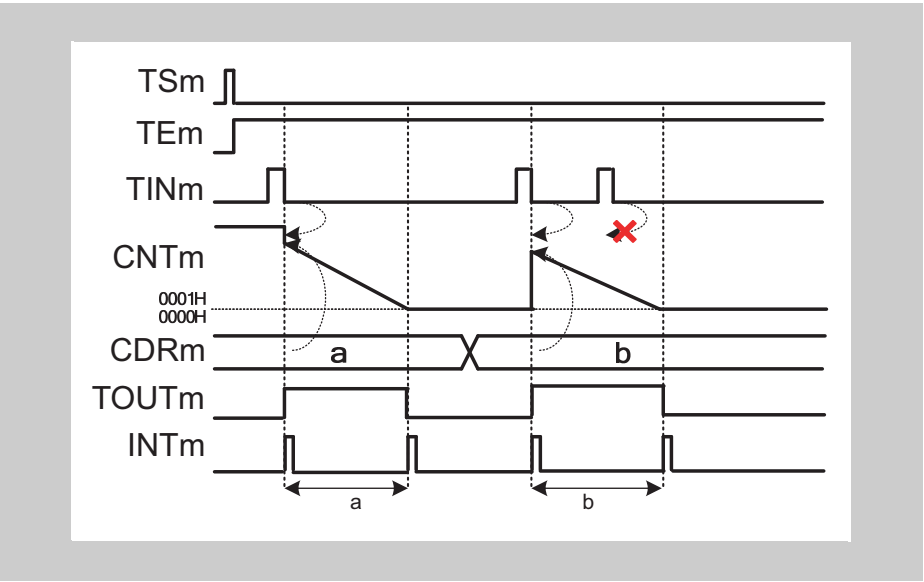


Figure 16-29 General timing diagram for One-Pulse Output Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-23 TAUBnCMORm settings for One-Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-24 TAUBnCMURm settings for One-Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement)

(c) Channel output mode

Table 16-25 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	1: Independent Channel Output Mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 16-11 “Channel output modes” on page 1123* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the One-Pulse Output Function. Therefore, these registers must be set to 0.

Table 16-26 Simultaneous rewrite settings for One-Pulse Output Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for One-Pulse Output Function

Table 16-27 Operating procedure for One-Pulse Output Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-23 "TAUBnCMORm settings for One-Pulse Output Function" on page 1151* and *Table 16-24 "TAUBnCMURm settings for One-Pulse Output Function" on page 1151* Set the value of the TAUBnCDRm register Set the channel output mode by setting the control bits as described in *Table 16-25 "Control bit settings for Independent Channel Output Mode 1" on page 1152*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value.
During operation	The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at all times.	INTTAUBnIm is generated when TAUBnCNTm starts and TAUBnTTOUTm is set to its active level. TAUBnCNTm counts down. When the counter reaches 0001_H: • INTTAUBnIm is generated • TAUBnTTOUTm is set to its inactive level. TAUBnCNTm stops counting and waits for a trigger. If a trigger occurs while TAUBnCNTm is counting, the trigger is ignored. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values.

Restart

16.15 Independent Channel Signal Measurement Functions

This chapter describes functions that measure the widths of an individual TAUBnTTINm pulse or the total width of successive TAUBnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *16.15.1 “TAUBnTTINm Input Pulse Interval Measurement Function”*
- *16.15.2 “TAUBnTTINm Input Signal Width Measurement Function”*
- *16.15.3 “Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)”*
- *16.15.4 “TAUBnTTINm Input Period Count Detection Function”*
- *16.15.5 “Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)”*
- *16.15.6 “TAUBnTTINm Input Pulse Interval Judgment Function”*
- *16.15.7 “TAUBnTTINm Input Signal Width Judgment Function”*

16.15.1 TAUBnTTINm Input Pulse Interval Measurement Function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUBnCSRm.OVF to measure the interval of the TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture Mode, refer to *Table 16-29 "TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Measurement Function" on page 1157*
- TAUBnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter TAUBnCNTm starts counting up from 0000_H. When a valid TAUBnTTINm edge is detected, the value of TAUBnCNTm is captured, transferred to TAUBnCDRm, and an interrupt INTTAUBnIm is generated. The counter resets to 0000_H and subsequently continues operation.

If the counter reaches $FFFF_H$ before a valid TAUBnTTINm edge is detected, it overflows to 0000_H. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUBnCDRm and TAUBnCSRm.OVF respectively depend on the values of bits TAUBnCMORm.COS[1:0]:

Table 16-28 Effects of an overflow

TAUBnCMORm.COS[1:0]	When overflow occurs		When a valid TAUBnTTINm input is then detected	
	TAUBnCDRm	TAUBnCSRm.OVF	TAUBnCDRm and TAUBnCNTm	TAUBnCSRm.OVF
00	Unchanged	0	TAUBnCNTm written to TAUBnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUBnCNTm set to 0, TAUBnCDRm unchanged	0
11		1		

If an overflow is set (TAUBnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUBnCSCm.CLOV = 1.

The combination of the value of TAUBnCDRm and TAUBnCSRm.OVF can be used to deduce the interval of the TAUBnTTINm signal. However, if an overflow occurs multiple times before a valid TAUBnTTINm input is detected, the overflow bit TAUBnCSRm.OVF cannot indicate this.

The function can be stopped by setting TAUBnTT.TTm = 1, which in turn sets TAUBnTE.TEm = 0. TAUBnCNTm stops but retains its value. While the function is stopped, TAUBnTTINm input valid edge detection and TAUBnCNTm capture are not performed.

The function can be restarted by setting TAUBnTS.TSm = 1. The counter is reset to 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the interrupt at start or restart is not generated. For details refer to *16.11 "TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)" on page 1131* .

Note When TAUBnCMORm.COS[1:0] = 11_B, the value of TAUBnCNTm is *not* written to TAUBnCDRm when the first valid TAUBnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUBnTTINm input pulse interval = count clock cycle x [(TAUBnCSRm.OVF x ($FFFF_H$ + 1)) + TAUBnCDRm capture value + 1]

(3) Block diagram and general timing diagram

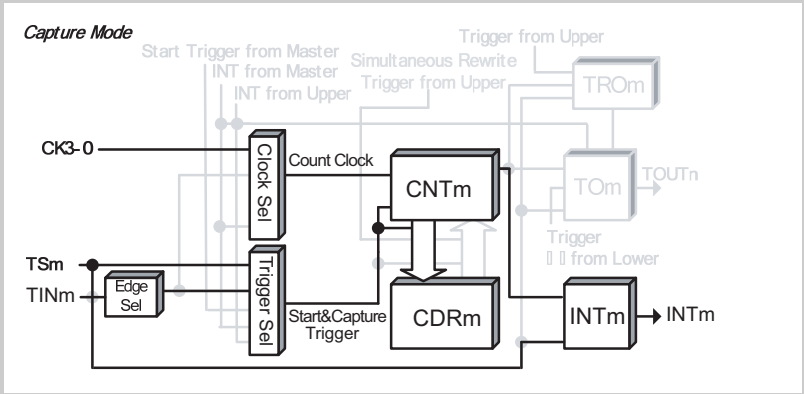


Figure 16-30 Block diagram for TAUBnTTINm Input Pulse Interval Measurement Function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)
- When a valid TAUBnTTINm input is detected after an overflow TAUBnCDRm is changed and TAUBnCSRm.OVF is set to 1 (TAUBnCMORm.COS[1:0] = 00_B)

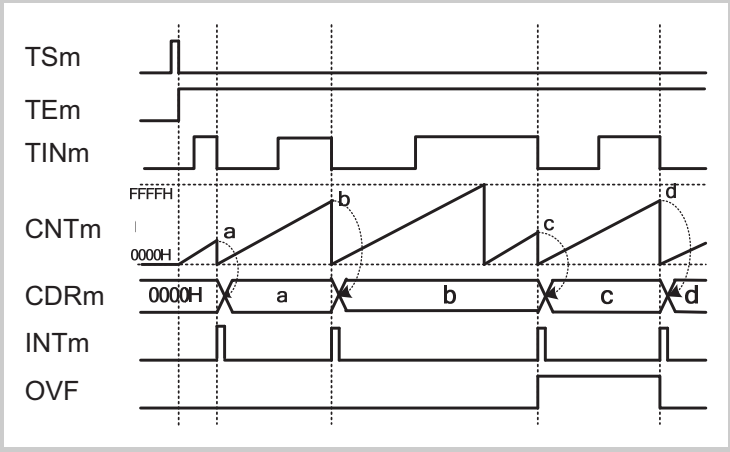


Figure 16-31 General timing diagram for TAUBnTTINm Input Pulse Interval Measurement Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-29 TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUBnTTINm input signal is the external capture trigger
COS[1:0]	See *Table 16-28 “Effects of an overflow” on page 1155*
MD[4:1]	0010: Capture Mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-30 TAUBnCMURm settings for TAUBnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simuBtaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Pulse Interval Measurement Function. Therefore, these registers must be set to 0.

Table 16-31 Simultaneous rewrite settings for TAUBnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm Input Pulse Interval Measurement Function

Table 16-32 Operating procedure for TAUBnTTINm Input Pulse Interval Measurement Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-29 "TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Measurement Function" on page 1157* and *Table 16-30 "TAUBnCMURm settings for TAUBnTTINm Input Pulse Interval Measurement Function" on page 1157* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm is cleared to 0000_H. INTTAUBnIm is generated when TAUBnCMORm.MD0 is set to 1.
During operation	Detection of TAUBnTTINm edges. The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm, and returns to 0000_H • INTTAUBnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUBnCMORm.COS[1:0] = 00_B

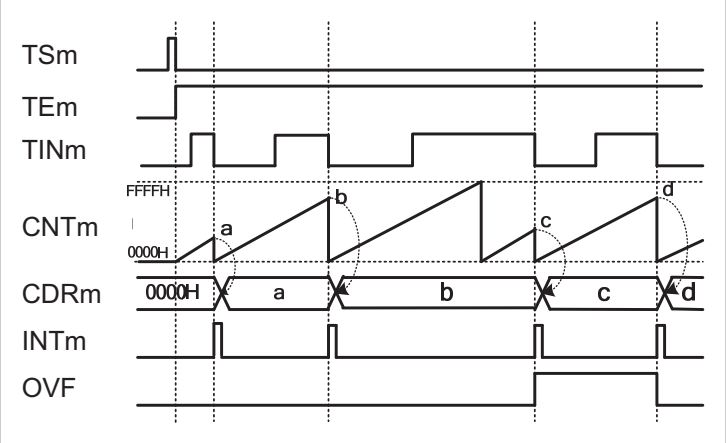


Figure 16-32 TAUBnCMORm.COS[1:0] = 00_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm and TAUBnCSRm.OVF is set to 1.

(b) TAUBnCMORm.COS[1:0] = 01_B

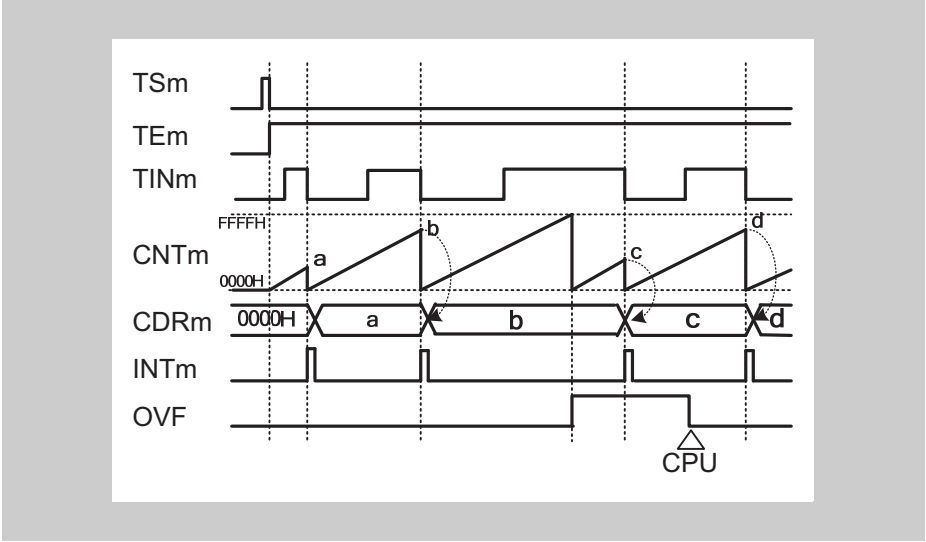


Figure 16-33 TAUBnCMORm.COS[1:0] = 01_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm.
- TAUBnCSRm.OVF is only cleared by a CPU command.

(c) TAUBnCMORm.COS[1:0] = 10_B

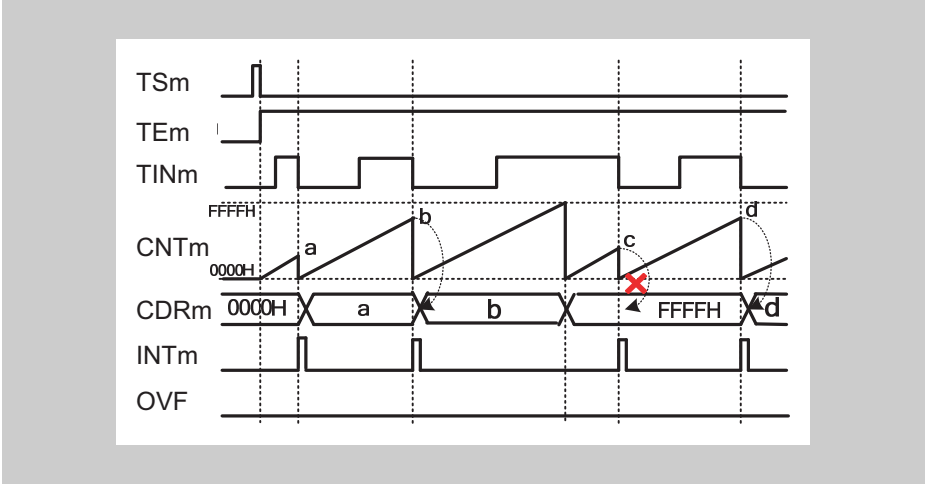


Figure 16-34 TAUBnCMORm.COS[1:0] = 10_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$ and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.

(d) TAUBnCMORm.COS[1:0] = 11_B

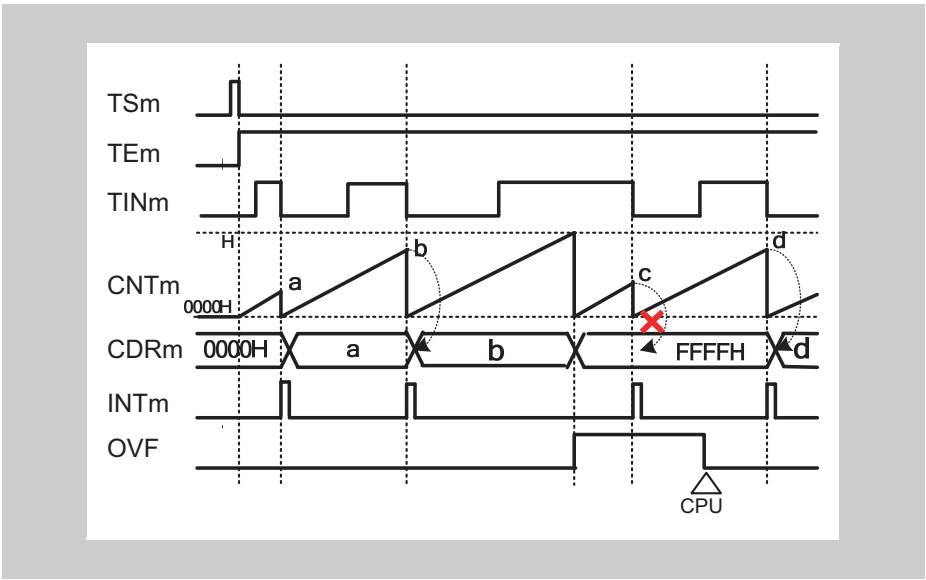


Figure 16-35 TAUBnCMORm.COS[1:0] = 11_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$, and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.
- TAUBnCSRm.OVF is cleared by a CPU command.

16.15.2 TAUBnTTINm Input Signal Width Measurement Function

(1) Overview

Summary This function measures the width of a TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & One Count Mode, refer to *Table 16-34 "TAUBnCMORm settings for TAUBnTTINm Input Signal Width Measurement Function" on page 1165*
- TAUBnTTOUTm is not used for this function
- TAUBnCMORm.MD0 must be set to 0

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When a valid TAUBnTTINm start edge is detected, the counter TAUBnCNTm starts counting up from 0000_H. When a valid TAUBnTTINm stop edge is detected, the value of TAUBnCNTm is captured, transferred to TAUBnCDRm, and an interrupt INTTAUBnIm is generated. The counter retains its value and awaits the next valid TAUBnTTINm input start edge.

If the counter reaches $FFFF_H$ before a valid TAUBnTTINm stop edge is detected, it overflows. The counter is reset to 0000_H and subsequently continues operation. The values transferred to TAUBnCDRm and TAUBnCSRm.OVF respectively depend on the values of bits TAUBnCMORm.COS[1:0]:

Table 16-33 Effects of an overflow

TAUBnCMORm.COS[1:0]	When overflow occurs		When a valid TAUBnTTINm input stop edge is detected	
	TAUBnCDRm	TAUBnCSRm.OVF	TAUBnCDRm and TAUBnCNTm	TAUBnCSRm.OVF
00	Unchanged	0	TAUBnCNTm written to TAUBnCDRm	1
01		1		
10	Set to $FFFF_H$	0	TAUBnCNTm stops counting TAUBnCDRm unchanged	0
11		1		

If an overflow is set (TAUBnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUBnCSCm.CLOV = 1.

The combination of the value of TAUBnCDRm and TAUBnCSRm.OVF can be used to deduce the width of the TAUBnTTINm signal. However, if an overflow occurs multiple times before a valid TAUBnTTINm input is detected, the overflow bit TAUBnCSRm.OVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUBnCMORm.COS[1:0] = 11_B, the value of TAUBnCNTm is *not* written to TAUBnCDRm when the first valid TAUBnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUBnTTINm input signal width = count clock cycle x [(TAUBnCSRm.OVF x ($FFFF_H$ + 1)) + TAUBnCDRm capture value + 1]

(3) Block diagram and general timing diagram

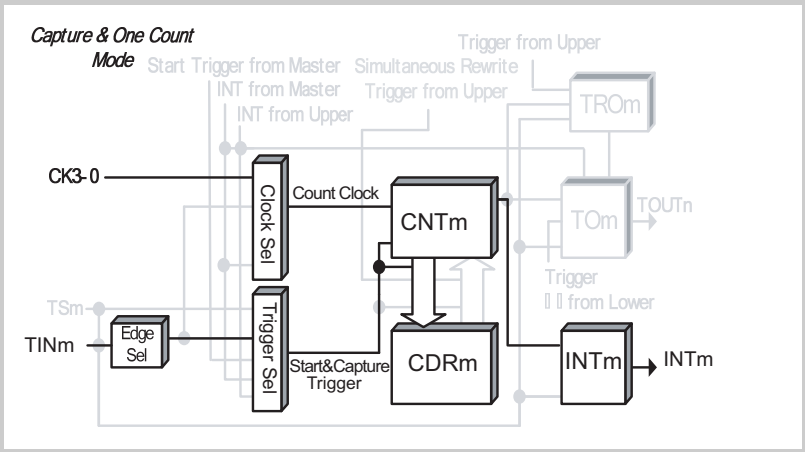


Figure 16-36 Block diagram for TAUBnTTINm Input Signal Width Measurement Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)
- When a valid TAUBnTTINm input is detected after an overflow TAUBnCDRm is changed and TAUBnCSRm.OVF is set to 1 (TAUBnCMORm.COS[1:0] = 00_B)

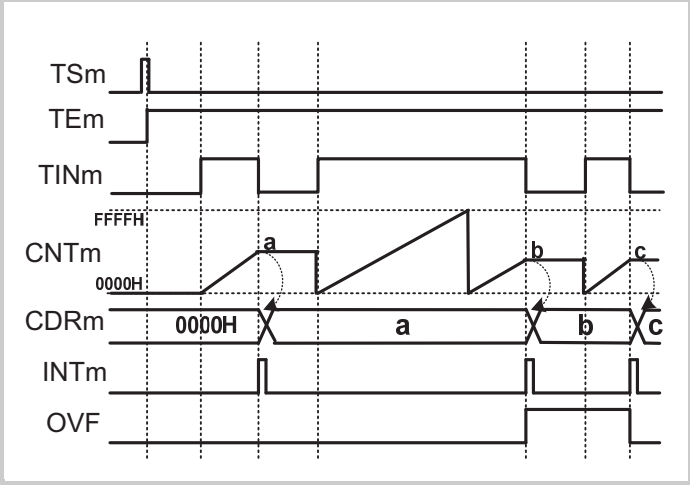


Figure 16-37 General timing diagram for TAUBnTTINm Input Signal Width Measurement Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-34 TAUBnCMORm settings for TAUBnTTINm Input Signal Width Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	See *Table 16-33 “Effects of an overflow” on page 1163*
MD[4:1]	0110: Capture & One Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-35 TAUBnCMURm settings for TAUBnTTINm Input Signal Width Measurement Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Signal Width Measurement Function. Therefore, these registers must be set to 0.

Table 16-36 Simultaneous rewrite settings for TAUBnTTINm Input Signal Width Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for TAUBnTTINm Input Signal Width Measurement Function

Table 16-37 Operating procedure for TAUBnTTINm Input Signal Width Measurement Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-34 "TAUBnCMORm settings for TAUBnTTINm Input Signal Width Measurement Function" on page 1165* and *Table 16-35 "TAUBnCMURm settings for TAUBnTTINm Input Signal Width Measurement Function" on page 1165* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a TAUBnTTINm start is detected, TAUBnCNTm starts to count up.
During operation	Detection of TAUBnTTINm edges. The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm, and retains its value • INTTAUBnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUBnCMORm.COS[1:0] = 00_B

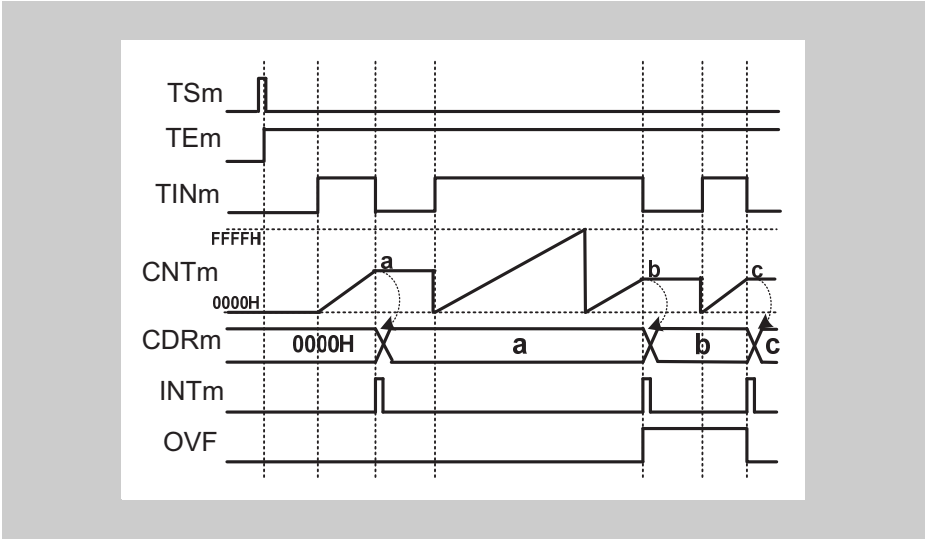


Figure 16-38 TAUBnCMORm.COS[1:0] = 00_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm and TAUBnCSRm.OVF is set to 1.

(b) TAUBnCMORm.COS[1:0] = 01_B

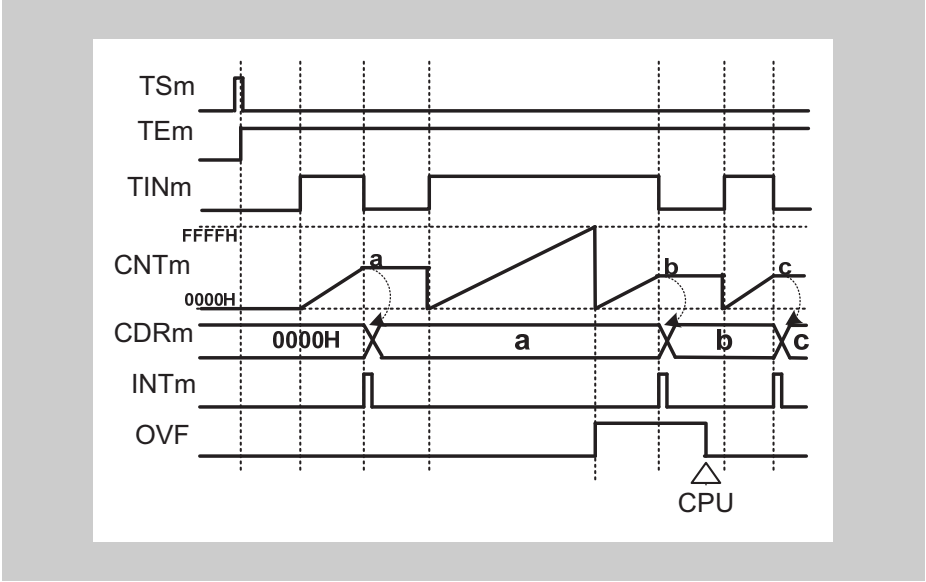


Figure 16-39 TAUBnCMORm.COS[1:0] = 01_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUBnCDRm remains unchanged and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, the value of TAUBnCNTm is written to TAUBnCDRm.
- TAUBnCSRm.OVF is only cleared by a CPU command.

(c) TAUBnCMORm.COS[1:0] = 10_B

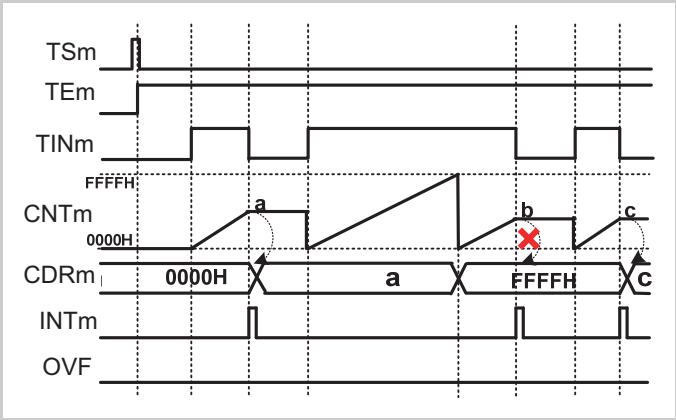


Figure 16-40 TAUBnCMORm.COS[1:0] = 10_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$ and TAUBnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.

(d) TAUBnCMORm.COS[1:0] = 11_B

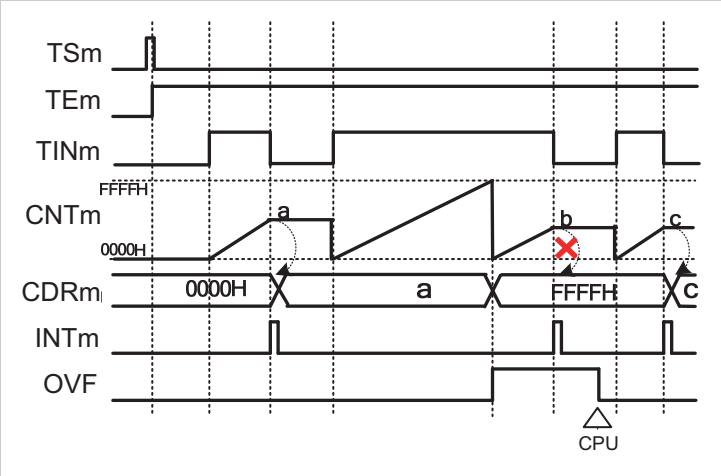


Figure 16-41 TAUBnCMORm.COS[1:0] = 11_B, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUBnCDRm is set to $FFFF_H$, and TAUBnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUBnTTINm input edge, TAUBnCNTm is reset to 0, but TAUBnCDRm and TAUBnCSRm.OVF remain unchanged.
- Thus, the next TAUBnTTINm input valid edge after the overflow is ignored.
- TAUBnCSRm.OVF is cleared by a CPU command.

16.15.3 Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

(1) Overview

Summary This function measures the width of an individual TAUBnTTINm input signal. An interrupt is generated if the TAUBnTTINm input width is longer than $FFFF_H$.

Prerequisites
- The operation mode must be set to One Count Mode, refer to *Table 16-38 "TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)" on page 1173*
- TAUBnTTOUTm is not used for this function
- The value of TAUBnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input start edge is detected. $FFFF_H$ is written to TAUBnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUBnTTINm input start edge is detected, TAUBnCNTm reloads $FFFF_H$ and starts to count down.

If the counter reaches 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

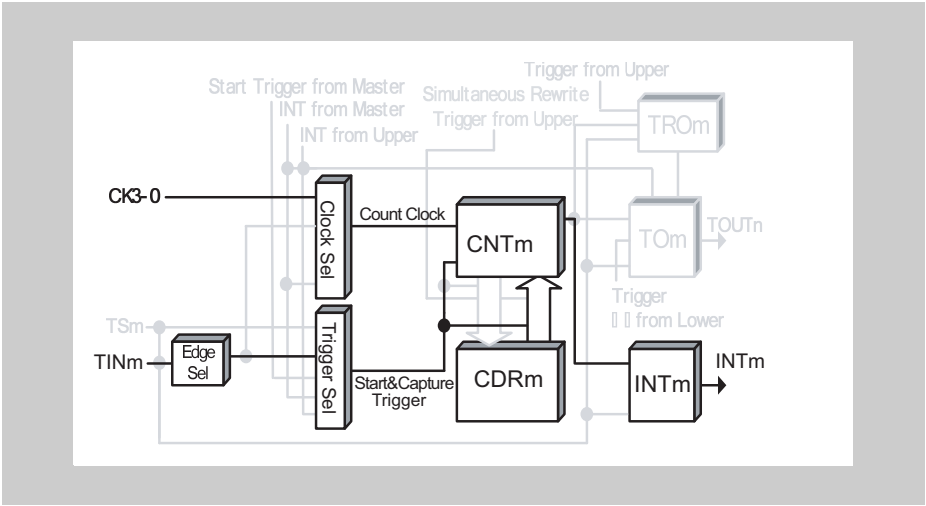


Figure 16-42 Block diagram for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

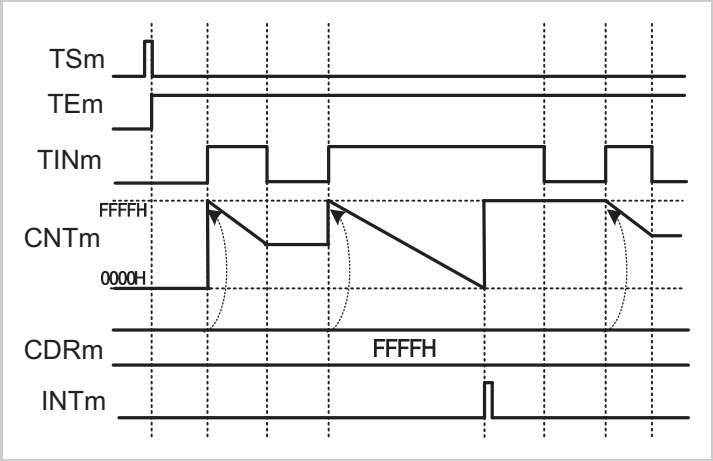


Figure 16-43 General timing diagram for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-38 TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-39 TAUBnCMURm settings Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement). Therefore, these registers must be set to 0.

(4) Operating procedure for Overflow Interrupt Output Function

Table 16-40 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(During TAUBnTTINm Width Measurement)

Table 16-41 Operating procedure for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-38 "TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)" on page 1173* and *Table 16-39 "TAUBnCMURm settings Overflow Interrupt Output Function (During TAUBnTTINm Width Measurement)" on page 1173* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$).
During operation	The TAUBnCNTm register can be read at any time. Detection of TAUBnTTINm edges.	TAUBnCNTm counts down. When the counter reaches 0000_H: • INTTAUBnIm is generated • TAUBnCNTm stops counting at $FFFF_H$ and waits for a trigger. When a reverse edge of TAUBnTTINm is detected during count operation: • TAUBnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.4 TAUBnTTINm Input Period Count Detection Function

(1) Overview

Summary This function measures the cumulative width of a TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & Gate Count Mode, refer to *Table 16-42 "TAUBnCMORm settings for TAUBnTTINm Input Period Count Detection Function" on page 1178*
- TAUBnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter awaits a valid TAUBnTTINm input edge.

When a valid TAUBnTTINm input start edge is detected, the counter starts to count from 0000_H.

When a valid TAUBnTTINm input stop edge is detected, the current TAUBnCNTm value is written to TAUBnCDRm and an interrupt (INTTAUBnIm) is generated. The counter stops and retains its value until the next valid TAUBnTTINm input start edge is detected.

If the counter reaches $FFFF_H$ the bit TAUBnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUBnCSRm.OVF is reset by the CPU by setting TAUBnCSCm.CLOV = 1.

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by the TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

Cumulative TAUBnTTINm input width =
count clock cycle × (($FFFF_H$ × TAUBnCSRm.OVF) + (TAUBnCDRm capture value + 1))

(3) Block diagram and general timing diagram

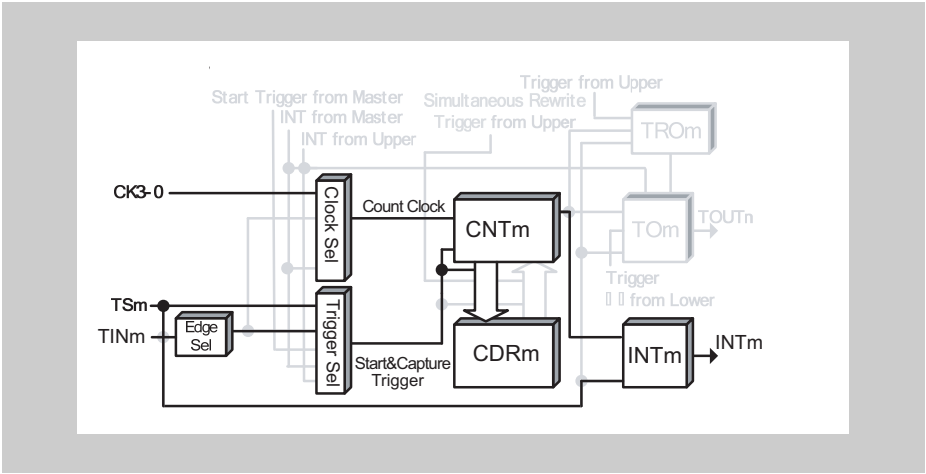


Figure 16-44 Block diagram for TAUBnTTINm Input Period Count Detection Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

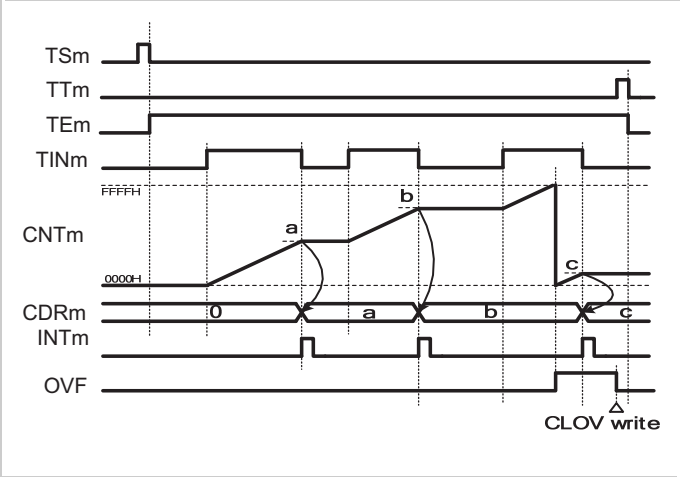


Figure 16-45 General timing diagram for TAUBnTTINm Input Period Count Detection Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-42 TAUBnCMORm settings for TAUBnTTINm Input Period Count Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	01: Overflow (TAUBnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1101: Capture & Gate Count Mode
MD0	0: INTTAUBnIm not generated at operation start and start trigger during count disabled

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-43 TAUBnCMURm settings for TAUBnTTINm Input Period Count Detection Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Period Count Detection Function. Therefore, these registers must be set to 0.

Table 16-44 Simultaneous rewrite settings for TAUBnTTINm Input Period Count Detection Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUBnTTINm Input Period Count Detection Function

Table 16-45 Operating procedure for TAUBnTTINm Input Period Count Detection Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-42 "TAUBnCMORm settings for TAUBnTTINm Input Period Count Detection Function" on page 1178* and *Table 16-43 "TAUBnCMURm settings for TAUBnTTINm Input Period Count Detection Function" on page 1178* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a start edge is detected, TAUBnCNTm is cleared to 0000_H and TAUBnCNTm starts to count up.
During operation	Detection of TAUBnTTINm edges. The TAUBnCDRm, TAUBnCNTm, and TAUBnCSRm registers can be read at any time. The TAUBnCSC.CLOV bit can be set to 1.	When a TAUBnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUBnCNTm starts to count up from the stop value. When TAUBnCNTm detects a capture edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUBnCDRm and INTTAUBnIm is generated. Counting stops at the "value transferred to TAUBnCDRm + 1" value and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. If the TAUBnCNTm reaches $FFFF_H$, the counter overflows and TAUBnCSR.OVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

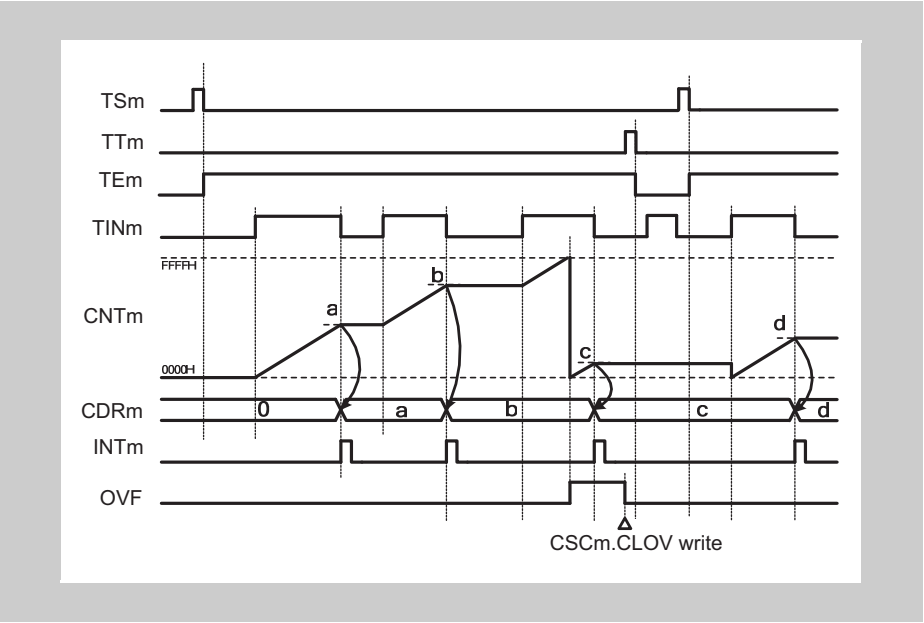


Figure 16-46 Operation stop and restart, TAUBnCMURm.TIS[1:0] = 11_B

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUBnTTINm input edges are ignored.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm restarts to count from 0000_H.

16.15.5 Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

(1) Overview

Summary This function measures the cumulative width of a TAUBnTTINm input signal. An interrupt is generated if the cumulative TAUBnTTINm input width is longer than $FFFF_H$.

Prerequisites

- The operation mode must be set to Gate Count Mode, refer to *Table 16-46 "TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)" on page 1184*
- TAUBnTTOUTm is not used for this function
- The value of TAUBnCDRm must be set to $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUBnTTINm input start edge is detected. $FFFF_H$ is written to TAUBnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUBnTTINm input start edge and then continues to count down from the current value.

When the counter reaches 0000_H an interrupt is generated. $FFFF_H$ is written to TAUBnCNTm and the counter continues to count down until a TAUBnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUBnCMURm.TIS[1:0] bits:

- If TAUBnCMURm.TIS[1:0] = 10_B, the TAUBnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUBnCMURm.TIS[1:0] = 11_B, the TAUBnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

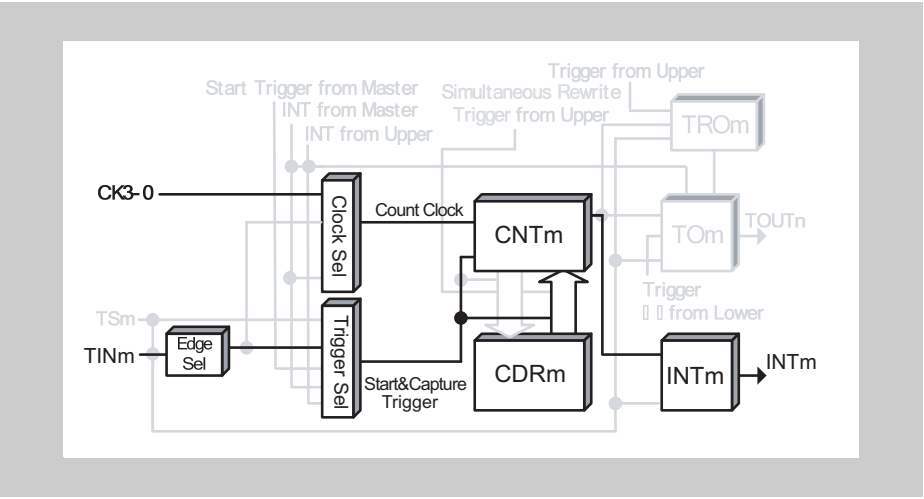


Figure 16-47 Block diagram for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUBnCMURm.TIS[1:0] = 11_B)

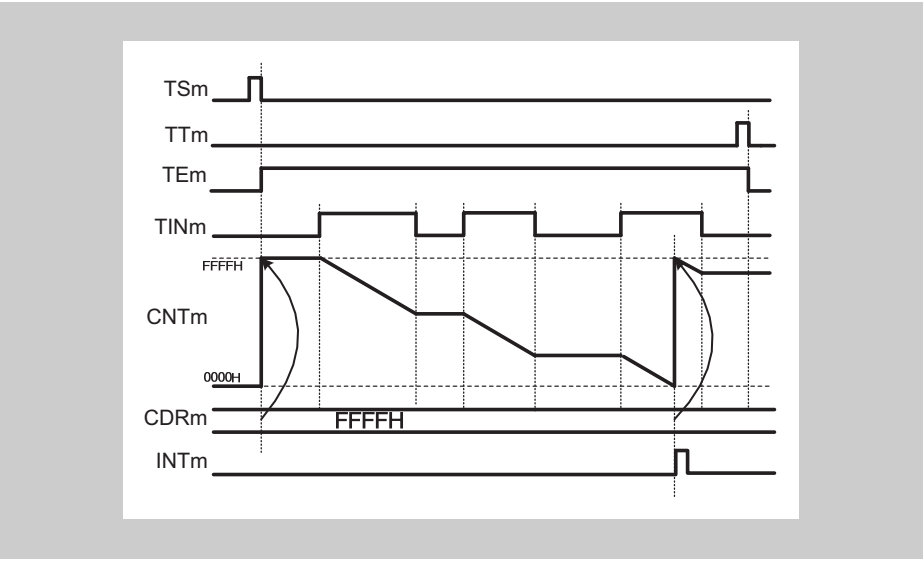


Figure 16-48 General timing diagram for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-46 TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1100: Gate Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-47 TAUBnCMURm settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection). Therefore, these registers must be set to 0.

Table 16-48 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(4) Operating procedure for Overflow Interrupt Output Function (during TAUBnTTINm input period count detection)

Table 16-49 Operating procedure for Overflow Interrupt Output Function (during TAUBnTTINm input period count detection)

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-46 "TAUBnCMORm settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)" on page 1184* and *Table 16-47 "TAUBnCMURm settings for Overflow Interrupt Output Function (During TAUBnTTINm Input Period Count Detection)" on page 1184* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the start edge. When a start edge is detected, TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$).
During operation	The TAUBnCNTm register can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • INTTAUBnIm is generated • TAUBnCNTm loads the TAUBnCDRm value ($FFFF_H$) and continues to count down. When a reverse edge of TAUBnTTINm is detected during count operation: • TAUBnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.6 TAUBnTTINm Input Pulse Interval Judgment Function

(1) Overview

Summary This function outputs the result of a comparison between the count value (TAUBnCNTm) and the value in the channel data register (TAUBnCDRm) when a TAUBnTTINm input pulse occurs. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true.

Prerequisites
- The operation mode must be set to Judge Mode, refer to *Table 16-50 "TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Judgment Function" on page 1189*
- TAUBnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When a TAUBnTTINm valid edge is detected or TAUBnTS.TSm is set to 1, the function compares the current values of TAUBnCNTm and TAUBnCDRm. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true. TAUBnCNTm reloads the value of TAUBnCDRm and subsequently continues operation, regardless of the result of the comparison.

If the counter reaches 0000_H before a TAUBnTTINm valid edge is detected, TAUBnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

Conditions The TAUBnCMORm.MD0 bit specifies the type of comparison:
- If TAUBnCMORm.MD0 = 0, INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm.
- If TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm.

(2) Block diagram and general timing diagram

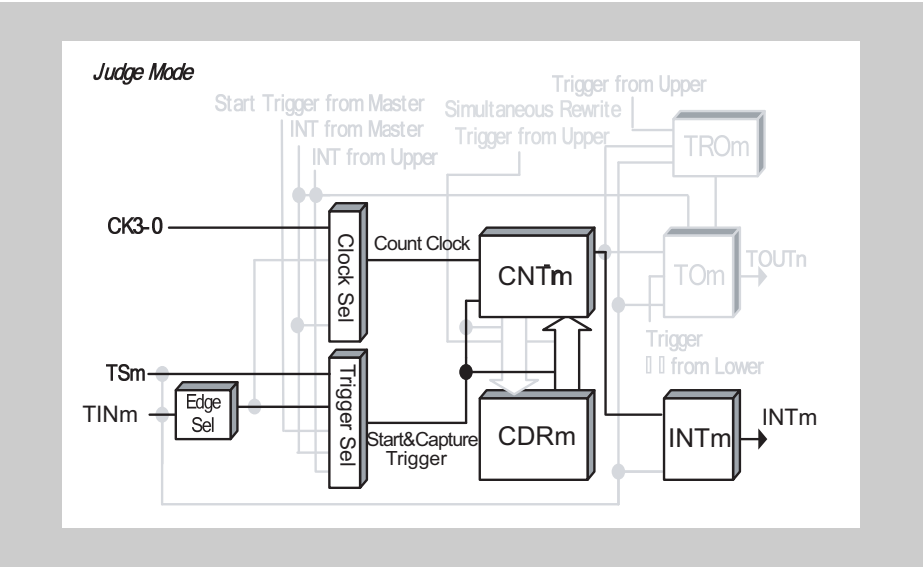


Figure 16-49 Block diagram for TAUBnTTINm Input Pulse Interval Judgment Function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

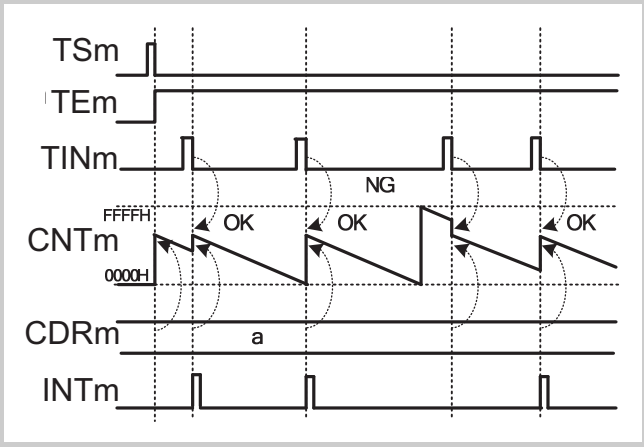


Figure 16-50 General timing diagram for TAUBnTTINm Input Pulse Interval Judgment Function

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-50 TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUBnTTINm input signal is the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0001: Judge Mode
MD0	0: INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm 1: INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-51 TAUBnCMURm settings for TAUBnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Pulse Interval Judgment Function. Therefore, these registers must be set to 0.

Table 16-52 Simultaneous rewrite settings for TAUBnTTINm Input Pulse Interval Judgment Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for for TAUBnTTINm Input Pulse Interval Judgment Function

Table 16-53 Operating procedure for TAUBnTTINm Input Pulse Interval Judgment Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-50 "TAUBnCMORm settings for TAUBnTTINm Input Pulse Interval Judgment Function" on page 1189* and *Table 16-51 "TAUBnCMURm settings for TAUBnTTINm Input Pulse Interval Judgment Function" on page 1189* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value.
During operation	Detection of TAUBnTTINm edges. The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm counts down. When a TAUBnTTINm input edge is detected • TAUBnCNTm reloads TAUBnCDRm and continues count operation. • TAUBnCNTm compares the values and judges the condition according to the TAUBnCMORm.MD0 setting. • If the condition is satisfied, INTTAUBnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.15.7 TAUBnTTINm Input Signal Width Judgment Function

(1) Overview

Summary

This function outputs the result of a comparison between the count value (TAUBnCNTm) and the value in the channel data register (TAUBnCDRm) at a valid stop edge of a TAUBnTTINm input signal. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true.

Prerequisites

- The operation mode must be set to Judge & One Count Mode, refer to *Table 16-54 "TAUBnCMORm settings for TAUBnTTINm Input Signal Width Judgment Function" on page 1194*
- TAUBnTTOUTm is not used for this function

Description

The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When a valid TAUBnTTINm input start edge is detected, the current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value.

When a TAUBnTTINm valid stop edge is detected, the function compares the current values of TAUBnCNTm and TAUBnCDRm. An interrupt signal INTTAUBnIm is generated if the result of the comparison is true. The counter TAUBnCNTm retains its value until the next TAUBnTTINm valid start edge is detected, regardless of the result of the comparison.

If the counter reaches 0000_H before a valid TAUBnTTINm stop edge is detected, TAUBnCNTm overflows and is set to $FFFF_H$. It then continues to count down.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

Conditions

- The TAUBnCMORm.MD0 bit specifies the type of comparison:
  - If TAUBnCMORm.MD0 = 0, INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm.
  - If TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm.
- The TAUBnCMURm.TIS[1:0] bits specify the type of width measurement:
  - For high width measurement, the start edge is a rising TAUBnTTINm edge and the stop edge is a falling TAUBnTTINm edge.
  - For low width measurement, the start edge is a falling TAUBnTTINm edge and the stop edge is a rising TAUBnTTINm edge.
- Forced restart is not possible for this function.

(2) Block diagram and general timing diagram

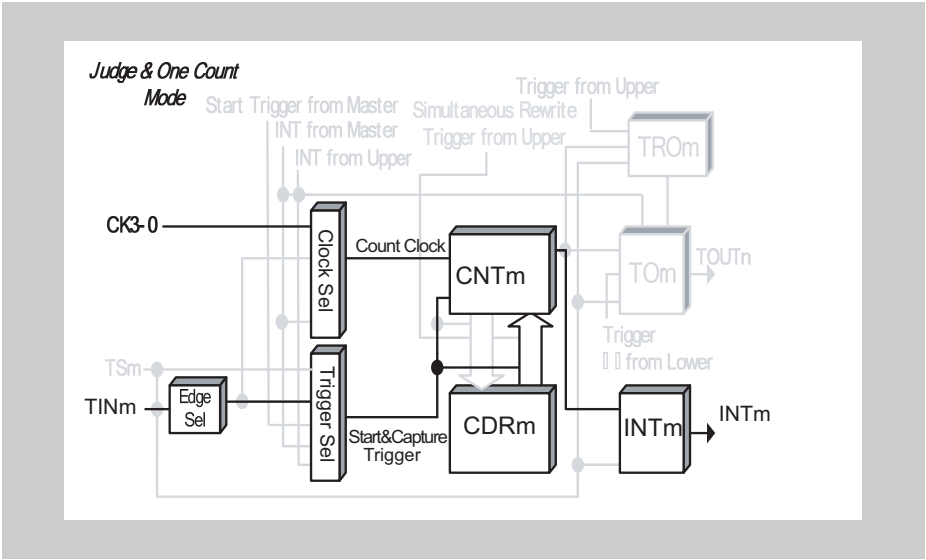


Figure 16-51 Block diagram for TAUBnTTINm Input Signal Width Judgment Function

The following settings apply to the general timing diagram:

- INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm (TAUBnCMORm.MD0 = 0)
- TAUBnTTINm valid start edge = rising edge, TAUBnTTINm valid stop edge = falling edge (TAUBnCMURm.TIS[1:0] = 11_B)

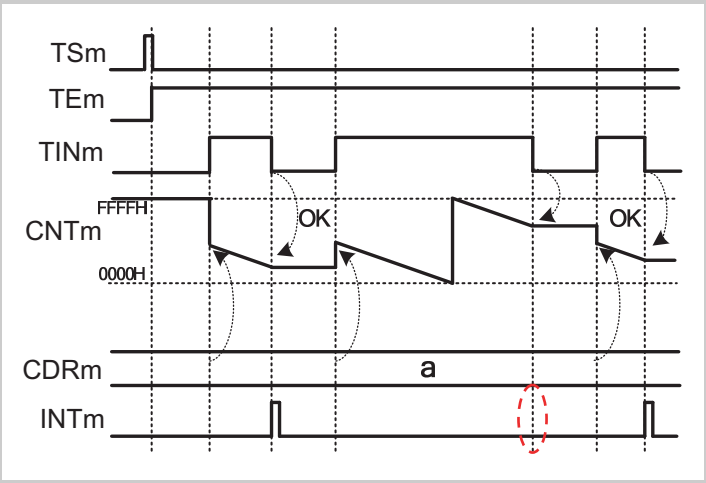


Figure 16-52 General timing diagram for TAUBnTTINm Input Signal Width Judgment Function

(3) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-54 TAUBnCMORm settings for TAUBnTTINm Input Signal Width Judgment Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUBnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0111: Judge & One Count Mode
MD0	0: INTTAUBnIm is generated when TAUBnCNTm ≤ TAUBnCDRm 1: INTTAUBnIm is generated when TAUBnCNTm > TAUBnCDRm

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-55 TAUBnCMURm settings for TAUBnTTINm Input Signal Width Judgment Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Signal Width Judgment Function. Therefore, these registers must be set to 0.

Table 16-56 Simultaneous rewrite settings for TAUBnTTINm Input Signal Width Judgment Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(4) Operating procedure for TAUBnTTINm Input Signal Width Judgment Function

Table 16-57 Operating procedure for TAUBnTTINm Input Signal Width Judgment Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-54 "TAUBnCMORm settings for TAUBnTTINm Input Signal Width Judgment Function" on page 1194* and *Table 16-55 "TAUBnCMURm settings for TAUBnTTINm Input Signal Width Judgment Function" on page 1194* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUBnTTINm start edge	TAUBnTE.TEm is set to 1 and TAUBnCNTm waits for detection of the TAUBnTTINm start edge. When a TAUBnTTINm start edge is detected, TAUBnCNTm loads the TAUBnCDRm value.
During operation	Detection of TAUBnTTINm edges The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm counts down. When a TAUBnTTINm stop edge is detected: • TAUBnCNTm stops and waits for detection of the TAUBnTTINm start edge. • TAUBnCNTm compares the values and judges the condition according to the TAUBnCMORm.MD0. • If the condition is satisfied, INTTAUBnIm is generated Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

16.16 Independent Channel Simultaneous Rewrite Functions

This chapter describes functions that carry out simultaneous rewrite:

- *16.16.1 “Simultaneous Rewrite Trigger Generation Function Type 1”*

16.16.1 Simultaneous Rewrite Trigger Generation Function Type 1

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is generated at regular intervals.

Prerequisites

- Two (or more) channels, each with simultaneous rewrite enabled (TAUBnRDE.RDEm = 1)
- The operation mode of the upper channel must be set to Interval Timer Mode, refer to *Table 16-58 "TAUBnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 1201*
- The operation mode of the lower channel(s) can be set as desired

Description The counters are started by setting the channel trigger bits (TAUBnTS.TSm and TAUBnTS.TSm_lower) to 1. This in turn sets TAUBnTE.TEm and TAUBnTE.TEm_lower = 1, enabling count operation. The current value of the data register buffer of the upper channel (TAUBnCDRm buf) is written to the counter (TAUBnCNTm) and the counter starts to count down from this value. The counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a counter reaches 0000_H, an interrupt is generated from the channel. The corresponding TAUBnCNTm then reloads the current TAUBnCDRm buffer value and subsequently continues operation.

If the channel where the interrupt occurs is specified as the trigger channel for simultaneous rewrite (TAUBnRDC.RDCm = 1) and is an upper channel, simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUBnRSF.RSFm = 1).

The values of the data registers are copied to the corresponding data register buffers. Each time a counter starts to count down, it reads the value in the data register buffer and counts down from this value.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel which is monitored for INTTAUBnIm is specified by setting TAUBnRDC.RDCm = 1 for the corresponding channel. The TAUBnRDC.RDCm bit must be 0 for all other channels in which simultaneous rewrite should take place.
- If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *16.11 "TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)" on page 1131* .

(2) Equations

Simultaneous rewrite trigger generation cycle = count clock cycle × (TAUBnCDRm + 1)

To control simultaneous rewrite, the following condition must be satisfied:

TAUBnCDRm = [(value of TAUBnCDRm of master channel subject to simultaneous rewrite + 1) × number of interrupts] - 1

That is, the ratio of TAUBnCDRm + 1 and TAUBnCDRm_master + 1 must be an integer. This integer corresponds to the number of interrupts.

(3) Block diagram and general timing diagram

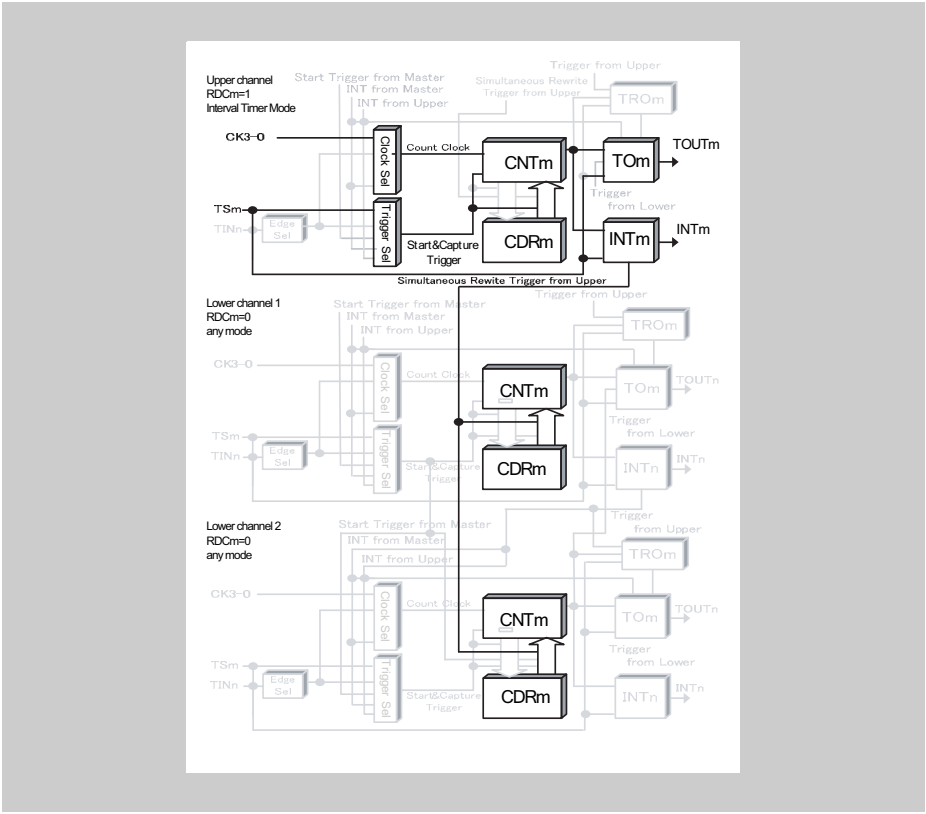


Figure 16-53 Block diagram for Simultaneous Rewrite Trigger Generation Function Type 1

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)

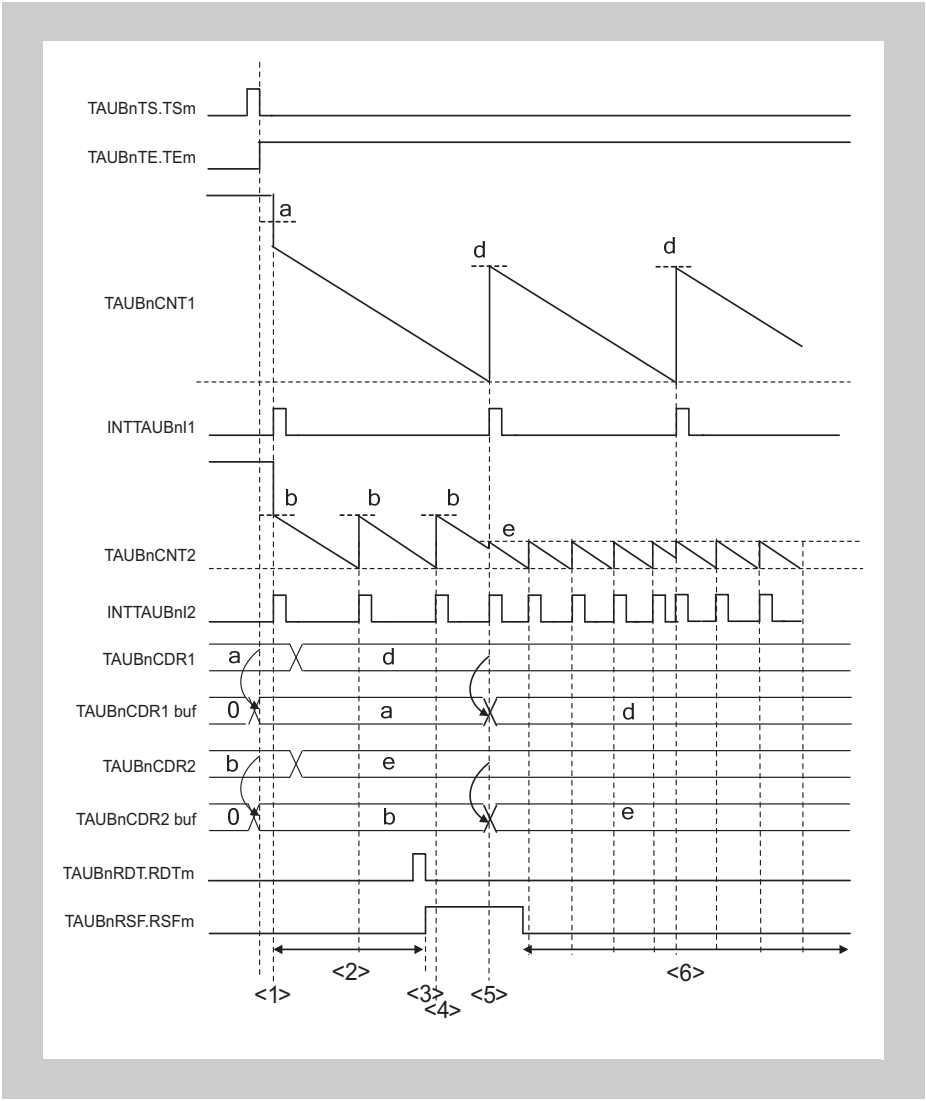


Figure 16-54 General timing diagram for Simultaneous Rewrite Trigger Generation Function Type 1

(4) Register settings for the upper channel

(a) TAUBnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-58 TAUBnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-59 TAUBnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the upper channel

Table 16-60 Simultaneous rewrite settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	1: Channel is monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger

(5) Register settings for the lower channel(s)

(a) TAUBnCMORm for the lower channel(s)

The TAUBnCMORm register of the lower channel(s) can be set arbitrarily.

(b) TAUBnCMURm for the lower channel(s)

The TAUBnCMURm register of the lower channel(s) can be set arbitrarily.

(c) Channel output mode for the lower channel(s)

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite for the lower channel(s)

Table 16-61 Simultaneous rewrite settings for the lower channel in Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDM.RDMm	0: The signal that controls simultaneous rewrite is loaded when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous-rewrite trigger

(6) Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

Table 16-62 Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers for the upper channel as described in *Table 16-58 "TAUBnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 1201* and *Table 16-59 "TAUBnCMURm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 1201* Set the TAUBnCMORm register and TAUBnCMURm registers for the lower channel as described in *5 "Register settings for the lower channel(s)"* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 = 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	TAUBnRDT.RDTm can be changed. TAUBnRSF.RSFm can be read at all times.	TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated • TAUBnTTOUTm toggles. Simultaneous rewrite is controlled when INTTAUBnIm is generated from the channel where TAUBnRDC.RDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnTTOUTm retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

16.17 Other Independent Channel Functions

This chapter describes a function that generates an interrupt when a certain number of TAUBnTTINm pulses has occurred, a function that divides the frequency of TAUBnTTINm, and a function that measures the duration between the function start and a TAUBnTTINm input signal:

- *16.17.1 “External Event Count Function”*
- *16.17.2 “Clock Divide Function”*
- *16.17.3 “TAUBnTTINm Input Position Detection Function”*

16.17.1 External Event Count Function

(1) Overview

Summary This function is used as an event timer. It generates an interrupt (INTTAUBnIm) when a specific number of TAUBnTTINm input pulses has occurred.

Prerequisites
- The operation mode must be set to Event Count Mode, refer to *Table 16-63 "TAUBnCMORm settings for External Event Count Function" on page 1207*
- TAUBnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. When the counter starts, the current value of TAUBnCDRm is written to TAUBnCNTm.

When a valid TAUBnTTINm input edge is detected, the value of TAUBnCNTm reduces by 1. TAUBnCNTm retains this value until a valid TAUBnTTINm input edge is detected or the counter is restarted.

When the counter value reaches 0000_H, INTTAUBnIm is generated. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm stops and retains its value. The counter can be restarted by setting TAUBnTS.TSm to 1. The counter can also be restarted without stopping it first (forced restart) by setting TAUBnTS.TSm to 1 during operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the counter starts to count down.

Conditions The type of edge used as the trigger is specified by the TAUBnCMURm.TIS[1:0] bits:
- If TAUBnCMURm.TIS[1:0] = 00_B, the falling edges are counted.
- If TAUBnCMURm.TIS[1:0] = 01_B, the rising edges are counted.
- If TAUBnCMURm.TIS[1:0] = 10_B, the rising and falling edges are counted.

(2) Equations

Number of valid edges,
detected before INTTAUBnIm is generated = TAUBnCDRm + 1

(3) Block diagram and general timing diagram

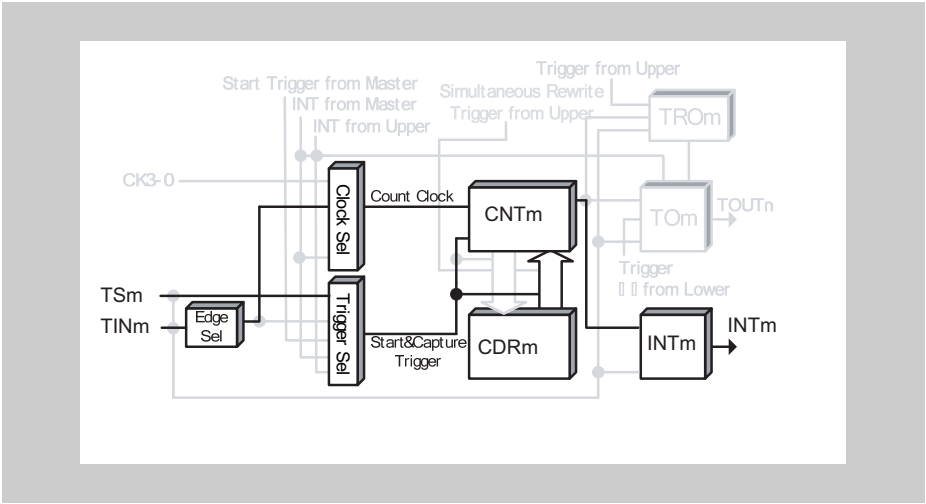


Figure 16-55 Block diagram for External Event Count Function

The following settings apply to the general timing diagram:

- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

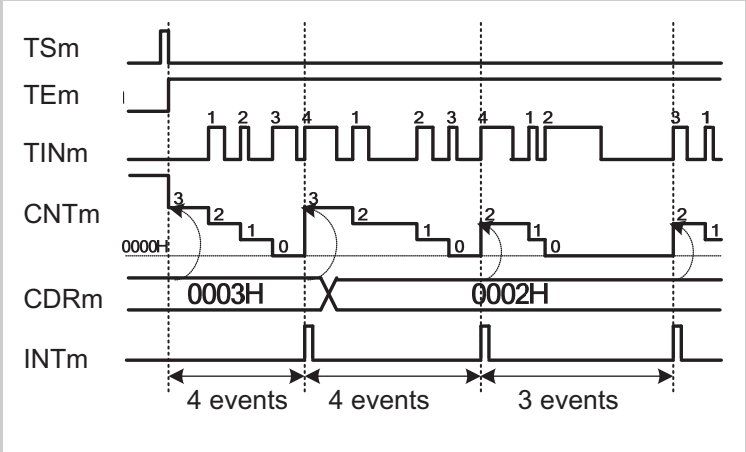


Figure 16-56 General timing diagram for External Event Count Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-63 TAUBnCMORm settings for External Event Count Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	1: Valid TAUBnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0011: Event Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-64 TAUBnCMURm settings for External Event Count Function

Bit name	Setting
TIS[1:0]	00: Falling edge 01: Rising edge 10: Rising and falling edge

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the External Event Count Function. Therefore, these registers must be set to 0.

Table 16-65 Simultaneous rewrite settings for External Event Count Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for External Event Count Function

Table 16-66 Operating procedure for External Event Count Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-63 "TAUBnCMORm settings for External Event Count Function" on page 1207* and *Table 16-64 "TAUBnCMURm settings for External Event Count Function" on page 1207* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value and waits for detection of the TAUBnTTINm input edge.
During operation	Detection of TAUBnTTINm edges. The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at any time.	TAUBnCNTm performs count-down operation each time a TAUBnTTINm input edge is detected. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and retains its current value.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H

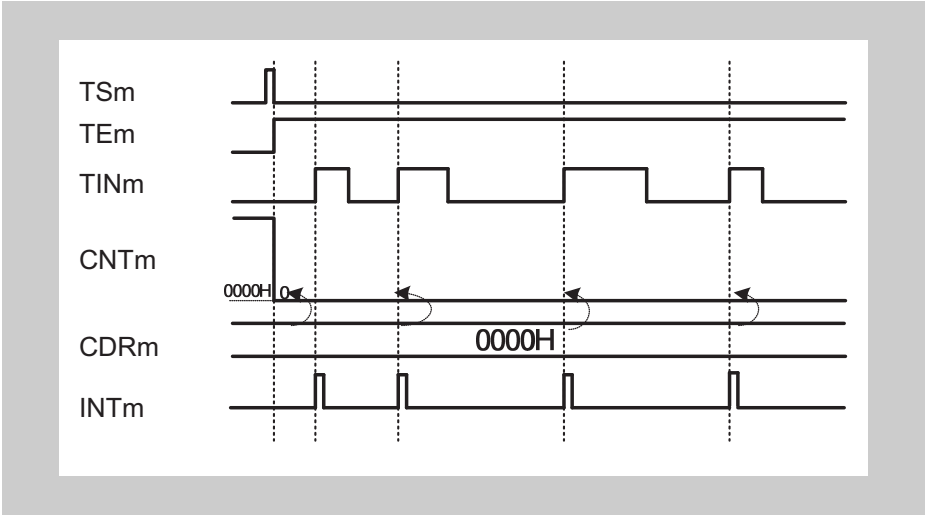


Figure 16-57 TAUBnCDRm = 0000H, TAUBnCMURm.TIS[1:0] = 01

- If 0000_H = TAUBnCDRm, 0000_H is written to TAUBnCNTm every time a valid TAUBnTTINm input edge is detected.

  This means, INTTAUBnIm is generated every time a valid TAUBnTTINm input edge is detected.

(b) Operation stop and restart

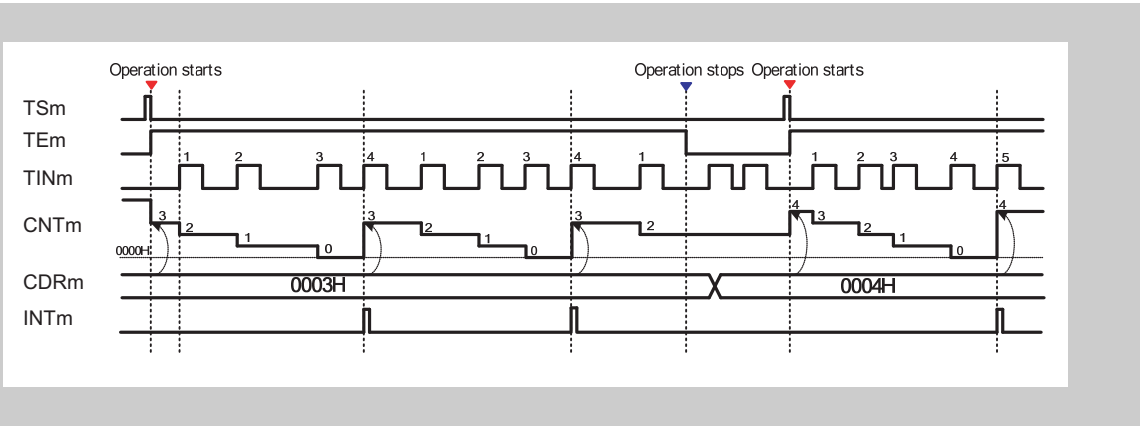


Figure 16-58 Operation stop and restart, TAUBnCMURm.TIS[1:0] = 01

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained. TAUBnTTINm continues and TAUBnCNTm ignores the valid edge.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm loads the TAUBnCDRm value and restarts count operation.

(c) Forced restart

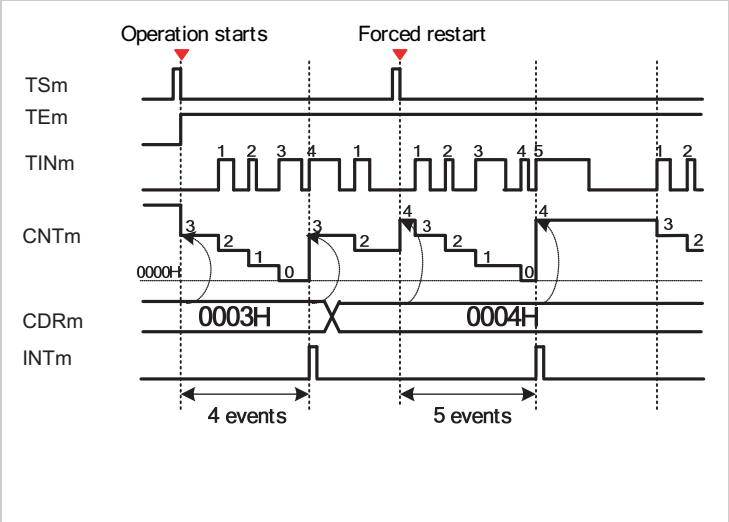


Figure 16-59 Forced restart, TAUBnCMURm.TIS[1:0] = 01

A forced restart applies a change to TAUBnCDRm immediately.

- The counter can be restarted (without stopping it first), by setting TAUBnTS.TSm to 1 during operation.
- The value of TAUBnCDRm is written to TAUBnCNTm and the counter awaits the next valid TAUBnTTINm input edge.

16.17.2 Clock Divide Function

(1) Overview

Summary This function is used as a frequency divider. The frequency of the input signal TAUBnTTINm is divided by a factor related to TAUBnCDRm, and the resulting signal is output to TAUBnTTOUTm.

Prerequisites

- TAUBnTTINm must have a fixed frequency
- The operation mode must be set to Interval Timer Mode, refer to *Table 16-67 "TAUBnCMORm settings for Clock Divide Function" on page 1214*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *16.9 "Channel Output Modes" on page 1122*

Description The counter is started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value, using TAUBnTTINm as the count clock.

When the counter value reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and subsequently continues operation.

The value of TAUBnCDRm can be rewritten at any time, and the changed value of TAUBnCDRm is applied the next time the function starts to count down.

The counter can be stopped by setting TAUBnTT.TTm = 1, which in turn sets TAUBnTE.TEm = 0. TAUBnCNTm and TAUBnTTOUTm stop but retain their values. The function can be restarted by setting TAUBnTS.TSm = 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUBnTTOUTm does not toggle. This results in an inverted TAUBnTTOUTm signal compared to when TAUBnCMORm.MD0 is set to 1. For details refer to *16.11 "TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)" on page 1131* .

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

(2) Equations

- When rising edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / [(TAUBnCDRm + 1) x 2]
- When falling edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / [(TAUBnCDRm + 1) x 2]
- When rising and falling edge detection is selected:
  TAUBnTTOUTm frequency = TAUBnTTINm frequency / (TAUBnCDRm + 1)

(3) Block diagram and general timing diagram

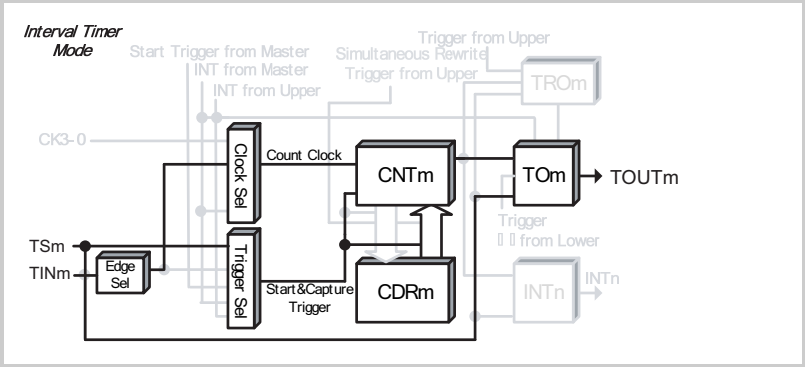


Figure 16-60 Block diagram for Clock Divide Function

The following settings apply to the general timing diagram:

- INTTAUBnIm generated at operation start (TAUBnCMORm.MD0 = 1)
- Rising edge detection (TAUBnCMURm.TIS[1:0] = 01_B)

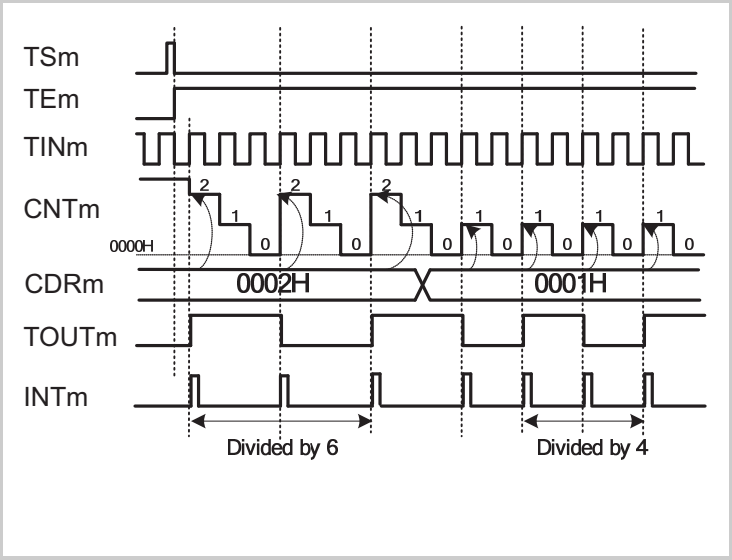


Figure 16-61 General timing diagram for Clock Divide Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-67 TAUBnCMORm settings for Clock Divide Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	1: Valid TAUBnTTINm input edge is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-68 TAUBnCMURm settings for Clock Divide Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 16-69 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic
TDE.TDEm	0: Disables dead time operation
TDL.TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUBnTOE.TOEm = 0. TAUBnTTOUTm can then be controlled independently of the interrupts. For details refer to *Table 16-11 “Channel output modes” on page 1123* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the Clock Divide Function. Therefore, these registers must be set to 0.

Table 16-70 Simultaneous rewrite settings for Clock Divide function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDM.RDMm	
RDC.RDCm	

(5) Operating procedure for Clock Divide Function

Table 16-71 Operating procedure for Clock Divide Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-67 "TAUBnCMORm settings for Clock Divide Function" on page 1214* and *Table 16-68 "TAUBnCMURm settings for Clock Divide Function" on page 1214* Set the value of the TAUBnCDRm register Set the channel output mode by setting the control bits as described in *Table 16-69 "Control bit settings for Independent Channel Output Mode 1" on page 1215*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. TAUBnCNTm loads the TAUBnCDRm value. When TAUBnCMORm.MD0 is set to 1, INTTAUBnIm is generated and TAUBnTTOUTm toggles.
During operation	The value of TAUBnCDRm can be changed at any time. The TAUBnCNTm register can be read at all times.	When a TAUBnTTINm input edge is detected, TAUBnCNTm counts down. When the counter reaches 0000_H: • TAUBnCNTm reloads the TAUBnCDRm value and continues count operation • INTTAUBnIm is generated • TAUBnTTOUTm toggles. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnTTOUTm retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm = 0000_H

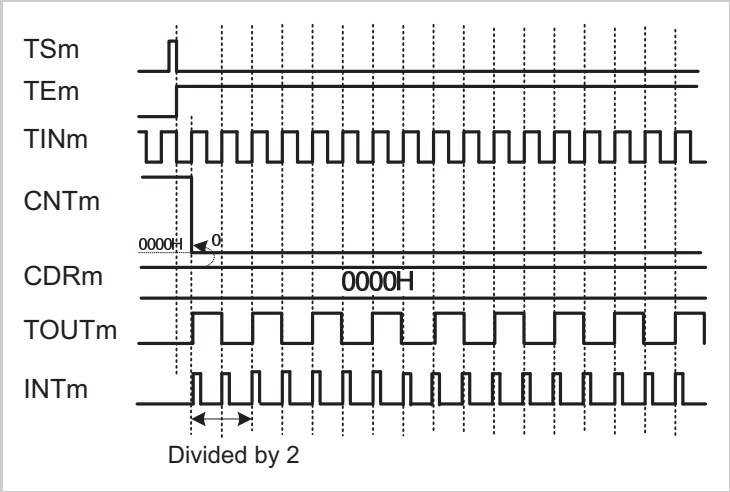


Figure 16-62 TAUBnCDRm = 0000_H, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

- If TAUBnCDRm is 0000_H, TAUBnCNTm is also always 0000_H.
- INTTAUBnIm is generated every count clock, resulting in TAUBnTTOUTm toggling every count clock.

(b) Restart

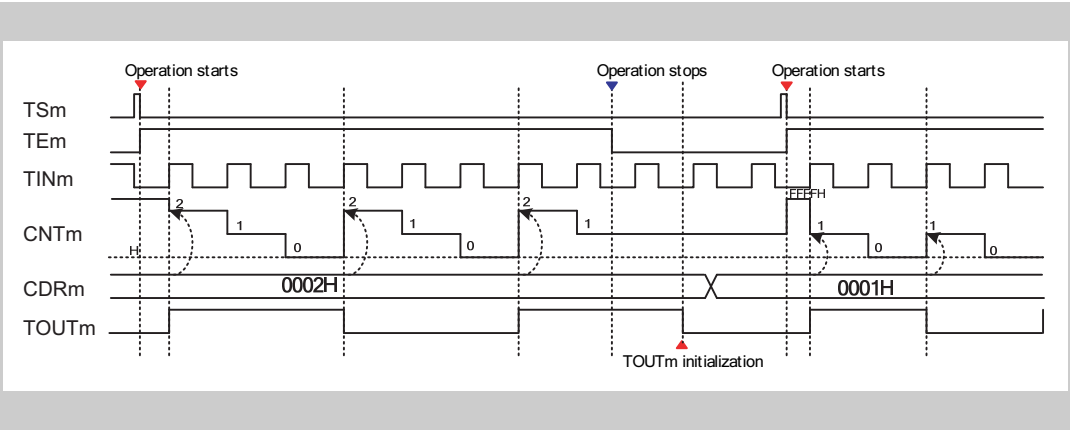


Figure 16-63 Restart, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

To reset the value of TAUBnTTOUTm:

- Set TAUBnTOE.TOEm = 0 when the counter is stopped (TAUBnTE.TEm = 0)
- Then write either 0 or 1 to TAUBnTO.TOm to set the new start value of TAUBnTTOUTm

(c) Forced restart

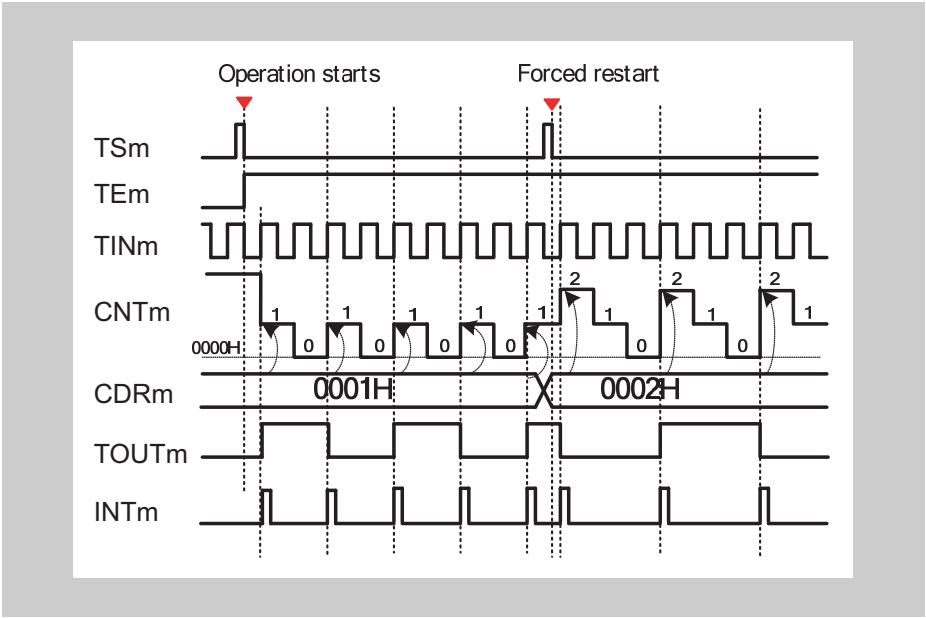


Figure 16-64 Forced restart, TAUBnCMORm.MD0 = 1, TAUBnCMURm.TIS[1:0] = 01

- The counter can be forcibly restarted (without stopping it first) by setting TAUBnTS.TSm = 1 during operation.
- The value of TAUBnCDRm is written to TAUBnCNTm and the count operation restarts.
- TAUBnTTOUTm restarts at the same level as before the forced restart.

16.17.3 TAUBnTTINm Input Position Detection Function

(1) Overview

Summary This function measures the duration between the function start and a TAUBnTTINm input signal.

Prerequisites
- The operation mode must be set to Count Capture Mode, refer to *Table 16-72 "TAUBnCMORm settings for TAUBnTTINm Input Position Detection Function" on page 1221*
- TAUBnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The counter starts to count from 0000_H. When a valid TAUBnTTINm input stop edge is detected, the current TAUBnCNTm value is written to TAUBnCDRm and an interrupt (INTTAUBnIm) is generated. The counter continues to count from the current value until the next valid TAUBnTTINm input edge is detected.

When the counter reaches $FFFF_H$, the bit TAUBnCSRm.OVF is set to 1 and the counter restarts from 0000_H. The value of TAUBnCSRm.OVF is reset by the CPU by TAUBnCSCm.CLOV = 1.

Note The input TAUBnTTINm is sampled at the frequency of the operation clock, specified by the TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions If the TAUBnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *16.11 "TAUBnTTOUTm toggle and INTTAUBnIm Generation when Counter start is triggered (MD0-bit)" on page 1131* .

(2) Equations

Function duration at a TAUBnTTINm input pulse =

count clock cycle × [($FFFF_H$+1 × TAUBnCSRm.OVF) + (TAUBnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

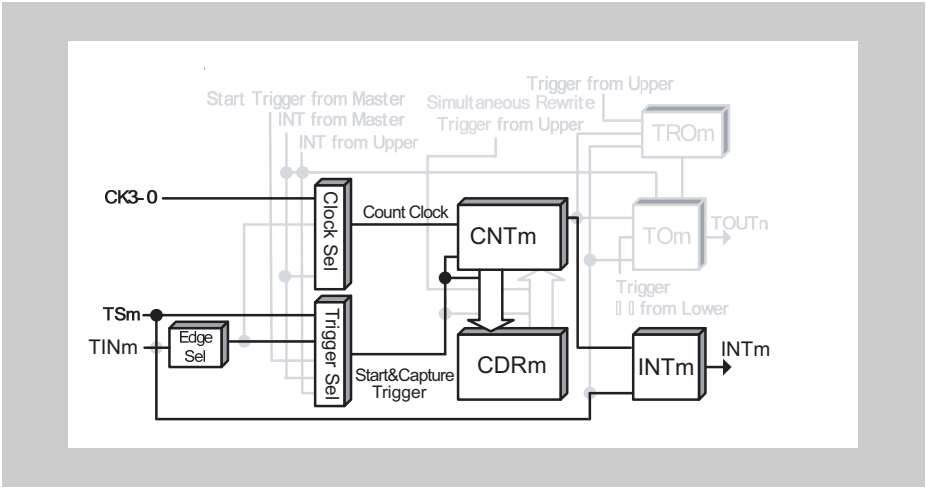


Figure 16-65 Block diagram for TAUBnTTINm Input Position Detection Function

The following settings apply to the general timing diagram:

- INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

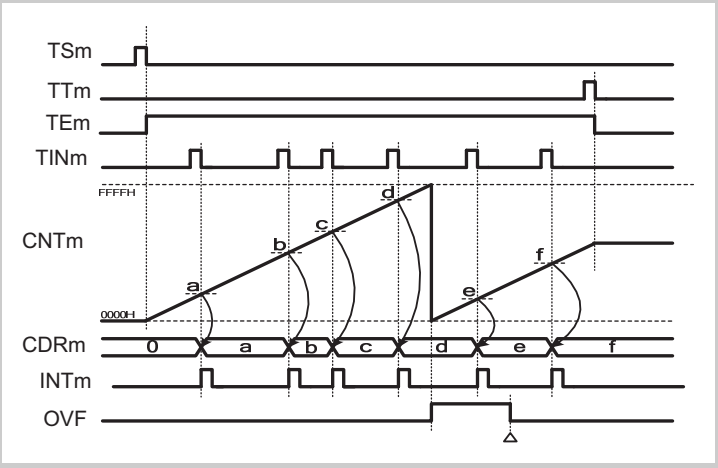


Figure 16-66 General timing diagram for TAUBnTTINm Input Position Detection Function

(4) Register settings

(a) TAUBnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-72 TAUBnCMORm settings for TAUBnTTINm Input Position Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS0	0: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the external capture trigger
COS[1:0]	01: Overflow (TAUBnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1011: Count Capture Mode
MD0	0: INTTAUBnIm not generated at operation start 1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-73 TAUBnCMURm settings for TAUBnTTINm Input Position Detection Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUBnRDE, TAUBnRDS, TAUBnRDM, and TAUBnRDC) cannot be used with the TAUBnTTINm Input Position Detection Function. Therefore, these registers must be set to 0.

Table 16-74 Simultaneous rewrite settings for TAUBnTTINm Input Position Detection Function

Bit name	Setting
RDEm	0: Disables simultaneous rewrite
RDSm	0: When simultaneous rewrite is disabled (TAUBnRDE.RDEm = 0), set these bits to 0
RDMm	
RDCm	

(5) Operating procedure for TAUBnTTINm Input Position Detection Function

Table 16-75 Operating procedure for TAUBnTTINm Input Position Detection Function

	Operation	Status of TAUBn
Initial channel setting	Set the TAUBnCMORm register and TAUBnCMURm registers as described in *Table 16-72 "TAUBnCMORm settings for TAUBnTTINm Input Position Detection Function" on page 1221* and *Table 16-73 "TAUBnCMURm settings for TAUBnTTINm Input Position Detection Function" on page 1221* Set the value of the TAUBnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm to 1. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is set to 1 and the counter starts. INTTAUBnIm is generated when TAUBnCMORm.MD0 is set to 1.
During operation	The TAUBnCMURm.TIS[1:0] bits can be changed at any time. The TAUBnCDRm and TAUBnCSRm registers can be read at any time.	TAUBnCNTm starts to count up from 0000_H. When a TAUBnTTINm valid edge is detected: • TAUBnCNTm transfers (captures) its value to TAUBnCDRm • TAUBnTTINm is output. • The counter value is not cleared to 0000_H and TAUBnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUBnTT.TTm to 1. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm stops and both it and TAUBnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

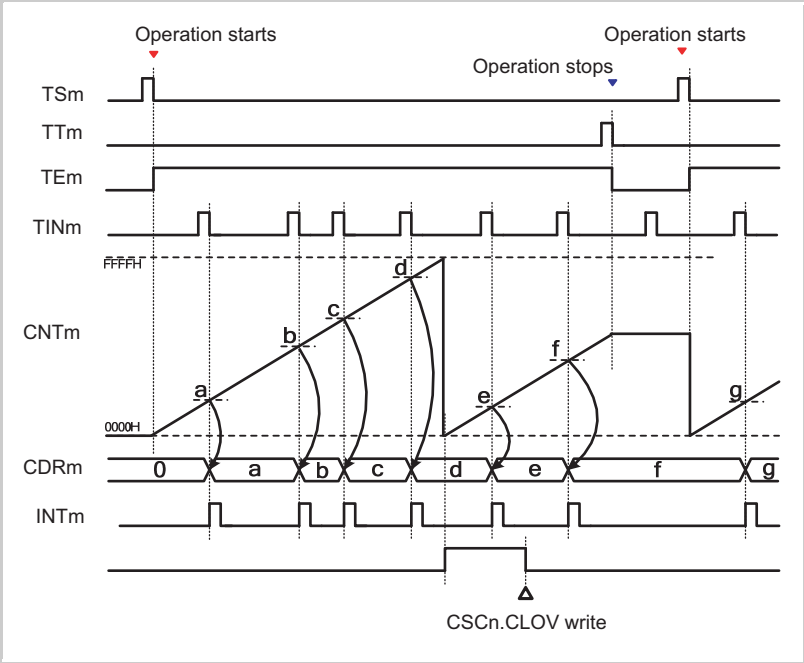


Figure 16-67 Operation stop and restart, TAUBnCMORm.MD0 = 0, TAUBnCMURm.TIS[1:0] = 00

- The counter can be stopped by setting TAUBnTT.TTm to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUBnTTINm input edges are ignored.
- The counter can be restarted by setting TAUBnTS.TSm to 1. TAUBnCNTm restarts to count from 0000_H.

16.18 Synchronous Channel Operation Functions

This section lists all the synchronous channel operation functions provided by the Timer Array Unit B. For a general overview of synchronous channel operation, see *16.4 “Functional Description” on page 1105* .

16.19 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes functions that generate PWM signals at regular intervals, that can be reset by a TAUBnTTINm input, with and without dead time.

- *16.19.1 “PWM Output Function”*
- *16.19.2 “Delay Pulse Output Function”*
- *16.19.3 “AD Conversion Trigger Output Function Type 1”*

16.19.1 PWM Output Function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUBnTTOUTm to be set. The pulse cycle is set in the master channel. The pulse width is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 16-76 “TAUBnCMORm settings for the master channel of the PWM Output Function” on page 1229*
- The operation mode of the slave channel(s) must be set to One Count Mode, refer to *Table 16-79 “TAUBnCMORm settings for the slave channel of the PWM Output Function” on page 1231*
- TAUBnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 1 (*16.9 “Channel Output Modes” on page 1122*)

Description The counters are started by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm = 1, enabling count operation. The current value of TAUBnCDRm is written to TAUBnCNTm and the counters start to count down from these values. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave) toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUBnIm is generated. The counter reloads the TAUBnCDRm value and counts down.

- Slave channel(s)

  The INTTAUBnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUBnCDRm (slave) is written to TAUBnCNTm (slave) and the counter starts to count down from this value. The TAUBnTTOUTm signal is set.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

Note If a forced restart is executed during operation, the counter value becomes invalid and TAUBnTTOUTm is not output as expected PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *16.8 “Simultaneous Rewrite” on page 1112*

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUBnCDRm (slave) / (TAUBnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

  TAUBnCDRm (slave) = 0000_H

– Duty cycle = 100 %

  TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

(3) Block diagram and general timing diagram

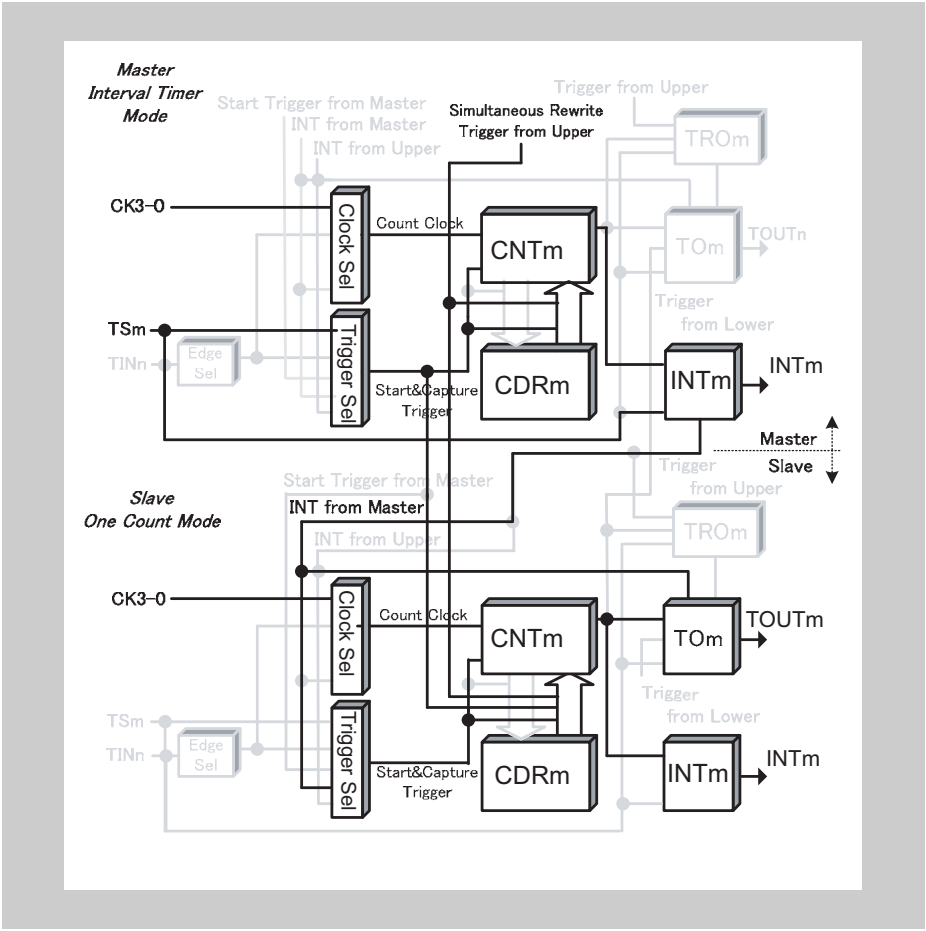


Figure 16-68 Block diagram for PWM Output Function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUBnTOL.TOLm = 0)

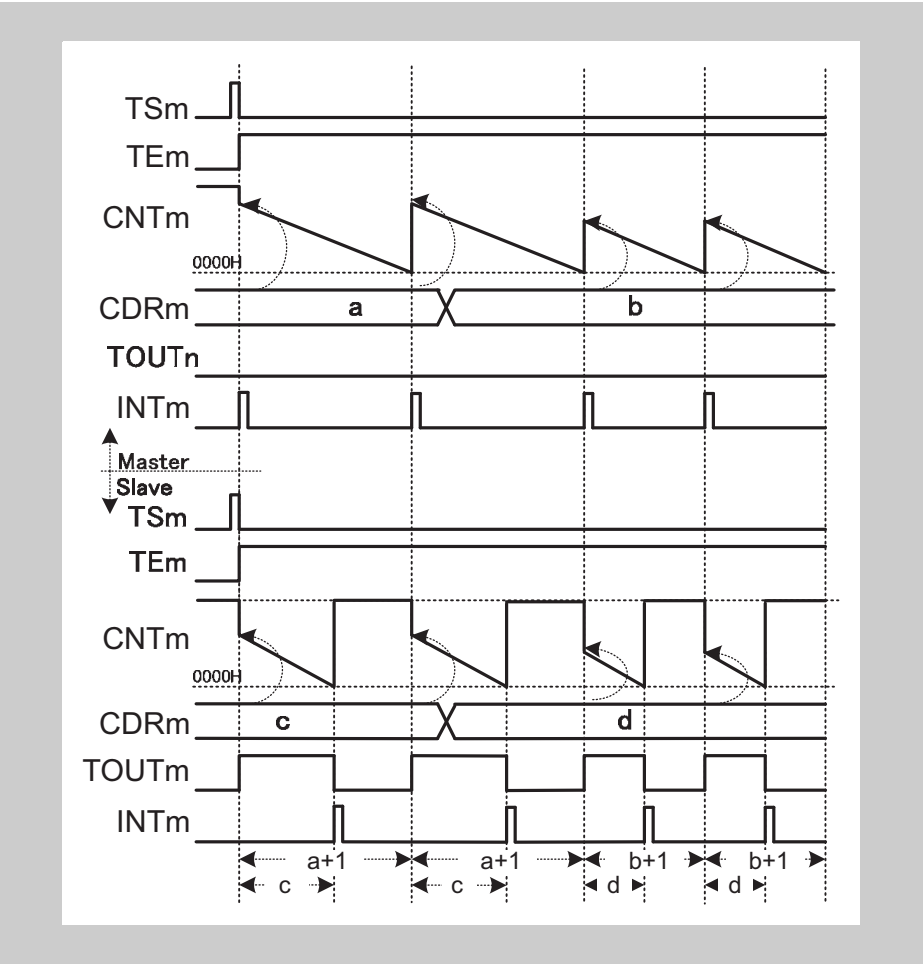


Figure 16-69 General timing diagram for PWM Output Function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUBnCDRm, whereas for the master channel the interval is the corresponding TAUBnCDRm + 1.

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-76 TAUBnCMORm settings for the master channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-77 TAUBnCMURm settings for the master channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-78 Simultaneous rewrite settings for the master channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUBnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-79 TAUBnCMORm settings for the slave channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUBnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-80 TAUBnCMURm settings for the slave channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 16-81 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-82 Simultaneous rewrite settings for the slave channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for PWM Output Function

Table 16-83 Operating procedure for PWM Output Function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1229* Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 1231* Set the values of the TAUBnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave) is set.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel loads TAUBnCDRm and counts down. When the counter reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave) reloads the TAUBnCDRm value and counts down • TAUBnTTOUTm (slave) is set When TAUBnCNTm (slave) reaches 0000_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is reset
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

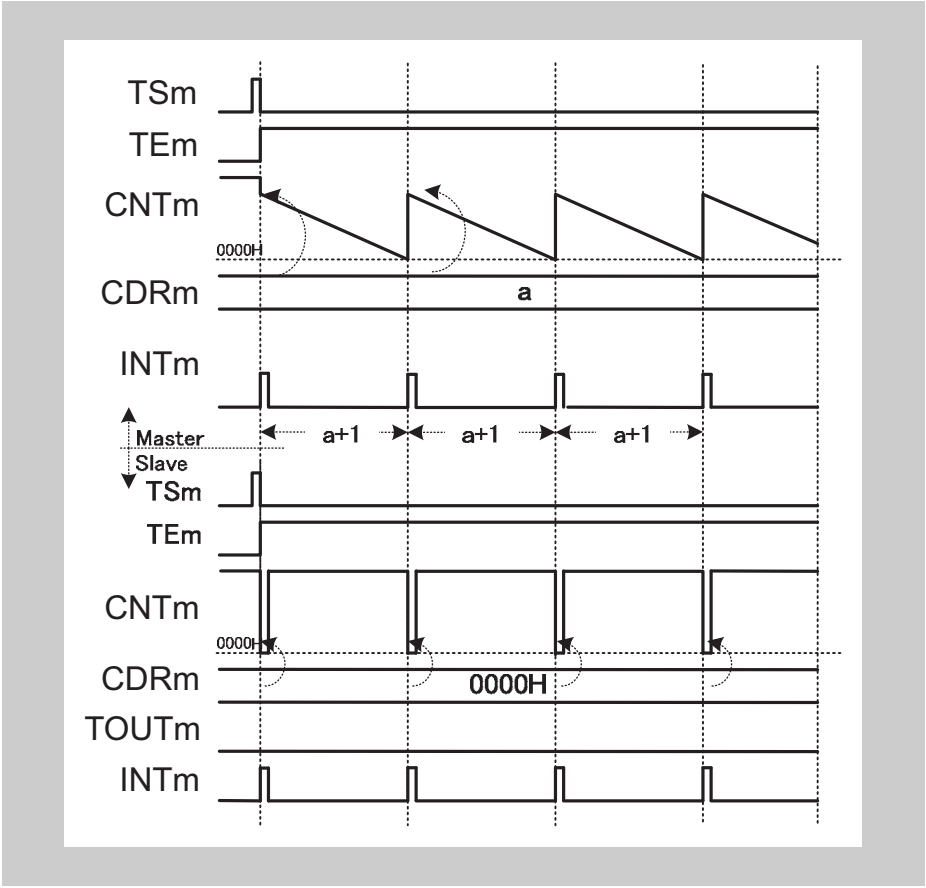


Figure 16-70 TAUBnCDRm (slave) = 0000_H, positive logic (TAUBnTOL.TOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUBnIm), 0000_H is written to TAUBnCNTm (slave). Therefore, TAUBnCNTm (slave) cannot start to count and TAUBnTTOUTm remains at not active state.
- TAUBnCNTm (slave) generates an interrupt every time the value of TAUBnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

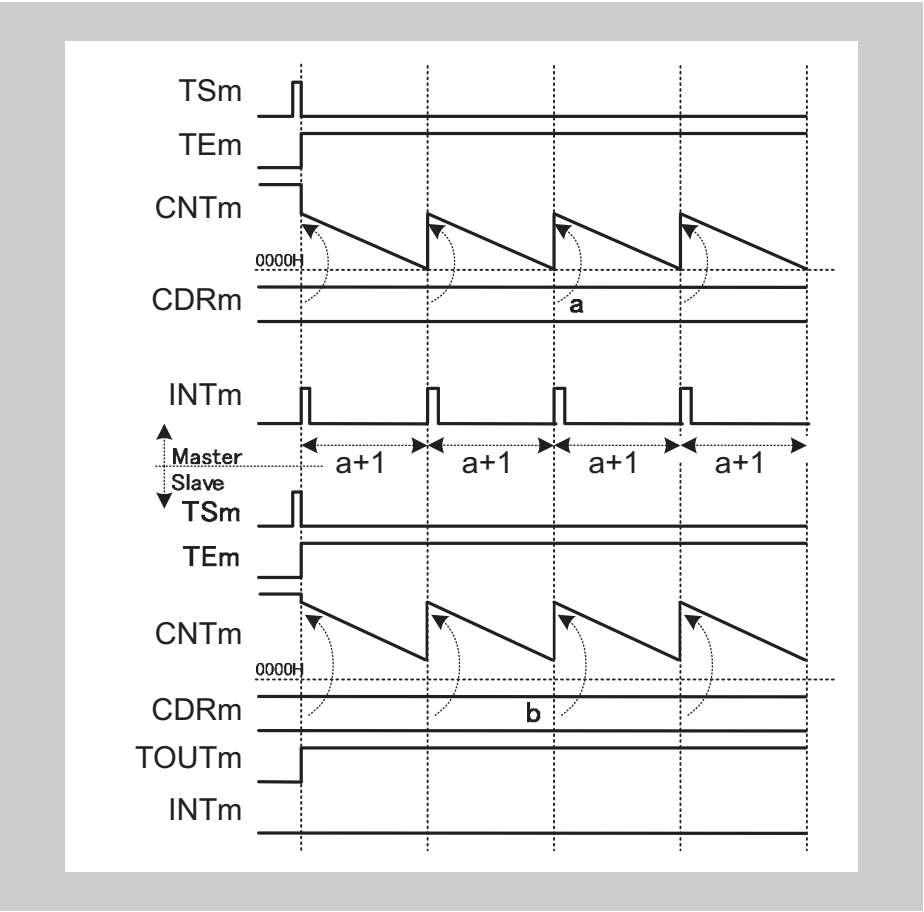


Figure 16-71 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1, positive logic (TAUBnTOL.TOLm (slave) = 0)

- If the value TAUBnCDRm (slave) is higher than the value TAUBnCDRm (master), the counter of the slave channel cannot reach 0000_H and cannot generate interrupts. The TAUBnTTOUTm remains at active state.

(c) Stop and restart operation

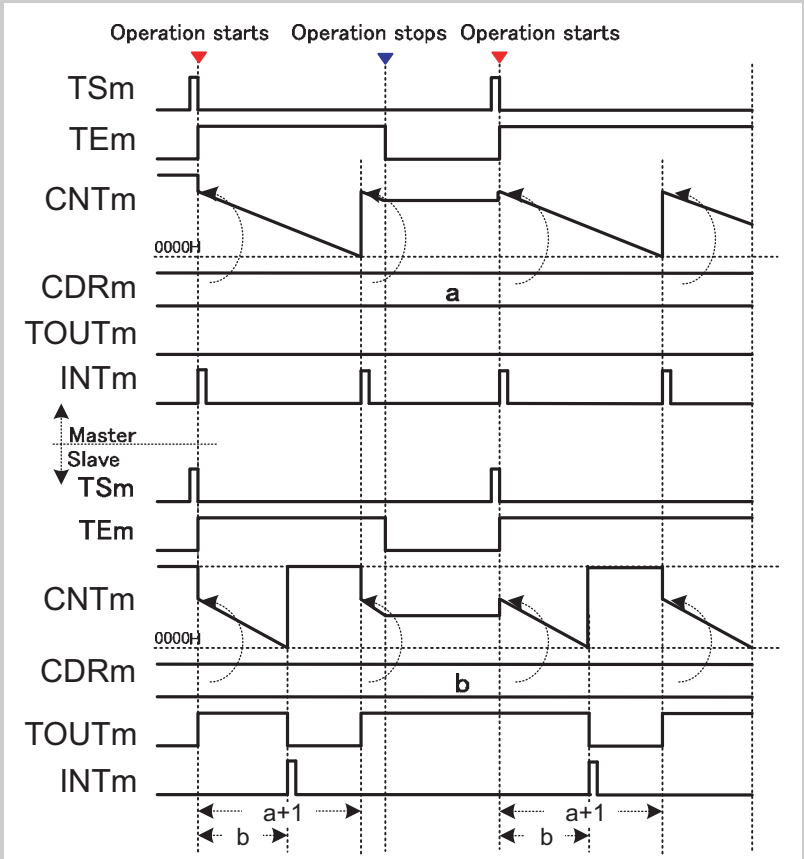


Figure 16-72 Stop and restart operation, positive logic (TAUBnTOL.TOLm (slave) = 0)

- The counter can be stopped by setting TAUBnTT.TTm of the master and slave channel(s) to 1, which in turn sets TAUBnTE.TEm to 0.
- TAUBnCNTm and TAUBnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUBnTS.TSm of master and slave channel(s) to 1. TAUBnCNTm of master and slave channel reload the current values of TAUBnCDRm and start to count down from these values.

16.19.2 Delay Pulse Output Function

(1) Overview

Summary

This function outputs two signals. The reference signal has a defined pulse width and pulse cycle specified using the master channel and slave channel 1. Slave channels 2 and 3 output the reference signal with a specified delay. The delay signal is identical to the reference signal, but delayed by amount specified in slave channel 2.

The signal values are specified in the following way:

- The pulse cycle is specified using the master channel.
- The duty cycle of the reference signal is specified using slave channel 1. The duty cycle of the delay signal is specified using slave channel 3.

  The values of TAUBnCDRm of these both channels have to be identical.
- The delay is specified in slave channel 2.

Prerequisites

- Four channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 16-84 “TAUBnCMORm settings for the master channel of the Delay Pulse Output Function” on page 1241*
- The operation mode of slave channel 1 and 2 must be set to One Count Mode, refer to *Table 16-87 “TAUBnCMORm settings for slave channel 1 of the Delay Pulse Output Function” on page 1243*
- The operation mode of slave channel 3 must be set to Pulse One Count Mode, refer to *Table 16-91 “TAUBnCMORm settings for slave channel 2 of the Delay Pulse Output Function” on page 1245*
- TAUBnTTOUTm is not used for the master channel and slave channel 2
- The channel output mode of slave channel 1 must be set to Synchronous Channel Output Mode 1 (refer to *16.9 “Channel Output Modes” on page 1122*)
- The channel output mode of slave channel 3 must be set to Independent Channel Output Mode 1 (refer to *16.9 “Channel Output Modes” on page 1122*)

Description

The counters of the channel group are started by setting the channel trigger bit (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation.

- Master channel:

  The current value of TAUBnCDRm is written to TAUBnCNTm and the counter starts to count down from this value. INTTAUBnIm is generated on the master channel.

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUBnIm is generated. The counter reloads the TAUBnCDRm value and counts down.
- Slave channels 1 and 2:

  When the slave channels 1 and 2 detect an interrupt from the master channel, they start to count down from the current value of TAUBnCDRm. The TAUBnTTOUTm signal (slave 1) is set.

– Slave channel 1:

When the counter of slave channel 1 reaches 0000_H, duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel.

– Slave channel 2:

When the counter of slave channel 2 reaches 0000_H, delay time has elapsed and INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of the master channel.

INTTAUBnIm (slave 2) triggers the counter of slave channel 3

- Slave channel 3:

When slave channel 3 detects an interrupt from slave channel 2, it starts to count down from the current value of TAUBnCDRm. INTTAUBnIm is generated and the TAUBnTTOUTm signal (slave 3) toggles.
When the counter of slave channel 3 reaches 0000_H, duty time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles again.
The output from slave channel 3 is the delayed PWM pulse

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channels, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channels stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

Conditions Simultaneous rewrite can be used with this function. Please refer to *16.8 “Simultaneous Rewrite” on page 1112*

Equations Pulse cycle = (TAUBnCDRm (master) + 1) × count clock cycle

Duty cycle = (TAUBnCDRm (slave 1 and 3)) × count clock cycle

Delay = (TAUBnCDRm (slave 2) + 1) × count clock cycle

(2) Block diagram and general timing diagram

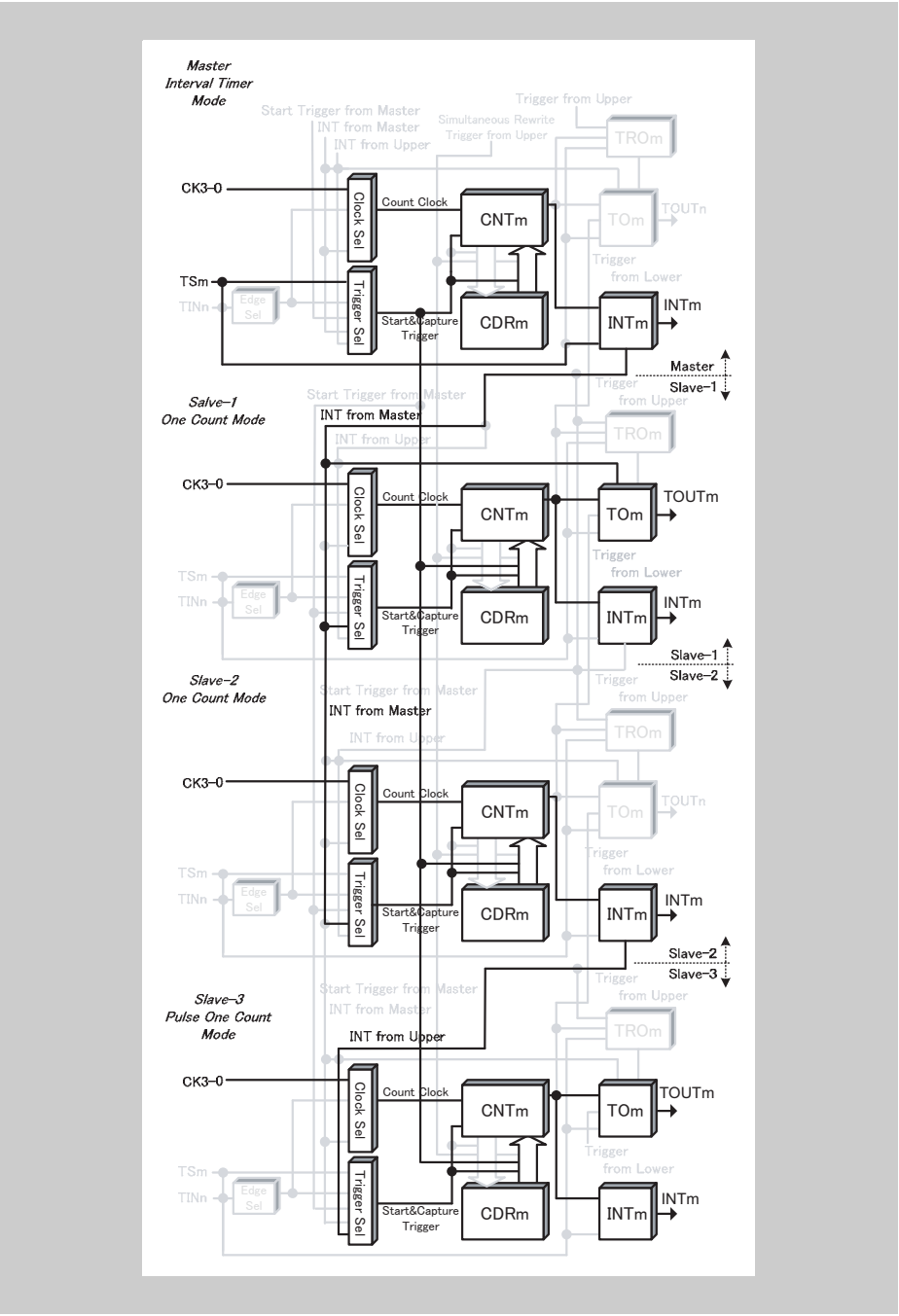


Figure 16-73 Block diagram for Delay Pulse Output Function

The following settings apply to the general timing diagram:

- All channels
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

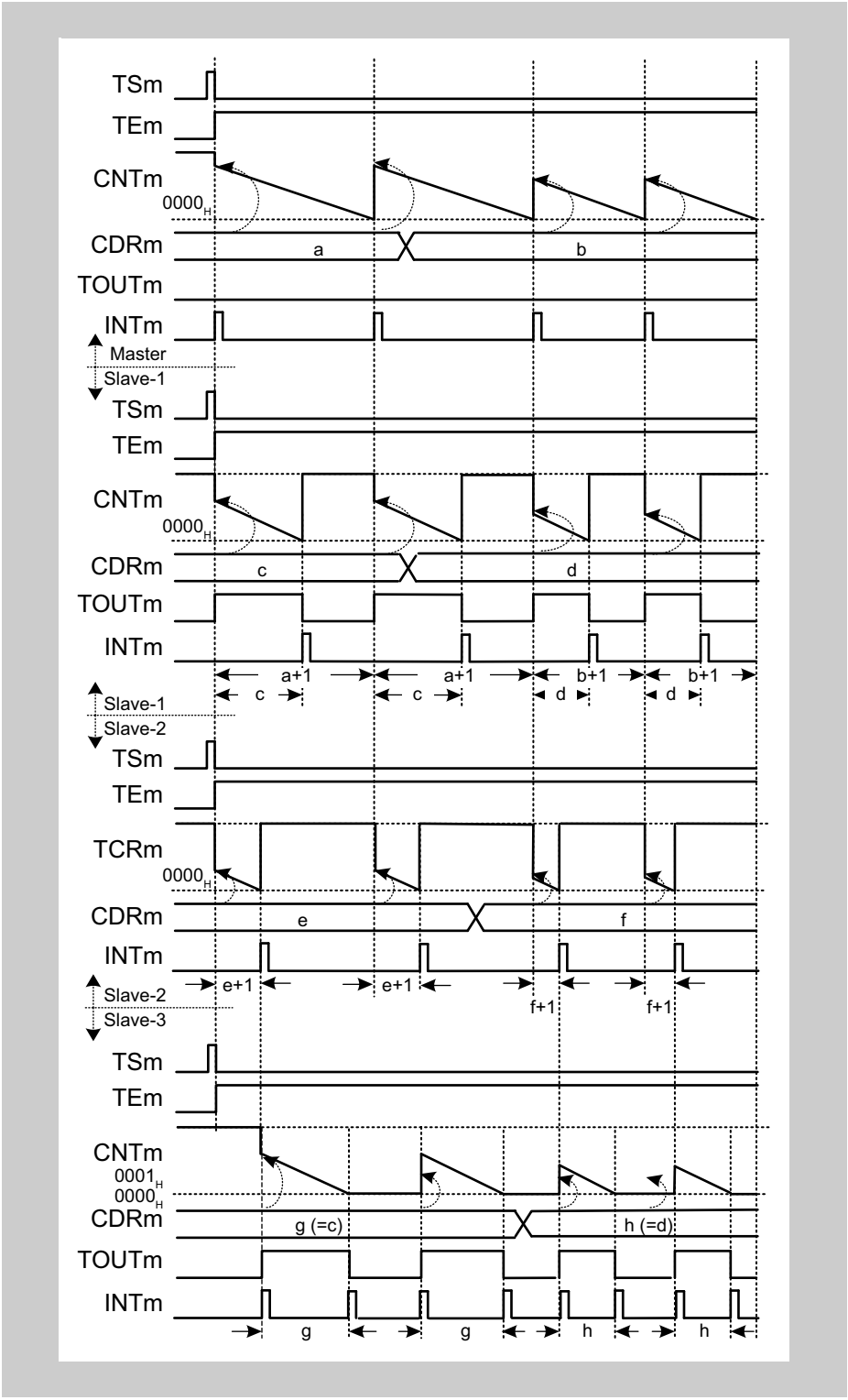


Figure 16-74 General timing diagram for Delay Pulse Output Function

(3) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-84 TAUBnCMORm settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUBnIm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-85 TAUBnCMURm settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by the master channel of this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-86 Simultaneous rewrite settings for the master channel of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for slave channel 1

(a) TAUBnCMORm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-87 TAUBnCMORm settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start (and enables start trigger during counting).

(b) TAUBnCMURm for slave channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-88 TAUBnCMURm settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 1

Table 16-89 Control bit settings for slave channel 1 of the Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for slave channel 1

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-90 Simultaneous rewrite settings for slave channel 1 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUBnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-91 TAUBnCMORm settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUBnIm at operation start (and enables start trigger during counting)

(b) TAUBnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-92 TAUBnCMURm settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-93 Simultaneous rewrite settings for slave channel 2 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUBnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-94 TAUBnCMORm settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	101: INTTAUBnIm of the upper channel (m-1) is the start trigger, regardless of the master setting
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start (and enables start trigger during counting)

(b) TAUBnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-95 TAUBnCMURm settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 16-96 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-97 Simultaneous rewrite settings for slave channel 3 of the Delay Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Delay Pulse Output Function

Table 16-98 Operating procedure for Delay Pulse Output Function (1/2)

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *3 "Register settings for the master channel" on page 1241* Slave channel 1: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *4 "Register settings for slave channel 1" on page 1243* Slave channel 2: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 2" on page 1245* Slave channel 3: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *6 "Register settings for slave channel 3" on page 1247* Set the values of the TAUBnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 16-98 Operating procedure for Delay Pulse Output Function (2/2)

	Operation	Status of TAUBn
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master channel and slave channels 1 and 2 start. INTTAUBnIm is generated on the master channel and TAUBnTTOUTm (slave 1) is set.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel and slave channels 1 and 2 load TAUBnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave 1 and slave 2) reload the TAUBnCDRm value and start counting down • TAUBnTTOUTm (slave 1) is set When TAUBnCNTm (slave 1) reaches 0000_H: • INTTAUBnIm (slave 1) is generated The output from slave channel 1 acts as the reference pulse • TAUBnTTOUTm (slave 1) is reset When TAUBnCNTm (slave 2) reaches 0000_H: • INTTAUBnIm (slave 2) is generated • TAUBnTTOUTm (slave 3) toggles • TAUBnCNTm (slave 3) reloads the TAUBnCDRm value and starts counting down When TAUBnCNTm (slave 3) reaches 0000_H: • INTTAUBnIm (slave 3) is generated • TAUBnTTOUTm (slave 3) toggles The output from slave channel 3 is the delayed PWM pulse
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle (slave 3) = 100 %

The following values apply to the figure below:

- TAUBnCDRm (master) = $000A_H$
- TAUBnCDRm (slave 1) = $000B_H$
- TAUBnCDRm (slave 2) = 0000_H
- TAUBnCDRm (slave 3) = $000B_H$

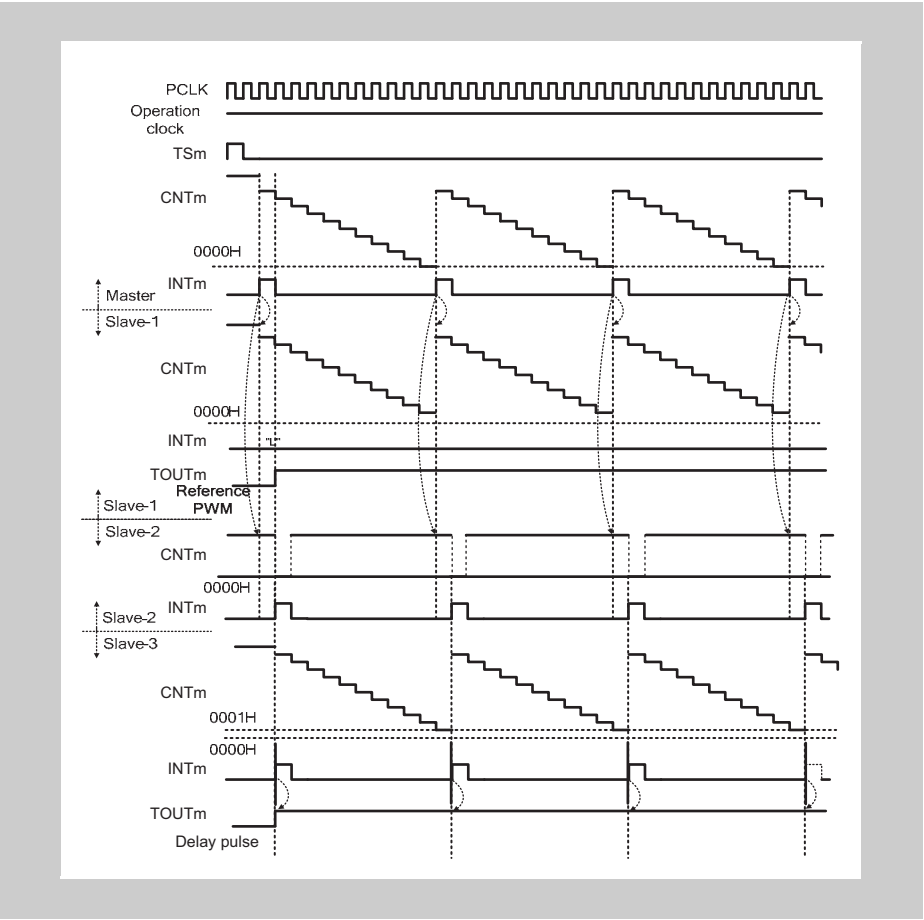


Figure 16-75 Duty cycle (slave 3) = 100 %

- If the value of TAUBnCDRm (slave 1 and 3) is higher than the value of TAUBnCDRm (master), the counter of the slave channels cannot reach 0000_H and cannot generate interrupts. TAUBnTTOUTm of channels 1 and 3 remain in the active state.

(b) TAUBnTTOUTm (slave 1) = TAUBnTTOUTm (slave 3)

The following values apply to the figure below:

- TAUBnCDRm (master) = $000A_H$
- TAUBnCDRm (slave 1) = 0005_H
- TAUBnCDRm (slave 2) = 0000_H
- TAUBnCDRm (slave 3) = 0005_H

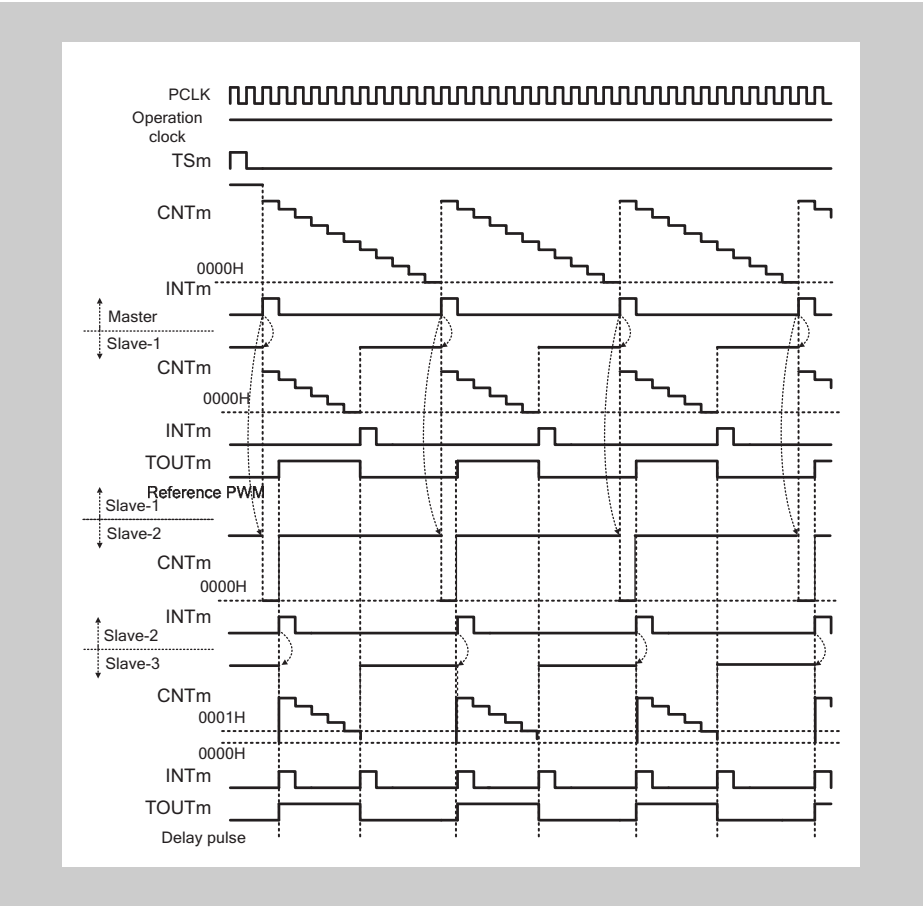


Figure 16-76 TAUBnTTOUTm (slave 1) = TAUBnTTOUTm (slave 3)

- If TAUBnCDRm (slave 2) = 0000_H, the counter of slave channel 3 starts counting one count clock later than the counter of slave channel 1. The reference pulse and the delay pulse are output with a delay of one clock count.

16.19.3 AD Conversion Trigger Output Function Type 1

(1) Overview

Summary This function is identical to *16.19.1 “PWM Output Function” on page 1226* except that TAUBnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to Direct Channel Output Mode.

(2) Block diagram and general timing diagram

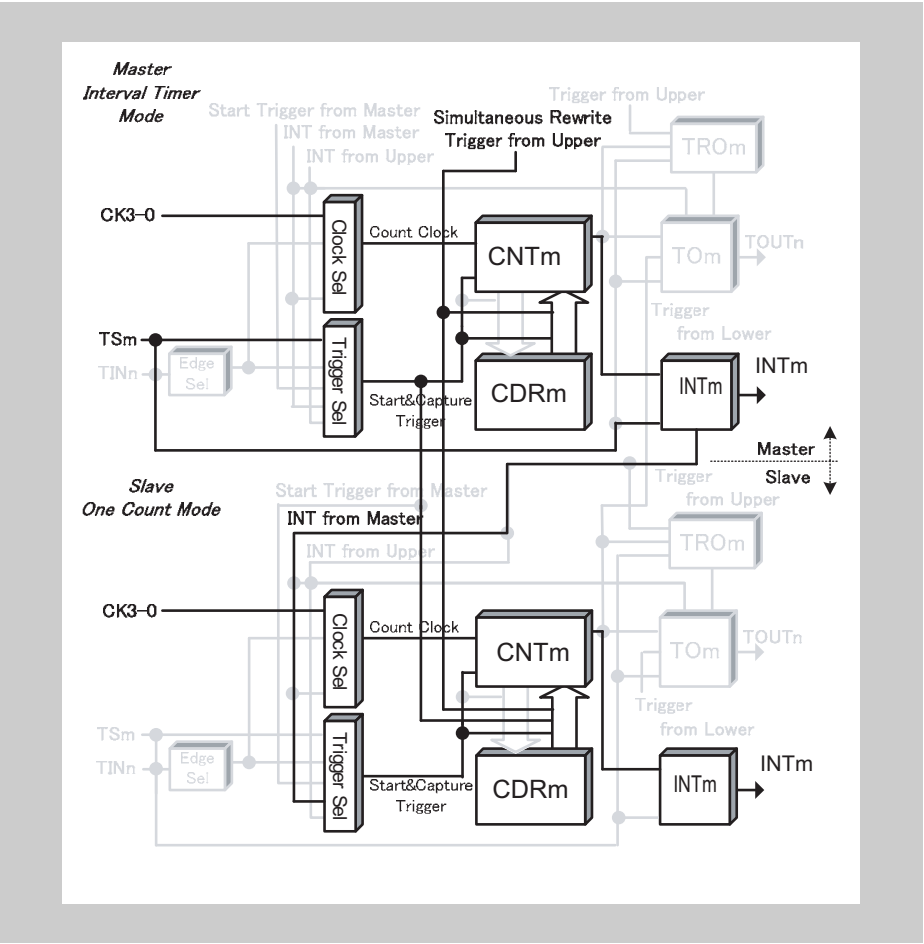


Figure 16-77 Block diagram for AD Conversion Trigger Output Function Type 1

(3) General timing diagram

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUBnTOL.TOLm = 0)

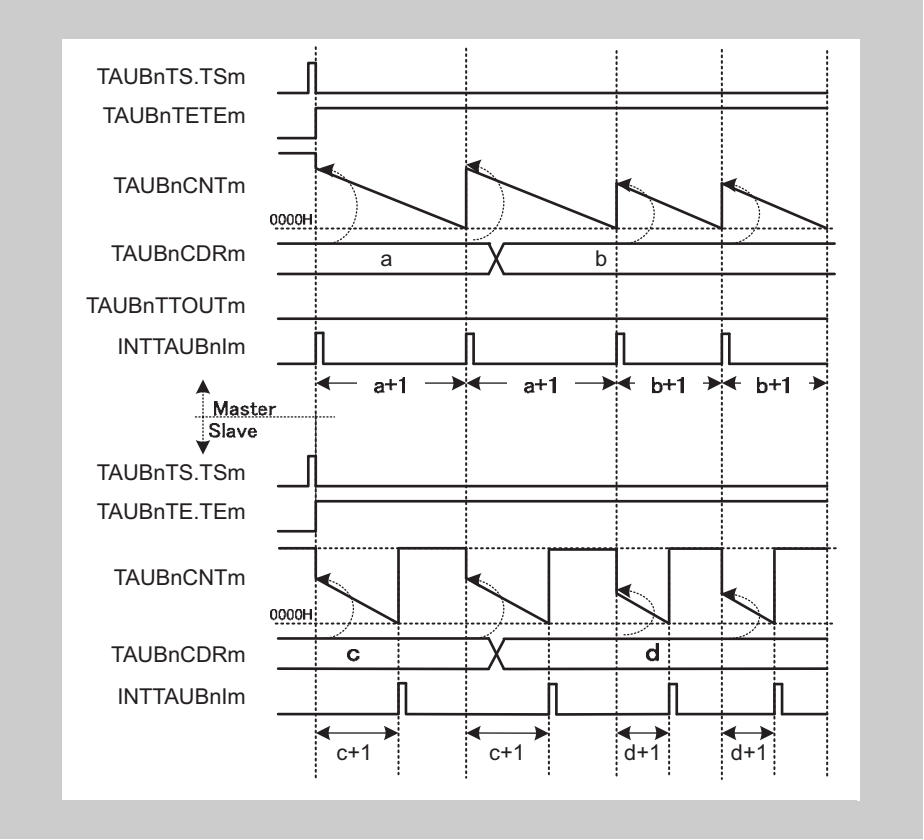


Figure 16-78 General timing diagram for AD Conversion Trigger Output Function Type 1

16.20 Synchronous PWM Signal Functions Triggered by an External Signal

This chapter describes functions that generate PWM signals and which are triggered by an external signal.

- *16.20.1 “One-Shot Pulse Output Function”*

16.20.1 One-Shot Pulse Output Function

(1) Overview

Summary This function outputs a signal pulse with a defined pulse width and a specific delay time compared to an external input signal pulse by using a master and a slave channel. The delay time is specified using the master channel. The pulse width is specified using the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to One Count Mode, refer to *Table 16-99 “TAUBnCMORm settings for the master channel of the One-Shot Pulse Output Function” on page 1259*
- The operation mode of the slave channel must be set to Pulse One Count Mode, refer to *Table 16-102 “TAUBnCMORm settings for the slave channel of the One-Shot Pulse Output Function” on page 1261*
- TAUBnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel must be set to Synchronous Channel Output Mode 2 (refer to *16.9 “Channel Output Modes” on page 1122*)
- TAUBnTTINm (master) has to be detected while TAUBnCNTm (master) and TAUBnCNTm (slave) await a trigger. Furthermore, the slave is only triggered by an interrupt from the master channel and not by TAUBnTTINm.

Description The counters are enabled by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation.

- Master channel:

  When the next valid TAUBnTTINm input edge is detected, the current value of TAUBnCDRm is written to TAUBnCNTm. The counter starts to count down from this value. If TAUBnCMORm.MD0 = 0, a trigger (TAUBnTTINm) which is detected within the delay time is ignored.
  When the counter of the master channel reaches 0000_H, INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next valid TAUBnTTINm input edge.

- Slave channel

  The INTTAUBnIm of the master channel triggers the counter of the slave channel. The current value of TAUBnCDRm (slave) is written to TAUBnCNTm (slave) and the counter starts to count down from this value. An interrupt is generated and the TAUBnTTOUTm signal is set.
  When the counter reaches 0001_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal is reset. The counter remains at 0001_H and awaits the next INTTAUBnIm of the master channel.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel, which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel stop but retain their values. The counters can be restarted by setting TAUBnTS.TSm to 1.

The counter of the master channel can be restarted without stopping it first (forced restart) by setting TAUBnTS.TSm to 1 during operation.

Notes

1. If a forced restart of the slave channel is executed during operation, the width of the output signal does not correspond to the value of TAUBnCDRm (slave).
2. The input TAUBnTTINm is sampled at the frequency of the operating clock, specified by TAUBnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUBnTTOUTm has an error of ± 1 operation clock cycle.

Conditions

- If TAUBnCMORn.MD0 of the master channel is set to 0, during counting detected TAUBnTTINm input edges are ignored.
- Simultaneous rewrite can be used with this function. Please refer to *16.8 "Simultaneous Rewrite" on page 1112*

Equations

Delay to input pulse = (TAUBnCDRm (master) + 1) × count clock cycle

Pulse width = (TAUBnCDRm (slave)) × count clock cycle

(2) Block diagram and general timing diagram

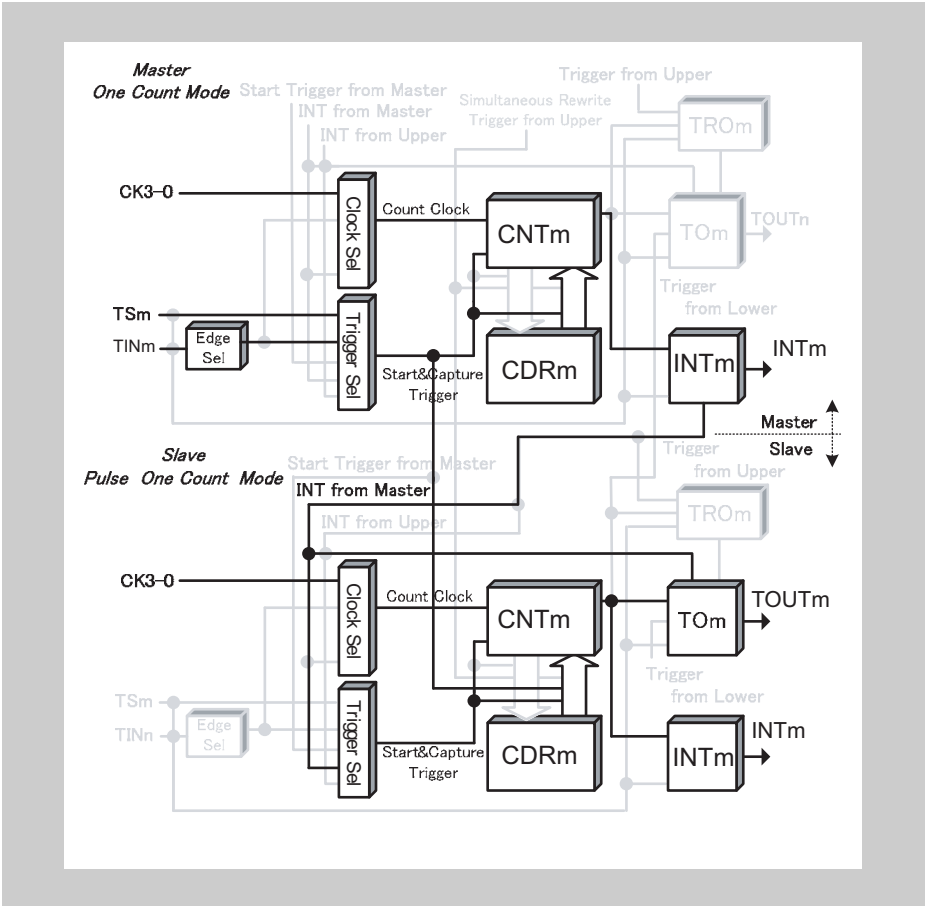


Figure 16-79 Block diagram for One-Shot Pulse Output Function

The following settings apply to the general timing diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

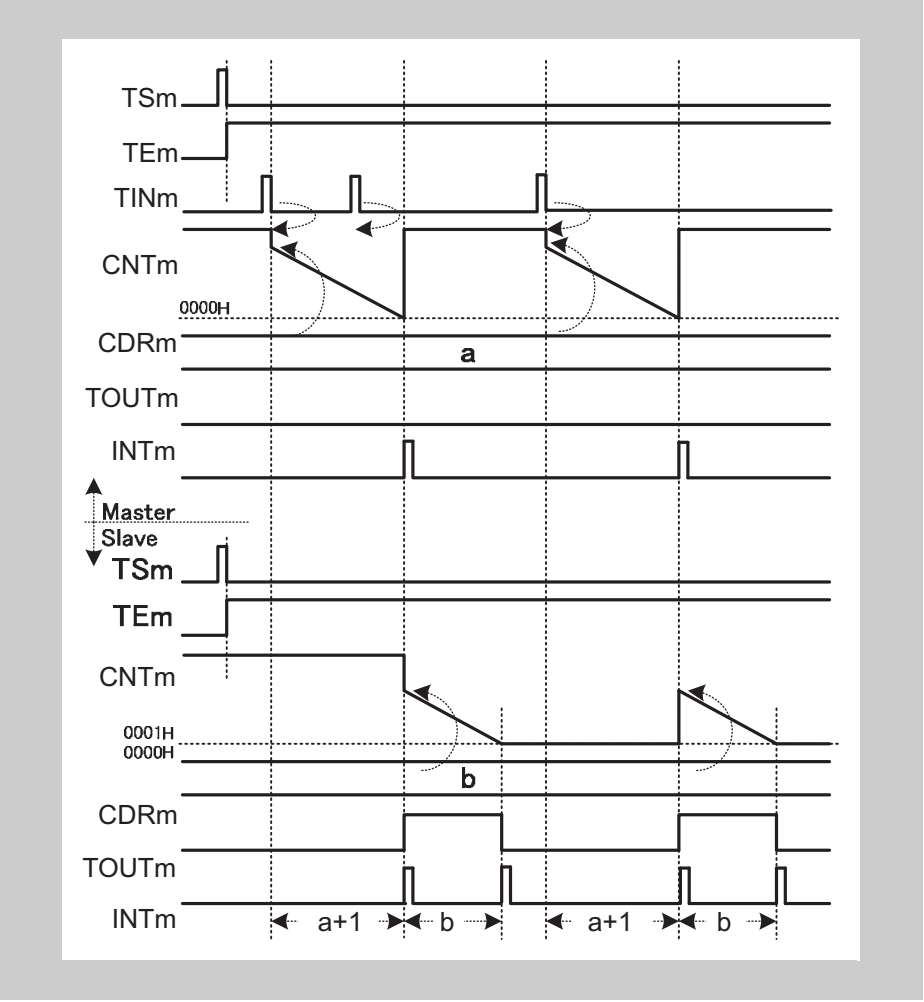


Figure 16-80 General timing diagram for One-Shot Pulse Output Function

(3) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-99 TAUBnCMORm settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	001: Valid TAUBnTTINm input edge signal is used as the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-100 TAUBnCMURm settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection 11: Setting prohibited

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-101 Simultaneous rewrite settings for the master channel of the One-Shot Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(4) Register settings for the slave channel

(a) TAUBnCMORm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-102 TAUBnCMORm settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: TAUBnTTOUTm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1010: Pulse One Count Mode
MD0	0: Disables start trigger detection during counting 1: Enables start trigger detection during counting The value of the MD0 bit of the master and slave channel must be identical.

(b) TAUBnCMURm for the slave channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-103 TAUBnCMURm settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel

Table 16-104 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-105 Simultaneous rewrite settings for the slave channel of the One-Shot Pulse Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is the control channel for simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Operating procedure for One-Shot Pulse Output Function

Table 16-106 Operating procedure for One-Shot Pulse Output Function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *3 "Register settings for the master channel" on page 1259* Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *4 "Register settings for the slave channel" on page 1261* Set the values of the TAUBnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the master channel awaits a TAUBnTTINm input. INTTAUBnIm is generated on the master channel.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	When a valid TAUBnTTINm input edge is detected, TAUBnCNTm of the master channel loads TAUBnCDRm and counts down. When the counter reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave) reloads the TAUBnCDRm value and starts to count down • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is set When TAUBnCNTm (slave) reaches 0001_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is reset If a TAUBnTTINm input is detected on the master channel while the counter is counting, the input is ignored when TAUBnCMORm.MD0 = 0.
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm output is initialized to the value set by TAUBnTO.TOm.

Restart

(6) Specific timing diagrams

(a) TAUBnCDRm (master) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

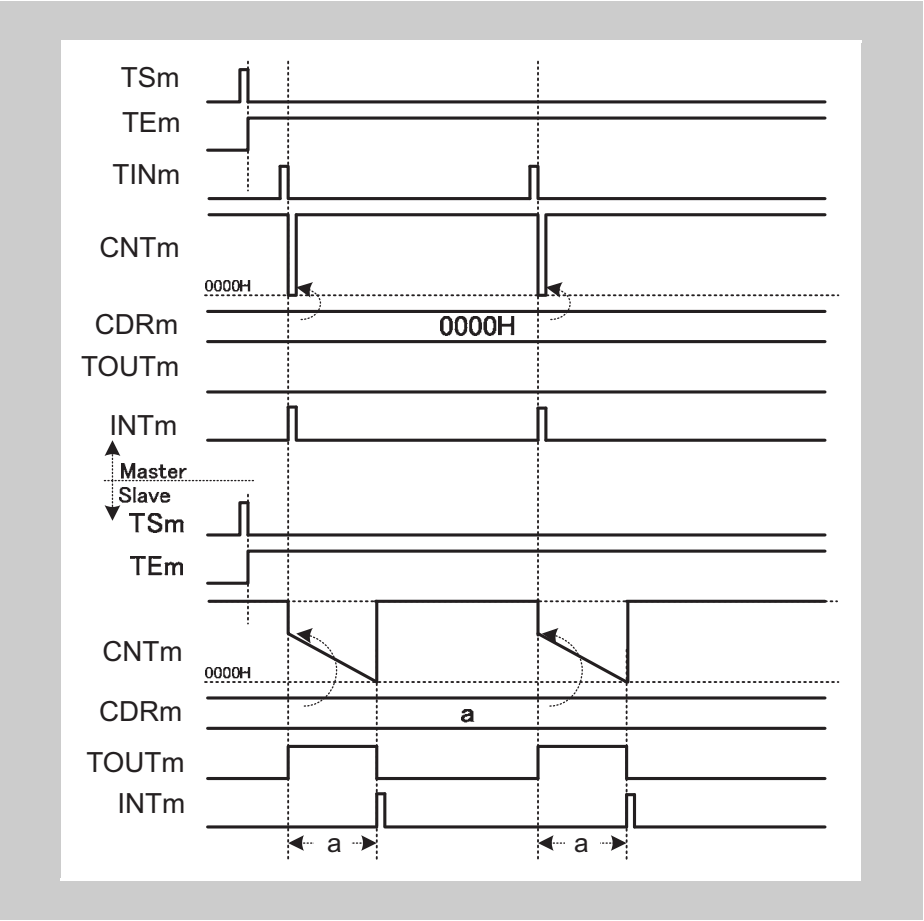


Figure 16-81 TAUBnCDRm (master) = 0000_H

- When a valid TAUBnTTINm input edge is detected, the value 0000_H is written to TAUBnCNTm (master). The counter is set to 0000_H for one count and returns to $FFFF_H$.

  Thus the slave channel starts to count down one count clock later to TAUBnTTINm (master).

(b) TAUBnCDRm (slave) = 0000_H

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

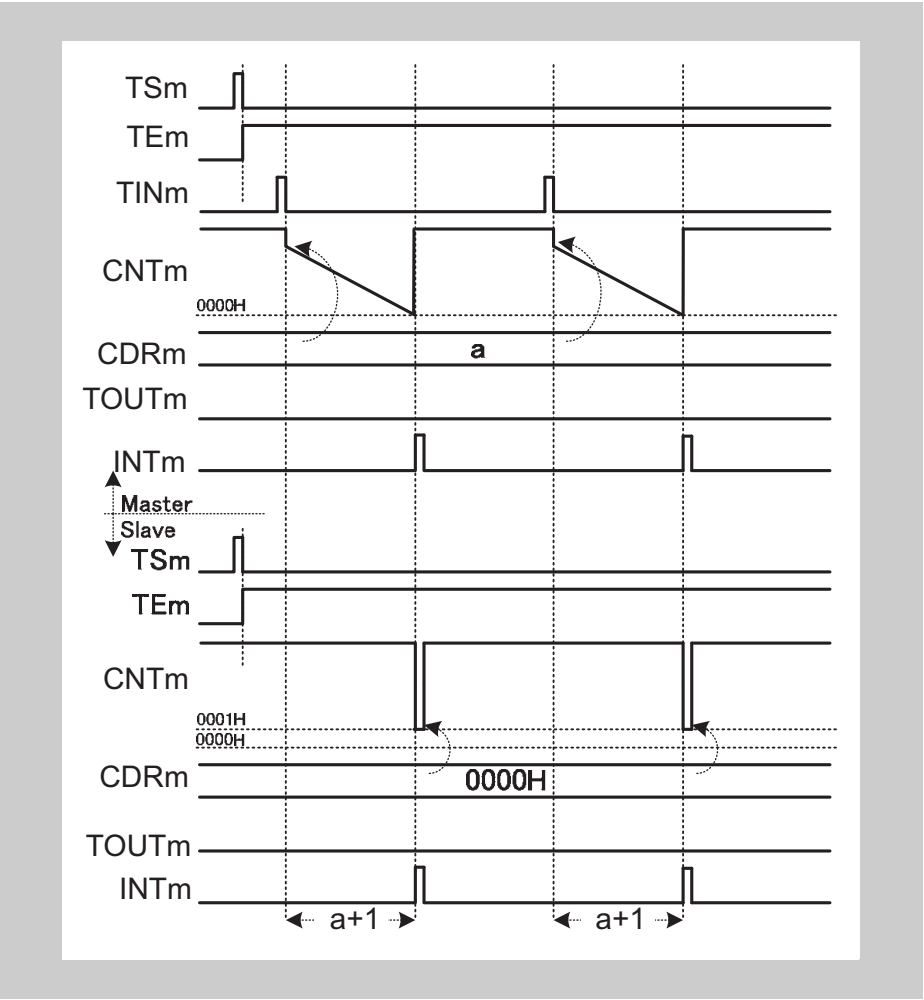


Figure 16-82 TAUBnCDRm (slave) = 0000_H

- The counter of the slave channel reloads the value 0000_H and returns to returns to $FFFF_H$ one clock count later.

  TAUBnTTOUTm remains at not active state, because the pulse width is zero.

(c) TAUBnCMORm.MD0 = 1

The following settings apply to this diagram:

- Start trigger detection enabled during counting (TAUBnCMORm.MD0 = 1)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

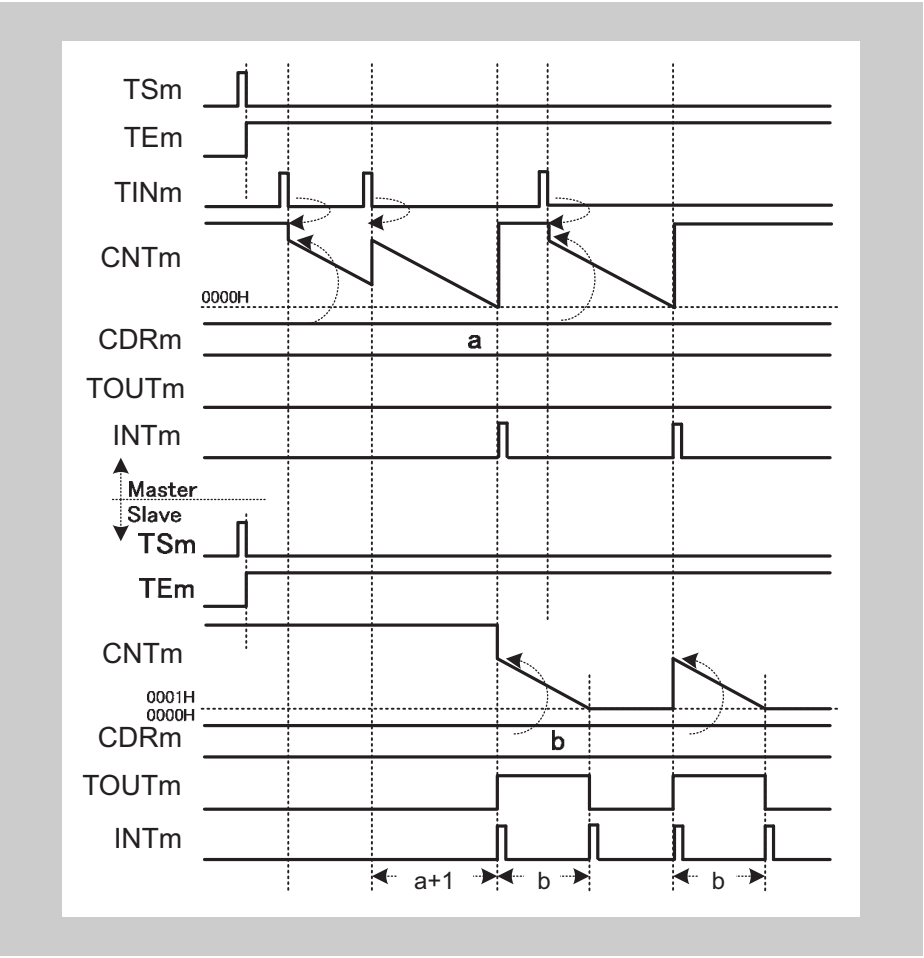


Figure 16-83 TAUBnCMORm.MD0 = 1

- If a valid TAUBnTTINm input edge is detected while the counter of the master channel counts down, TAUBnCNTm reloads the value of TAUBnCDRm. The counter restarts to count down.

  This means the delay is extended by the value of TAUBnCNTm at the time a valid TAUBnTTINm input edge is detected.

(d) Restarting the master channel while the slave channel is counting

The following settings apply to this diagram:

- Start trigger detection disabled during counting (TAUBnCMORm.MD0 = 0)
- Falling edge detection (TAUBnCMURm.TIS[1:0] = 00_B)

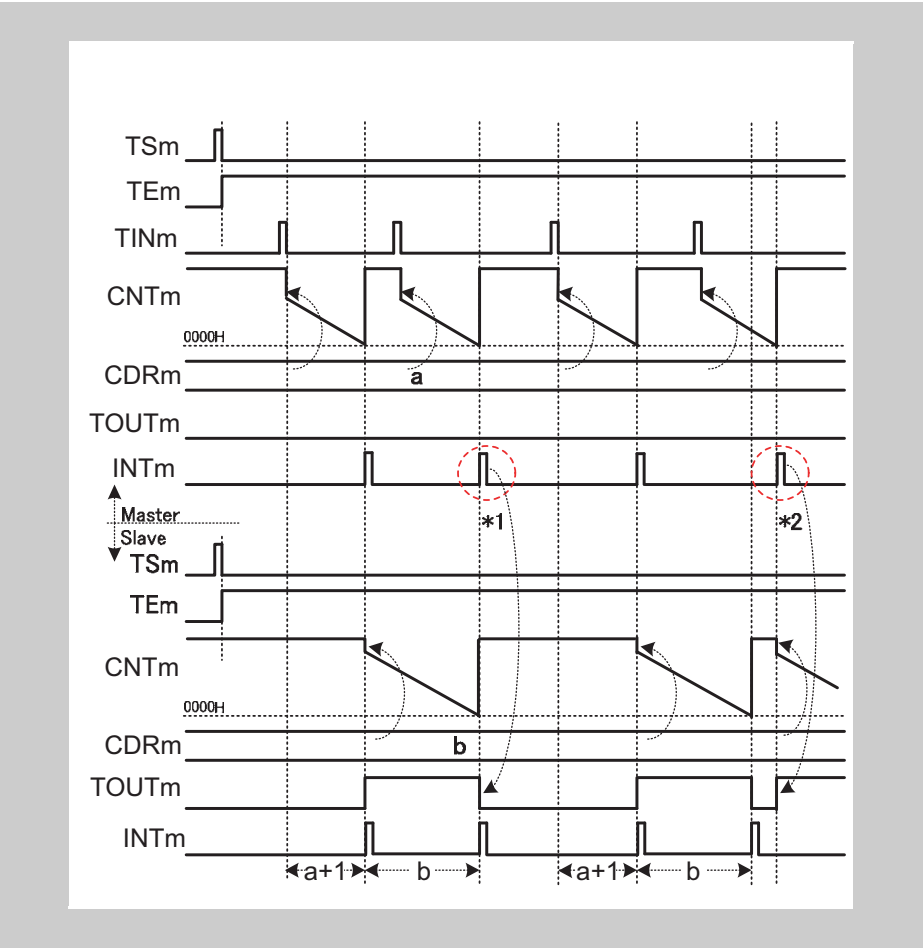


Figure 16-84 Interval of TAUBnTTINm ≤ delay time + pulse width +1

- If the master channel generates an interrupt before the counter of the slave channel has reached 0001_H or exactly when 0001_H is reached (*1), the interrupt (master) is ignored.
- If an interrupt of the master channel occurs when the counter of the slave channel awaits the next trigger, the value of TAUBnCDRm (slave) is reloaded. An interrupt is generated and TAUBnTTOUTm toggles. If TAUBnCNTm (master) has started to count down while the TAUBnCNTm (slave) is still counting (*2), TAUBnTTOUTm is not output with the expected delay time.
- To generate the correct one-shot pulse, the start trigger for the master channel must be detected while the master and slave channels are waiting for the start trigger, and not while they are counting.

16.21 Synchronous Triangle PWM Functions

This chapter describes functions that generate a triangle PWM output.

- *16.21.1 “Triangle PWM Output Function”*
- *16.21.2 “Triangle PWM Output Function with Dead Time”*
- *16.21.3 “AD Conversion Trigger Output Function Type 2”*

16.21.1 Triangle PWM Output Function

(1) Overview

Summary

This function generates multiple triangle PWM outputs by using a master and one or more slave channels. It enables the pulse cycle (frequency) and the duty cycle of TAUBnTTOUTm to be set using the master and slave channel(s) respectively.

The slave channel generates a carrier cycle from two pulse cycles. The first pulse of the master channel controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUBnCNTm (slave) means that signal duration of TAUBnTTOUTm (slave) is double that of the difference between TAUBnCDRm (master) +1 and TAUBnCDRm (slave).

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 16-107 “TAUBnCMORm settings for the master channel of the Triangle PWM Output Function” on page 1273*
- The operation mode of the slave channel(s) must be set to Up Down Count Mode, refer to *Table 16-111 “TAUBnCMORm settings for the slave channel of the Triangle PWM Output Function” on page 1275*
- The channel output mode of the master channel must be set to Independent Channel Output Mode 1 (refer to *16.9 “Channel Output Modes” on page 1122*)
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 2 (refer to *16.9 “Channel Output Modes” on page 1122*)
- The following settings establish TAUBnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUBnCMORm.MD0 (master) bit is set to 0,TAUBnTO.TOm must be set to 1 while TAUBnTOE.TOEm is 0.
  - If the TAUBnCMORm.MD0 (master) bit is set to 1, TAUBnTO.TOm must be set to 0 while TAUBnTOE.TOEm is 0.

Description

The counters are started by setting the channel trigger bit (TAUBnTS.TSm) to 1 for every channel. This in turn sets TAUBnTE.TEm, enabling count operation. The current values of TAUBnCDRm (master and slave) are written to TAUBnCNTm (master and slave) and the counters start to count down from these values. Depending on the setting of the master channel TAUBnCMORm.MD0 bit an interrupt is generated and TAUBnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. TAUBnCNTm then reloads the TAUBnCDRm value and counts down.

- Slave channel:

  The INTTAUBnIm of the master channel triggers the counter of the slave channel:

  – If the slave counter currently counts down, it changes count direction .

  – If the slave counter currently counts up, the value of TAUBnCDRm is reloaded and the counter counts down.

  When the counter of the slave channel reaches 0001_H while counting up or down, INTTAUBnIm is generated and the TAUBnTTOUTm (slave) signal toggles:

  – It is set in the count-down status

  – It is reset in the count-up status

  The counter continues to count down or up and awaits the next INTTAUBnIm of the master channel.

TAUBnTTOUTm can be switched between positive and negative phase setting TAUBnTOL.TOLm during operation.

The counters can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values.

Note If a forced restart is executed during operation, TAUBnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *16.8 “Simultaneous Rewrite” on page 1112*

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUBnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUBnCDRm (master) + 1 - TAUBnCDRm (slave))/ (TAUBnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUBnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

(3) Block diagram and general timing diagram

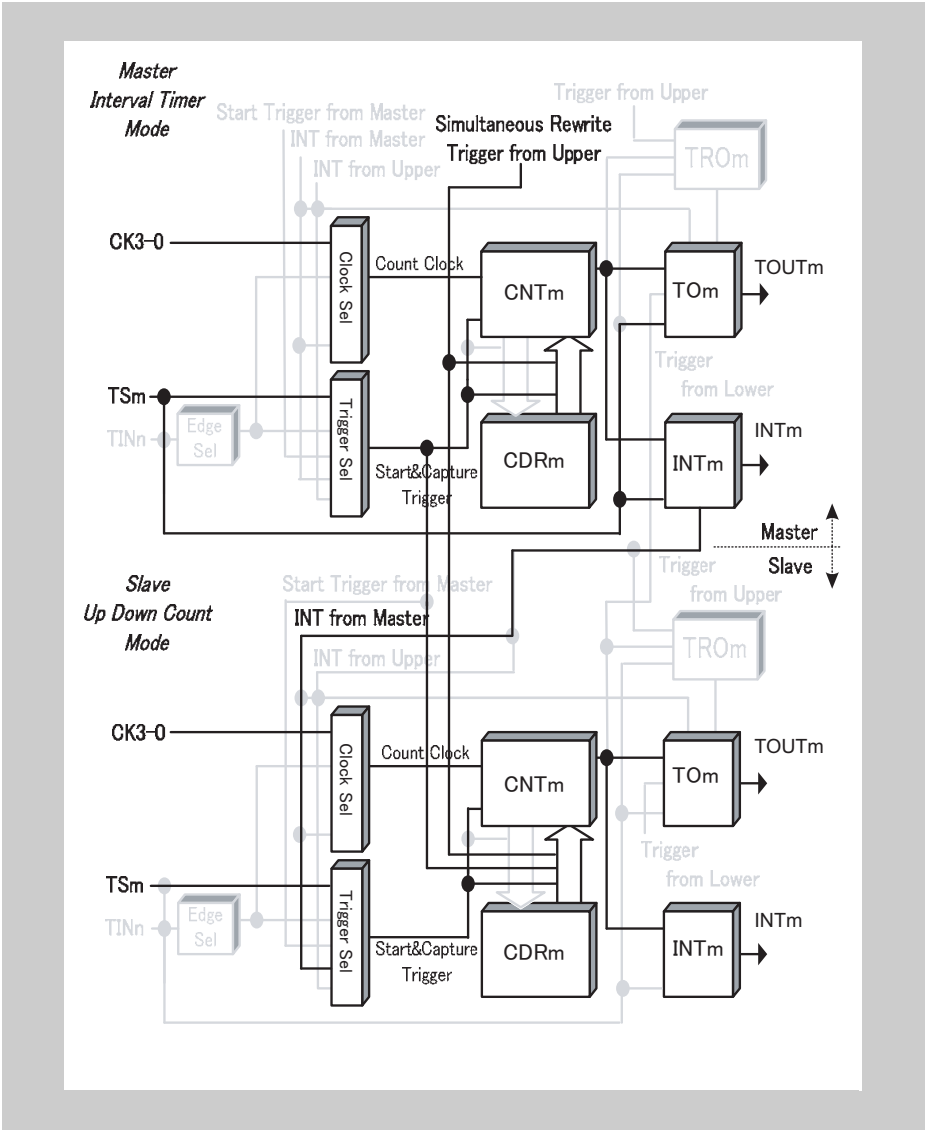


Figure 16-85 Block diagram for Triangle PWM Output Function

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

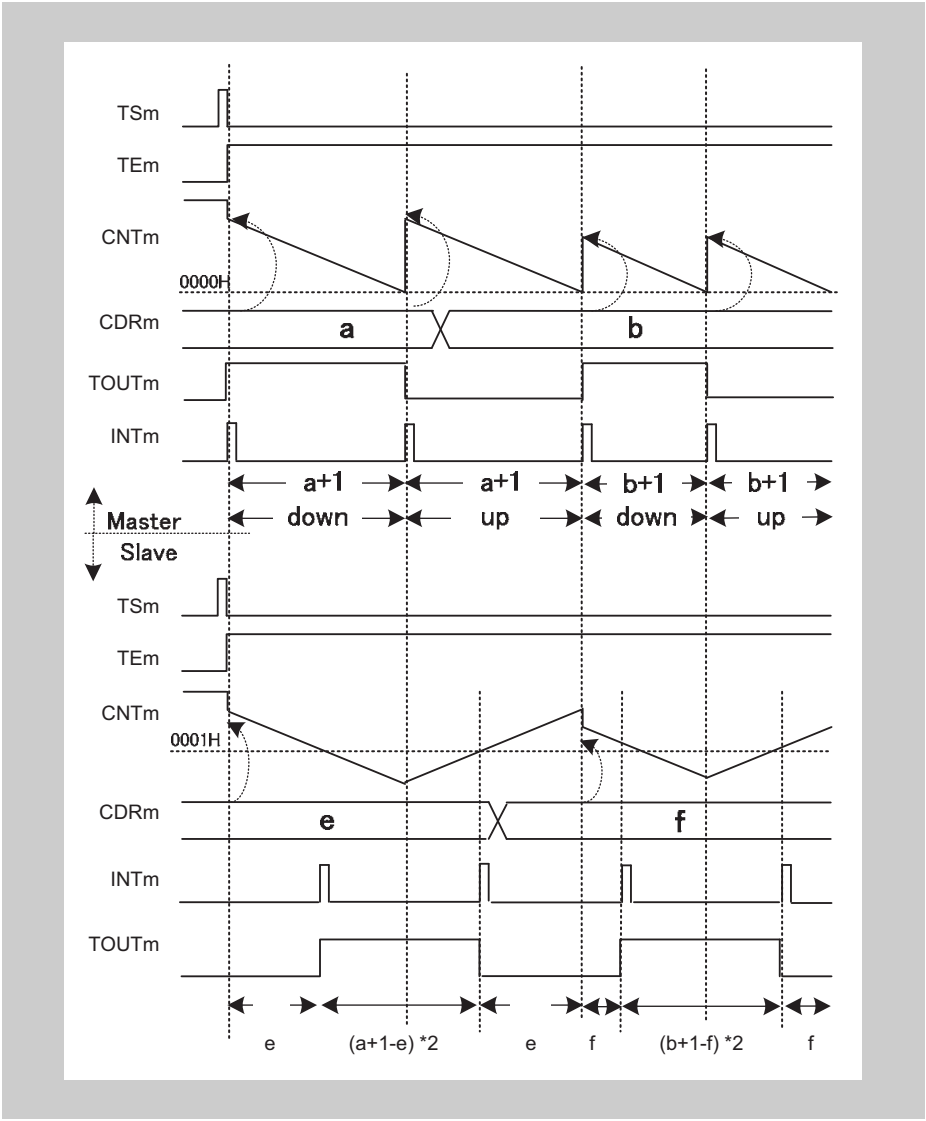


Figure 16-86 General timing diagram for Triangle PWM Output Function

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-107 TAUBnCMORm settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-108 TAUBnCMURm settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 16-109 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-110 Simultaneous rewrite settings for the master channel of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUBnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-111 TAUBnCMORm settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-112 TAUBnCMURm settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 16-113 Control bit settings for Synchronous Channel Output Mode 2

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-114 Simultaneous rewrite settings for the slave channel of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for Triangle PWM Output Function

Table 16-115 Operating procedure for Triangle PWM Output Function

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1273* Slave channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 1275* Set the values of the TAUBnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel, depending on TAUBnCMORm.MD0.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master and slave channel loads TAUBnCDRm and counts down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnTTOUTm (master) toggles • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation. • TAUBnCNTm (slave) reloads the TAUBnCDRm value or counts in the reverse direction. When TAUBnCNTm of the slave = 0001_H: • INTTAUBnIm (slave) is generated • TAUBnTTOUTm (slave) is set (in count-down status) or reset (in count-up status)
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm is initialized to the value set by TAUBnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnCDRm = a = 5_H
- Slave channel:
  - TAUBnCDRm = 6_H

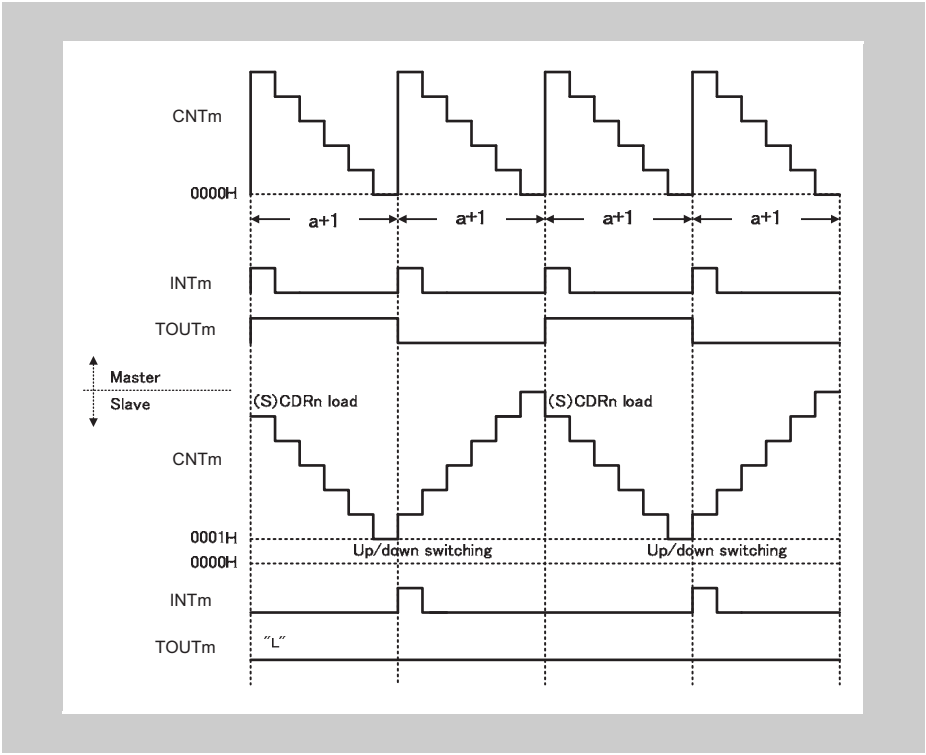


Figure 16-87 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

- If TAUBnCDRm (slave) ≥ TAUBnCDRm (master) the counter of slave channel cannot reach 0001_H during *counting down*. The set signal is never detected, so TAUBnTTOUTm remains at low state.

(b) Duty cycle = 100 %

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnCDRm = a = 5_H
- Slave channel:
  - TAUBnCDRm = 0_H

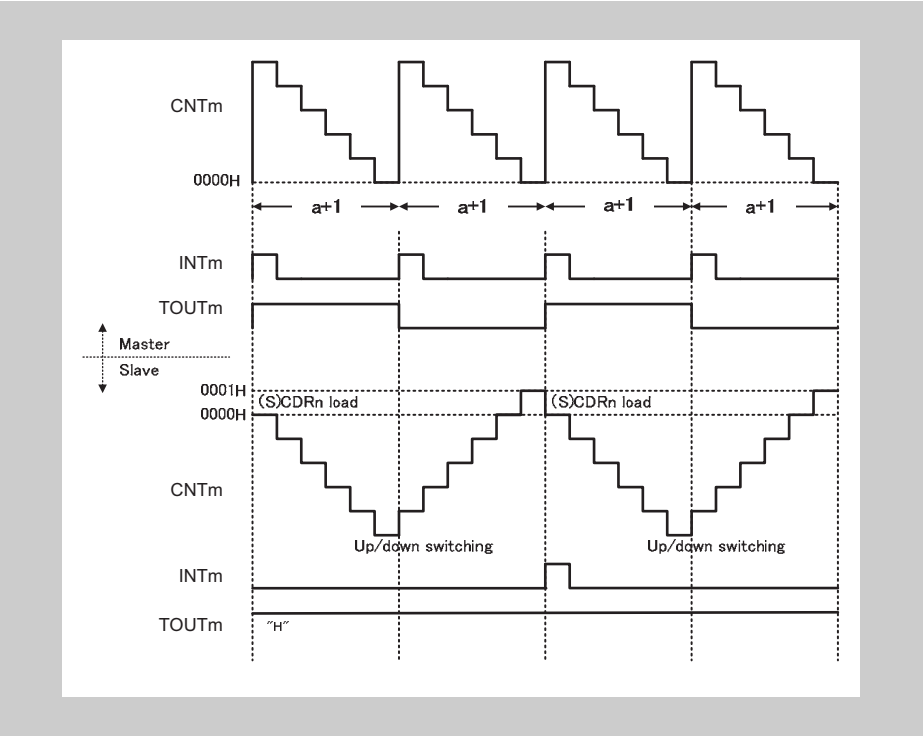


Figure 16-88 TAUBnCDRm (slave) = 0000_H

- If TAUBnCDRm (slave) =0000_H the counter of slave channel cannot reach 0001_H during *counting up*. The reset signal is never detected, so TAUBnTTOUTm remains at high state.

16.21.2 Triangle PWM Output Function with Dead Time

(1) Overview

Summary

This function generates multiple triangle PWM outputs with a defined dead time by using a master and two or more slave channels. The resulting PWM signals are output via TAUBnTTOUTm of the slave channels 2 and 3. It enables the pulse cycle (frequency) and the duty cycle of TAUBnTTOUTm to be set using the master and slave channel(s) respectively.

Slave channel 2 generates a carrier cycle from two pulse cycles from the master channel. The first pulse controls the down status and the second pulse controls the up status of the slaves counter.

Counting up and down TAUBnCNTm (slave 2) means that signal duration of TAUBnTTOUTm (slave) is double that of the difference between TAUBnCDRm (master) +1 and TAUBnCDRm (slave 2).

An interrupt on slave 2 causes TAUBnTTOUTm of the slave channels to toggle. Depending on the settings of TAUBnTDL.TDLm, delay time is added to positive or negative logic side of the signal (i.e. whether TAUBnTTOUTm toggles immediately or after dead time has elapsed). The duration of the dead time is specified by slave channel 3.

Prerequisites

- Three channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 16-117 “TAUBnCMORm settings for the master channel of the Triangle PWM Output Function with Dead Time” on page 1285*
- Slave channel 1 is not used for this function. This ensures that slave channel 2 is an odd channel, and slave channel 3 is an even channel.
- The operation mode of slave channel 2 must be set to Up Down Mode, refer to *Table 16-121 “TAUBnCMORm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time” on page 1287*
  Furthermore, slave channel 2 must be an even channel
- The operation mode of slave channel 3 must be set to One Count Mode, refer to *Table 16-125 “TAUBnCMORm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time” on page 1289*
  Furthermore, slave channel 3 must be an odd channel
- The channel output mode of the master channel must be set to Independent Channel Output Mode 1 (refer to *16.9 “Channel Output Modes” on page 1122*)
- The channel output mode of the slave channels 2 and 3 must be set to Synchronous Channel Output Mode 2 with Dead Time Output (refer to *16.9 “Channel Output Modes” on page 1122*)
- The following settings establish TAUBnTTOUTm at high level for the down status of the carrier cycle.
  - If the TAUBnCMORm.MD0 (master) bit is set to 0,TAUBnTO.TOm must be set to 1 while TAUBnTOE.TOEm is 0.
  - If the TAUBnCMORm.MD0 (master) bit is set to 1, TAUBnTO.TOm must be set to 0 while TAUBnTOE.TOEm is 0.

Note Slave channel 1 is not used for Triangle PWM Output Function with Dead Time.

Description

The counters are started by setting the channel trigger bits (TAUBnTS.TSm) to 1. This in turn sets TAUBnTE.TEm, enabling count operation. The current values of TAUBnCDRm is written to TAUBnCNTm and the counters start to

count down from these values. Depending on the setting of the master channel TAUBnCMORm.MD0 bit an interrupt is generated and TAUBnTTOUTm signal of the master toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, INTTAUBnIm is generated and the TAUBnTTOUTm signal toggles. The counter reloads the TAUBnCDRm value and counts down.

- Slave channel 2:

  The INTTAUBnIm of the master channel triggers the counter of the slave channel 2:

  – If the slave counter currently counts down, it changes count direction.

  – If the slave counter currently counts up, the value of TAUBnCDRm is reloaded and the counter counts down.

  The counter continues to count down or up and awaits the next INTTAUBnIm of the master channel.

- Slave channel 3:

  INTTAUBnIm of slave channel 2 triggers the counter of slave channel 3. The current value of TAUBnCDRm (slave 3) is written to TAUBnCNTm (slave 3) and the counter starts to count down from this value.

  When the counter reaches 0000_H, dead time has elapsed, INTTAUBnIm is generated. The counter returns to $FFFF_H$ and awaits the next INTTAUBnIm of slave channel 2.

An interrupt on slave channel 2 causes TAUBnTTOUTm to toggle as follows:

- It is set by the interrupt when slave channel 2 is counting down
- It is reset by the interrupt when slave channel 2 is counting up.

However, the TAUBnTDL.TDLm settings of the corresponding channel specify whether it is set/reset immediately, or after dead time has elapsed, as shown in *Table 16-116 “Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2” on page 1282* .

The TAUBnTOL.TOLm settings specify whether set corresponds to a high signal (TAUBnTOL.TOLm = 0) or a low signal (TAUBnTOL.TOLm = 1).

In Triangle PWM Output Function with Dead Time, TAUBnTTOUTm can be switched between positive and negative phase by setting TAUBnTOL.TOLm during operation.

The counter can be stopped by setting TAUBnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUBnTE.TEm to 0. TAUBnCNTm and TAUBnTTOUTm of master and slave channel(s) stop but retain their values.

TAUBnCDRm value of slave channel 2 can be set to 0000_H to output 100 % TAUBnTTOUTm.

Note If a forced restart is executed during operation, TAUBnTTOUTm is not output as a triangle PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *16.8 "Simultaneous Rewrite" on page 1112*

TAUBnTOL.TOLm bits should be set before the counter starts, and slave channels 2 and 3 should have opposite TAUBnTOL.TOLm settings or opposite TAUBnTDL.TDLm settings.

Table 16-116 Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2

TAUBnTDL.TDLm	Count direction of slave channel 2 when interrupt is generated	TAUBnTTOUTm toggles
0	Down	Set after dead time has elapsed
	Up	Reset immediately
1	Down	Set immediately
	Up	Reset after dead time has elapsed

Notes

1. When the set and reset condition for TAUBnTTOUTm occur simultaneously:
   - If TAUBnTOL.TOLm = 0, the reset condition has priority
   - If TAUBnTOL.TOLm = 1, the set condition has priority .
2. To generate a two-phase PWM output with added dead time, the settings of TAUBnTOL.TOLm bit of slave channels 2 and 3 must not be changed.

(2) Equations

Pulse cycle = (TAUBnCDRm (master) + 1) x count clock cycle

Carrier cycle (down/up) = (TAUBnCDRm (master) + 1) × 2 × count clock cycle

Duty cycle [%] =
[(TAUBnCDRm (master) + 1 - TAUBnCDRm (slave))/ (TAUBnCDRm (master) + 1)] x 100

– Duty cycle = 100 %

  TAUBnCDRm (slave) = 0000_H

– Duty cycle = 0 %

  TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

PWM signal width (positive phase) = [((TAUBnCDRm (master) + 1 - TAUBnCDRm (slave 2) × 2) - (TAUBnCDRm (slave 3) + 1)] × count clock cycle

PWM signal width (negative phase) = [((TAUBnCDRm (master) + 1 - TAUBnCDRm (slave 2) × 2) + (TAUBnCDRm (slave 3) + 1)] × count clock cycle

(3) Block diagram and general timing diagram

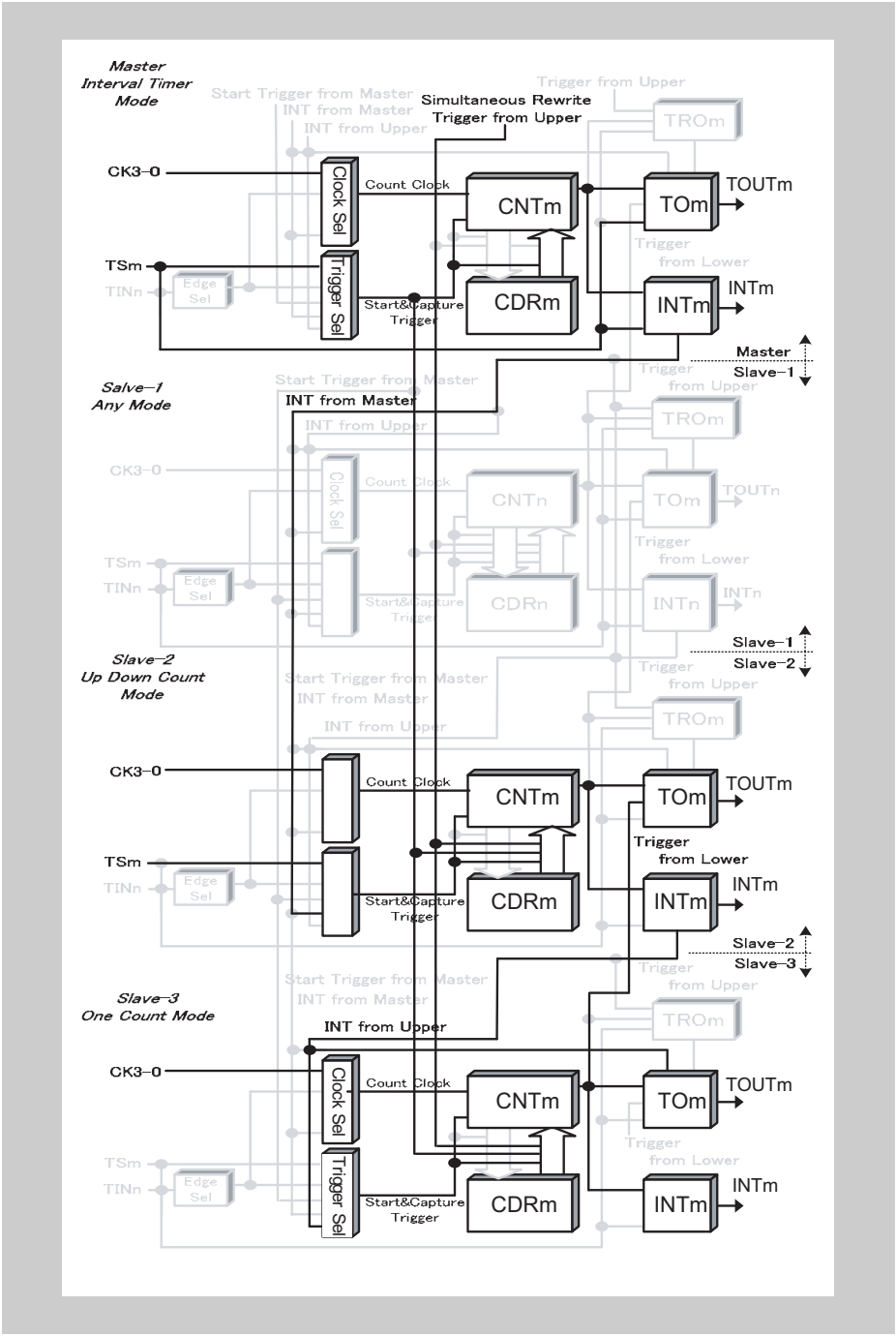


Figure 16-89 Block diagram for Triangle PWM Output Function with Dead Time

The following settings apply to the general timing diagram:

- Master channel:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
- Slave channel 2:
  - INTTAUBnIm not generated at operation start (TAUBnCMORm.MD0 = 0)
  - TAUBnTDL.TDLm = 0
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)
  - TAUBnTDL.TDLm = 1
  - Negative logic (TAUBnTOL.TOLm = 1)

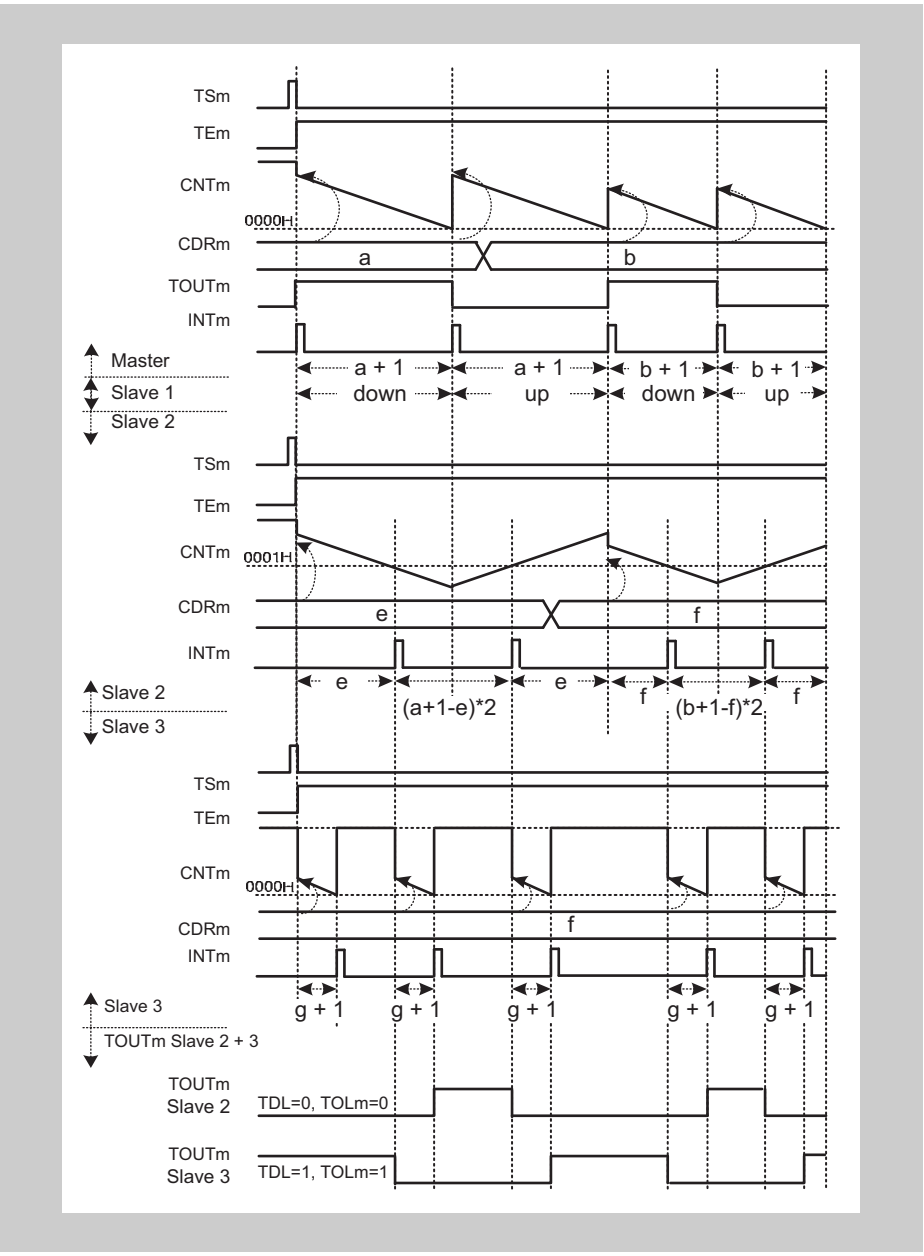


Figure 16-90 General timing diagram for Triangle PWM Output Function with Dead Time

(4) Register settings for the master channel

(a) TAUBnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-117 TAUBnCMORm settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUBnIm not generated and TAUBnTTOUTm does not toggle at operation start 1: Generates INTTAUBnIm and toggles TAUBnTTOUTm at operation start

(b) TAUBnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-118 TAUBnCMURm settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

Table 16-119 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	0: Independent channel output
TOCm	0: Operation mode 1 (= Toggle mode if TAUBnTOM.TOMm = 0)
TOLm	0: Positive logic
TDEm	0: Disables dead time operation
TDLm	0: When dead time operation is disabled (TAUBnTDE.TDEm = 0), set these bits to 0

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-120 Simultaneous rewrite settings for the master channel of the Triangle PWM Output Function with Dead Time

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for slave channel 2

(a) TAUBnCMORm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-121 TAUBnCMORm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1001: Up Down Count Mode
MD0	0: INTTAUBnIm not generated at operation start

(b) TAUBnCMURm for slave channel 2

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-122 TAUBnCMURm settings for slave channel 2 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 2

Table 16-123 Control bit settings for Synchronous channel Output Mode 2 with Dead Time Output

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase

(d) Simultaneous rewrite for slave channel 2

The simultaneous rewrite settings of the master and slave channels must be identical.

Table 16-124 Simultaneous rewrite settings for slave channel 2 of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Register settings for slave channel 3

(a) TAUBnCMORm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 16-125 TAUBnCMORm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS0	0: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	111: The up/down output trigger signal TAUBnTUDSm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Enables start trigger detection during counting

(b) TAUBnCMURm for slave channel 3

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 16-126 TAUBnCMURm settings for slave channel 3 of the Triangle PWM Output Function with Dead Time

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for slave channel 3

Table 16-127 Control bit settings for Synchronous channel Output Mode 2 with Dead Time Output

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	1: Operation mode 2
TOLm	0: Positive logic 1: Inverted logic
TDEm	1: Enables dead time operation
TDLm	0: Dead time is added to the positive phase 1: Dead time is added to the negative phase

(d) Simultaneous rewrite for slave channel 3

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 16-128 Simultaneous rewrite settings for slave channel 3 of the Triangle PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDM.RDMm	1: The simultaneous rewrite trigger signal is generated when the master channel starts counting and the corresponding slave channel is at the peak of a triangular wave
RDC.RDCm	0: Channel is not monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger. If TAUBnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(7) Operating procedure for Triangle PWM Output Function with Dead Time

Table 16-129 Operating procedure for Triangle PWM Output Function with Dead Time (1/2)

	Operation	Status of TAUBn
Initial channel setting	Master channel: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1285* Slave channel 2: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *5 "Register settings for slave channel 2" on page 1287* Slave channel 3: set the TAUBnCMORm and TAUBnCMURm registers and the channel output mode as described in *6 "Register settings for slave channel 3" on page 1289* Set the values of the TAUBnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)

Table 16-129 Operating procedure for Triangle PWM Output Function with Dead Time (2/2)

	Operation	Status of TAUBn
Start operation	Set TAUBnTS.TSm of the master and slave channels to 1 simultaneously. TAUBnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUBnIm is generated on the master channel., depending on the setting of TAUBnCMORm.MD0.
During operation	TAUBnCDRm can be changed at any time. TAUBnCNTm and TAUBnRSF.RSFm can be read at any time. TAUBnRDT.RDTm can be changed during operation.	TAUBnCNTm of the master channel and slave channel 2 load TAUBnCDRm and count down. When the counter of the master channel reaches 0000_H: • INTTAUBnIm (master) is generated • TAUBnCNTm (master) reloads the TAUBnCDRm value and continues count operation • TAUBnCNTm (slave 2) reloads the TAUBnCDRm value or counts in the reverse direction When TAUBnCNTm (slave 2) reaches 0001_H: • INTTAUBnIm (slave 2) is generated • TAUBnTTOUTm is reset (TAUBnTOL.TOLm=0) • TAUBnCNTm of slave channel 3 reloads the TAUBnCDRm value and counts down When TAUBnCNTm of slave channel 3 = 0000_H: • INTTAUBnIm is generated • TAUBnTTOUTm is set (TAUBnTOL.TOLm=0) Whether TAUBnTTOUTm (slave 2 and 3) are set/reset with INTTAUBnIm (slave 2) simultaneously or after dead time has elapsed depends on the settings in TAUBnTDL.TDLm and TAUBnTOL.TOLm (refer to *Table 16-116 "Behavior of TAUBnTTOUTm when an interrupt occurs on slave channel 2" on page 1282*).
Stop operation	Set TAUBnTT.TTm of the master and slave channels to 1 simultaneously. TAUBnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUBnTE.TEm is cleared to 0 and the counter stops. TAUBnCNTm and TAUBnTTOUTm stop and retain their current values. When TAUBnTOE.TOEm is 0, TAUBnTTOUTm of slave channels 2 and 3 is initialized to the value set by TAUBnTO.TOm.

Restart

(8) Specific timing diagrams

(a) Duty cycle = 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

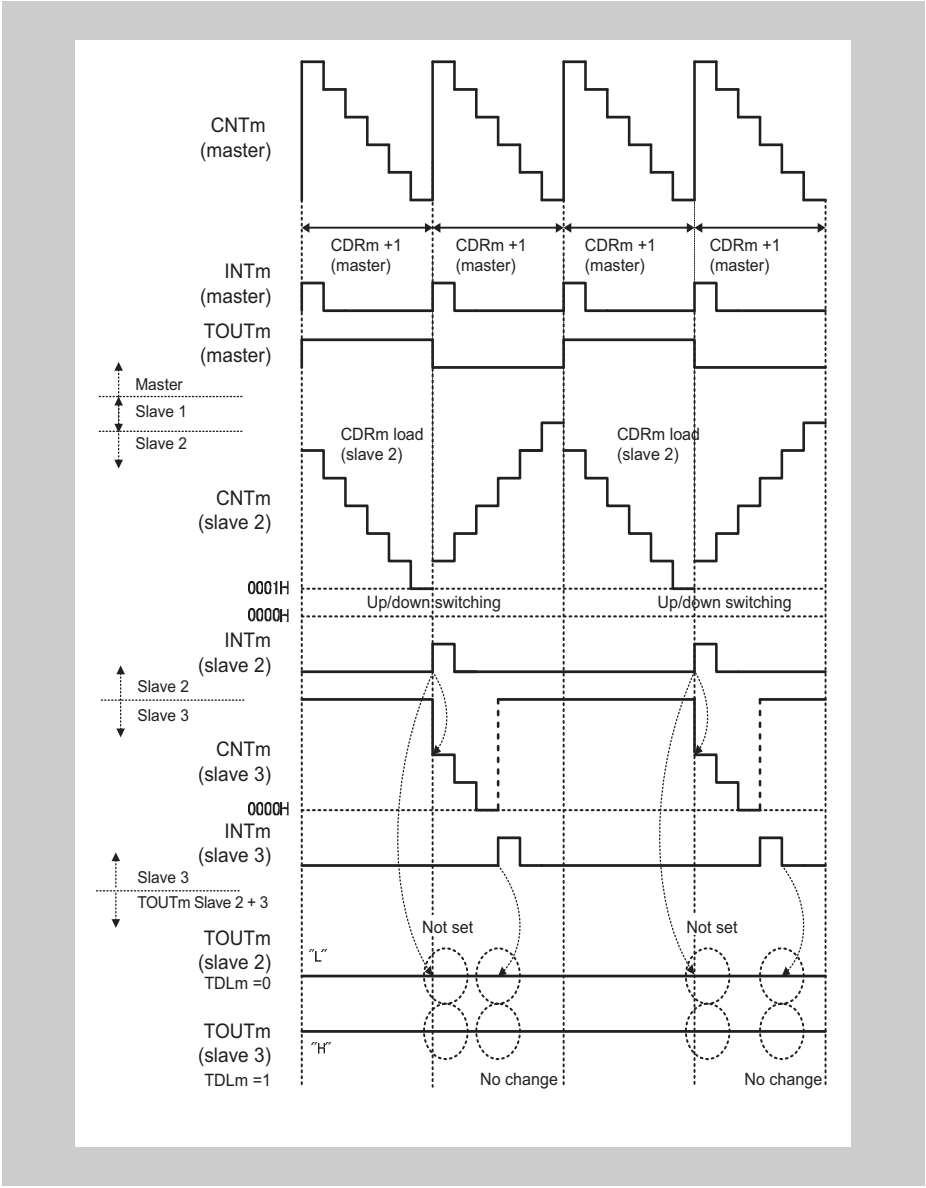


Figure 16-91 TAUBnCDRm (slave) ≥ TAUBnCDRm (master) + 1

- If TAUBnCDRm (slave 2) ≥ TAUBnCDRm (master) the counter of slave channel cannot reach 0000_H during counting down. Therefore TAUBnTTOUTm cannot toggle, i.e. it remains at its initial state. The interrupt from slave channel 2 occurs during count up, therefore it is a reset signal.

(b) Duty cycle = 100 %

The following settings apply to the diagram below:

- Slave channels 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channels 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

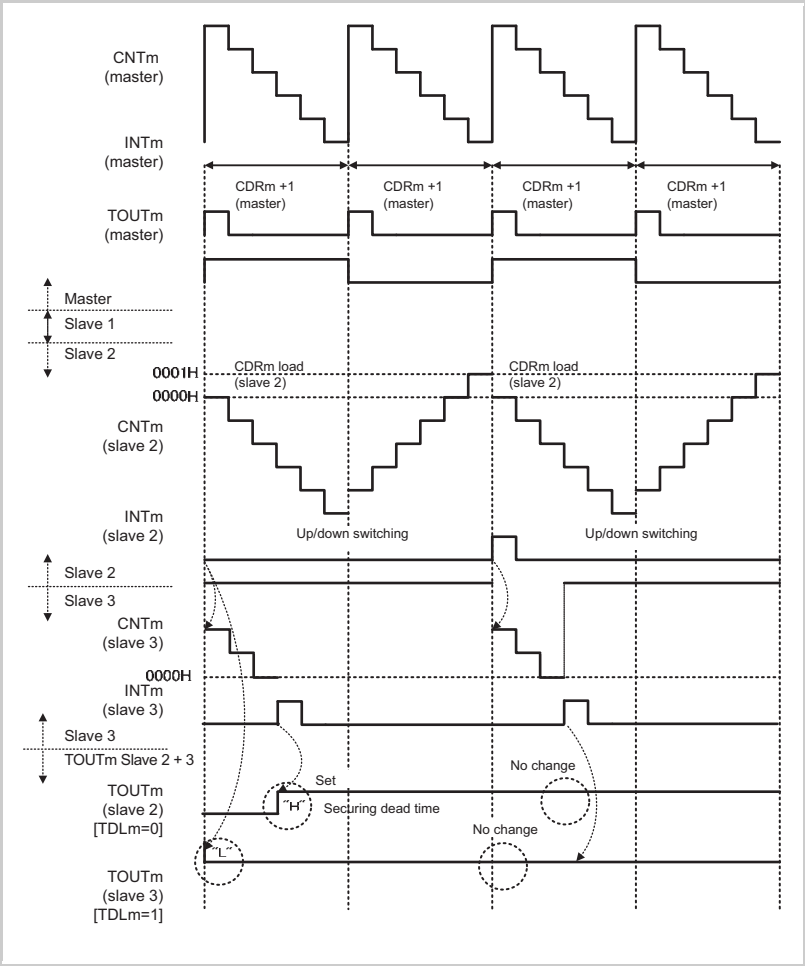


Figure 16-92 TAUBnCDRm (slave) = 0000_H

- If TAUBnCDRm (slave 2) =0000_H the counter of slave channel cannot reach 0001_H while counting up and therefore cannot generate an INTTAUBnIm while counting up.
  - The set conditions for a channel in which TAUBnTDL.TDLm = 0 are met after dead time has elapsed. TAUBnTTOUTm toggles but remains in the new state because the reset conditions never occur for such a channel.
  - Slave channel 3 in the diagram above is set when the counter starts. However, the reset conditions for a channel in which TAUBnTDL.TDLm = 1 never occur so TAUBnTTOUTm remains in its initial state for such a slave channel. In slave 3 in the diagram above, the initial state is high because TAUBnTOL.TOLm = 1.

(c) TAUBnTTOUTm (slave 2) = 0 % and TAUBnTTOUTm (slave 3) > 0 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

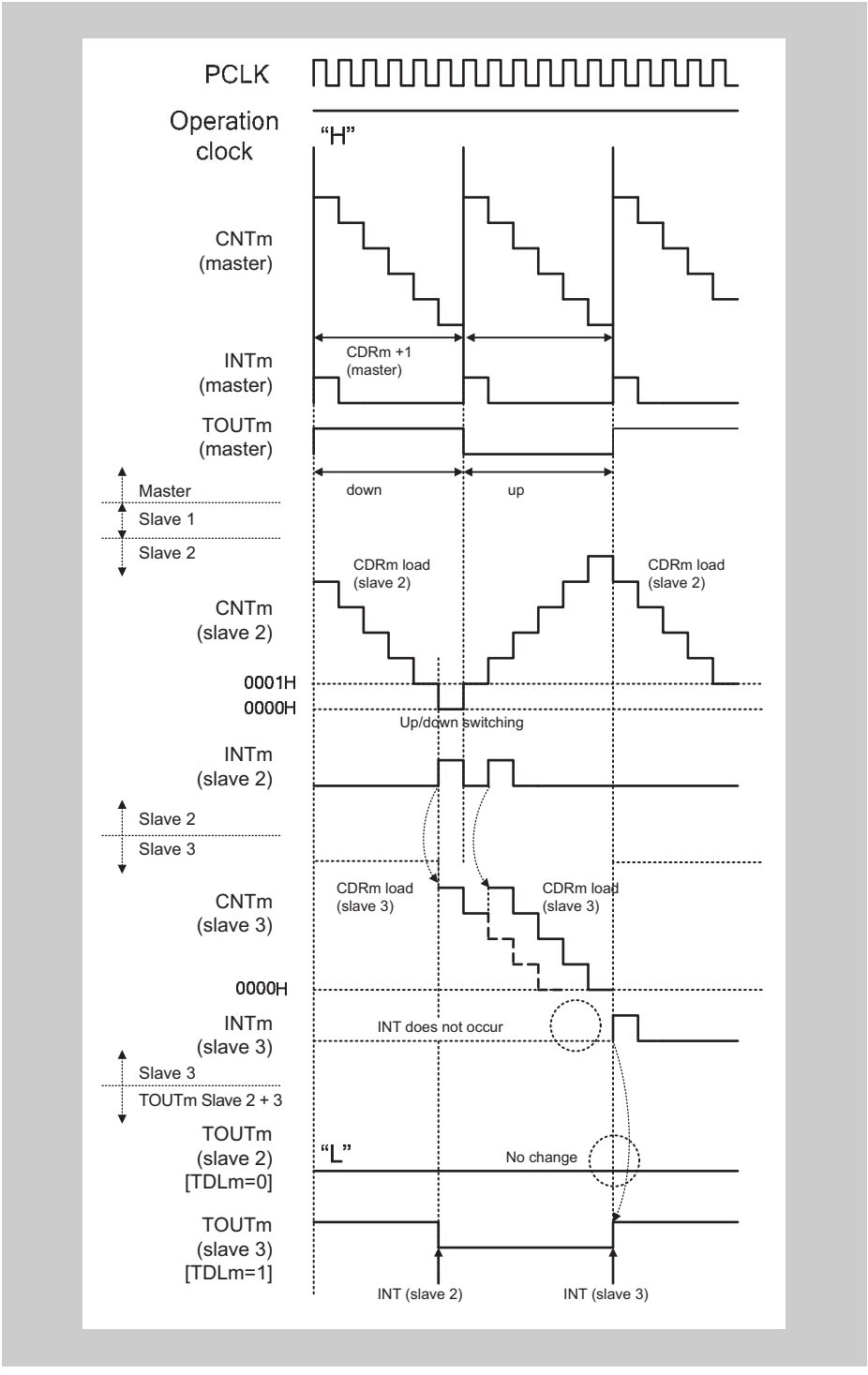


Figure 16-93 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0005_H, TAUBnCDRm (slave 3) = 0004_H

- When the counter of slave channel 2 reaches 0000_H, INTTAUBnIm (slave 2) is generated. The counter of slave channel 3 starts to count down.
- If another INTTAUBnIm (slave 2) is generated while the counter of slave channel 3 is still counting down, the value of TAUBnCDRm (slave 3) is reloaded and the counter restarts counting down from this value.
- In the diagram above, the first interrupt on channel 2 occurs while the counter is counting down, and the second whilst it is counting up.
- After the first interrupt, a slave for which TAUBnTDL.TDLm = 0 waits for dead time to elapse before setting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a reset signal, meaning that a channel for which TAUBnTDL.TDLm = 0 always remains inactive.
- TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 1 is set and reset as normal when the corresponding INTTAUBnIm is generated.

(d) TAUBnTTOUTm (slave 2) > 0 % and TAUBnTTOUTm (slave 3) = 100 %

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

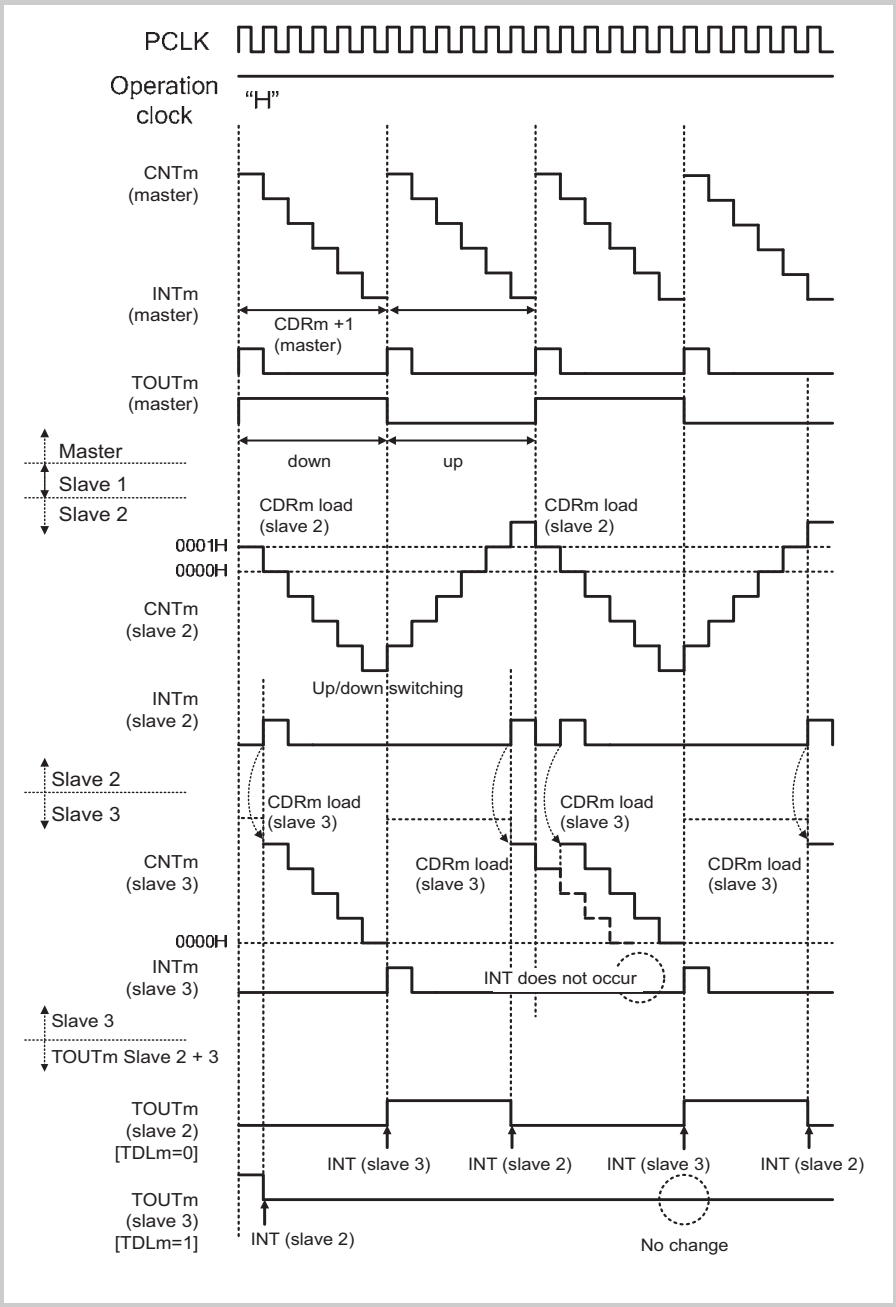


Figure 16-94 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0002_H, TAUBnCDRm (slave 3) = 0004_H
PWM signal width (negative phase) ≥ Carrier cycle

- After the second interrupt, a slave for which TAUBnTDL.TDLm = 1 waits for dead time to elapse before resetting. However, before the dead time has elapsed, another interrupt occurs on slave 2, this time while the counter is counting up. This acts as a set signal, meaning that a channel for which TAUBnTDL.TDLm = 1 always remains active.
- TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 0 is set and reset as normal when the corresponding INTTAUBnIm is generated.

(e) Inhibited INTTAUBnIm to set TAUBnTTOUTm positive phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

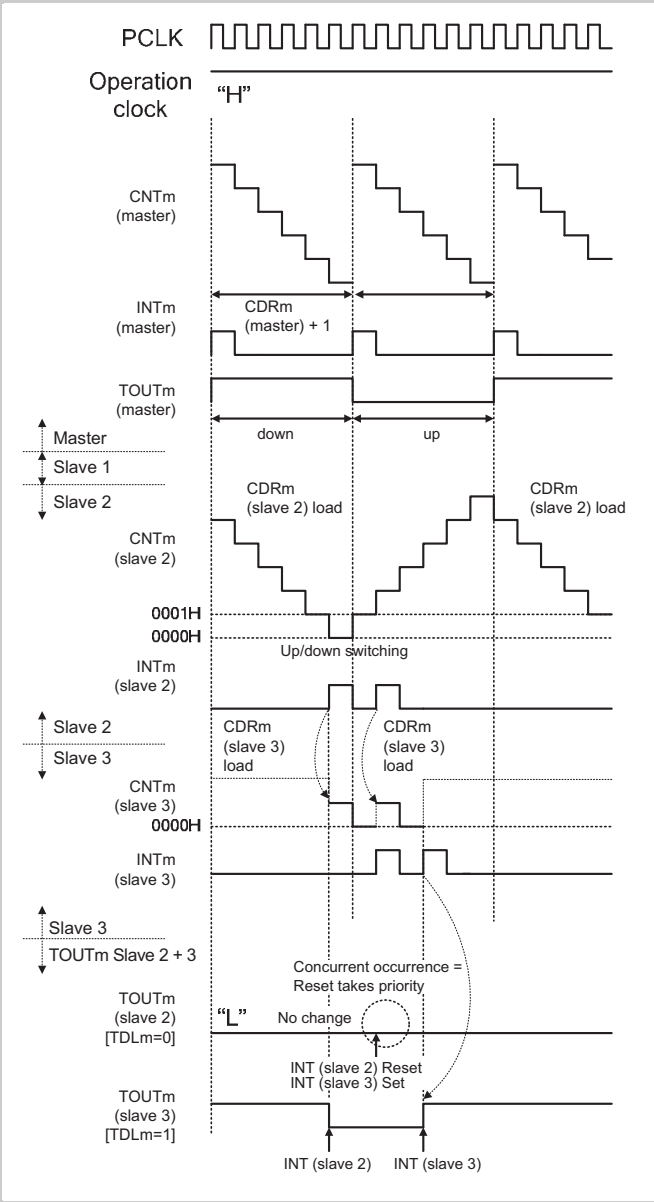


Figure 16-95 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0005_H, TAUBnCDRm (slave 3) = 0001_H
PWM signal width (positive phase) = 0

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUBnIm to set the TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 0 (slave channel 2 in this example).
- If channel 2 generates an INTTAUBnIm to reset TAUBnTTOUTm simultaneously, this reset signal has priority (assuming TAUBnTOL.TOLm = 0, otherwise the set signal has priority).
- Therefore, TAUBnTTOUTm of a slave channel for which TAUBnTDL.TDLm = 0 remains in its initial state.

(f) Inhibited INTTAUBnIm to set TAUBnTTOUTm negative phase period

The following settings apply to the diagram below:

- Slave channel 2:
  - Positive logic (TAUBnTOL.TOLm = 0)
- Slave channel 3:
  - Negative logic (TAUBnTOL.TOLm = 1)

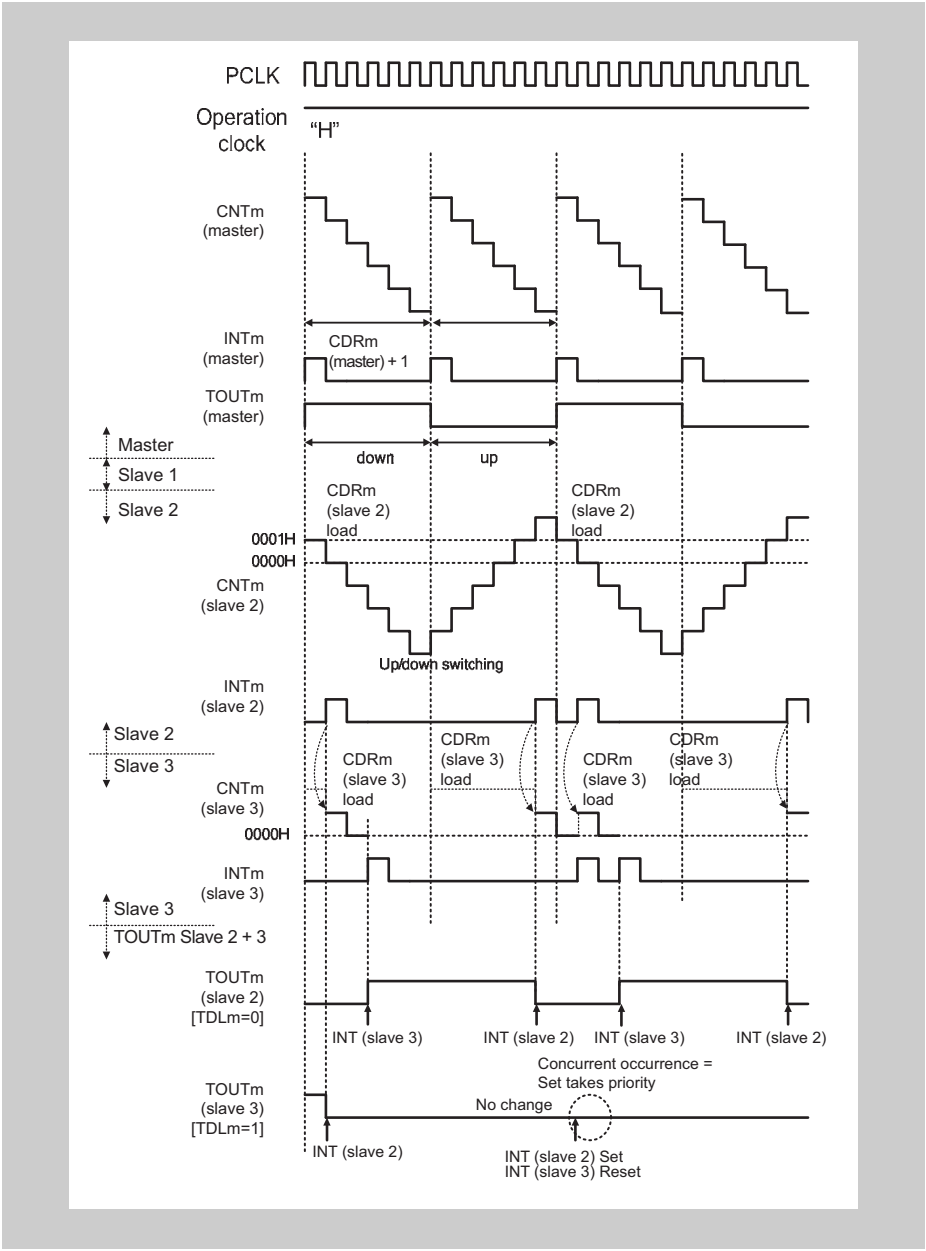


Figure 16-96 TAUBnCDRm (master) = 0005_H, TAUBnCDRm (slave 2) = 0001_H, TAUBnCDRm (slave 3) = 0001_H
PWM signal width (negative phase) = carrier cycle

- The counter of slave channel 3 reaches 0000_H and generates an INTTAUBnIm to set the TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 1 (slave 3 in this example).
- If channel 2 generates an INTTAUBnIm to reset TAUBnTTOUTm simultaneously, the set signal has priority (assuming TAUBnTOL.TOLm = 1, otherwise the reset signal has priority).
- Therefore, TAUBnTTOUTm of slave channel for which TAUBnTDL.TDLm = 1 remains in its initial state.

16.21.3 AD Conversion Trigger Output Function Type 2

(1) Overview

Summary This function is identical to *16.21.1 "Triangle PWM Output Function" on page 1269* except that TAUBnTTOUTm is not output.

This is achieved by setting the channel output mode of the slave to Direct Channel Output Mode.

(2) Block diagram and general timing diagram

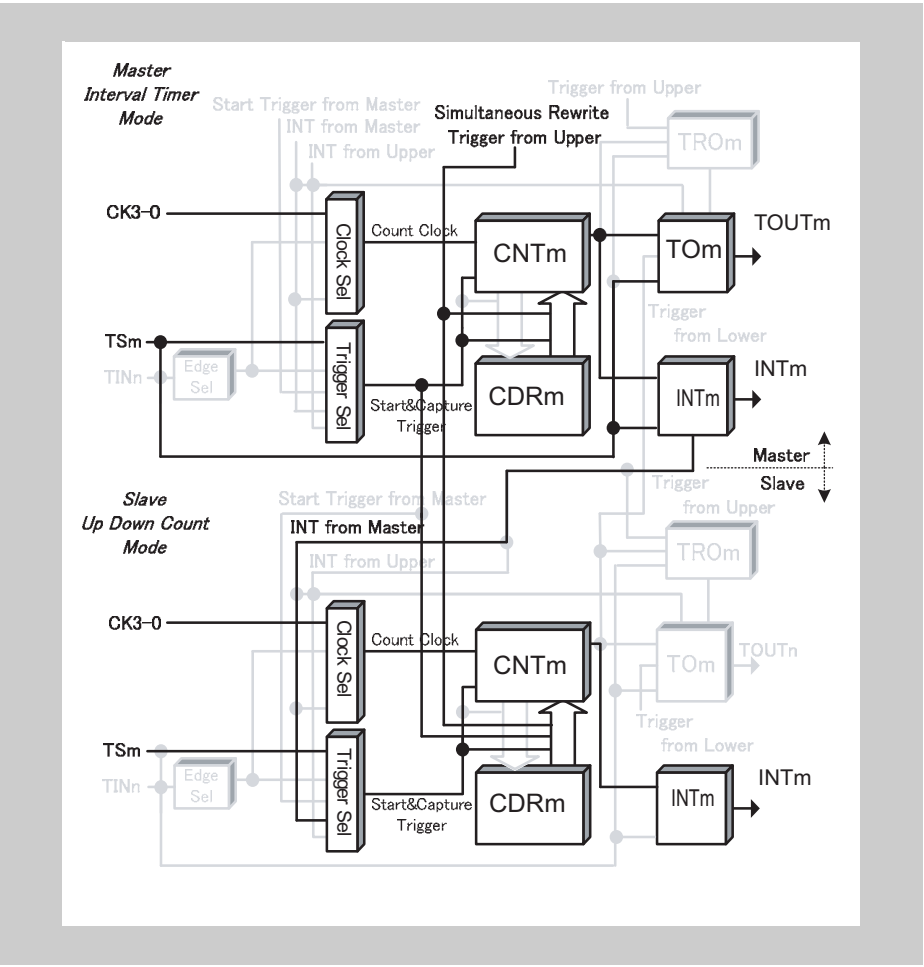


Figure 16-97 Block diagram for AD Conversion Trigger Output Function Type 2

The following settings apply to the general timing diagram:

- Master channel
  - INTTAUBnIm is generated at operation start (TAUBnCMORm.MD0 = 1)

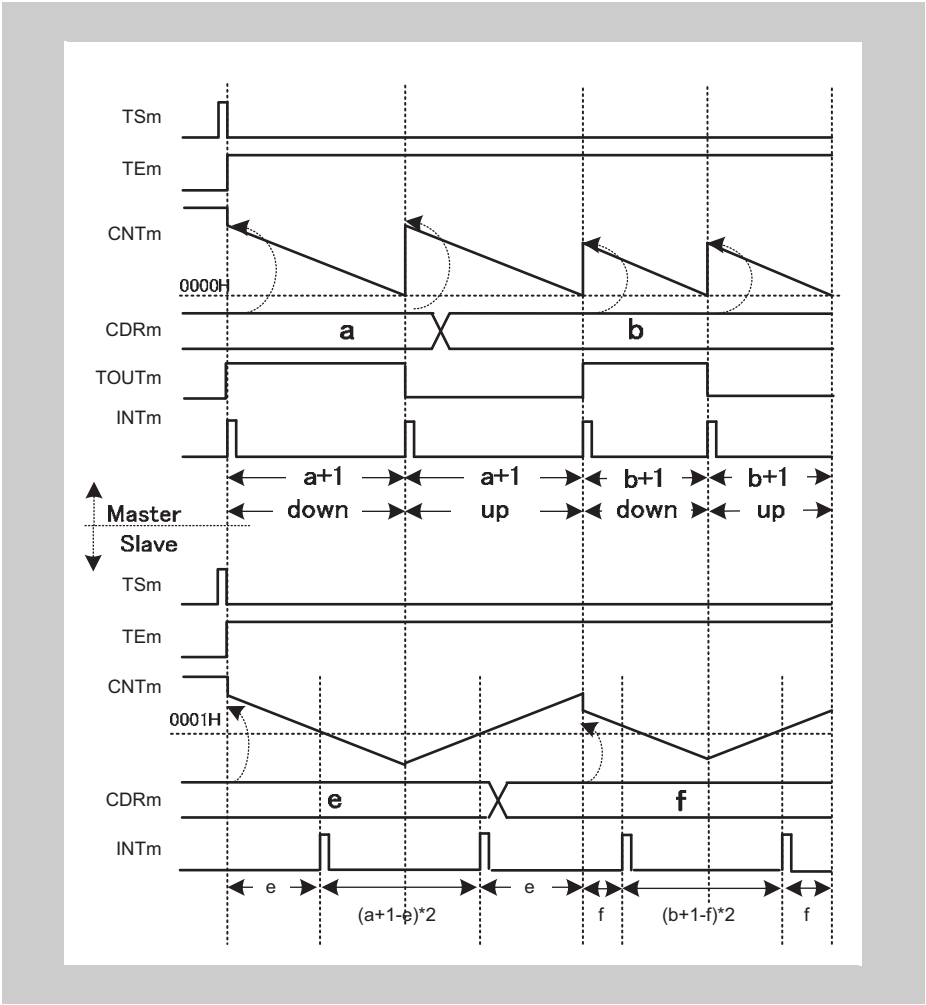


Figure 16-98 General timing diagram for AD Conversion Trigger Output Function Type 2

16.22 Registers

This section contains a description of all the registers of the 16-bit Timer Array Unit B.

16.22.1 TAUBn registers overview

The TAUBn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 15.

Table 16-130 TAUBn registers overview

Register name	Shortcut	Address
TAUBn prescaler registers		
TAUBn prescaler clock select register	TAUBnTPS	<TAUBn_base> + 240_H
TAUBn control registers		
TAUBn channel data register m	TAUBnCDRm	<TAUBn_base> + m x 4_H
TAUBn channel counter register m	TAUBnCNTm	<TAUBn_base> + 80_H + m x 4_H
TAUBn channel mode OS register m	TAUBnCMORm	<TAUBn_base> + 200_H + m x 4_H
TAUBn channel mode user register m	TAUBnCMURm	<TAUBn_base> + $C0_H$ + m x 4_H
TAUBn channel status register m	TAUBnCSRm	<TAUBn_base> + 140_H + m x 4_H
TAUBn channel status clear trigger register m	TAUBnCSCm	<TAUBn_base> + 180_H + m x 4_H
TAUBn channel start trigger register	TAUBnTS	<TAUBn_base> + $1C4_H$
TAUBn channel enable status register	TAUBnTE	<TAUBn_base> + $1C0_H$
TAUBn channel stop trigger register	TAUBnTT	<TAUBn_base> + $1C8_H$
TAUBn output registers		
TAUBn channel output enable register	TAUBnTOE	<TAUBn_base> + $5C_H$
TAUBn channel output register	TAUBnTO	<TAUBn_base> + 58_H
TAUBn channel output mode register	TAUBnTOM	<TAUBn_base> + 248_H
TAUBn channel output configuration register	TAUBnTOC	<TAUBn_base> + $24C_H$
TAUBn channel output active level register	TAUBnTOL	<TAUBn_base> + 040_H
TAUBn channel dead time output enable register	TAUBnTDE	<TAUBn_base> + 250_H
TAUBn channel dead time output level register	TAUBnTDL	<TAUBn_base> + 54_H
TAUBn reload data registers		
TAUBn channel reload data enable register	TAUBnRDE	<TAUBn_base> + 260_H
TAUBn channel reload data mode register	TAUBnRDM	<TAUBn_base> + 264_H
TAUBn channel reload data control CH select register	TAUBnRDS	<TAUBn_base> + 268_H
TAUBn channel reload data control register	TAUBnRDC	<TAUBn_base> + $26C_H$
TAUBn channel reload data trigger register	TAUBnRDT	<TAUBn_base> + 44_H
TAUBn channel reload status register	TAUBnRSF	<TAUBn_base> + 48_H
TAUBn emulation register		
TAUBn emulation register	TAUBnEMU	<TAUBn_base> + 290_H

<TAUBn_base> The <TAUBn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

16.22.2 TAUBn prescaler registers details

(1) TAUBnTPS - TAUBn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3 for all channels.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 240_H

Initial Value $FFFF_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS3[3:0]				PRS2[3:0]				PRS1[3:0]				PRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-131 TAUBnTPS register contents (1/2)

Bit position	Bit name	Function
15 to 12	PRS3[3:0]	Specifies the CK3 clock. **PRS3[3:0] / CK3 clock** 0000_B / PCLK/2^0 0001_B / PCLK/2^1 0010_B / PCLK/2^2 0011_B / PCLK/2^3 ... / ... 1110_B / PCLK/2^{14} 1111_B / PCLK/2^{15} These bits can only be rewritten when all counters using CK3 are stopped (TAUBnTE.TEm = 0).
11 to 8	PRS2[3:0]	Specifies the CK2 clock. **PRS2[3:0] / CK2 clock** 0000_B / PCLK/2^0 0001_B / PCLK/2^1 0010_B / PCLK/2^2 0011_B / PCLK/2^3 ... / ... 1110_B / PCLK/2^{14} 1111_B / PCLK/2^{15} These bits can only be rewritten when all counters using CK2 are stopped (TAUBnTE.TEm = 0).

Table 16-131 TAUBnTPS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUBnTE.TEm = 0).</td></tr>
<tr><td>3 to 0</td><td>PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUBnTE.TEm = 0).</td></tr>
</table>

Note The TAUBn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

16.22.3 TAUBn control registers details

(1) TAUBnCDRm - TAUBn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUBnCMORm.MD[4:0].

Access This register can be read/written in 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUBn_base> + 0_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDR[15:0]															
R/W															

Table 16-132 TAUBnCDRm register contents

Bit position	Bit name	Function
15 to 0	CDR[15:0]	Data register for the capture/compare value.

(2) TAUBnCNTm - TAUBn channel counter register

This register is the channel m counter register.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + 80_H + m x 4_H

Initial Value 0000_H or $FFFF_H$ The initial value depends on the operation mode, see *Table 16-134 "TAUBnCNTm read values after the counter is re-enabled" on page 1309* .

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R															

Table 16-133 TAUBnCNTm register contents

Bit position	Bit name	Function
15 to 0	CNT[15:0]	16-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUBnTS.TSm and TAUBnTT.TTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUBnTT.TTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUBnTE.TEm= 0) and re-enabled (TAUBnTS.TSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUBnTS.TSm = 1) for modes where the counter waits for a start trigger.

Table 16-134 TAUBnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUBnCNTm value		
		After reset	After stop trigger	After one count
Interval Timer mode	Count down	$FFFF_H$	Stop value	-
Judge mode	Count down	$FFFF_H$	Stop value	-
Capture mode	Count up	0000_H	Stop value	-
Event Count mode	Count down	$FFFF_H$	Stop value	-
One Count mode	Count down	$FFFF_H$	Stop value	$FFFF_H$
Capture & One Count mode	Count up	0000_H	Stop value	Captured value + 1 (TAUBnCDRm)
Judge & One Count mode	Count down	$FFFF_H$	Stop value	TAUBnCNTm value - 1
Up Down Count mode	Count up/down	$FFFF_H$	Stop value	-
Pulse One Count mode	Count down	$FFFF_H$	Stop value	0000_H
Count Capture Mode	Count up	0000_H	Stop value	-
Gate Count Mode	Count down	$FFFF_H$	Stop value	Stop value
Capture & Gate Count Mode	Count up	0000_H	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUBnCMORm.MD[4:0].

(3) TAUBnCMORm - TAUBn channel mode OS register

This register controls channel m operation. It specifies, for example, the operation clock, count clock, and master/slave function.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 200_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	CCS0	MAS	STS[2:0]			COS[1:0]		-	MD[4:0]				
R/W		R	R/W	R/W	R/W			R/W		R	R/W				

Table 16-135 TAUBnCMORm register contents (1/3)

Bit position	Bit name	Function
15,14	CKS[1:0]	Selects the operation clock. The operation clock is used for the TAUBnTTINm input edge detection circuit. It can also be used as the count clock depending on bit TAUBnCMORm.CCS0.
12	CCS0	Selects the count clock for TAUBnCNTm counter: 0: Operation clock as specified by TAUBnCMORm.CKS[1:0] 1: Valid edge of TAUBnTTINm input signal
11	MAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

CKS1	CKS0	Selected operation clock
0	0	CK0
0	1	CK1
1	0	CK2
1	1	CK3

Table 16-135 TAUBnCMORm register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>STS[2:0]</td><td>Selects the external start trigger:
<table>
<tr><th>STS2</th><th>STS1</th><th>STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUBnTTINm input signal. TAUBnCMURm.TIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUBnTTINm input signal is the start trigger and the reverse edge is the stop (capture) trigger</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INTTAUBnI of the master channel</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>INTTAUBnI of the upper channel (m-1), regardless of the master setting</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>Dead-time output signal TAUBnTTDL of the TAUBnTTOUTm generation unit</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>Up/down output trigger signal TAUBnTUDSm of the master channel.</td></tr>
</table></td></tr>
<tr><td>7, 6</td><td>COS[1:0]</td><td>Specifies when the capture register TAUBnCDRm and the overflow flag TAUBnCSRm.OVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>COS1</th><th>COS0</th><th>Capture register</th><th>TAUBnCSRm.OVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUBnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUBnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUBnCSRm.OVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUBnCSR.OVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUBnTTINm input valid edge and upon counter overflow:
• TAUBnTTINm input valid edge: Counter value is written to TAUBnCDRm
• Overflow: $FFFF_H$ is written to TAUBnCDRm. The next TAUBnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
</table></td></tr>
</table>

Table 16-135 TAUBnCMORm register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>4 to 0</td><td>MD[4:0]</td><td>Specifies the operation mode.
<table>
<tr><th>MD4</th><th>MD3</th><th>MD2</th><th>MD1</th><th>MD0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1/0</td><td>Interval Timer mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1/0</td><td>Judge mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Capture mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>Event Count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1/0</td><td>One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Capture & One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>1/0</td><td>Judge & One Count mode</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Up Down Count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Pulse One Count mode</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Count Capture mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>Gate Count mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Capture & Gate Count mode</td></tr>
</table>
</td></tr>
</table>

Mode	Role of the MD0 bit
Interval Timer mode Capture mode Count Capture mode	Specifies whether an INTTAUBnIm is generated when the counter is triggered: 0: No INTTAUBnIm generated 1: INTTAUBnIm generated
Event Count mode Up Down Count mode	This bit must be set to 0: 0: No INTTAUBnIm generated when the counter is triggered 1: -
One Count mode Gate Count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUBnIm and TAUBnTTOUTm are not output when the counter is triggered.
Pulse One Count mode	Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUBnIm and TAUBnTTOUTm are output when the counter is triggered.
Capture & One Count mode Capture & Gate Count mode	This bit must be set to 0: 0: No INTTAUBnIm generated when the counter is triggered. Start trigger is disabled during counting. 1: -
Judge mode Judge One Count mode	Specifies when INTTAUBnIm is generated: 0: When TAUBnCNTm ≤ TAUBnCDRm 1: When TAUBnCNTm > TAUBnCDRm

(4) TAUBnCMURm - TAUBn channel mode user register

This register specifies the type of valid edge detection used for the TAUBnTTINm input.

Access This register can be read/written in 8-bit units.

Address <TAUBn_base> + $C0_H$ + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	TIS[1:0]	
R	R	R	R	R	R	R/W	R/W

Table 16-136 TAUBnCMURm register contents

Bit position	Bit name	Function
1, 0	TIS[1:0]	Specifies the valid edge of the TAUBnTTINm input: TIS1 / TIS0 / Description 0 / 0 / Falling edge 0 / 1 / Rising edge 1 / 0 / Rising and falling edges (low-width measurement selection). Start trigger: falling edge. Stop trigger (capture): rising edge 1 / 1 / Rising and falling edges (high-width measurement selection). Start trigger: rising edge. Stop trigger (capture): falling edge. This setting is only valid if the start trigger selection is set to TAUBnCMORm.STS[2:0] = 010_B • To detect rising and falling edges when TAUBnCMORm.STS[2:0] is not set to 010_B, set TAUBnCMURm.TIS[1:0] = 10_B. • Edge detection for TAUBnTTINm input signals is performed based on the operation clock selected by TAUBnCMORm.CKS[1:0].

(5) TAUBnCSRm - TAUBn channel status register

This register indicates the count direction and the overflow status of channel m's counter.

Access This register can be read in 8-bit units.

Address <TAUBn_base> + 140_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	CSF	OVF
R	R	R	R	R	R	R	R

Table 16-137 TAUBnCSRm register contents

Bit position	Bit name	Function
1	CSF	Indicates the count direction: 0: Counts up 1: Counts down The read value of this bit is only valid in the following mode: • Up Down Count mode For channel 0 this bit is fixed to 0.
0	OVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & One Count mode • Count Capture mode • Capture & Gate Count mode The function of this bit depends on the setting of control bits TAUBnCMORm.COS[1:0]. OVF is not be set when TAUBnCMORm.COS[1:0] = 10_B.

(6) TAUBnCSCm - TAUBn channel status clear register

This register is a trigger register for clearing the overflow flag TAUBnCSRm.OVF of a channel m.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUBn_base> + 180_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	0	CLOV
R	R	R	R	R	R	R	W

Table 16-138 TAUBnCSCm register contents

Bit position	Bit name	Function
0	CLOV	0: No function 1: Clears the overflow flag TAUBnCSRm.OVF

(7) TAUBnTS - TAUBn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + $1C4_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS 15	TS 14	TS 13	TS 12	TS 11	TS 10	TS 09	TS 08	TS 07	TS 06	TS 05	TS 04	TS 03	TS 02	TS 01	TS 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-139 TAUBnTS register contents

Bit position	Bit name	Function
15 to 0	TSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUBnTE.TEm = 1. When the counter is enabled, this bit immediately returns to 0. TAUBnTE.TEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUBnTE - TAUBn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + $1C0_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE15	TE14	TE13	TE12	TE11	TE10	TE09	TE08	TE07	TE06	TE05	TE04	TE03	TE02	TE01	TE00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 16-140 TAUBnTE register contents

Bit position	Bit name	Function
15 to 0	TEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting TAUBnTS.TSm to 1 or trigger input detection TAUBnTSSTm = 1 sets this bit to 1. Setting TAUBnTT.TTm to 1 resets this bit to 0.

(9) TAUBnTT - TAUBn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + $1C8_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TT15	TT14	TT13	TT12	TT11	TT10	TT09	TT08	TT07	TT06	TT05	TT04	TT03	TT02	TT01	TT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-141 TAUBnTT register contents

Bit position	Bit name	Function
15 to 0	TTm	Stops the counter of channel m: 0: No function 1: Stops the counter and sets TAUBnTE.TEm = 0. When the counter has stopped, this bit immediately returns to 0. TAUBnCNTm stops counting and TAUBnCNTm, TAUBnTO.TOm, and TAUBnTTOUTm all retain the values they had before the counter was stopped.

16.22.4 TAUBn output registers details

(1) TAUBnTOE - TAUBn channel output enable register

This register enables and disables Direct Channel Output Mode.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + $5C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOE 15	TOE 14	TOE 13	TOE 12	TOE 11	TOE 10	TOE 09	TOE 08	TOE 07	TOE 06	TOE 05	TOE 04	TOE 03	TOE 02	TOE 01	TOE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-142 TAUBnTOE register contents

Bit position	Bit name	Function
15 to 0	TOEm	Enables/disables Direct Channel Output Mode: 0: Enables Direct Channel Output Mode (TAUBnTTOUT m output is controlled by the application software) 1: Disables Direct Channel Output Mode (TAUBnTTOUT m output is controlled by the timer)

(2) TAUBnTOM - TAUBn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 248_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM 15	TOM 14	TOM 13	TOM 12	TOM 11	TOM 10	TOM 09	TOM 08	TOM 07	TOM 06	TOM 05	TOM 04	TOM 03	TOM 02	TOM 01	TOM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-143 TAUBnTOM register contents

Bit position	Bit name	Function
15 to 0	TOMm	Specifies the timer controlled channel output mode, if Direct Channel Output Mode is disabled (TAUBnTOE.TOEm = 1): 0: Independent Channel Output Mode 1: Synchronous Channel Output Mode The output mode depends on several channel output control bits, as can be seen in *Table 16-11 “Channel output modes” on page 1123* .

(3) TAUBnTOC - TAUBn channel output configuration register

This register specifies the output mode of each channel in combination with TAUBnTOMm.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + $24C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOC 15	TOC 14	TOC 13	TOC 12	TOC 11	TOC 10	TOC 09	TOC 08	TOC 07	TOC 06	TOC 05	TOC 04	TOC 03	TOC 02	TOC 01	TOC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-144 TAUBnTOC register contents

Bit position	Bit name	Function
15 to 0	TOCm	Specifies the output mode: 0: Operation mode 1 1: Operation mode 2 The output mode also depends on TAUBnTOM.TOMm, as can be seen in the following table.

TOMm	TOCm	Description
0	0	Toggle mode: TAUBnTTOUTm toggles when INTTAUBnIm occurs.
	1	Set/reset mode: TAUBnTTOUTm set when INTTAUBnIm occurs upon count start and reset when INTTAUBnIm occurs due to detection of a match between TAUBnCNTm and TAUBnCDRm, or, in One-Shot Pulse Output Function when the counter is 0.
1	0	Synchronous Channel Operation Mode 1: TAUBnTTOUTm set when INTTAUBnI occurs on the master channel and reset when INTTAUBnI occurs on the slave channel.
	1	Synchronous Channel Operation Mode 2: TAUBnTTOUTm set when INTTAUBnIm occurs while the slave channel is counting down and reset when INTTAUBnIm occurs while the slave channel is counting up.

(4) TAUBnTDE - TAUBn channel dead time output enable register

This register enables/disables dead time operation for each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUBnTE.TEm = 0).

Address <TAUBn_base> + 250_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDE 15	TDE 14	TDE 13	TDE 12	TDE 11	TDE 10	TDE 09	TDE 08	TDE 07	TDE 06	TDE 05	TDE 04	TDE 03	TDE 02	TDE 01	TDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-145 TAUBnTDE register contents

Bit position	Bit name	Function
15 to 0	TDEm	Enables/disables dead time control operation of channel m: 0: Disables dead time operation 1: Enables dead time operation The same settings must be set for the even and the odd slave channel that comprise a set. These bits only apply when: • TAUBnTOE.TOEm, TAUBnTOM.TOMm, and TAUBnTOC.TOCm = 1.

(5) TAUBnTDL - TAUBn channel dead time output level register

This register selects the phase period to which dead time is added.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 54_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDL 15	TDL 14	TDL 13	TDL 12	TDL 11	TDL 10	TDL 09	TDL 08	TDL 07	TDL 06	TDL 05	TDL 04	TDL 03	TDL 02	TDL 01	TDL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-146 TAUBnTDL register contents

Bit position	Bit name	Function
15 to 0	TDLm	Selects the phase period to which dead time is added: 0: Positive phase period 1: Negative phase period These bits only apply when: • TAUBnTOE.TOEm, TAUBnTOM.TOMm, TAUBnTOC.TOCm, and TAUBnTDE.TDEm = 1.

16.22.5 TAUBn channel output level registers details

(1) TAUBnTO - TAUBn channel output register

This register specifies and reads the level of TAUBnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 58_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO09	TO08	TO07	TO06	TO05	TO04	TO03	TO02	TO01	TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-147 TAUBnTO register contents

Bit position	Bit name	Function
15 to 0	TOm	Specifies/reads the level of TAUBnTTOUTm: 0: Low 1: High Only TOm bits for which Independent Channel Output function is disabled (TAUBnTOEm = 0) can be written.

(2) TAUBnTOL - TAUBn channel output level register

This register specifies the output logic of the channel output bit (TAUBnTO.TOm).

Access This register can be read/written in 16-bit units.

Address <TAUBn_base> + 040_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOL 15	TOL 14	TOL 13	TOL 12	TOL 11	TOL 10	TOL 09	TOL 08	TOL 07	TOL 06	TOL 05	TOL 04	TOL 03	TOL 02	TOL 01	TOL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-148 TAUBnTOL register contents

Bit position	Bit name	Function
15 to 0	TOLm	Specifies the output logic of the channel m output bit (TAUBnTO.TOm): 0: Positive logic (active high) 1: Inverted logic (active low) These bits apply in all channel output modes except Direct Channel Output Mode and channel output modes with real-time output.

16.22.6 TAUBn simultaneous rewrite register details

(1) TAUBnRDE - TAUBn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUBnCDRm. It also enables simultaneous rewrite of the data register TAUBnTOLm for the PWM output function and the triangle PWM output function.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 260_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDE 15	RDE 14	RDE 13	RDE 12	RDE 11	RDE 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-149 TAUBnRDE register contents

Bit position	Bit name	Function
15 to 0	RDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUBnRDM - TAUBn channel reload data mode register

This register selects when the signal that controls simultaneous rewrite is loaded.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 264_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDM 15	RDM 14	RDM 13	RDM 12	RDM 11	RDM 10	RDM 09	RDM 08	RDM 07	RDM 06	RDM 05	RDM 04	RDM 03	RDM 02	RDM 01	RDM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-150 TAUBnRDM register contents

Bit position	Bit name	Function
15 to 0	RDMm	Selects when the signal that triggers simultaneous is generated: 0: When the master channel counter starts counting 1: At the top of a triangle wave cycle These bits only apply when TAUBnRDE.RDEm = 1 and TAUBnRDS.RDSm = 0.

(3) TAUBnRDS - TAUBn channel reload data control channel select register

This register selects the control channel for simultaneous rewrite.

Access This register can be read/written in 16-bit or 1-bit units. It can only be written when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 268_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDS 15	RDS 14	RDS 13	RDS 12	RDS 11	RDS 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-151 TAUBnRDS register contents

Bit position	Bit name	Function
15 to 0	RDSm	Specifies which channel is monitored for the simultaneous rewrite trigger: 0: Master channel 1: Another upper channel

(4) TAUBnRDC - TAUBn channel reload data control register

This register specifies the channel that generates the INTTAUBnIm signal that triggers simultaneous rewrite.

Access This register can be read/written in 16-bit units. It can only be written when TAUBnTE.TEm = 0

Address <TAUBn_base> + $26C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDC 15	RDC 14	RDC 13	RDC 12	RDC 11	RDC 10	RDC 09	RDC 08	RDC 07	RDC 06	RDC 05	RDC 04	RDC 03	RDC 02	RDC 01	RDC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-152 TAUBnRDC register contents

Bit position	Bit name	Function
15 to 0	RDCm	Specifies whether the channel is monitored for an INTTAUBnIm signal that is used as the simultaneous rewrite trigger: 0: Channel is not monitored 1: Channel is monitored These bits only apply when TAUBnRDS.RDSm = 1.

(5) TAUBnRDT - TAUBn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUBn_base> + 044_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT 15	RDT 14	RDT 13	RDT 12	RDT 11	RDT 10	RDT 09	RDT 08	RDT 07	RDT 06	RDT 05	RDT 04	RDT 03	RDT 02	RDT 01	RDT 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 16-153 TAUBnRDT register contents

Bit position	Bit name	Function
15 to 0	RDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUBnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger. TAUBnRDT.RDTm immediately returns to 0.

(6) TAUBnRSF - TAUBn channel reload status register

This flag register indicates that simultaneous rewrite is possible.

Access This register can be read in 16-bit units.

Address <TAUBn_base> + 048_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSF 15	RSF 14	RSF 13	RSF 12	RSF 11	RSF 10	RSF 09	RSF 08	RSF 07	RSF 06	RSF 05	RSF 04	RSF 03	RSF 02	RSF 01	RSF 00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 16-154 TAUBnRSF register contents

Bit position	Bit name	Function
15 to 0	RSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewrite disabled 1: Simultaneous rewrite enabled

16.22.7 TAUBn emulation register

(1) TAUBnEMU - TAUB emulation register

This register controls whether the TAUBn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units only when TAUBnTE.TEm = 0.

Address <TAUBn_base> + 290_H

Initial Value 00_H

7	6	5	4	3	2	1	0
TAUBn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 16-155 TAUBnEMU register contents

Bit position	Bit name	Function
7	TAUBn SVSDIS	Emulation control 0: TAUBn can be stopped during emulation 1: TAUBn continuous operating during emulation

Chapter 17 Timer Array Unit C (TAUC)

This chapter contains a generic description of the Timer Array Unit C (TAUC).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

17.1 V850E2/Fx4-H TAUC Features

Instances

This microcontroller has following number of instances of the Timer Array Unit C.

Table 17-1 Instances of TAUC

Timer Array Unit C	V850E2/FK4-H	V850E2/FL4-H
Instance	4	6
Name	TAUC3 to TAUC6	TAUC3 to TAUC7

Instances index n

Throughout this chapter, the individual instances of a Timer Array Unit C is identified by the index "n" (n = 3 to 7), for example, TAUCnTOM for the TAUCn channel output mode register.

Channel index m

The Timer Array Unit C has 16 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 15), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2, 4, 6, 8, 10, 12, 14) are denoted as CHm_even.
The odd numbered channels (m = 1, 3, 5, 7 , 9, 11, 13, 15) are denoted as CHm_odd.

Register addresses

All TAUCn register addresses are given as address offsets to the individual base address <TAUCn_base>.
The base address <TAUCn_base> of each TAUCn is listed in the following table:

Table 17-2 Register base addresses <TAUCn_base>

TAUCn instance	<TAUCn_base> address
TAUC3	FF80 $B000_H$
TAUC4	FF80 $C000_H$
TAUC5	FF80 $D000_H$
TAUC6	FF80 $E000_H$
TAUC7	FF80 $F000_H$

Clock supply All Timer Array Units C provide one clock input:

Table 17-3 TAUCn clock supply

TAUCn instance	TAUCn clock	Connected to
TAUC3	PCLK	Clock Controller CKSCLK_106
TAUC4	PCLK	Clock Controller CKSCLK_105
TAUC5	PCLK	Clock Controller CKSCLK_111
TAUC6	PCLK	Clock Controller CKSCLK_111
TAUC7	PCLK	Clock Controller CKSCLK_111

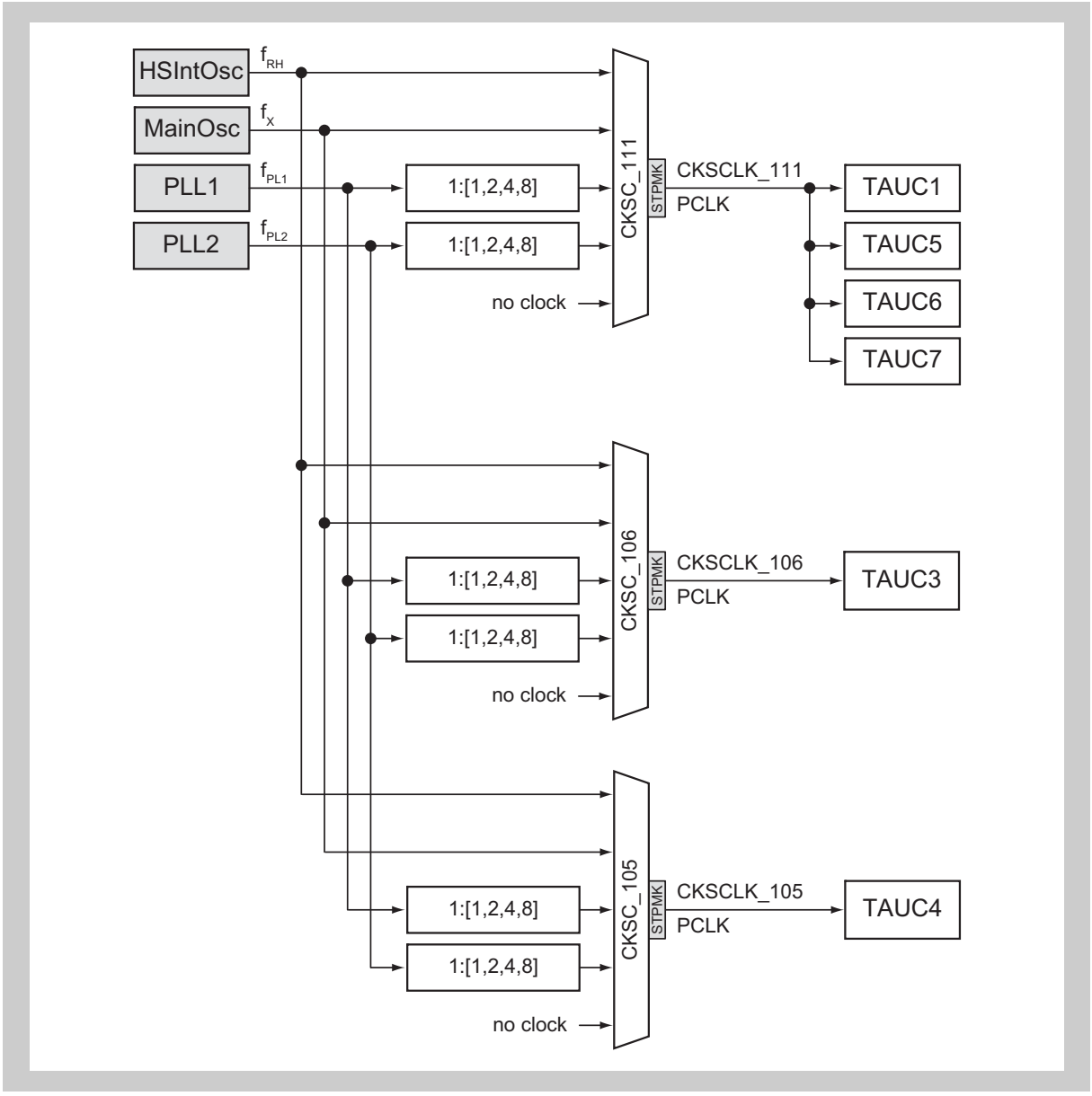


Figure 17-1 TAUC clock supply

Interrupts The Timer Array Unit C can generate the following interrupt requests:

Table 17-4 TAUCn interrupt requests

TAUCn signals	Function	Connected to
TAUC3:		
INTTAUC3I0 to INTTAUC3I5	Channel 0 to 15 interrupt	Interrupt Controller INTTAUC3I0 to INTTAUC3I15
TAUC4:		
INTTAUC4I0 to INTTAUC4I5	Channel 0 to 15 interrupt	Interrupt Controller INTTAUC4I0 to INTTAUC4I15
TAUC5:		
INTTAUC5I0 to INTTAUC5I5	Channel 0 to 15 interrupt	not connected
TAUC6:		
INTTAUC6I0 to INTTAUC6I5	Channel 0 to 15 interrupt	not connected
TAUC7:		
INTTAUC7I0 to INTTAUC7I5	Channel 0 to 15 interrupt	not connected

TAUC H/W reset The Time Array Units C and their registers are initialized by the following reset signal:

Table 17-5 TAUCn reset signal

TAUCn	Reset signal
TAUCn	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

Output signals The output signals of the Timer Array Unit C are used for various purposes, as listed in the following table.

Note √ / – : used / not used for the function

Table 17-6 TAUC3 output signals

TAUC3 signal	Connected to	
	Port	PWM delay[a]
TAUC3TTOUT0	–	–
TAUC3TTOUT1	TAUC3O1	√
TAUC3TTOUT2	TAUC3O2	√
TAUC3TTOUT3	–	–
TAUC3TTOUT4	–	–
TAUC3TTOUT5	TAUC3O5	√
TAUC3TTOUT6	TAUC3O6	√
TAUC3TTOUT7	–	–
TAUC3TTOUT8	–	–
TAUC3TTOUT9	TAUC3O9	√
TAUC3TTOUT10	TAUC3O10	√
TAUC3TTOUT11	–	–
TAUC3TTOUT12	–	–
TAUC3TTOUT13	TAUC3O13	√
TAUC3TTOUT14	TAUC3O14	√
TAUC3TTOUT15	–	–

a) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Table 17-7 TAUC4 output signals

TAUC4 signal	Connected to	
	Port	PWM delay[a]
TAUC4TTOUT0	–	–
TAUC4TTOUT1	TAUC4O1	√
TAUC4TTOUT2	TAUC4O2	√
TAUC4TTOUT3	–	–
TAUC4TTOUT4	–	–
TAUC4TTOUT5	TAUC4O5	√
TAUC4TTOUT6	TAUC4O6	√
TAUC4TTOUT7	–	–
TAUC4TTOUT8	–	–
TAUC4TTOUT9	TAUC4O9	√
TAUC4TTOUT10	TAUC4O10	√
TAUC4TTOUT11	–	–
TAUC4TTOUT12	–	–
TAUC4TTOUT13	TAUC4O13	√
TAUC4TTOUT14	TAUC4O14	√
TAUC4TTOUT15	–	–

a) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Table 17-8 TAUC5 output signals

TAUC5 signal	Connected to	
	Port	PWM delay[a]
TAUC5TTOUT0	–	–
TAUC5TTOUT1	TAUC5O1	√
TAUC5TTOUT2	TAUC5O2	√
TAUC5TTOUT3	–	–
TAUC5TTOUT4	–	–
TAUC5TTOUT5	TAUC5O5	√
TAUC5TTOUT6	TAUC5O6	√
TAUC5TTOUT7	–	–
TAUC5TTOUT8	–	–
TAUC5TTOUT9	TAUC5O9	√
TAUC5TTOUT10	TAUC5O10	√
TAUC5TTOUT11	–	–
TAUC5TTOUT12	–	–
TAUC5TTOUT13	TAUC5O13	√
TAUC5TTOUT14	TAUC5O14	√
TAUC5TTOUT15	–	–

a) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Table 17-9 TAUC6 output signals

TAUC6 signal	Connected to	
	Port	PWM delay[a]
TAUC6TTOUT0	–	–
TAUC6TTOUT1	TAUC6O1	√
TAUC6TTOUT2	TAUC6O2	√
TAUC6TTOUT3	–	–
TAUC6TTOUT4	–	–
TAUC6TTOUT5	TAUC6O5	√
TAUC6TTOUT6	TAUC6O6	√
TAUC6TTOUT7	–	–
TAUC6TTOUT8	–	–
TAUC6TTOUT9	TAUC6O9	√
TAUC6TTOUT10	TAUC6O10	√
TAUC6TTOUT11	–	–
TAUC6TTOUT12	–	–
TAUC6TTOUT13	TAUC6O13	√
TAUC6TTOUT14	TAUC6O14	√
TAUC6TTOUT15	–	–

a) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

Table 17-10 TAUC7 output signals

TAUC7 signal	Connected to	
	Port	PWM delay[a]
TAUC7TTOUT0	–	–
TAUC7TTOUT1	TAUC7O1	√
TAUC7TTOUT2	TAUC7O2	√
TAUC7TTOUT3	–	–
TAUC7TTOUT4	–	–
TAUC7TTOUT5	TAUC7O5	√
TAUC7TTOUT6	TAUC7O6	√
TAUC7TTOUT7	–	–
TAUC7TTOUT8	–	–
TAUC7TTOUT9	TAUC7O9	√
TAUC7TTOUT10	TAUC7O10	√
TAUC7TTOUT11	–	–
TAUC7TTOUT12	–	–
TAUC7TTOUT13	TAUC7O13	√
TAUC7TTOUT14	TAUC7O14	√
TAUC7TTOUT15	–	–

a) Refer to chapter *22 “PWM Diagnostic” on page 1690* for further details.

17.2 Functional Overview

Features summary

The TAUC has the following functions:

- 16 channels
- 16-bit counter and 16-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of PWM output signal
- Interrupt generation

The following figure shows the main components of the TAUC:

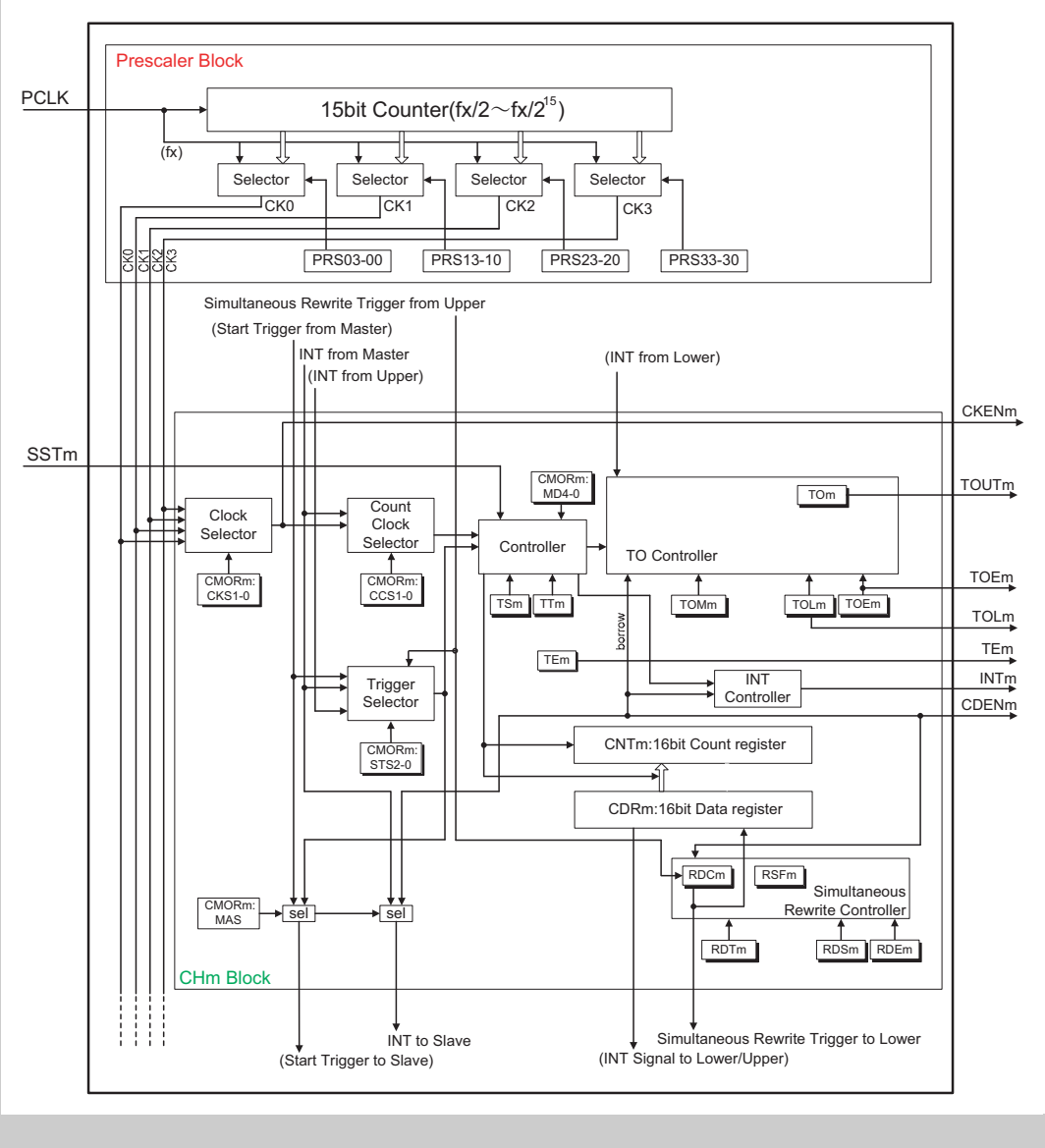


Figure 17-2 Block diagram of the TAUC

The prefix “TAUCn” has been omitted from the register names for the sake of clarity in the above figure.

17.2.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.

  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.

  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.

  Examples is “Interval Timer Mode”.

- **Channel output mode**

  The channel output mode defines the operation of TAUCnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Examples is “Independent Channel Output Mode”.

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

  It defines the operation mode, start trigger, count clock and the channel output mode.

  Example is “PWM Output Function”

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 5, channel 3 is an upper channel and channel 9 is a lower channel.

17.3 Functional Description

The Timer Array Unit C is used to perform count or timer operations and to output PWM signals which depend on the result of the operation. It contains one prescaler block for count clock generation and 16 channels, each equipped with a 16 bit counter TAUCnCNTm and a 16-bit data register TAUCnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in different operation modes, either independently or in combination with other channels (synchronously), i.e. multiple channels depend on each other with one master and one or more slave channels.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler block

The prescaler block provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

Count clocks CK0 to CK3 are derived from PCLK by a configurable prescaler division factor of 2^0 to 2^{15}.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUCnIm from master channel

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUCnCMORm.MD[4:0])
- Counter start enable (TAUCnTS.TSm) and counter stop (TAUCnTT.TTm)

  When counter start is enabled, status flag TAUCnTE.TEm is set.

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUCnTE.TEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUCnTSSTm
- INTTAUCnIm from the master

Simultaneous rewrite controller

Simultaneous rewrite control is a special function that can be used in synchronous operation modes. The data registers of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller ensures that new data register values of all channels become effective at the same time.

TAUCnTO Controller

The output control of every channel enables the generation of various output signal forms such as PWM signals.

Signals

The TAUC has various input and output signals. A full list can be found in the first section of this chapter under the keyword “I/O signals”.

17.4 General Operating Procedure

The following lists the general operation procedure for the TAUCn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. All circuits and registers of all channels are initialized. The control register of TAUCnTTOUTm is also initialized and outputs a low level.

1. Set the TAUCnTPS register to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUCn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUCnTS.TSm bit to 1.
   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
   The function is in operation.
4. If desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUCnTT.TTm bit to 0.

Note A detailed description of the required control bits and the operation of the individual functions is given in *17.11 “Independent Channel Operation Functions” on page 1354* .

17.5 Operation Modes

The TAUC contains 2 operation modes. These determine the basic behavior of a channel.

One operation mode can be set for each channel. It is specified using the TAUCnCMOR.MD[4:0] bits.

When choosing a function, these settings cannot be specified individually, but are grouped under the term operation mode. If a function uses multiple channels, the operation mode of each channel must be set correctly for the function to work correctly.

For more information about the operation modes required by each function, refer to the required function in *17.11 “Independent Channel Operation Functions” on page 1354* .

17.6 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *17.6.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *17.6.2 "Simultaneous start and stop of synchronous channel counters" on page 1339*
- *17.7 "Simultaneous Rewrite" on page 1340*

17.6.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2, CH4, …) can be set as master channels. Any channel apart from CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 and the lower channels (CH4, CH5, …) can be set as slave channels.
- If multiple master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH4 are master channels, CH1 to CH3 can be set as slave channels for CH0, but CH5 to CH7 cannot.

Operation clock

- The same operation clock must be set for the slave channel and the master channel. This is achieved using the TAUCnCMORm.CKS[1:0] bits of the slave and master channel.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

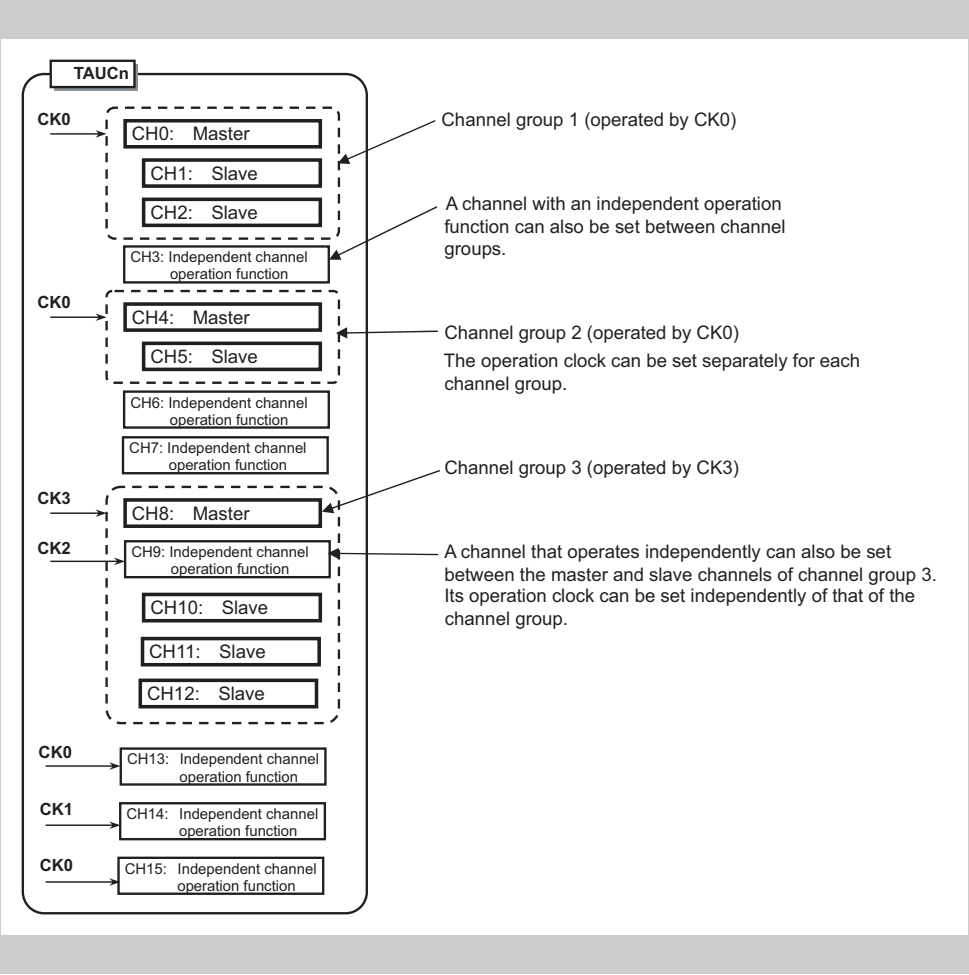


Figure 17-3 Grouping of the channels and assignment of operation clocks

INTTAUCnIm, start trigger, and count clock

- Master channels can transfer an interrupt request (INTTAUCnI), the start trigger, and the count clock to slave channels.
- Slave channels can use INTTAUCnI, the start trigger, and the count clock of the master channels but cannot transfer their INTTAUCnI, start trigger, or count clock to the lower channels.
- A master channel cannot use INTTAUCnI, the start trigger, or the count clock of the higher master channels.

17.6.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a TAUC unit, and between TAUC units.

(1) Simultaneous start and stop within a TAUC unit

- To simultaneously start synchronized channels, the TAUCnTS.TSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUCnTT.TTm bits of the channels must be set at the same time.

Writing to the TAUCnTS.TSm bits sets the corresponding TAUCnTE.TEm bits to 1, enabling counting. TAUCnTS.TSm = 1 only enables the corresponding counter to start; the exact time that it starts depends on the operation mode.

(2) Simultaneous start between TAUC units

Counters in different TAUC units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal. The simultaneous start trigger register is then sent to the TAUCnTSSTm input.

17.7 Simultaneous Rewrite

17.7.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUCnCDRm and TAUCnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by:

- The counter on the master channel or upper channel (depending on the selected operation mode) reaching a certain value
- INTTAUCnI being issued on the upper channel specified by TAUCnRDC.RDCm

There are three methods for simultaneous rewrite. These are listed in the following table, along with how to specify them and when they cause simultaneous rewrite to be triggered.

Table 17-11 Simultaneous rewrite methods and when they are triggered

Method	Simultaneous rewrite triggered when	TAUCn RDE. RDEm	TAUCn RDS. RDSm
-	No simultaneous rewrite	0	0
A	The master channel (re)starts counting	1	0
C1	INTTAUCnIm is generated on an upper channel specified by TAUCnRDC.RDCm	1	1
TOLm	For TOLm, the following table shows whether TOLm can be rewritten during operation. The TOLm rewrite method is the same as that of CDRn.		

The following table lists which of these four methods is available for each channel operation function. For more information about the individual channel operation functions, see the corresponding sections in *17.11 “Independent Channel Operation Functions” on page 1354* .

Table 17-12 Channel operation functions and methods they use

Function	A	B	C1	TOLm
Simultaneous Rewrite Trigger Output Function Type 1			X	
PWM Output Function	X		X	X

17.7.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

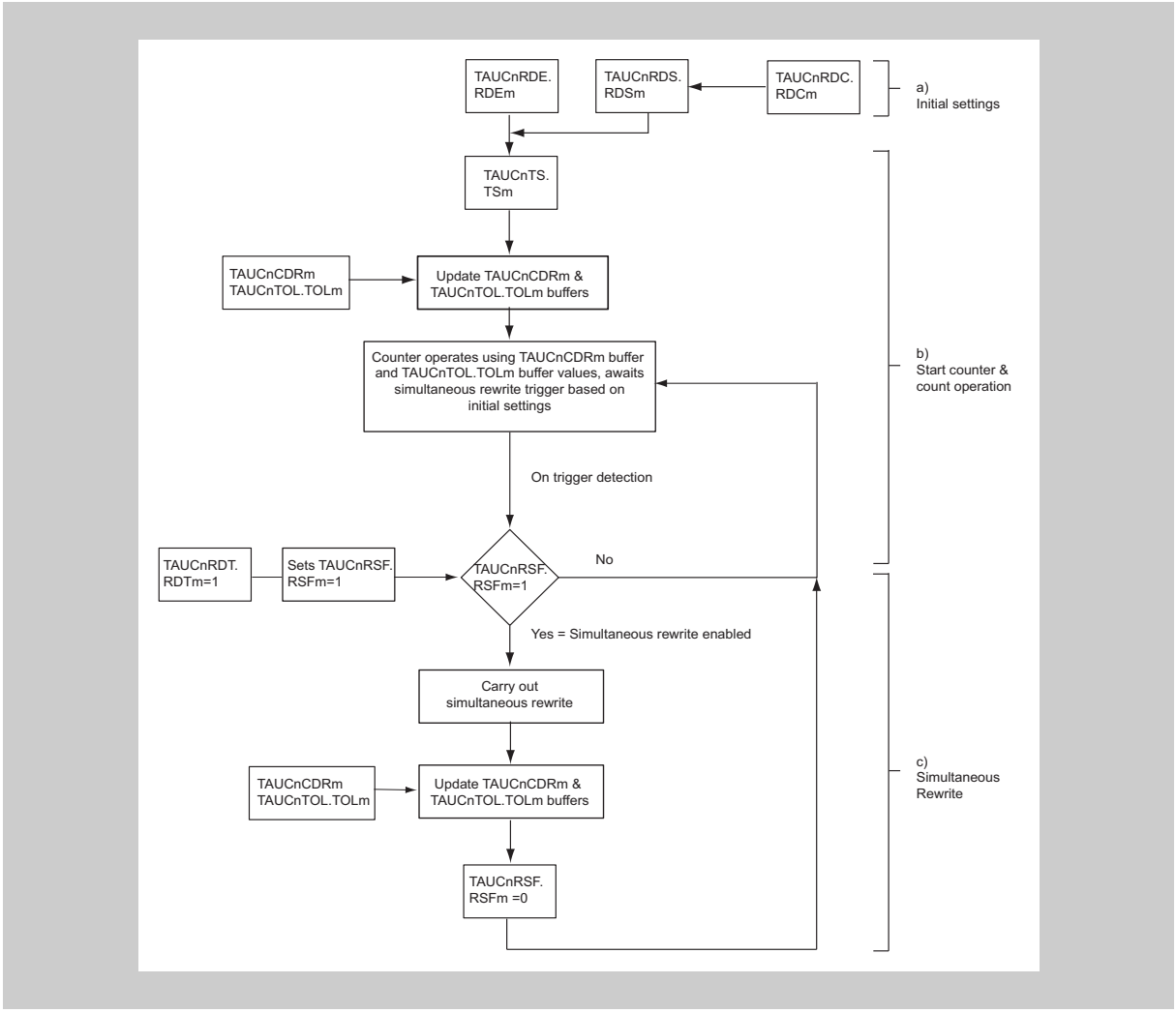


Figure 17-4 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUCnRDE.RDEm = 1
- To select the type of simultaneous rewrite, set TAUCnRDS.RDSm according to the values in *Table 17-11 “Simultaneous rewrite methods and when they are triggered” on page 1340*
- To select which upper channel is monitored for the simultaneous rewrite trigger use TAUCnRDC.RDCm (prerequisite: TAUCnRDS.RDSm is set to upper channel)

(2) Start counter and count operation

- To start all the TAUCnCNTm counters in the channel group, set the corresponding TAUCnTS.TSm bits to 1. TAUCnTOL.TOLm and the values in the data registers (TAUCnCDRm) are written to the corresponding TAUCnTOL.TOLm buffer (TAUCnTOL.TOLm buf) and data buffer registers (TAUCnCDRm buf) and the counters start.
- Setting the reload data trigger bit (TAUCnRDT.RDTm) to 1 sets the reload flag (TAUCnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUCnRDT.RDTm then immediately returns to 0, but TAUCnRSF.RSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUCnRSF.RSFm bit is checked to see if simultaneous rewrite is enabled (TAUCnRSF.RSFm = 1). If it is, simultaneous rewrite is carried out. Otherwise the value of the TAUCnRSF.RSFm bit is re-evaluated the next time the trigger is detected.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUCnRSF.RSFm = 1), the current values of the data registers are copied to their buffers. These values are then written to the corresponding counters and the values are applied the next time the counter starts or restarts.
- The TAUCnRSF.RSFm bit is set to 0, and the system awaits the next simultaneous rewrite trigger.

17.7.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUCnRDE.RDEm, TAUCnRDS.RDSm and TAUCnRDC.RDCm cannot be changed while the counter is in operation (TAUCnTE.TEm = 1).
- TAUCnTOL.TOLm can only be rewritten during operation when in PWM output function or triangle PWM output function. For all other output functions, TAUCnTOL.TOLm must be written before the counter starts. If it is rewritten in another function, TAUCnTTOUTm outputs an invalid wave.
- When an upper channel is used as the channel issuing the simultaneous rewrite trigger (TAUCnRDS.RDSm = 1), the TAUCnRDC.RDCm bit controls all the lower channels. This means that if the TAUCnRDC.RDCm bits of CH2 and CH7 are set to 1 and the TAUCnRDC.RDCm bits of other channels are set to 0, CH2 and CH7 serve as simultaneous rewrite trigger generation channels. CH2 controls the lower channels CH3 to CH6, and CH7 controls the lower channels CH8 to CH15.
- If simultaneous rewrite is enabled and an upper channel is selected for the simultaneous rewrite trigger (TAUCnRDE.RDEm and TAUCnRDS.RDSm = 1) but no upper channel is set (TAUCnRDC.RDC[15:0] = 0), simultaneous rewrite cannot take place.

17.7.4 Types of simultaneous rewrite

In the following section the four simultaneous rewrite methods are explained using timing diagrams.

(1) Simultaneous rewrite when the master channel (re)starts counting (method A)

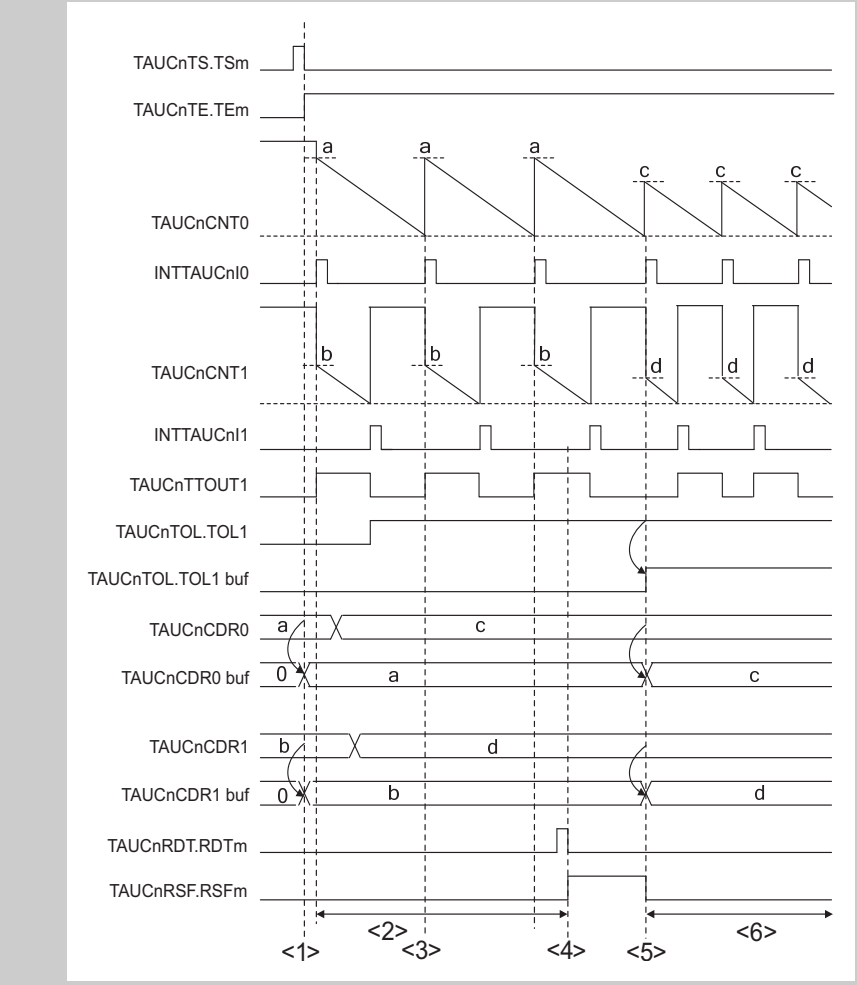


Figure 17-5 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite method A is applied.

Description:

1. When the counter starts, the value of TAUCnCDRm is copied to the TAUCnCDRm buffer and the value of TAUCnTOL.TOLm is copied to the TAUCnTOL.TOLm buffer. The TAUCnCDRm buffer value is written to the counter.
2. The TAUCnCDRm and TAUCnTOL.TOLm registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUCnRSF.RSFm = 0).
4. The reload data trigger bit (TAUCnRDT.RDTm) is set to 1 which sets the status flag (TAUCnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewrite is triggered by counter TAUCnCNT0 starting to count down. The TAUCnCDRm value is written to the TAUCnCDRm buffer and the TAUCnTOL.TOLm value is written to the TAUCnTOL.TOLm buffer.
   – The counter starts to count down from the value in the TAUCnCDRm buffer and the TAUCnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUCnTOL.TOLm becomes effective.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUCnCDRm and TAUCnTOL.TOLm can be changed again.

(2) Simultaneous rewrite when INTTAUCnIm is generated on an upper channel specified by TAUCnRDC.RDCm (method C1)

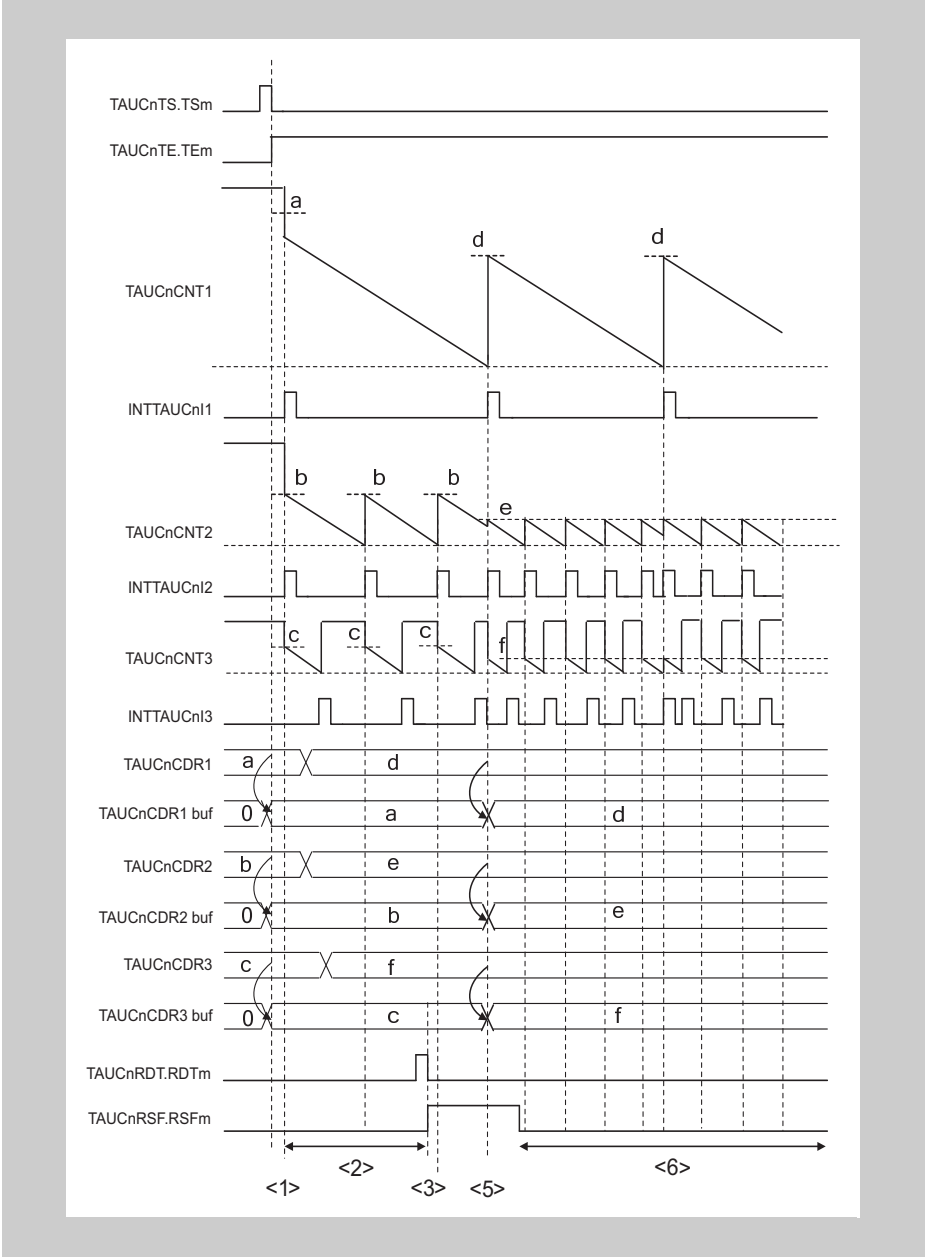


Figure 17-6 Simultaneous rewrite when INTTAUCnIm is generated on an upper channel specified by TAUCnRDC.RDCm

Setup:

- CH1 is an upper channel, counting down. CH2 is a master channel, CH3 is the slave channel, and simultaneous rewrite method C1 is applied. The TAUCnRDC register specifies which upper channel is monitored for an INTTAUCnI trigger.

Description:

1. When the counter starts, the value of TAUCnCDRm is copied to the TAUCnCDRm buffer. The buffer value is written to the counter.
2. The TAUCnCDRm register can be written at any time, but the value does not affect the counter as the counter reads the buffer value.
3. The reload data trigger bit (TAUCnRDT.RDTm) is set to 1 which sets the status flag (TAUCnRSF.RSFm = 1), enabling simultaneous rewrite.
4. Even though simultaneous rewrite is enabled, it does not take place because it is only triggered by an interrupt on channel 1.
5. Simultaneous rewrite is triggered by INT1 which is caused by counter 1 reaching 0000_H. The TAUCnCDRm values are written to the corresponding TAUCnCDRm buffers, the counters start to count down from the values in the TAUCnCDRm buffers, and the TAUCnRSF.RSFm bit is reset to 0.
6. The counter counts down and awaits the next simultaneous rewrite trigger. The values of the TAUCnCDRm registers can be changed again.

17.8 Channel Output Modes

The output of the TAUCnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (Direct Channel Output Mode, TAUCnTOE.TOEm = 0)

  When controlled by software, the output register bit (TAUCnTO.TOm) can be written and the value of the bit is transferred to the output pin (TAUCnTTOUTm).

- By TAUC signals (TAUCnTOE.TOEm = 1)

  When operated by TAUC signals, the output level of TAUCnTTOUTm is set or reset or toggled by internal signals. The value of TAUCnTO.TOm is updated accordingly to reflect the value of TAUCnTTOUTm.

  – Independently (Independent Channel Output Mode, TAUCnTOM.TOMm = 0)

    When operated independently, the output of the TAUCnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUCnTOM.TOMm = 0).

  – Synchronously (Synchronous Channel Output Mode, TAUCnTOM.TOMm = 1)

    When operated synchronously, the output of the TAUCnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUCnTOM.TOMm = 1).

The TAUCnTO.TOm bit can always be read to determine the current value of TAUCnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 17-13 “Channel output modes” on page 1348* .

The channel output modes are described in detail in

- *17.8.2 “Channel output modes controlled independently by TAUCn signals” on page 1350*
- *17.8.3 “Channel output modes controlled synchronously by TAUCn signals” on page 1350* .

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUCnTOL.TOLm.

The value of the TAUCnTOL.TOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function or triangle PWM output function. Otherwise, changes to TAUCnTOL.TOLm result in an invalid TAUCnTTOUTm signal.

Refer to *17.7 “Simultaneous Rewrite” on page 1340* .

The various channel output modes and the channel output control bits are listed in the following table.

Multiple outputs For a function on channel m that uses its output and the outputs of other channels (q):

- If channel m requires a certain operation mode for channel q, set the operation mode on channel q.
- If channel m does not require a certain operation mode for channel q, the counter on channel q must be disabled (TAUCnTE.TEq = 0), i.e. no operation mode can be used on channel q, even one that does not generate an output.

Table 17-13 Channel output modes

Channel output mode	TAUCnTOE.TOEm	TAUCnTOM.TOMm
By software		
Direct Channel Output Mode	0	x
By timer signals, independently (Independent Channel Output Mode)		
Independent Channel Output Mode 1	1	0
By timer signals, synchronously (Synchronous Channel Output Mode)		
Synchronous Channel Output Mode 1	1	1

- All combinations not listed in this table are forbidden.
- Bits marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUCnTE.TE = 1):

- TAUCnTOE.TOEm,
- TAUCnTOM.TOMm

17.8.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUCnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUCnTOE.TOEm = 0).

1. Set TAUCnTO.TOm to specify the initial level of the TAUCnTTOUTm output.
2. Set the channel output mode using *Table 17-13 “Channel output modes” on page 1348* and the output logic using the TAUCnTOL.TOLm bit.
3. Start the counter (TAUCnTS.TSm = 1).

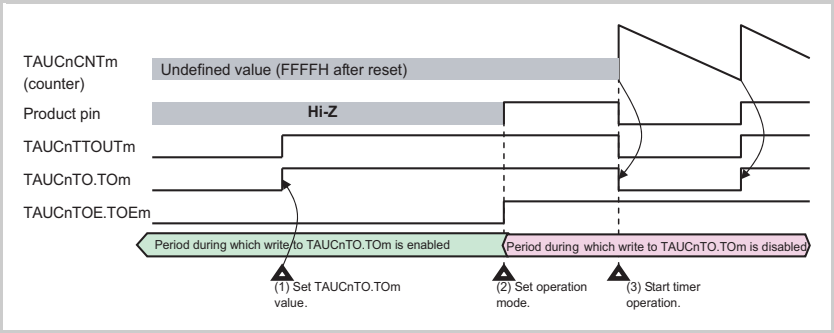


Figure 17-7 General procedure for specifying a TAUCnTTOUTm channel output mode

The following figure shows a general illustration of how the output changes when the counter is enabled:

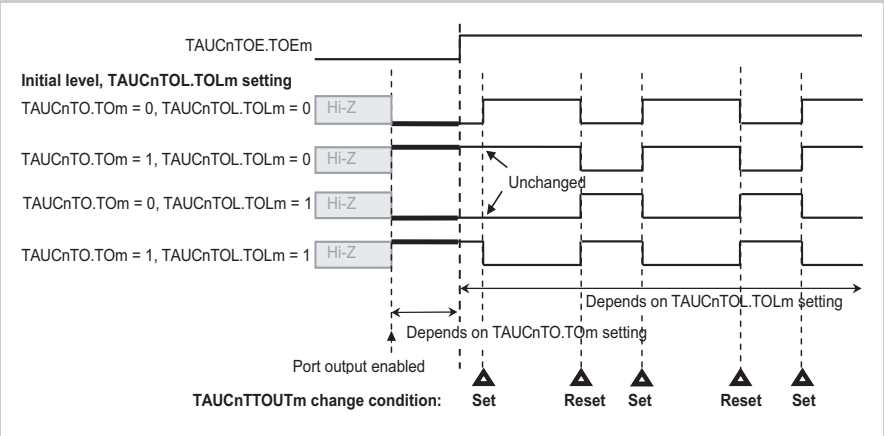


Figure 17-8 General change of the TAUCnTTOUTm output

- TAUCnTO.TOm sets the initial value of TAUCnTTOUTm and can be changed while TAUCnTOE.TOEm = 0.
- TAUCnTOL.TOLm specifies whether the set signal sets TAUCnTO.TOm to high (TAUCnTOL.TOLm = 0) or low (inverted logic, TAUCnTOL.TOLm = 1).

17.8.2 Channel output modes controlled independently by TAUCn signals

This chapter lists the channel output modes that are controlled independently by TAUCn signals. The control bits used to specify a mode are listed in *Table 17-13 “Channel output modes” on page 1348* .

(1) Independent Channel Output Mode 1

Set/reset conditions In this output mode, TAUCnTTOUTm toggles when INTTAUCnIm is detected. The value of TAUCnTOL.TOLm is ignored.

Prerequisites None, other than those in *Table 17-13 “Channel output modes” on page 1348* .

17.8.3 Channel output modes controlled synchronously by TAUCn signals

This chapter lists the channel output modes that are controlled synchronously by TAUCn signals. The control bits used to specify a mode are listed in *Table 17-13 “Channel output modes” on page 1348* .

(1) Synchronous Channel Output Mode 1

Set/reset conditions In this output mode, INTTAUCnIm of the master channel serves as the set signal and INTTAUCnIm of the slave channel as the reset signal. If INTTAUCnIm of the master channel and INTTAUCnIm of the slave channel are generated at the same time, INTTAUCnIm of the slave channel (reset signal) has priority over INTTAUCnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites None, other than those in *Table 17-13 “Channel output modes” on page 1348* .

17.9 Start Timing of Operating Modes

This chapter describes when the counters of the different operating modes start after the TAUCnTS.TSm bit is set to 1.

In all modes, the value of the data register and whether or not an interrupt is issued depends on the individual mode and corresponding register settings.

17.9.1 Interval Timer Mode

The counter starts at the start of the next count clock cycle after TAUCnTS.TSm is set to 1. The value of data register is also loaded at the point the counter starts.

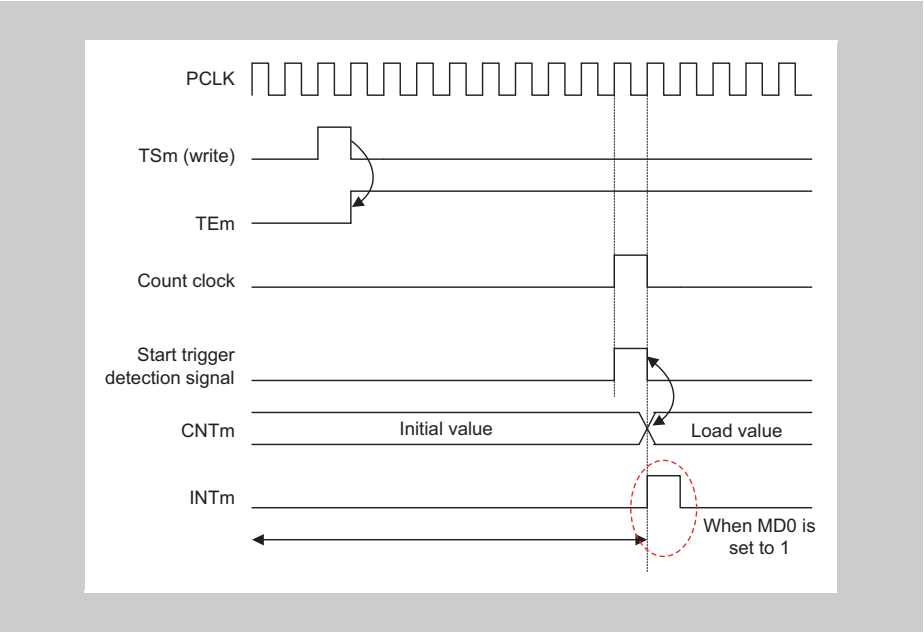


Figure 17-9 Start timing of Interval Timer Mode, Judge Mode, Capture Mode, Up Down Count Mode

17.9.2 Event Mode

The value of the data register is loaded as soon as TAUCnTS.TSm is set to 1. The counter also starts immediately. The value of the data register changes with the subsequent count clock cycles.

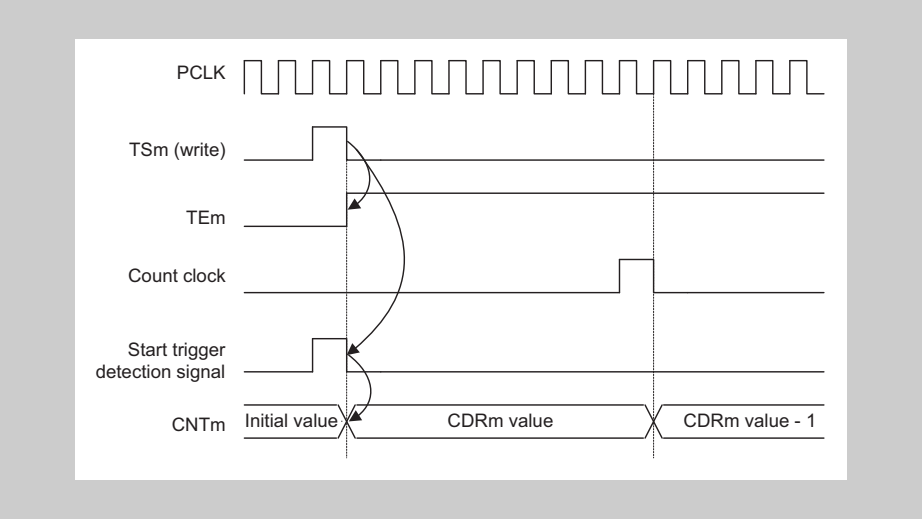


Figure 17-10 Start timing of Event Mode

17.10 TAUCnTTOUTm toggle and INTTAUCnIm Generation when Counter start is triggered (MD0-bit)

It is possible to specify whether an INTTAUCnIm is generated when the counter starts or gets restarted, using the TAUCnCMOR.MD0 bit. The effect of the bit depends on the selected mode, as shown in the following table. The effects of INTTAUCnIm on TAUCnTTOUTm depend on the selected channel operation function.

Table 17-14 Effect of CMOR.MD0 bit on generation of INTTAUCnIm when counter is triggered

Mode	TAUCnCMOR. MD0 bit	INTTAUCnIm generated when counter is (re)startedl
Interval Timer Mode	0	No
	1	Yes
One Count Mode	0/1	No, regardless of setting of TAUCnCMOR.MD0 bit.

Note As an example see figure *17-18 "Forced restart operation, TAUCnCMORm.MD0 = 1"* and figure *17-19 "Forced restart operation, TAUCnCMORm.MD0 = 0"*.
Refer to the description of *"Role of the MD0 bit"* also.

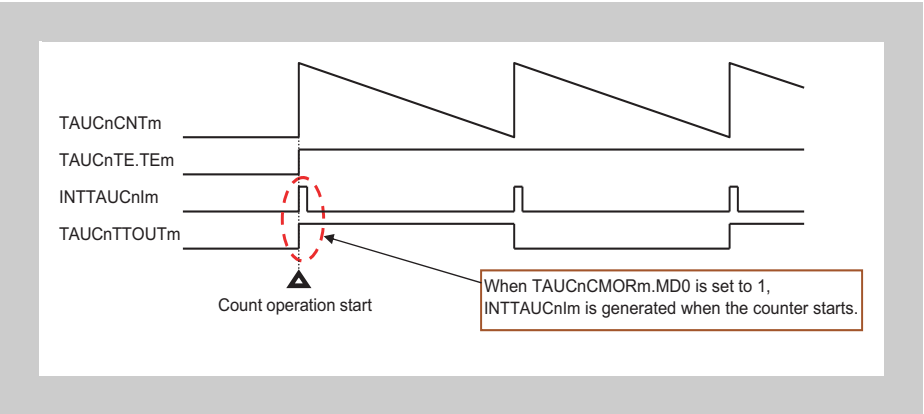


Figure 17-11 INTTAUCnIm generated when counter starts

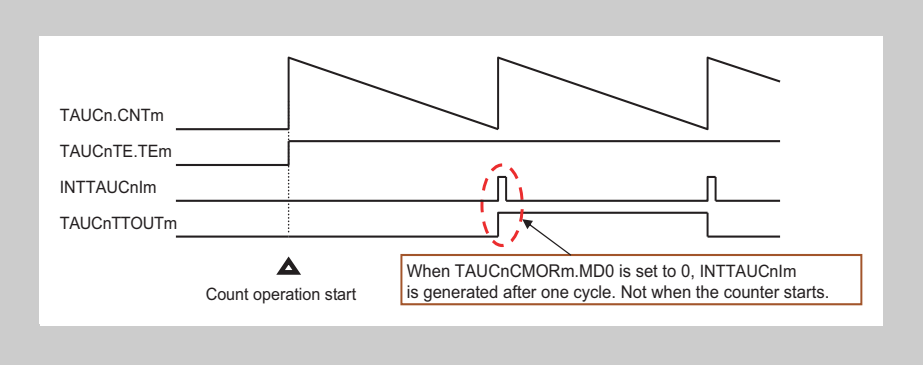


Figure 17-12 INTTAUCnIm not generated when counter starts

17.11 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by the Timer Array Unit B. For a general overview of independent channel operation, see *17.3 “Functional Description” on page 1334* .

17.12 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals or with a specified delay.

- *17.12.1 “Interval Timer Function”*

17.12.1 Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUCnIm) at regular intervals. When an interrupt is generated, the TAUCnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 17-15 "TAUCnCMORm settings for Interval Timer Function" on page 1357*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *17.8 "Channel Output Modes" on page 1347*

Description The counter is started by setting the channel trigger bit (TAUCnTS.TSm) to 1. This in turn sets TAUCnTE.TEm = 1, enabling count operation. The current value of TAUCnCDRm is written to TAUCnCNTm and the counter starts to count down from this value.

When the counter reaches 0000_H, INTTAUCnIm is generated and the TAUCnTTOUTm signal toggles. TAUCnCNTm then reloads the TAUCnCDRm value and subsequently continues operation.

The value of TAUCnCDRm can be rewritten at any time, and the changed value of TAUCnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUCnTT.TTm to 1, which in turn sets TAUCnTE.TEm to 0. TAUCnCNTm and TAUCnTTOUTm stop but retain their values. The counter can be reset by setting TAUCnTS.TSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUCnTS.TSm to 1 during operation.

Conditions If the TAUCnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUCnTTOUTm does not toggle. This results in an inverted TAUCnTTOUTm signal compared to when TAUCnCMORm.MD0 is set to 1. For details refer to *17.10 "TAUCnTTOUTm toggle and INTTAUCnIm Generation when Counter start is triggered (MD0-bit)" on page 1353* .

(2) Equations

INTTAUCnIm cycle = count clock cycle x (TAUCnCDRm + 1)

TAUCnTTOUTm square wave cycle = count clock cycle x (TAUCnCDRm + 1) x 2

(3) Block diagram and general timing diagram

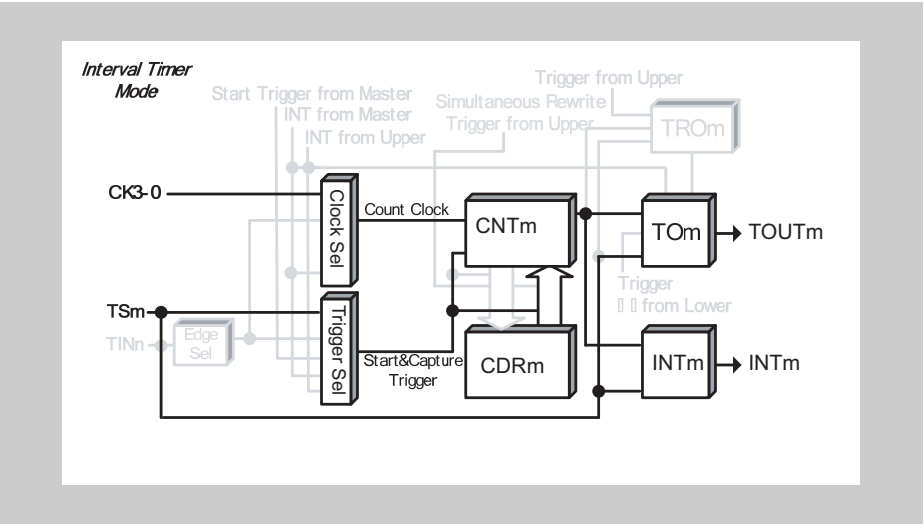


Figure 17-13 Block diagram for Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUCnIm generated at operation start (TAUCnCMORm.MD0 = 1)

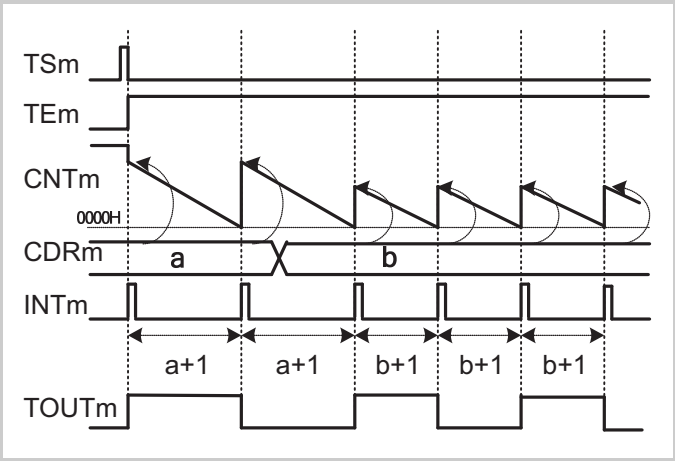


Figure 17-14 General timing diagram for Interval Timer Function

(4) Register settings

(a) TAUCnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	-	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 17-15 TAUCnCMORm settings for Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUCnIm not generated and TAUCnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUCnIm and toggles TAUCnTTOUTm at operation start or restart

(b) Channel output mode

Table 17-16 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOL.TOLm	0: Positive logic

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUCnTOE.TOEm = 0. TAUCnTTOUTm can then be controlled independently of the interrupts. For details refer to *17-13 “Channel output modes” on page 1348* .

(c) Simultaneous rewrite

The simultaneous rewrite registers (TAUCnRDE, TAUCnRDS and TAUCnRDC) cannot be used with the Interval Timer Function. Therefore, these registers must be set to 0.

Table 17-17 Simultaneous rewrite settings for Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDS.RDSm	0: When simultaneous rewrite is disabled (TAUCnRDE.RDEm = 0), set these bits to 0
RDC.RDCm	

(5) Operating procedure for Interval Timer Function

Table 17-18 Operating procedure for Interval Timer Function

	Operation	Status of TAUCn
Initial channel setting	Set the TAUCnCMORm register as described in *Table 17-15 "TAUCnCMORm settings for Interval Timer Function" on page 1357* Set the value of the TAUCnCDRm register Set the channel output mode by setting the control bits as described in *Table 17-16 "Control bit settings for Independent Channel Output Mode 1" on page 1358*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUCnTS.TSm to 1. TAUCnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm is set to 1 and the counter starts. TAUCnCNTm loads the TAUCnCDRm value. When TAUCnCMORm.MD0 = 1, INTTAUCnIm is generated and TAUCnTTOUTm toggles.
During operation	The TAUCnCDRm register value can be changed at any time. The TAUCnCNTm register can be read at all times.	TAUCnCNTm counts down. When the counter reaches 0000_H: • TAUCnCNTm reloads the TAUCnCDRm value and continues count operation • INTTAUCnIm is generated and TAUCnTTOUTm toggles.
Stop operation	Set TAUCnTT.TTm to 1. TAUCnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm is cleared to 0 and the counter stops. TAUCnCNTm and TAUCnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUCnCDRm = 0000_H, count clock = PCLK/2

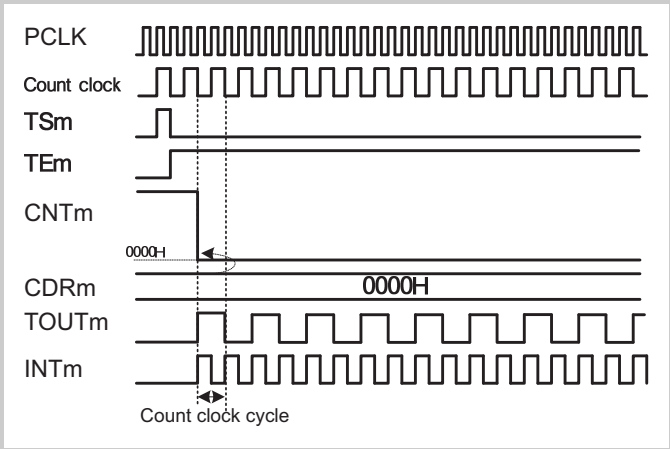


Figure 17-15 TAUCnCDRm = 0000_H, count clock = PCLK/2

- If TAUCnCDRm = 0000_H and the count clock = PCLK/2^1, the TAUCnCDRm value is written to TAUCnCNTm every count clock, meaning that TAUCnCNTm is always 0000_H.
- INTTAUCnIm is generated every count clock, resulting in TAUCnTTOUTm toggling every count clock.

(b) TAUCnCDRm = 0000_H, count clock = PCLK

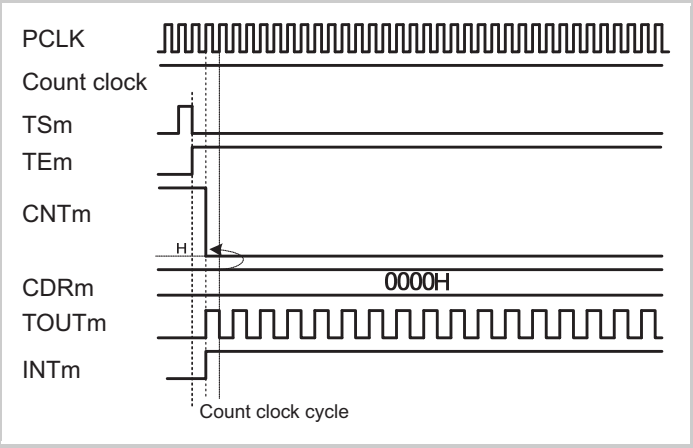


Figure 17-16 TAUCnCDRm = 0000_H, count clock = PCLK

- If TAUCnCDRm = 0000_H and the count clock = PCLK, the TAUCnCDRm value is written to TAUCnCNTm every PCLK clock, meaning that TAUCnCNTm is always 0000_H.
- INTTAUCnIm is generated continuously, resulting in TAUCnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

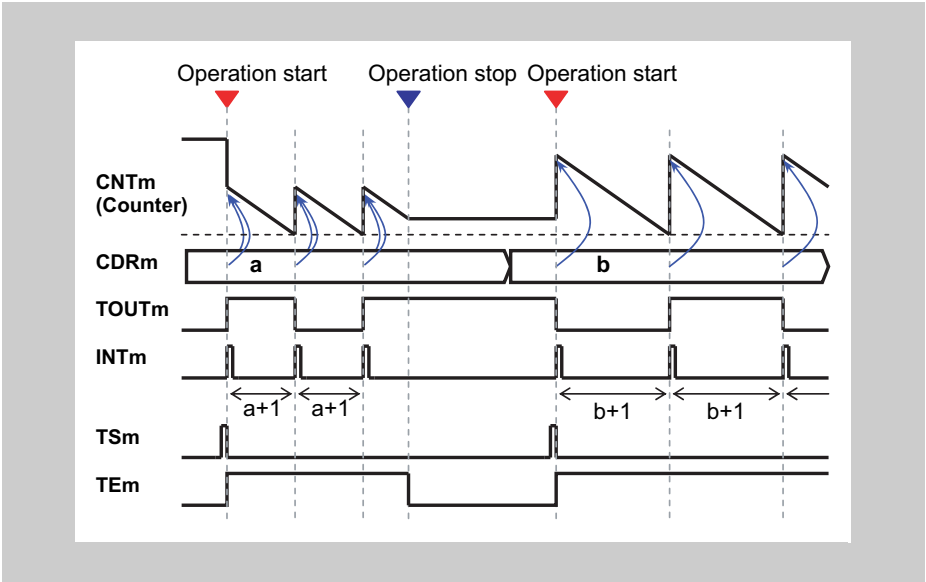


Figure 17-17 Operation stop and restart, TAUCnCMORm.MD0 = 1

- The counter can be stopped by setting TAUCnTT.TTm to 1, which in turn sets TAUCnTE.TEm to 0.
- TAUCnCNTm and TAUCnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUCnTS.TSm to 1.

(d) Forced restart

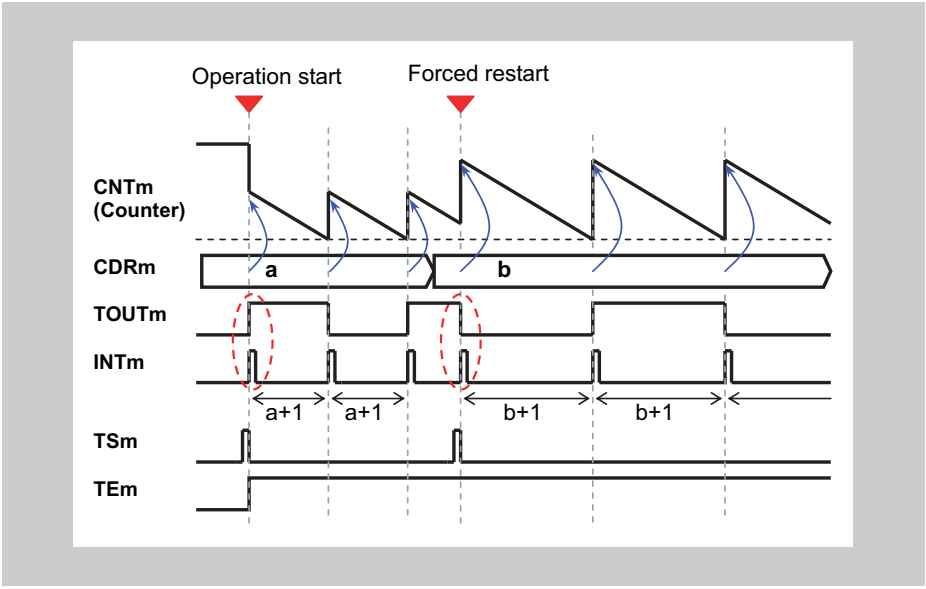


Figure 17-18 Forced restart operation, TAUCnCMORm.MD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUCnTS.TSm to 1 during operation.
- If the TAUCnCMORm.MD0 bit is set to 1, an interrupt at start or restart is generated and the output TOUTm toggles.

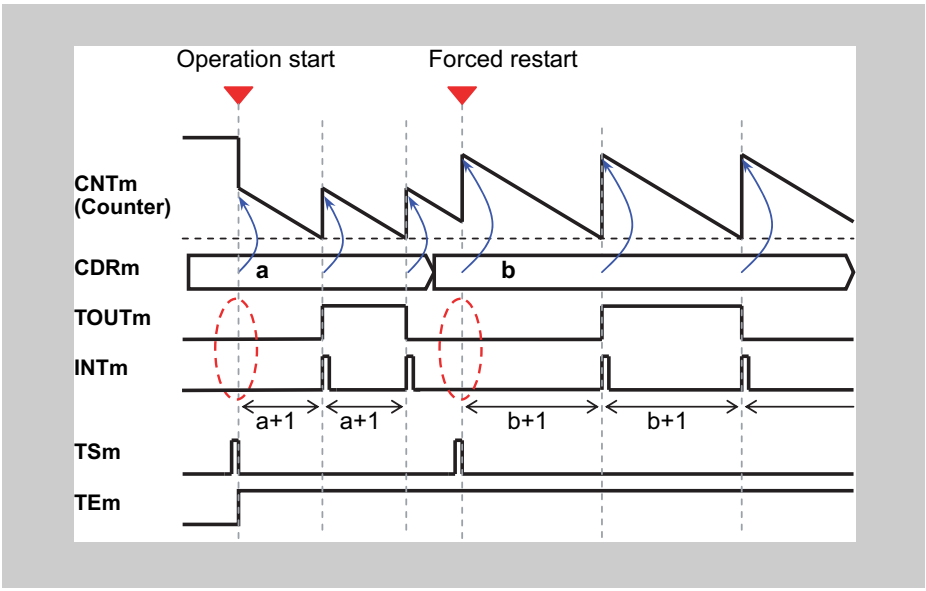


Figure 17-19 Forced restart operation, TAUCnCMORm.MD0 = 0

- The counter can be forcibly restarted (without stopping it first) by setting TAUCnTS.TSm to 1 during operation.
- If the TAUCnCMORm.MD0 bit is set to 0, the interrupt at start or restart is not generated and TOUTm does not toggle..

17.13 Independent Channel Simultaneous Rewrite Functions

This chapter describes functions that carry out simultaneous rewrite:

- *17.13.1 “Simultaneous Rewrite Trigger Generation Function Type 1”*

17.13.1 Simultaneous Rewrite Trigger Generation Function Type 1

(1) Overview

Summary This function generates an interrupt on a specific channel that can be used by lower channels as a simultaneous rewrite trigger. The interrupt is generated at regular intervals.

Prerequisites

- Two (or more) channels, each with simultaneous rewrite enabled (TAUCnRDE.RDEm = 1)
- The operation mode of the upper channel must be set to Interval Timer Mode, refer to *Table 17-19 "TAUCnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 1367*
- The operation mode of the lower channel(s) can be set as desired

Description The counters are started by setting the channel trigger bits (TAUCnTS.TSm and TAUCnTS.TSm_lower) to 1. This in turn sets TAUCnTE.TEm and TAUCnTE.TEm_lower = 1, enabling count operation. The current value of the data register buffer of the upper channel (TAUCnCDRm buf) is written to the counter (TAUCnCNTm) and the counter starts to count down from this value. The counter(s) of the lower channel(s) start to count as specified by their selected operating modes.

When a counter reaches 0000_H, an interrupt is generated from the channel. The corresponding TAUCnCNTm then reloads the current TAUCnCDRm buffer value and subsequently continues operation.

If the channel where the interrupt occurs is specified as the trigger channel for simultaneous rewrite (TAUCnRDC.RDCm = 1) and is an upper channel, simultaneous rewrite takes place on all lower channels in which simultaneous rewrite is currently possible (TAUCnRSF.RSFm = 1).

The values of the data registers are copied to the corresponding data register buffers. Each time a counter starts to count down, it reads the value in the data register buffer and counts down from this value.

The value of a data register can be changed at any time, but it is only transferred to the corresponding data register buffer when simultaneous rewrite occurs.

Conditions

- The channel which is monitored for INTTAUCnIm is specified by setting TAUCnRDC.RDCm = 1 for the corresponding channel. The TAUCnRDC.RDCm bit must be 0 for all other channels in which simultaneous rewrite should take place.
- If the TAUCnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *17.10 "TAUCnTTOUTm toggle and INTTAUCnIm Generation when Counter start is triggered (MD0-bit)" on page 1353* .

(2) Equations

Simultaneous rewrite trigger generation cycle = count clock cycle × (TAUCnCDRm + 1)

To control simultaneous rewrite, the following condition must be satisfied:

TAUCnCDRm = [(value of TAUCnCDRm of master channel subject to simultaneous rewrite + 1) × number of interrupts] - 1

That is, the ratio of TAUCnCDRm + 1 and TAUCnCDRm_master + 1 must be an integer. This integer corresponds to the number of interrupts.

(3) Block diagram and general timing diagram

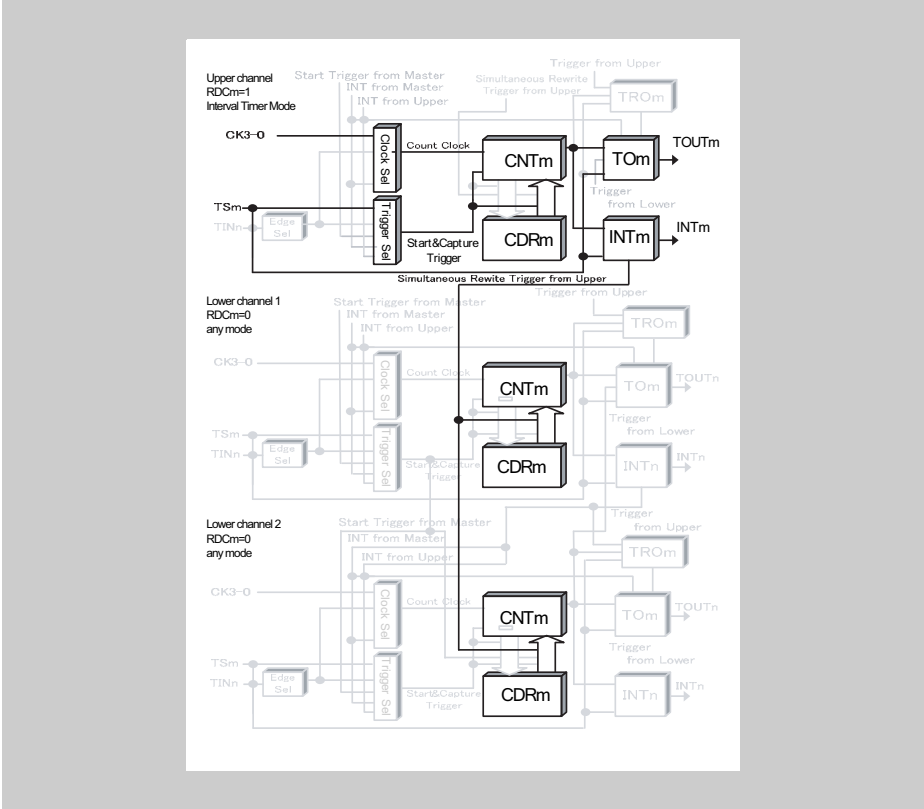


Figure 17-20 Block diagram for Simultaneous Rewrite Trigger Generation Function Type 1

The following settings apply to the general timing diagram:

- INTTAUCnIm generated at operation start (TAUCnCMORm.MD0 = 1)

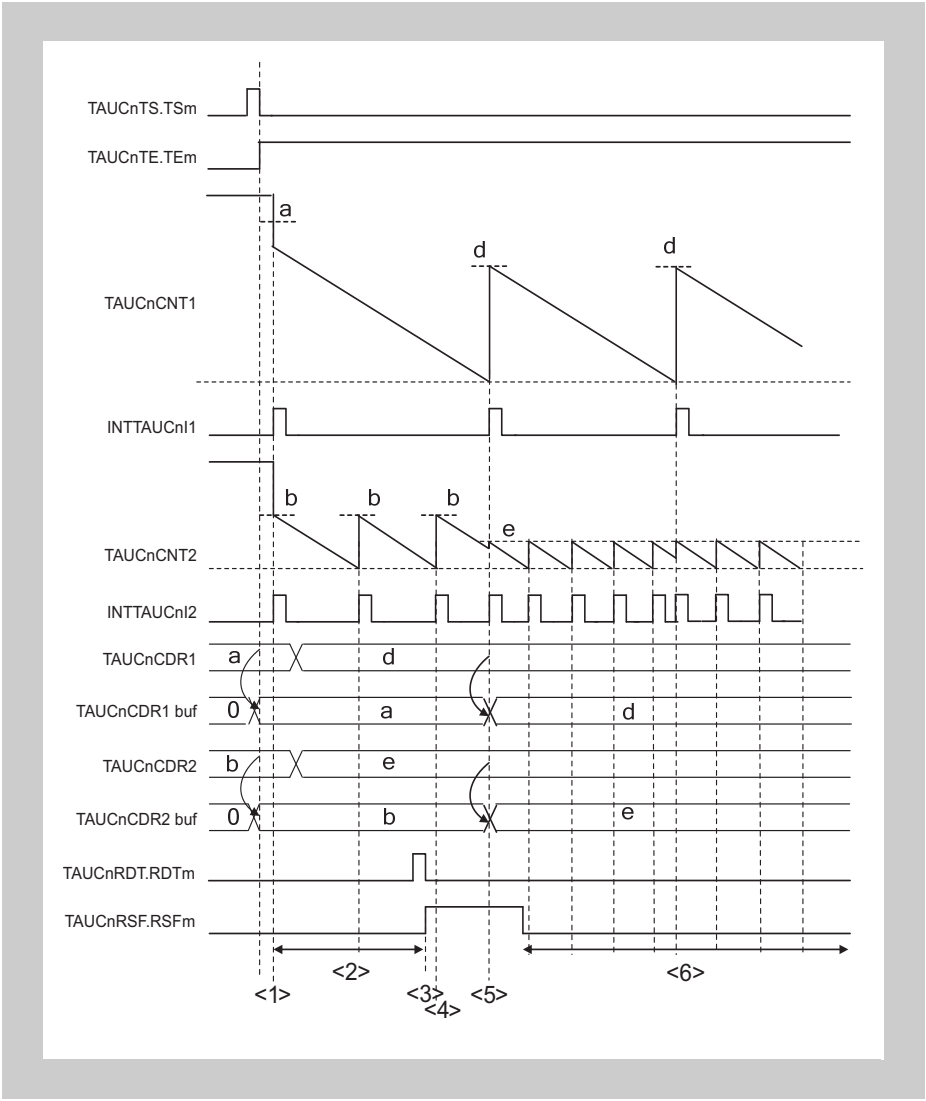


Figure 17-21 General timing diagram for Simultaneous Rewrite Trigger Generation Function Type 1

(4) Register settings for the upper channel

(a) TAUCnCMORm for the upper channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	-	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 17-19 TAUCnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUCnIm not generated at operation start 1: Generates INTTAUCnIm at operation start

(b) Channel output mode for the upper channel

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(c) Simultaneous rewrite for the upper channel

Table 17-20 Simultaneous rewrite settings for Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDC.RDCm	1: Channel is monitored for an INTTAUCnIm signal that is used as the simultaneous rewrite trigger

(5) Register settings for the lower channel(s)

(a) TAUCnCMORm for the lower channel(s)

The TAUCnCMORm register of the lower channel(s) can be set arbitrarily.

(b) Channel output mode for the lower channel(s)

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(c) Simultaneous rewrite for the lower channel(s)

Table 17-21 Simultaneous rewrite settings for the lower channel in Simultaneous Rewrite Trigger Generation Function Type 1

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	1: Selects an upper channel as the control channel for simultaneous rewrite
RDC.RDCm	0: Channel is not monitored for an INTTAUCnIm signal that is used as the simultaneous-rewrite trigger

(6) Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

Table 17-22 Operating procedure for Simultaneous Rewrite Trigger Generation Function Type 1

	Operation	Status of TAUCn
Initial channel setting	Set the TAUCnCMORm register for the upper channel as described in *Table 17-19 "TAUCnCMORm settings for Simultaneous Rewrite Trigger Generation Function Type 1" on page 1367* Set the TAUCnCMORm register for the lower channel as described in *5 "Register settings for the lower channel(s)"* Set the value of the TAUCnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUCnTS.TSm to 1. TAUCnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm is set to 1 and the counter starts. TAUCnCNTm loads the TAUCnCDRm value. When TAUCnCMORm.MD0 = 1, INTTAUCnIm is generated and TAUCnTTOUTm toggles.
During operation	TAUCnRDT.RDTm can be changed. TAUCnRSF.RSFm can be read at all times.	TAUCnCNTm counts down. When the counter reaches 0000_H: • TAUCnCNTm reloads the TAUCnCDRm value and continues count operation • INTTAUCnIm is generated • TAUCnTTOUTm toggles. Simultaneous rewrite is controlled when INTTAUCnIm is generated from the channel where TAUCnRDC.RDCm is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUCnTT.TTm to 1. TAUCnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm is cleared to 0 and the counter stops. TAUCnCNTm stops and both it and TAUCnTTOUTm retain their current values. When TAUCnTOE.TOEm is 0, TAUCnTTOUTm output is initialized to the value set by TAUCnTO.TOm.

Restart

17.14 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes functions that generate PWM signals at regular intervals.

- *17.14.1 “PWM Output Function”*

17.14.1 PWM Output Function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUCnTTOUTm to be set. The pulse cycle is set in the master channel. The pulse width is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 17-23 “TAUCnCMORm settings for the master channel of the PWM Output Function” on page 1374*
- The operation mode of the slave channel(s) must be set to One Count Mode, refer to *Table 17-25 “TAUCnCMORm settings for the slave channel of the PWM Output Function” on page 1376*
- TAUCnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 1 (*17.8 “Channel Output Modes” on page 1347*)

Description The counters are started by setting the channel trigger bits (TAUCnTS.TSm) to 1. This in turn sets TAUCnTE.TEm = 1, enabling count operation. The current value of TAUCnCDRm is written to TAUCnCNTm and the counters start to count down from these values. INTTAUCnIm is generated on the master channel and TAUCnTTOUTm (slave) toggles.

- Master channel:

  When the counter of the master channel reaches 0000_H, pulse cycle time has elapsed and INTTAUCnIm is generated. The counter reloads the TAUCnCDRm value and counts down.

- Slave channel(s)

  The INTTAUCnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUCnCDRm (slave) is written to TAUCnCNTm (slave) and the counter starts to count down from this value. The TAUCnTTOUTm signal is set.

  When the counter reaches 0000_H, i.e. duty time has elapsed, INTTAUCnIm is generated and the TAUCnTTOUTm signal is reset. The counter returns to $FFFF_H$ and awaits the next INTTAUCnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUCnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUCnTE.TEm to 0. TAUCnCNTm and TAUCnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUCnTS.TSm to 1.

Note If a forced restart is executed during operation, the counter value becomes invalid and TAUCnTTOUTm is not output as expected PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *17.7 “Simultaneous Rewrite” on page 1340*

(2) Equations

Pulse cycle = (TAUCnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUCnCDRm (slave) / (TAUCnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

  TAUCnCDRm (slave) = 0000_H

– Duty cycle = 100 %

  TAUCnCDRm (slave) ≥ TAUCnCDRm (master) + 1

(3) Block diagram and general timing diagram

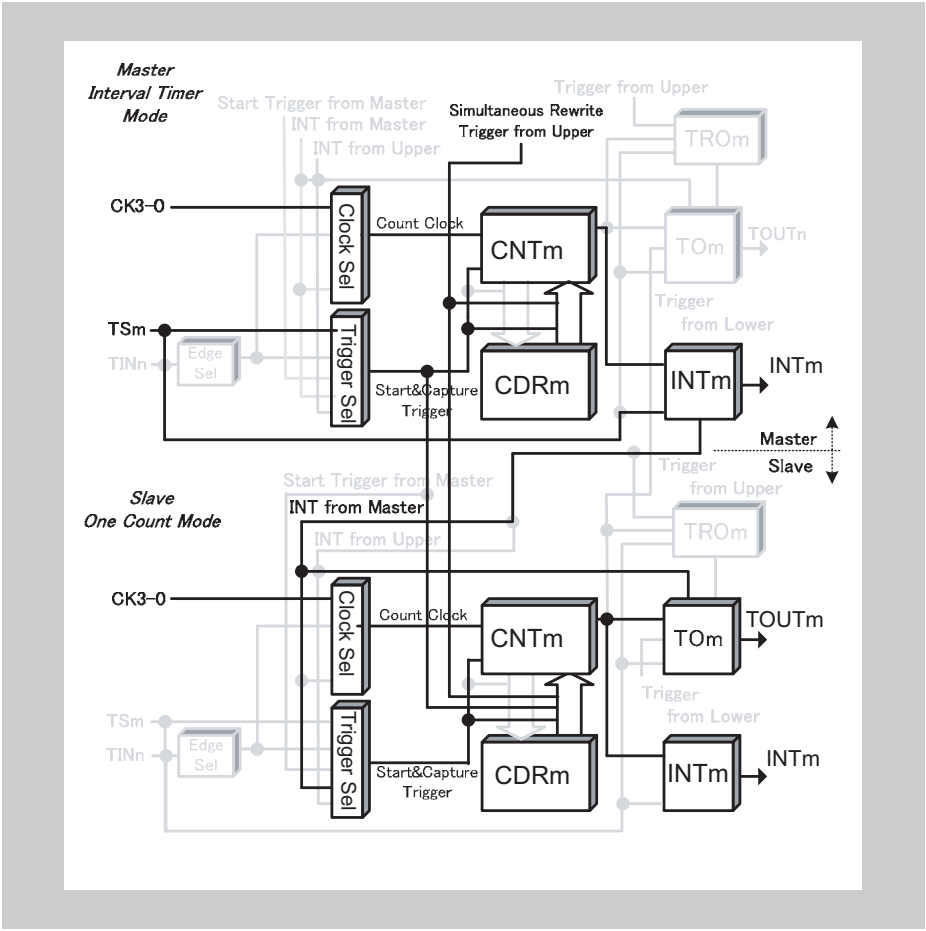


Figure 17-22 Block diagram for PWM Output Function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUCnTOL.TOLm = 0)

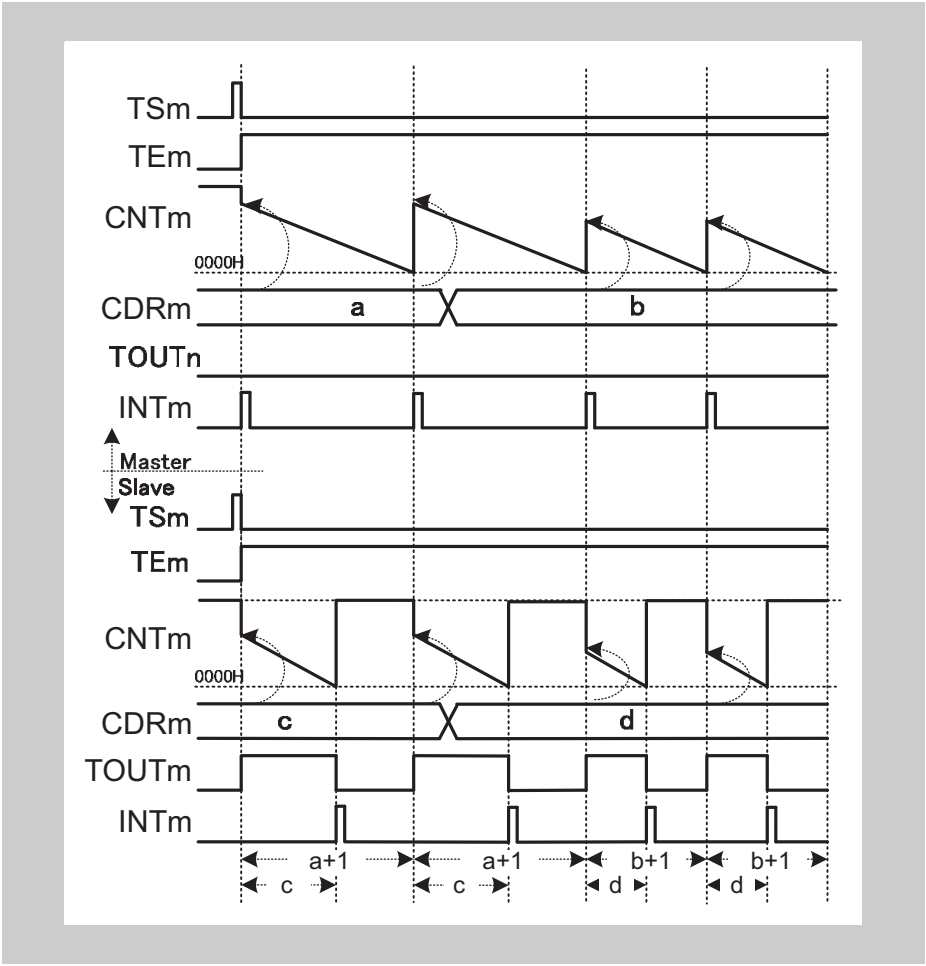


Figure 17-23 General timing diagram for PWM Output Function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUCnCDRm, whereas for the master channel the interval is the corresponding TAUCnCDRm + 1.

(4) Register settings for the master channel

(a) TAUCnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	-	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 17-23 TAUCnCMORm settings for the master channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUCnIm at operation start

(b) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(c) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 17-24 Simultaneous rewrite settings for the master channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDC.RDCm	0: Channel is not monitored for an INTTAUCnIm signal that is used as the simultaneous rewrite trigger. If TAUCnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(5) Register settings for the slave channel(s)

(a) TAUCnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	-	MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 17-25 TAUCnCMORm settings for the slave channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUCnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUCnIm and toggles TAUCnTTOUTm at operation start

(b) Channel output mode for the slave channel(s)

Table 17-26 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOLm	0: Positive logic 1: Inverted logic

(c) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 17-27 Simultaneous rewrite settings for the slave channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDS.RDSm	0: The master channel is monitored for the simultaneous rewrite trigger 1: An upper channel outside the channel group is monitored for the simultaneous rewrite trigger
RDC.RDCm	0: Channel is not monitored for an INTTAUCnIm signal that is used as the simultaneous rewrite trigger. If TAUCnRDS.RDSm = 0, the master channel is monitored for the simultaneous rewrite trigger regardless of the value of this bit.

(6) Operating procedure for PWM Output Function

Table 17-28 Operating procedure for PWM Output Function

	Operation	Status of TAUCn
Initial channel setting	Master channel: set the TAUCnCMORm register and the channel output mode as described in *4 "Register settings for the master channel" on page 1374* Slave channel: set the TAUCnCMORm register and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 1376* Set the values of the TAUCnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUCnTS.TSm of the master and slave channels to 1 simultaneously. TAUCnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUCnIm is generated on the master channel and TAUCnTTOUTm (slave) is set.
During operation	TAUCnCDRm can be changed at any time. TAUCnCNTm and TAUCnRSF.RSFm can be read at any time. TAUCnRDT.RDTm can be changed during operation.	TAUCnCNTm of the master channel loads TAUCnCDRm and counts down. When the counter reaches 0000_H: • INTTAUCnIm (master) is generated • TAUCnCNTm (master) reloads the TAUCnCDRm value and continues count operation • TAUCnCNTm (slave) reloads the TAUCnCDRm value and counts down • TAUCnTTOUTm (slave) is set When TAUCnCNTm (slave) reaches 0000_H: • INTTAUCnIm (slave) is generated • TAUCnTTOUTm (slave) is reset
Stop operation	Set TAUCnTT.TTm of the master and slave channels to 1 simultaneously. TAUCnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUCnTE.TEm is cleared to 0 and the counter stops. TAUCnCNTm and TAUCnTTOUTm stop and retain their current values. When TAUCnTOE.TOEm is 0, TAUCnTTOUTm output is initialized to the value set by TAUCnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

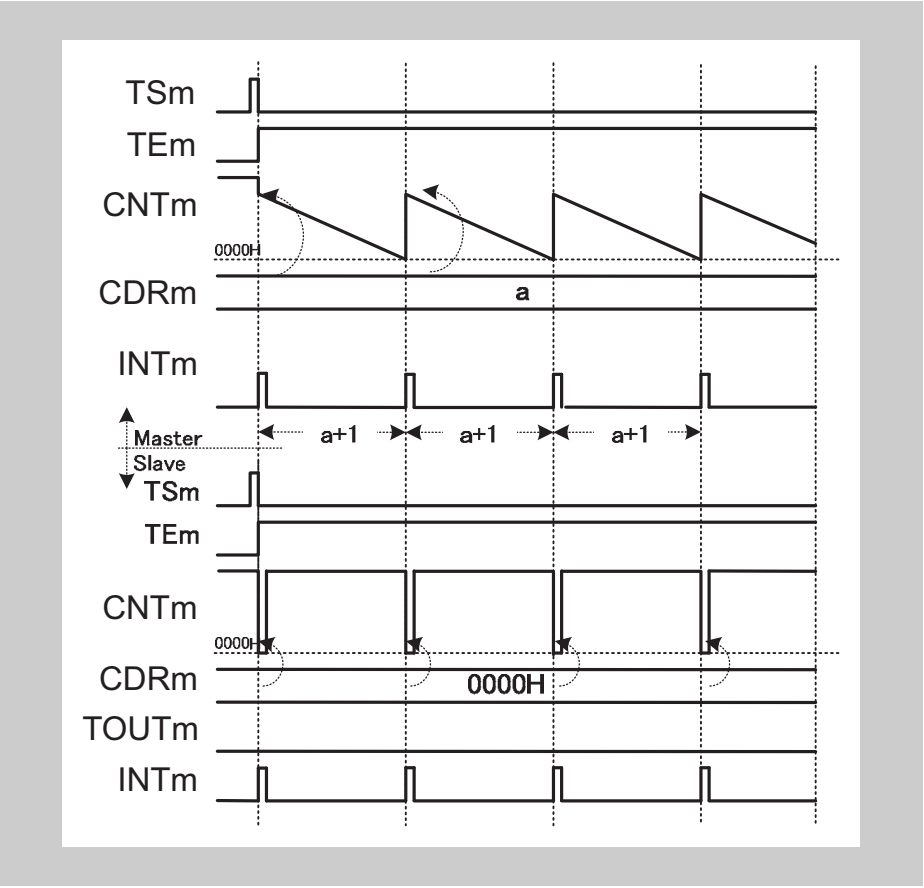


Figure 17-24 TAUCnCDRm (slave) = 0000_H, positive logic (TAUCnTOL.TOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUCnIm), 0000_H is written to TAUCnCNTm (slave). Therefore, TAUCnCNTm (slave) cannot start to count and TAUCnTTOUTm remains at not active state.
- TAUCnCNTm (slave) generates an interrupt every time the value of TAUCnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

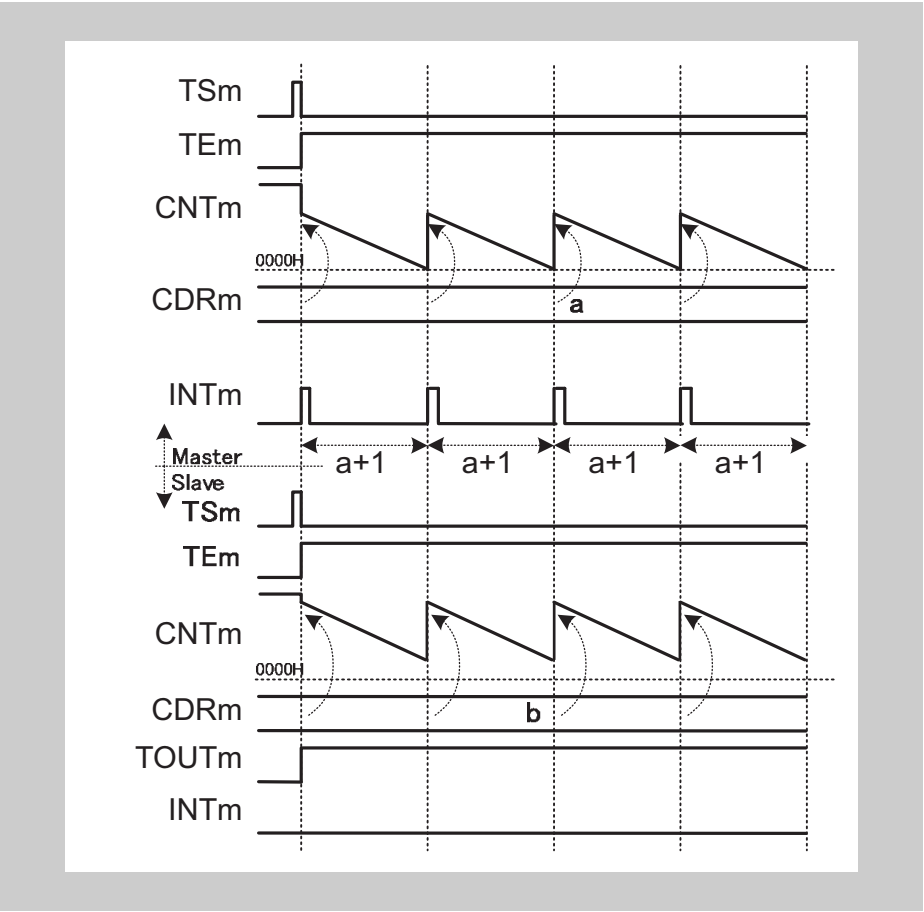


Figure 17-25 TAUCnCDRm (slave) ≥ TAUCnCDRm (master) + 1, positive logic (TAUCnTOL.TOLm (slave) = 0)

- If the value TAUCnCDRm (slave) is higher than the value TAUCnCDRm (master), the counter of the slave channel cannot reach 0000_H and cannot generate interrupts. The TAUCnTTOUTm remains at active state.

(c) Stop and restart operation

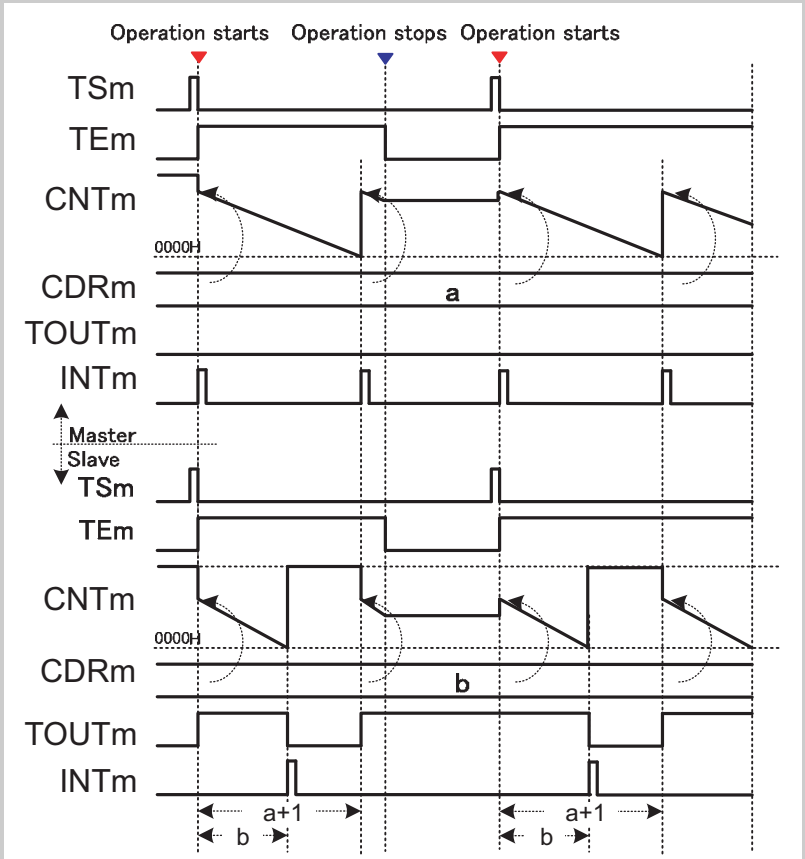


Figure 17-26 Stop and restart operation, positive logic (TAUCnTOL.TOLm (slave) = 0)

- The counter can be stopped by setting TAUCnTT.TTm of the master and slave channel(s) to 1, which in turn sets TAUCnTE.TEm to 0.
- TAUCnCNTm and TAUCnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUCnTS.TSm of master and slave channel(s) to 1. TAUCnCNTm of master and slave channel reload the current values of TAUCnCDRm and start to count down from these values.

17.15 Registers

This section contains a description of all the registers of the 16-bit Timer Array Unit B.

17.15.1 TAUCn registers overview

The TAUCn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 15.

Table 17-29 TAUCn registers overview

Register name	Shortcut	Address
TAUCn prescaler registers		
TAUCn prescaler clock select register	TAUCnTPS	<TAUCn_base> + 240_H
TAUCn control registers		
TAUCn channel data register m	TAUCnCDRm	<TAUCn_base> + m x 4_H
TAUCn channel counter register m	TAUCnCNTm	<TAUCn_base> + 80_H + m x 4_H
TAUCn channel mode OS register m	TAUCnCMORm	<TAUCn_base> + 200_H + m x 4_H
TAUCn channel start trigger register	TAUCnTS	<TAUCn_base> + $1C4_H$
TAUCn channel enable status register	TAUCnTE	<TAUCn_base> + $1C0_H$
TAUCn channel stop trigger register	TAUCnTT	<TAUCn_base> + $1C8_H$
TAUCn output registers		
TAUCn channel output enable register	TAUCnTOE	<TAUCn_base> + $5C_H$
TAUCn channel output register	TAUCnTO	<TAUCn_base> + 58_H
TAUCn channel output mode register	TAUCnTOM	<TAUCn_base> + 248_H
TAUCn channel output active level register	TAUCnTOL	<TAUCn_base> + 040_H
TAUCn reload data registers		
TAUCn channel reload data enable register	TAUCnRDE	<TAUCn_base> + 260_H
TAUCn channel reload data control CH select register	TAUCnRDS	<TAUCn_base> + 268_H
TAUCn channel reload data control register	TAUCnRDC	<TAUCn_base> + $26C_H$
TAUCn channel reload data trigger register	TAUCnRDT	<TAUCn_base> + 44_H
TAUCn channel reload status register	TAUCnRSF	<TAUCn_base> + 48_H
TAUCn emulation register		
TAUCn emulation register	TAUCnEMU	<TAUCn_base> + 290_H

<TAUCn_base> The <TAUCn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

17.15.2 TAUCn prescaler registers details

(1) TAUCnTPS - TAUCn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3 for all channels.

Access This register can be read/written in 16-bit units.

Address <TAUCn_base> + 240_H

Initial Value $FFFF_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS3[3:0]				PRS2[3:0]				PRS1[3:0]				PRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-30 TAUCnTPS register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 12</td><td>PRS3[3:0]</td><td>Specifies the CK3 clock.
<table>
<tr><th>PRS3[3:0]</th><th>CK3 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK3 are stopped (TAUCnTE.TEm = 0).</td></tr>
<tr><td>11 to 8</td><td>PRS2[3:0]</td><td>Specifies the CK2 clock.
<table>
<tr><th>PRS2[3:0]</th><th>CK2 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK2 are stopped (TAUCnTE.TEm = 0).</td></tr>
</table>

Table 17-30 TAUCnTPS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUCnTE.TEm = 0).</td></tr>
<tr><td>3 to 0</td><td>PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>PCLK/2^0</td></tr>
<tr><td>0001_B</td><td>PCLK/2^1</td></tr>
<tr><td>0010_B</td><td>PCLK/2^2</td></tr>
<tr><td>0011_B</td><td>PCLK/2^3</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>PCLK/2^{14}</td></tr>
<tr><td>1111_B</td><td>PCLK/2^{15}</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUCnTE.TEm = 0).</td></tr>
</table>

Note The TAUCn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

17.15.3 TAUCn control registers details

(1) TAUCnCDRm - TAUCn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUCnCMORm.MD[4:0].

Access This register can be read/written in 16-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUCn_base> + 0_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDR[15:0]															
R/W															

Table 17-31 TAUCnCDRm register contents

Bit position	Bit name	Function
15 to 0	CDR[15:0]	Data register for the capture/compare value.

(2) TAUCnCNTm - TAUCn channel counter register

This register is the channel m counter register.

Access This register can be read in 16-bit units.

Address <TAUCn_base> + 80_H + m x 4_H

Initial Value 0000_H or $FFFF_H$ The initial value depends on the operation mode, see *Table 17-33 "TAUCnCNTm read values after the counter is re-enabled" on page 1386* .

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R															

Table 17-32 TAUCnCNTm register contents

Bit position	Bit name	Function
15 to 0	CNT[15:0]	16-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUCnTS.TSm and TAUCnTT.TTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUCnTT.TTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUCnTE.TEm= 0) and re-enabled (TAUCnTS.TSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUCnTS.TSm = 1) for modes where the counter waits for a start trigger.

Table 17-33 TAUCnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUCnCNTm value		
		After reset	After stop trigger	After one count
Interval Timer mode	Count down	$FFFF_H$	Stop value	-
One Count mode	Count down	$FFFF_H$	Stop value	$FFFF_H$

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUCnCMORm.MD[4:0].

(3) TAUCnCMORm - TAUCn channel mode OS register

This register controls channel m operation. It specifies, for example, the operation clock, count clock, and master/slave function.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUCnTE.TEm = 0).

Address <TAUCn_base> + 200_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		-	-	MAS	STS[2:0]			COS[1:0]		-	MD[4:0]				
R/W		R	R	R/W	R/W			R/W		R	R/W				

Table 17-34 TAUCnCMORm register contents (1/2)

Bit position	Bit name	Function
15,14	CKS[1:0]	Selects the operation clock: CKS1 / CKS0 / Selected operation clock 0 / 0 / CK0 0 / 1 / CK1 1 / 0 / CK2 1 / 1 / CK3
11	MAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.
10 to 8	STS[2:0]	Selects the external start trigger: STS2 / RFU (STS1) / RFU (STS0) / Description 0 / 0 / 0 / Software trigger 1 / 0 / 0 / INTTAUCnI of the master channel
7, 6	RFU (COS[1:0])	COS1 / COS0 / / / / / 0 / 0 / / / /
4 to 0	MD[4:0]	Specifies the operation mode. MD4 / MD3 / MD2 / MD1 / MD0 / Description 0 / 0 / 0 / 0 / 1/0 / Interval Timer mode 0 / 1 / 0 / 0 / 1/0 / One Count mode

Table 17-34 TAUCnCMORm register contents (2/2)

Bit position	Bit name	Function
Mode		Role of the MD0 bit
Interval Timer mode		Specifies whether an INTTAUCnIm is generated when the counter is triggered: 0: No INTTAUCnIm generated 1: INTTAUCnIm generated
One Count mode		Enables/disables start trigger detection during counting: 0: Disabled 1: Enabled INTTAUCnIm and TAUCnTTOUTm are not output when the counter is triggered.

(4) TAUCnTS - TAUCn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUCn_base> + $1C4_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TS15	TS14	TS13	TS12	TS11	TS10	TS09	TS08	TS07	TS06	TS05	TS04	TS03	TS02	TS01	TS00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 17-35 TAUCnTS register contents

Bit position	Bit name	Function
15 to 0	TSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUCnTE.TEm = 1. When the counter is enabled, this bit immediately returns to 0. TAUCnTE.TEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(5) TAUCnTE - TAUCn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 16-bit units.

Address <TAUCn_base> + $1C0_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE15	TE14	TE13	TE12	TE11	TE10	TE09	TE08	TE07	TE06	TE05	TE04	TE03	TE02	TE01	TE00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 17-36 TAUCnTE register contents

Bit position	Bit name	Function
15 to 0	TEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting TAUCnTS.TSm to 1 or trigger input detection TAUCnTSSTm = 1 sets this bit to 1. Setting TAUCnTT.TTm to 1 resets this bit to 0.

(6) TAUCnTT - TAUCn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUCn_base> + $1C8_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TT15	TT14	TT13	TT12	TT11	TT10	TT09	TT08	TT07	TT06	TT05	TT04	TT03	TT02	TT01	TT00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 17-37 TAUCnTT register contents

Bit position	Bit name	Function
15 to 0	TTm	Stops the counter of channel m: 0: No function 1: Stops the counter and sets TAUCnTE.TEm = 0. When the counter has stopped, this bit immediately returns to 0. TAUCnCNTm stops counting and TAUCnCNTm, TAUCnTO.TOm, and TAUCnTTOUTm all retain the values they had before the counter was stopped.

17.15.4 TAUCn output registers details

(1) TAUCnTOE - TAUCn channel output enable register

This register enables and disables Direct Channel Output Mode.

Access This register can be read/written in 16-bit units.

Address <TAUCn_base> + $5C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOE 15	TOE 14	TOE 13	TOE 12	TOE 11	TOE 10	TOE 09	TOE 08	TOE 07	TOE 06	TOE 05	TOE 04	TOE 03	TOE 02	TOE 01	TOE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-38 TAUCnTOE register contents

Bit position	Bit name	Function
15 to 0	TOEm	Enables/disables Direct Channel Output Mode: 0: Enables Direct Channel Output Mode (TAUCnTTOUT m output is controlled by the application software) 1: Disables Direct Channel Output Mode (TAUCnTTOUT m output is controlled by the timer)

(2) TAUCnTOM - TAUCn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUCnTE.TEm = 0).

Address <TAUCn_base> + 248_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM 15	TOM 14	TOM 13	TOM 12	TOM 11	TOM 10	TOM 09	TOM 08	TOM 07	TOM 06	TOM 05	TOM 04	TOM 03	TOM 02	TOM 01	TOM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-39 TAUCnTOM register contents

Bit position	Bit name	Function
15 to 0	TOMm	Specifies the timer controlled channel output mode, if Direct Channel Output Mode is disabled (TAUCnTOE.TOEm = 1): 0: Independent Channel Output Mode 1: Synchronous Channel Output Mode The output mode depends on several channel output control bits, as can be seen in *Table 17-13 “Channel output modes” on page 1348* .

17.15.5 TAUCn channel output level registers details

(1) TAUCnTO - TAUCn channel output register

This register specifies and reads the level of TAUCnTTOUTm.

Access This register can be read/written in 16-bit units.

Address <TAUCn_base> + 58_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO15	TO14	TO13	TO12	TO11	TO10	TO09	TO08	TO07	TO06	TO05	TO04	TO03	TO02	TO01	TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-40 TAUCnTO register contents

Bit position	Bit name	Function
15 to 0	TOm	Specifies/reads the level of TAUCnTTOUTm: 0: Low 1: High Only TOm bits for which Independent Channel Output function is disabled (TAUCnTOEm = 0) can be written.

(2) TAUCnTOL - TAUCn channel output level register

This register specifies the output logic of the channel output bit (TAUCnTO.TOm).

Access This register can be read/written in 16-bit units.

Address <TAUCn_base> + 040_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOL 15	TOL 14	TOL 13	TOL 12	TOL 11	TOL 10	TOL 09	TOL 08	TOL 07	TOL 06	TOL 05	TOL 04	TOL 03	TOL 02	TOL 01	TOL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-41 TAUCnTOL register contents

Bit position	Bit name	Function
15 to 0	TOLm	Specifies the output logic of the channel m output bit (TAUCnTO.TOm): 0: Positive logic (active high) 1: Inverted logic (active low) These bits apply in all channel output modes except Direct Channel Output Mode.

17.15.6 TAUCn simultaneous rewrite register details

(1) TAUCnRDE - TAUCn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUCnCDRm. It also enables simultaneous rewrite of the data register TAUCnTOLm for the PWM output function and the triangle PWM output function.

Access This register can be read/written in 16-bit units. It can only be written when TAUCnTE.TEm = 0.

Address <TAUCn_base> + 260_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDE 15	RDE 14	RDE 13	RDE 12	RDE 11	RDE 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-42 TAUCnRDE register contents

Bit position	Bit name	Function
15 to 0	RDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUCnRDS - TAUCn channel reload data control channel select register

This register selects the control channel for simultaneous rewrite.

Access This register can be read/written in 16-bit or 1-bit units. It can only be written when TAUCnTE.TEm = 0.

Address <TAUCn_base> + 268_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDS 15	RDS 14	RDS 13	RDS 12	RDS 11	RDS 10	RDE 09	RDE 08	RDE 07	RDE 06	RDE 05	RDE 04	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-43 TAUCnRDS register contents

Bit position	Bit name	Function
15 to 0	RDSm	Specifies which channel is monitored for the simultaneous rewrite trigger: 0: Master channel 1: Another upper channel

(3) TAUCnRDC - TAUCn channel reload data control register

This register specifies the channel that generates the INTTAUCnIm signal that triggers simultaneous rewrite.

Access This register can be read/written in 16-bit units. It can only be written when TAUCnTE.TEm = 0

Address <TAUCn_base> + $26C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDC 15	RDC 14	RDC 13	RDC 12	RDC 11	RDC 10	RDC 09	RDC 08	RDC 07	RDC 06	RDC 05	RDC 04	RDC 03	RDC 02	RDC 01	RDC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-44 TAUCnRDC register contents

Bit position	Bit name	Function
15 to 0	RDCm	Specifies whether the channel is monitored for an INTTAUCnIm signal that is used as the simultaneous rewrite trigger: 0: Channel is not monitored 1: Channel is monitored These bits only apply when TAUCnRDS.RDSm = 1.

(4) TAUCnRDT - TAUCn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 16-bit units. It is always read as 0000_H.

Address <TAUCn_base> + 044_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDT 15	RDT 14	RDT 13	RDT 12	RDT 11	RDT 10	RDT 09	RDT 08	RDT 07	RDT 06	RDT 05	RDT 04	RDT 03	RDT 02	RDT 01	RDT 00
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 17-45 TAUCnRDT register contents

Bit position	Bit name	Function
15 to 0	RDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUCnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger. TAUCnRDT.RDTm immediately returns to 0.

(5) TAUCnRSF - TAUCn channel reload status register

This flag register indicates that simultaneous rewrite is possible.

Access This register can be read in 16-bit units.

Address <TAUCn_base> + 048_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSF 15	RSF 14	RSF 13	RSF 12	RSF 11	RSF 10	RSF 09	RSF 08	RSF 07	RSF 06	RSF 05	RSF 04	RSF 03	RSF 02	RSF 01	RSF 00
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 17-46 TAUCnRSF register contents

Bit position	Bit name	Function
15 to 0	RSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewrite disabled 1: Simultaneous rewrite enabled

17.15.7 TAUCn emulation register

(1) TAUCnEMU - TAUC emulation register

This register controls whether the TAUCn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units only when TAUCnTE.TEm = 0.

Address <TAUCn_base> + 290_H

Initial Value 00_H

7	6	5	4	3	2	1	0
TAUCn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 17-47 TAUCnEMU register contents

Bit position	Bit name	Function
7	TAUCn SVSDIS	Emulation control 0: TAUCn can be stopped during emulation 1: TAUCn continuous operating during emulation

Chapter 18 Timer Array Unit J (TAUJ)

This chapter contains a generic description of the Timer Array Unit J (TAUJ).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

18.1 V850E2/Fx4-H TAUJ Features

Instances

This microcontroller has following number of instances of the Timer Array Unit J.

Table 18-1 Instances of TAUJ

Timer Array Unit J	
Instance	2
Name	TAUJ0, TAUJ1

Instances index n

Throughout this chapter, the individual instances of a Timer Array Unit J is identified by the index "n" (n = 0, 1), for example, TAUJnTOM for the TAUJn channel output mode register.

Channel index m

The Timer Array Unit J has 4 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 3), thus a certain channel is denoted as CHm.
The even numbered channels (m = 0, 2) are denoted as CHm_even.
The odd numbered channels (m = 1, 3) are denoted as CHm_odd.

Register addresses

All TAUJ n register addresses are given as address offsets to the individual base address <TAUJ n_base>.
The <TAUJ n_base> address of each TAUJn are listed in the following table:

Table 18-2 Register base addresses <TAUJn_base>

TAUJn instance	<TAUJn_base> address
TAUJ 0	FF81 1000_H
TAUJ 1	FF81 2000_H

Clock supply All Timer Array Units J provide one clock input.

Table 18-3 TAUJn clock supply

TAUJn instance	TAUJn clock	Connected to
TAUJ0	PCLK	Clock Controller CKSCLK_A03
TAUJ1	PCLK	Clock Controller CKSCLK_A04

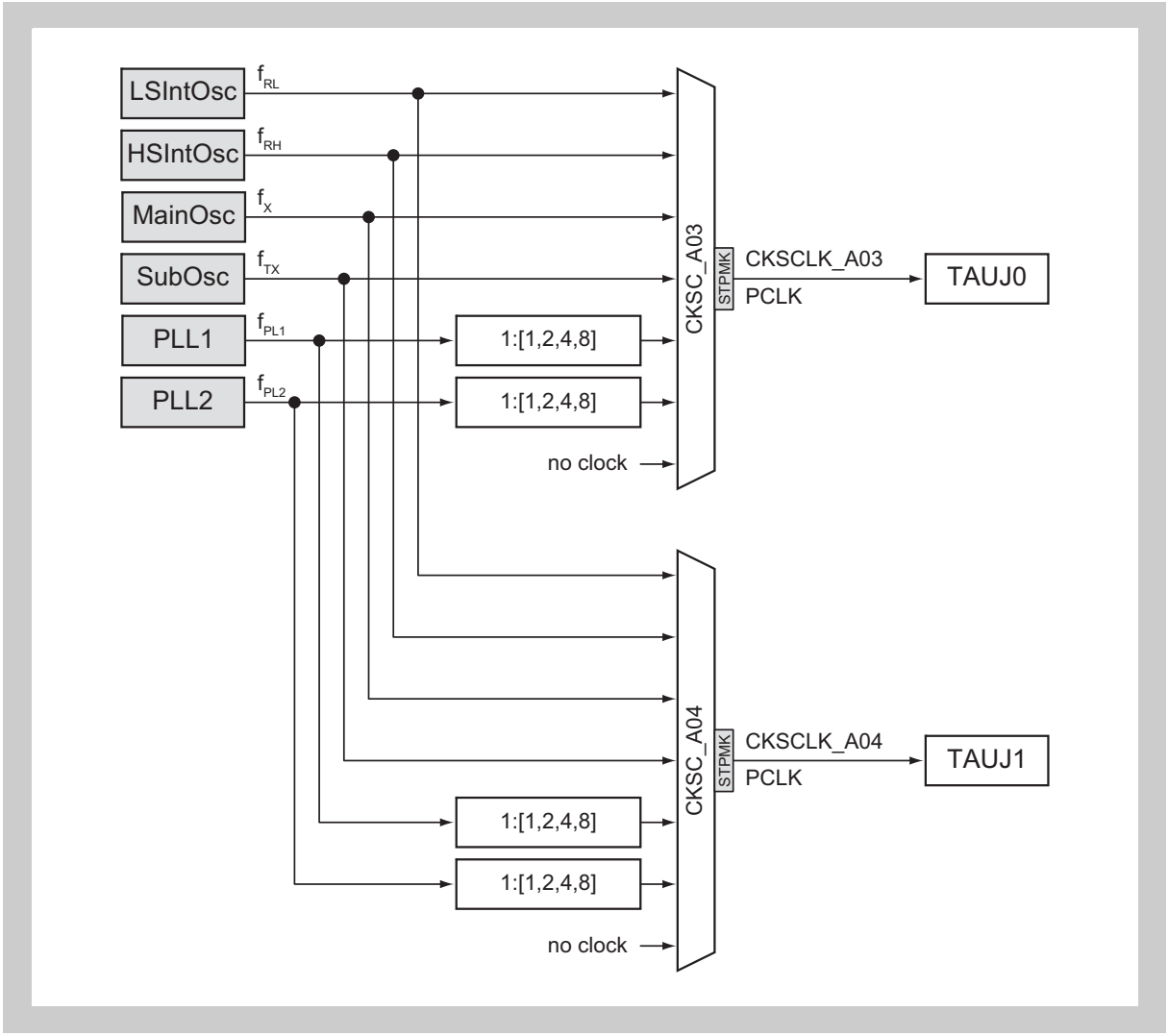


Figure 18-1 TAUJ clock supply

Interrupts and DMA

The Time Array Unit J can generaste the following interrupt and DMA/DTS requests:

Table 18-4 TAUJn interrupt and DMA/DTS requests

TAUJn signal	Function	Connected to
TAUJ0:		
INTTAUJ0I0	Channel 0 interrupt	Interrupt Controller INTTAUJ0I0[a] DMA Controller trigger 59
INTTAUJ0I1	Channel 1 interrupt	Interrupt Controller INTTAUJ0I1[a] DMA Controller trigger 60
INTTAUJ0I2	Channel 2 interrupt	Interrupt Controller INTTAUJ0I2[a] DMA Controller trigger 61 DTS Controller trigger 63
INTTAUJ0I3	Channel 3 interrupt	Interrupt Controller INTTAUJ0I3[ab] DMA Controller trigger 62
TAUJ1:		
INTTAUJ1I0	Channel 0 interrupt	Interrupt Controller INTTAUJ1I0[a] DMA Controller trigger 63
INTTAUJ1I1	Channel 1 interrupt	Interrupt Controller INTTAUJ1I1[a] DMA Controller trigger 64
INTTAUJ1I2	Channel 2 interrupt	Interrupt Controller INTTAUJ1I2[a] DMA Controller trigger 65
INTTAUJ1I3	Channel 3 interrupt	Interrupt Controller INTTAUJ1I3[ab] DMA Controller trigger 66

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

b) These signals can be used to as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

TAUJ H/W reset

The Time Array Units J and their registers are initialized by the following reset signal:

Table 18-5 TAUJn reset signal

TAUJn	Reset signal
TAUJn	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the Timer Array Unit J are used for various purposes, as listed in the following table.

Note √ / – : used / not used for the function

Table 18-6 TAUJ0 I/O signals

TAUJ0 signal	Connected to		
	Miscellaneous[a]	Port	Encoder Timer[b]
TAUJ0 input channels: All TAUJ0 port input signals are passed through a noise filter, refer to *2.5 "Port Filters" on page 143* for further details.			
TAUJ0TTIN0	–	TAUJ0I0	√
TAUJ0TTIN1	–	TAUJ0I1	√
TAUJ0TTIN2	RTCA1HZ	TAUJ0I2	–
TAUJ0TTIN3	RTCA1HZ	TAUJ0I3	–
TAUJ0 output channels:			
TAUJ0TTOUT0	–	TAUJ0O0	–
TAUJ0TTOUT1	–	TAUJ0O1	–
TAUJ0TTOUT2	–	TAUJ0O2	–
TAUJ0TTOUT3	–	TAUJ0O3	–

a) Refer to section *18.2 "TAUJ Input Selection" on page 1400* for further details.
b) Refer to chapter *21 "Encoder Timer" on page 1694* and section *18.2 "TAUJ Input Selection" on page 1400* for further details.

Table 18-7 TAUJ1 I/O signals

TAUJ1 signal	Connected to		
	Miscellaneous[a]	Port	Encoder Timer[b]
TAUJ1 input channels: All TAUJ1 port input signals are passed through a noise filter, refer to *2.5 "Port Filters" on page 143* for further details.			
TAUJ1TTIN0	–	TAUJ0I0	–
TAUJ1TTIN1	–	TAUJ0I1	–
TAUJ1TTIN2	RTCA1HZ	TAUJ0I2	–
TAUJ1TTIN3	RTCA1HZ	TAUJ0I3	–
TAUJ1 output channels:			
TAUJ1TTOUT0	–	TAUJ1O0	–
TAUJ1TTOUT1	–	TAUJ1O1	–
TAUJ1TTOUT2	–	TAUJ1O2	–
TAUJ1TTOUT3	–	TAUJ1O3	–

a) Refer to section *18.2 "TAUJ Input Selection" on page 1400* for further details.
b) Refer to chapter *21 "Encoder Timer" on page 1694* and section *18.2 "TAUJ Input Selection" on page 1400* for further details.

18.2 TAUJ Input Selection

18.2.1 TAUJ0/TAUJ1 input selection

The TAUJ0 and TAUJ1 have several options to connect its input signals:

- Real-Time Clock 1 second interval output RTCA1HZ for clock corrections
- FCN0, FCN1 time stamp output signals TSOUT for timing measurements
- connections to Encoder Timer
  Refer to the chapter *"Encoder Timer"* for further details

The following figure depicts the TAUJ0 and TAUJ1 input selection scheme:

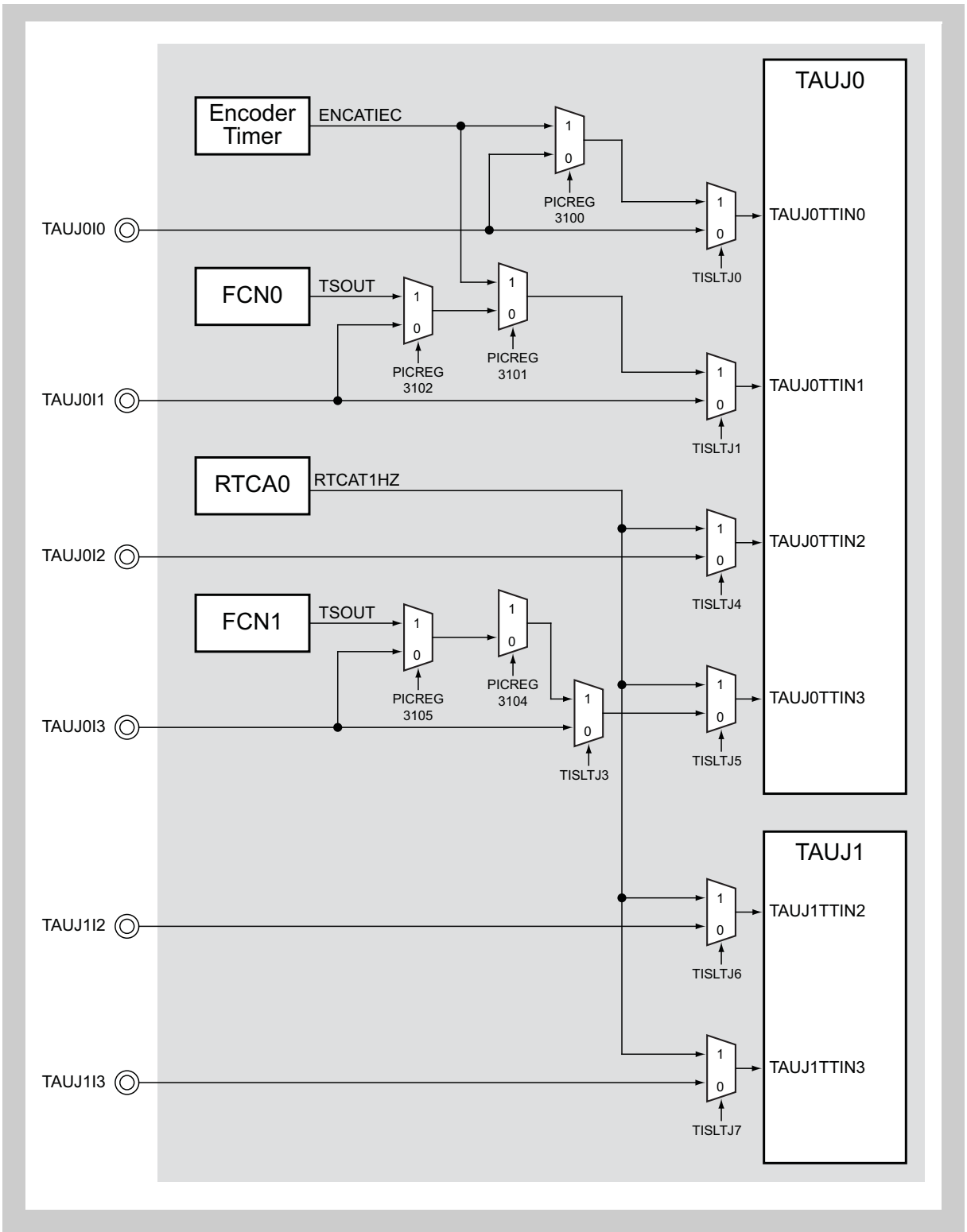


Figure 18-2 TAUJ0 and TAUJ1 input selector scheme

The following table shows the optional inputs to several TAUJ inputs:

Table 18-8 TAUJ input selections

TAUJn input	Input options	Selection control
TAUJ0TTIN0	Port TAUJ0I0	TISLTJ.TISLTJ0 = 0
	Connected to Encoder Timer (ENCA)[a]	TISLTJ.TISLTJ0 = 1 PIC0REG31.PIC0REG3100 = 1
TAUJ0TTIN1	Port TAUJ0I1	TISLTJ.TISLTJ1 = 0
	Connected to Encoder Timer (ENCA)[a]	TISLTJ.TISLTJ1 = 1 PIC0REG31.PIC0REG3101 = 1
	FCN0 time stamp output TSOUT	TISLTJ.TISLTJ1 = 1 PIC0REG31.PIC0REG3101 = 1 PIC0REG31.PIC0REG3102 = 1
TAUJ0TTIN2	Port TAUJ0I2	TISLTJ.TISLTJ4 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ4 = 1
TAUJ0TTIN3	Port TAUJ0I3	TISLTJ.TISLTJ5 = 0 TISLTJ.TISLTJ3 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ5 = 1
	FCN1 time stamp output TSOUT	TISLTJ.TISLTJ5 = 0 TISLTJ.TISLTJ3 = 1 PIC0REG31.PIC0REG3104 = 0 PIC0REG31.PIC0REG3105 = 1
TAUJ1TTIN2	Port TAUJ1I2	TISLTJ.TISLTJ6 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ6 = 1
TAUJ1TTIN3	Port TAUJ1I3	TISLTJ.TISLTJ7 = 0
	RTCA0 RTCAT1HZ (Real-Time Clock 1 Hz output)	TISLTJ.TISLTJ7 = 1

a) Refer to chapter *"Encoder Timer"* additional input selector settings.

(1) TISLTJ - Timer input selection register J

This register selects the input signals to several TAUJn inputs.

Access This register can be read/written in 8-bit units.

Address FF77 3000$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
TISLTJ7	TISLTJ6	TISLTJ5	TISLTJ4	TISLTJ3	0	TISLTJ1	TISLTJ0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-9 TISLTJ register contents

Bit position	Bit name	Function
7	TISLTJ7	Selection of TAUJ1TTIN3: 0: port TAUJ1I3 1: RTCA0 RTCAT1HZ
6	TISLTJ6	Selection of TAUJ1TTIN2: 0: port TAUJ1I2 1: RTCA0 RTCAT1HZ
5 3	TISLTJ5 TISLTJ3	Selection of TAUJ1TTIN3: TISLTJ5 / TISLTJ3 / Input signal 0 / 0 / Port TAUJ0I3 0 / 1 / Selection of PIC0REGREG31.PIC0REGREG3104 1 / x / RTCA0 RTCAT1HZ
4	TISLTJ4	Selection of TAUJ0TTIN2: 0: port TAUJ0I2 1: RTCA0 RTCAT1HZ
1	TISLTJ1	Selection of TAUJ0TTIN1: 0: port TAUJ0I1 1: signal selection by PIC0REGREG31.PIC0REGREG3101
0	TISLTJ0	Selection of TAUJ0TTIN0: 0: port TAUJ0I0 1: signal selection by PIC0REGREG31.PIC0REGREG3100

(2) PIC0REG31 - Timer I/O control register 31

Access This register can be read or written in 32-bit units.

Address FF81 C0C0$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	PIC0REG3113	PIC0REG3112	PIC0REG3111	PIC0REG3110	PIC0REG3109	PIC0REG3108
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PIC0REG3107	PIC0RG3106	0	0	0	PIC0REG3102	PIC0REG3101	PIC0RG3100
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-10 PIC0REG31 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13 to 6</td><td>PIC0REG3113 to PIC0REG3106</td><td>These bits are used for the selection of the TAUA0TTIN0 to TAUA0TTIN2 inputs for trigger pulse width measurements of Encoder Timer signals.[a]</td></tr>
<tr><td>5
4</td><td>PIC0REG3105,
PIC0REG3104</td><td>Selection of TAUJ0TTIN3 input.
Selections by PIC0REG3104 and PIC0REG3105 are only effective, if TISLTJ.TISLTJ5 = 0 and TISLTJ.TISLTJ3 = 1.
<table>
<tr><th>PIC0REG3105</th><th>PIC0REG3104</th><th>Input signal</th></tr>
<tr><td>0</td><td>0</td><td>Port TAUJ0I3</td></tr>
<tr><td>1</td><td>0</td><td>FCN1 time stamp output TSOUT</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>2
1</td><td>PIC0REG3102
PIC0REG3101</td><td>Selection of TAUJ0TTIN1 input.
Selections by PIC0REG3101 and PIC0REG3102 are only effective, if TISLTJ.TISLTJ1 = 1.
<table>
<tr><th>PIC0REG3102</th><th>PIC0REG3101</th><th>Input signal</th></tr>
<tr><td>0</td><td>0</td><td>Port TAUJ0I1</td></tr>
<tr><td>0</td><td>1</td><td rowspan="2">ENCATIEC signal output via DT[a]</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>FCN0 time stamp output TSOUT</td></tr>
</table></td></tr>
<tr><td>0</td><td>PIC0REG3100</td><td>Selection of TAUJ0TTIN0 input:
0: port TAUJ0I0
1: select the ENCATIEC signal output via DT[a]</td></tr>
</table>

a) Refer to the section *"Trigger Pulse Width Measurement"* in the *"Encoder Timer"* chapter for details.

18.3 Functional Overview

Features summary

The TAUJ has the following functions:

- 4 channels
- 32-bit counter and 32-bit data register per channel
- Independent channel operation
- Synchronous channel operation (master and slave operation)
- Generation of different types of output signal
- Counter can be triggered by external signal
- Interrupt generation

The following figure shows the main components of the TAUJ:

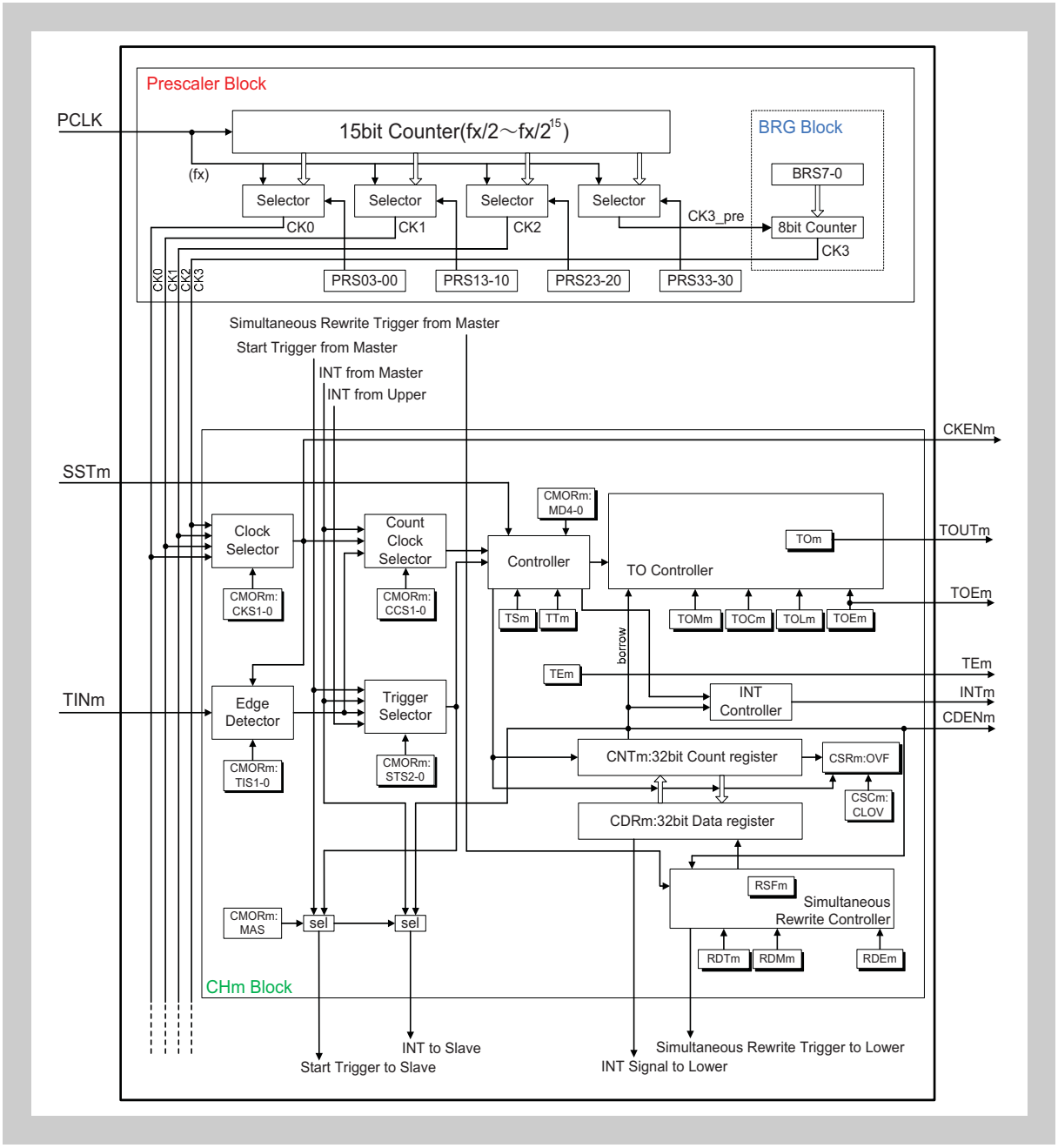


Figure 18-3 Block diagram of the TAUJ

The prefix “TAUJn” has been omitted from the register names for the sake of clarity in the above figure.

18.3.1 Terms

In this chapter, the following terms are used:

- **Independent / synchronous channel operation**

  Independent or synchronous channel operation describes the dependency of channels on each other:

  – If a channel operates independent of all other channels, this is called independent channel operation.

  – If a channel operates depending on other channels, this is called synchronous channel operation.

- **Channel group**

  In synchronous channel operation, all channels that depend on each other are referred to as a “channel group”.

  A channel group has one master channel and one or more slave channels.

- **Operation mode**

  An operation mode can be selected for every channel m. The operation mode defines the *basic* operation and features of a channel.

  In synchronous channel operation, every channel in the channel group can operate in a different operation mode.

  Examples are “Capture Mode”, and “Interval Timer Mode”.

- **Channel output mode**

  The channel output mode defines the operation of TAUJnTTOUTm

  – of a single channel (independent output operation) or

  – of all channels in a channel group (synchronous output operation).

  Example: “Independent Channel Output Mode 1”

- **Channel operation function**

  The channel operation function defines the *complete* function and all features

  – of a single channel (independent channel operation) or

  – of all channels in a channel group (synchronous channel operation).

  It defines the operation mode, start and capture trigger, count clock, and the channel output mode.

  Examples are “Interval Timer Function”, “TINm Input Position Detection Function”, and “TINm Input Period Count Detection Function”

- **Upper / lower channel**

  Depending on the channel number m, a neighboring channel can be referred to as “upper” or “lower” channel:

  – Upper channel: Channel with a smaller channel number

  – Lower channel: Channel with a higher channel number

  Example:
  For channel 2, channel 1 is an upper channel and channel 3 is a lower channel.

18.4 Functional Description

The Timer Array Unit J is used to perform various count or timer operations and to output a signal which depends on the result of the operation. It contains one prescaler block for count clock generation and 4 channels, each equipped with a 32-bit counter TAUJnCNTm and a 32-bit data register TAUJnCDRm to hold the start or compare value of the counter.

It also contains several control and status registers.

Independent and synchronous operation

Every channel can operate in two operation modes, either independently or in combination with other channels (synchronously), i.e. multiple channels depend on each other with one master and one or more slave channels.

When a channel is operated independently, its operation mode and functions are not affected by those of other channels. When a channel is operated synchronously it is either a master or a slave. A master channel can have multiple slaves, and the state of one channel affects that of the other channels. For example, this means that one channel can control when another starts to count, is reset, etc.

The following describes the functional blocks:

Prescaler block

The prescaler block provides up to 4 clock signals (CK0 to CK3) that can be used as count clocks for all channels.

Count clocks CK0 to CK2 are derived from PCLK by a configurable prescaler division factor of 2^0 to 2^{15}. The fourth count clock CK3 can be adjusted more precisely by an additional division factor that is not a power of 2.

Clock and count clock selection

For every channel, the count clock selector selects which of the following is used as the clock source:

- One of the clocks CK0 to CK3 (selected by the clock selector)
- INTTAUJnIm from master channel
- Edge detected TAUJnTTINm input signal

Controller

The controller controls the main operations of the counter:

- Operation mode (selected by bits TAUJnCMORm.MD[4:0])
- Counter start enable (TAUJnTS.TSm) and counter stop (TAUJnTT.TTm)

  When counter start is enabled, status flag TAUJnTE.TEm is set.

Trigger selector

Depending on the selected operation mode, the counter starts automatically when it is enabled (TAUJnTE.TEm = 1), or it waits for an external start trigger signal. Any of the following signals can be used as the start trigger:

- Synchronous channel start trigger input TAUJnTSSTm
- TAUJnTTINm input valid edge
- INTTAUJnIm from the master channel

Simultaneous rewrite controller	Simultaneous rewrite control is a special function that can be used in synchronous operation modes. The data registers of all channels in a channel group can be rewritten at any time. The simultaneous rewrite controller ensures that new data register values of all channels become effective at the same time.
TAUJnTO Controller	The output control of every channel enables the generation of various output signal forms such as PWM signals.
Signals	The TAUJ has various input and output signals.A full list can be found in the first section of this chapter under the keyword “I/O signals”.

18.5 General Operating Procedure

The following lists the general operation procedure for the TAUJn:

After reset release, the operation of each channel is stopped. Clock supply is started and writing to each register is enabled. All circuits and registers of all channels are initialized. The control register of TAUJnTTOUTm is also initialized and outputs a low level.

1. Set the TAUJnTPS and TAUJnBRS registers to specify the clock frequency of CK0 to CK3.
2. Configure the desired TAUJn function:
   – Set the operation mode
   – Set the channel output mode
   – Set any other control bits
3. Enable the counter by setting the TAUJnTS.TSm bit to 1.
   The counter starts to count immediately, or when an appropriate trigger is detected, depending on the bit settings.
   The function is in operation.
4. If desired, and if possible for the configured function, stop the counter or perform a forced restart operation.
5. Stop the function by setting the TAUJnTT.TTm bit to 0.

Note A detailed description of the required control bits and the operation of the individual functions is given in the following chapters:

18.15 “Independent Channel Interrupt Functions” on page 1435

18.16 “Independent Channel Signal Measurement Fuctions” on page 1449

18.17 “Other Independent Channel Functions” on page 1479

18.18 “Synchronous PWM Signal Functions Triggered at Regular Intervals” on page 1486

18.6 Operation Modes

The TAUJ contains 7 operation modes. These determine the basic behavior of a channel, for example whether a timer counts up or down, whether the data register TAUJnCDRm acts as a compare register or stores the initial value of the counter, etc.

One operation mode can be set for each channel. It is specified using the TAUJnCMORm.MD[4:0] bits.

When choosing a function, these settings cannot be specified individually, but are grouped under the term operation mode. If a function uses multiple channels, the operation mode of each channel must be set correctly for the function to work correctly.

Note For more information about the operation modes required by each function, refer to the required function in the following chapters:

18.15 “Independent Channel Interrupt Functions” on page 1435

18.16 “Independent Channel Signal Measurement Fuctions” on page 1449

18.17 “Other Independent Channel Functions” on page 1479

18.18 “Synchronous PWM Signal Functions Triggered at Regular Intervals” on page 1486

18.7 Concepts of Synchronous Channel Operation

In synchronous channel operation, multiple channels depend on each other, or are affected by changes in another channel. Therefore, several rules apply for the use of synchronous channel functions. These rules are detailed in *18.7.1 "Rules"*.

Two special features for synchronous channel operation are detailed in the following subchapters:

- *18.7.2 "Simultaneous start and stop of synchronous channel counters" on page 1415*
- *18.8 "Simultaneous Rewrite" on page 1416*

18.7.1 Rules

Number of masters and slaves

- Only even channels (CH0, CH2) can be set as master channels. Any channel apart from CH0 can be set as a slave channel.
- Only channels lower than the master channel can be set as slave channels, and several slave channels can be set for one master channel.
  Example: If CH2 is a master channel, CH3 can be set as slave channel.
- If two master channels are used, slave channels cannot cross the master channels.
  Example: If CH0 and CH2 are master channels, CH1 can be set as slave channels for CH0, but CH3 cannot.

Operation clock

- The same operation clock must be set for the slave channel and the master channel. This is achieved using the TAUJnCMORm.CKS[1:0] bits of the slave and master channel.

The basic concepts of master/slave usage and operation clocks are illustrated in the following figure.

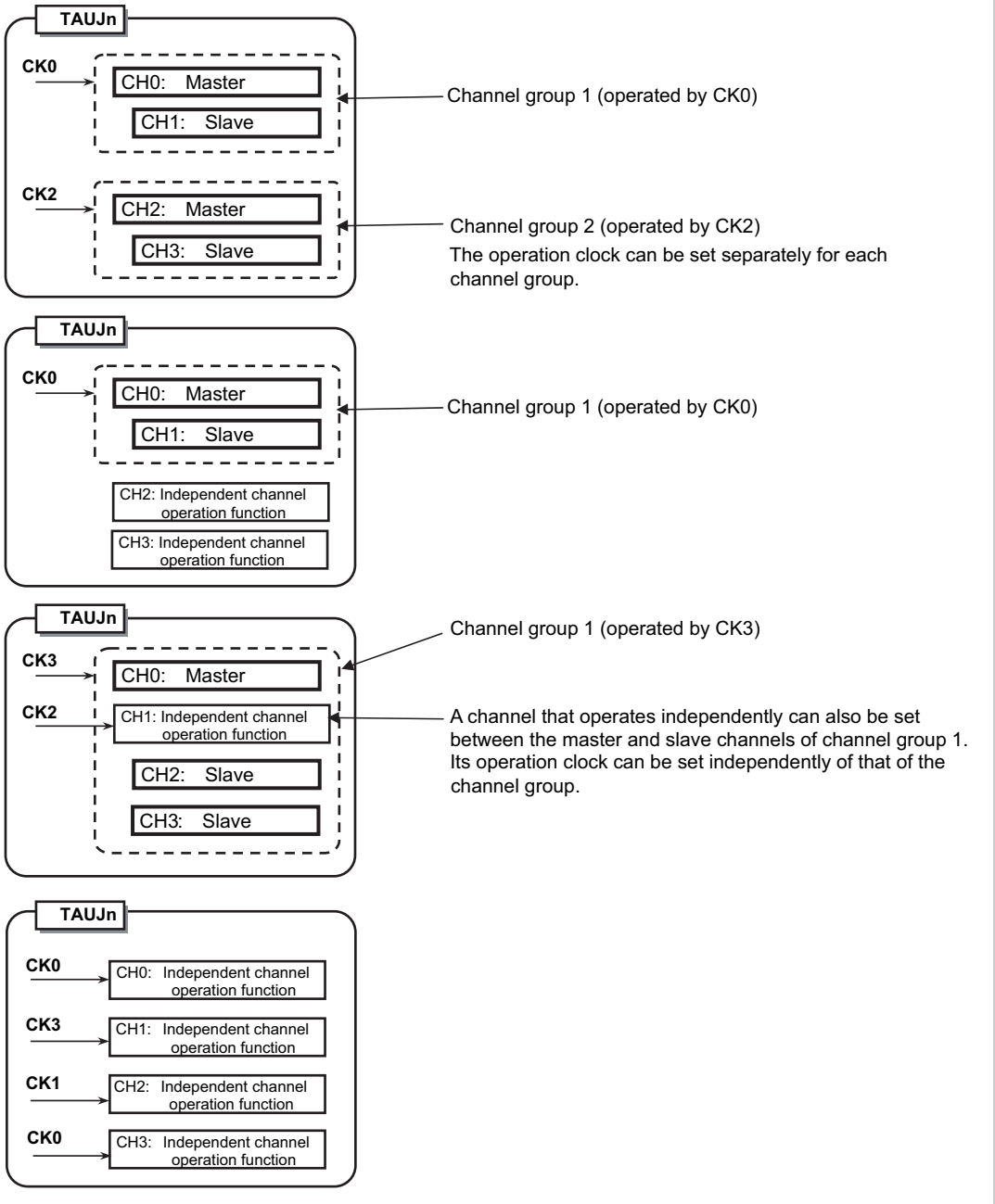


Figure 18-4 Grouping of the channels and assignment of operation clocks

INTTAUJnIm, start trigger, and count clock

- Master channels can transfer an interrupt request (INTTAUJnI), the start trigger, and the count clock to slave channels.
- Slave channels can use INTTAUJnI, the start trigger, and the count clock of the master channels but cannot transfer their INTTAUJnI, start trigger, or count clock to the lower channels.
- A master channel cannot use INTTAUJnI, the start trigger, or the count clock of the higher master channels.

18.7.2 Simultaneous start and stop of synchronous channel counters

Channels that are operated synchronously can be started and stopped simultaneously, both within a TAUJ unit, and between TAUJ units.

(1) Simultaneous start and stop within a TAUJ unit

- To simultaneously start synchronized channels, the TAUJnTS.TSm bits of the channels must be set at the same time.
- To simultaneously stop synchronized channels, the TAUJnTT.TTm bits of the channels must be set at the same time.

Writing to the TAUJnTS.TSm bits sets the corresponding TAUJnTE.TEm bits to 1, enabling counting. TAUJnTS.TSm = 1 only enables the corresponding counter to start; the exact time that it starts depends on the operation mode.

(2) Simultaneous start between TAUJ units

Counters in different TAUJ units can also be started simultaneously if the corresponding counters are enabled before receiving the simultaneous trigger signal. The simultaneous start trigger register is then sent to the TAUJnTSSTm input.

18.8 Simultaneous Rewrite

18.8.1 Introduction

Simultaneous rewrite describes the ability to change the compare/start value and the output logic of multiple channels at the same time.

The corresponding data and control registers (TAUJnCDRm and TAUJnTOLm) can nevertheless be written at any time. The new value does not affect the counter operation or the output signal until simultaneous rewrite is triggered.

Simultaneous rewrite can be triggered by the counter on the master channel reaching a certain value.

The following table shows the settings for simultaneous rewrite (TAUJnRDM.RDMm = 0).

Table 18-11 Simultaneous rewrite settings

Method	Simultaneous rewrite triggered when	TAUJnRDE.RDEm
-	No simultaneous rewrite	0
A	The master channel (re)starts counting	1

18.8.2 How to control simultaneous rewrite

The following figure shows the general procedure for simultaneous rewrite. The three main blocks (Initial settings, Start counter & count operation, and Simultaneous rewrite) are explained afterwards.

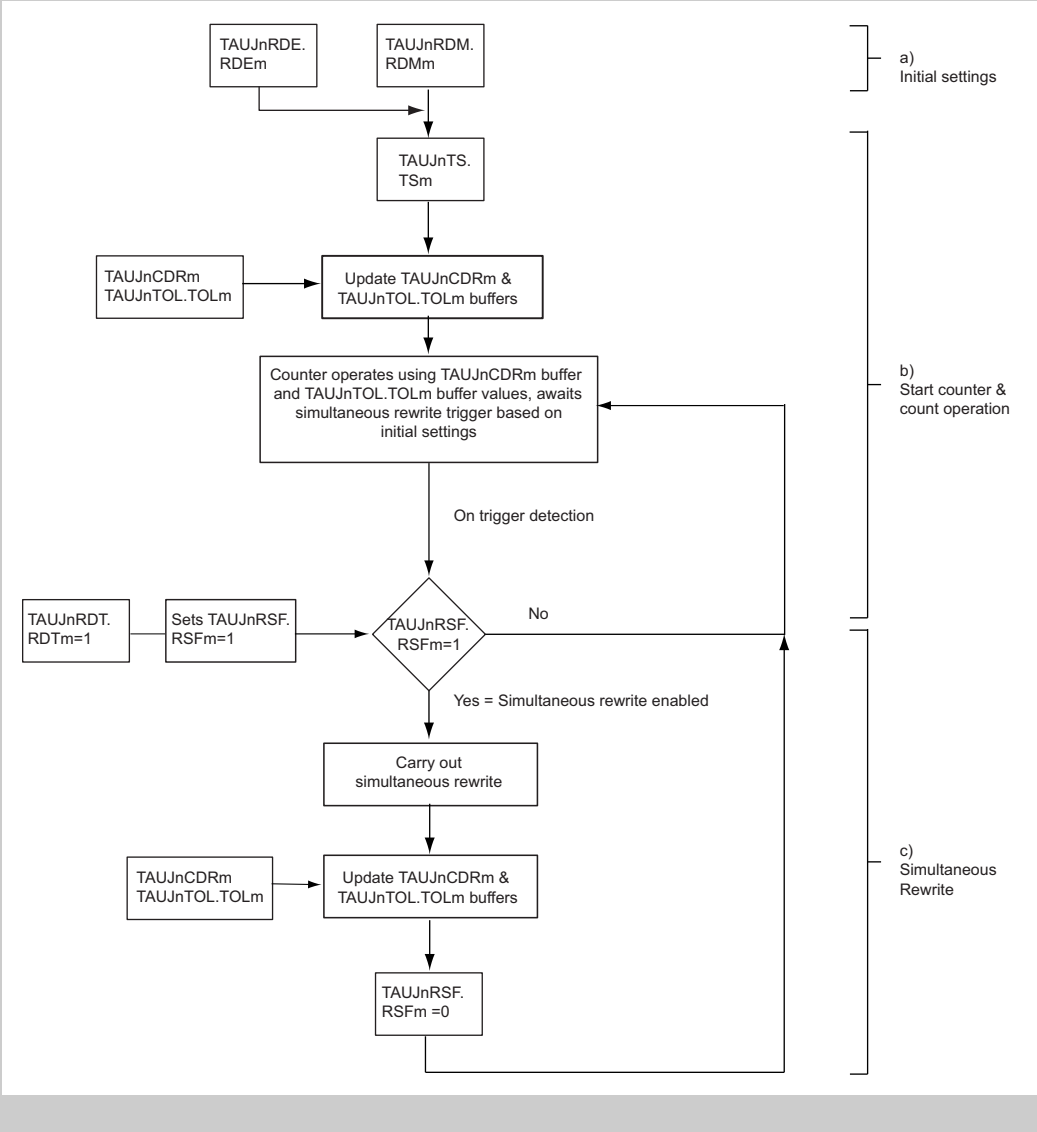


Figure 18-5 General procedure for simultaneous rewrite

(1) Initial settings

- To enable simultaneous rewrite in channel m, set TAUJnRDE.RDEm = 1
- To select simultaneous rewrite when the master channel starts counting, set TAUJnRDM.RDMm = 0

(2) Start counter and count operation

- To start all the TAUJnCNTm counters in the channel group, set the corresponding TAUJnTS.TSm bits to 1. TAUJnTOL.TOLm and the values in the data registers (TAUJnCDRm) are written to the corresponding TOLm buffer (TAUJnTOL.TOLm buf) and data buffer registers (TAUJnCDRm buf)

and the counters start.

- Setting the reload data trigger bit (TAUJnRDT.RDTm) to 1 sets the reload flag (TAUJnRSF.RSFm) to 1, enabling simultaneous rewrite. TAUJnRDT.RDTm then immediately returns to 0, but TAUJnRSF.RSFm remains at 1 until simultaneous rewrite has taken place.
- When the specified trigger for simultaneous rewrite is detected, the TAUJnRSF.RSFm bit is checked to see if simultaneous rewrite is enabled (TAUJnRSF.RSFm = 1). If it is, simultaneous rewrite is carried out. Otherwise the value of the TAUJnRSF.RSFm bit is re-evaluated the next time the trigger is detected.

(3) Simultaneous rewrite

- When the simultaneous rewrite trigger is detected and simultaneous rewrite is enabled (TAUJnRSF.RSFm = 1), the current values of the data registers are copied to their buffers. These values are then written to the corresponding counters and the values are applied the next time the counter starts or restarts.
- The TAUJnRSF.RSFm bit is set to 0, and the system awaits the next simultaneous rewrite trigger.

18.8.3 Other general rules of simultaneous rewrite

The following rules also apply:

- TAUJnRDE.RDEm and TAUJnRDM.RDMm cannot be changed while the counter is in operation (TAUJnTE.TEm = 1).
- TAUJnTOL.TOLm can only be rewritten during operation when in PWM output function. For all other output functions, TAUJnTOL.TOLm must be written before the counter starts. If it is rewritten in another function, TAUJnTTOUTm outputs an invalid wave.

18.8.4 Simultaneous rewrite procedure

Simultaneous rewrite is executed, when the master channel (re)starts counting. The simultaneous rewrite procedure is described in the following figure.

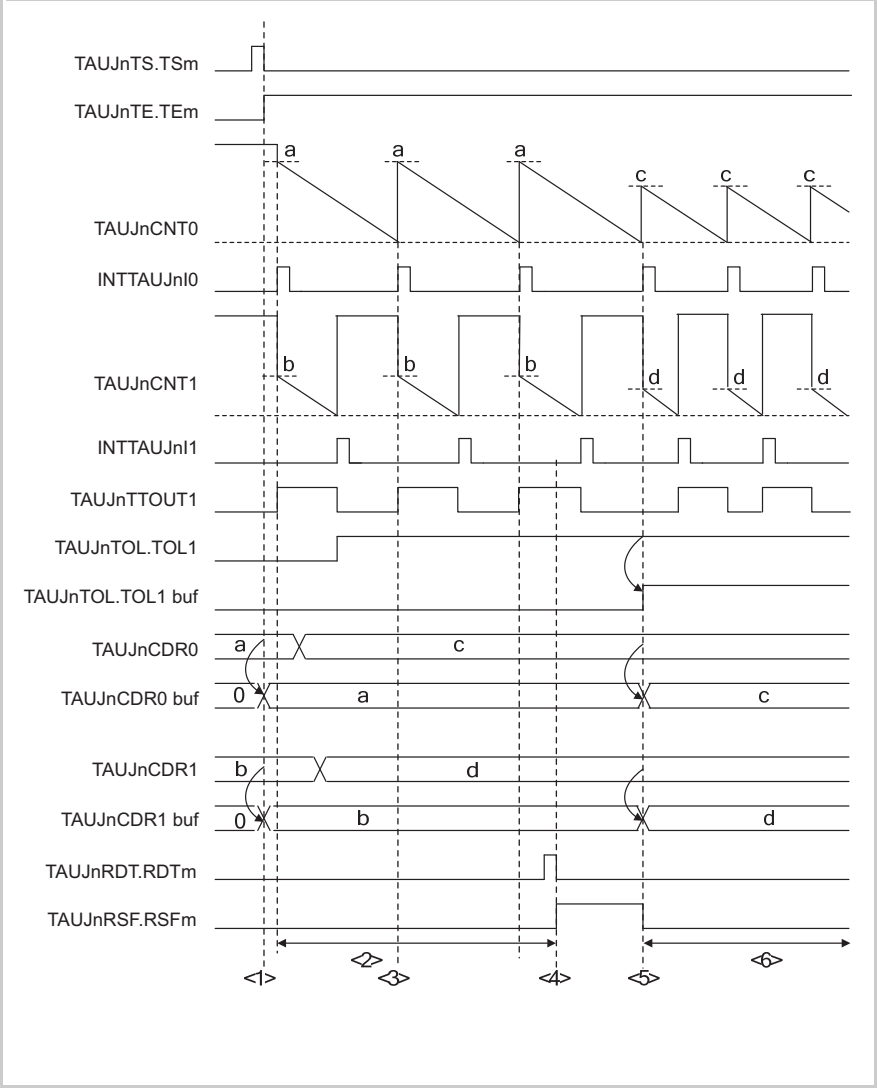


Figure 18-6 Simultaneous rewrite when the master channel (re)starts counting

Setup:

- CH0 is the master channel, counting down, CH1 represents an arbitrary slave channel, and simultaneous rewrite is applied when the master channel starts counting.

Description:

1. When the counter starts, the value of TAUJnCDRm is copied to the TAUJnCDRm buffer and the value of TAUJnTOL.TOLm is copied to the TAUJnTOL.TOLm buffer. The TAUJnCDRm buffer value is written to the counter.
2. The TAUJnCDRm and TAUJnTOL.TOLm registers can be written at any time, but the values do not affect the counter as the counter reads the buffer values.
3. CH0 restarts counting, but simultaneous rewrite does not occur because it is disabled (TAUJnRSF.RSFm = 0).
4. The reload data trigger bit (TAUJnRDT.RDTm) is set to 1 which sets the status flag (TAUJnRSF.RSFm = 1), enabling simultaneous rewrite.
5. Simultaneous rewrite is triggered by counter TAUJnCNT0 starting to count down. The TAUJnCDRm value is written to the TAUJnCDRm buffer and the TAUJnTOL.TOLm value is written to the TAUJnTOL.TOLm buffer.
   – The counter starts to count down from the value in the TAUJnCDRm buffer and the TAUJnRSF.RSFm bit is reset to 0.
   – The output logic specified by TAUJnTOL.TOLm becomes effective.
6. The counters count down and await the next simultaneous rewrite trigger. The values of TAUJnCDRm and TAUJnTOL.TOLm can be changed again.

18.9 Channel Output Modes

The output of the TAUJnTTOUTm pin can be controlled in two ways, the latter of which can be further split into individual modes:

- By software (Direct Channel Output Mode, TAUJnTOE.TOEm = 0)

  When controlled by software, the output register bit (TAUJnTO.TOm) can be written and the value of the bit is transferred to the output pin (TAUJnTTOUTm).

- By TAUJ signals (TAUJnTOE.TOEm = 1)

  When operated by TAUJ signals, the output level of TAUJnTTOUTm is set or reset or toggled by internal signals. The value of TAUJnTO.TOm is updated accordingly to reflect the value of TAUJnTTOUTm.

  – Independently (Independent Channel Output Mode, TAUJnTOM.TOMm = 0)

    When operated independently, the output of the TAUJnTTOUTm pin is only affected by settings of channel m. Therefore, independent channel operation must be selected (TAUJnTOM.TOMm = 0).

  – Synchronously (Synchronous Channel Output Mode, TAUJnTOM.TOMm = 1)

    When operated synchronously, the output of the TAUJnTTOUTm pin is affected by settings of channel m and those of other channels. Therefore, synchronous channel operation must be selected for all participating channels (TAUJnTOM.TOMm = 1).

The TAUJnTO.TOm bit can always be read to determine the current value of TAUJnTTOUTm, regardless of whether the pin is controlled by software, operated independently, or operated synchronously.

Control bits

The settings of the control bits required to select a specific channel output mode are listed in *Table 18-12 “Channel output modes” on page 1422* .

The channel output modes are described in detail in

- *18.9.2 “Channel output modes controlled independently by TAUJn signals” on page 1424* to
- *18.9.3 “Channel output modes controlled synchronously by TAUJn signals” on page 1425* .

Output logic

Positive logic or inverted logic of the output is specified by control bit TAUJnTOL.TOLm.

The value of the TAUJnTOL.TOLm bit must be set before the counter is started. It can only be changed during operation in PWM output function. Otherwise, changes to TAUJnTOL.TOLm result in an invalid TAUJnTTOUTm signal.

Refer to *18.8 “Simultaneous Rewrite” on page 1416* .

The channel output modes and the channel output control bits are listed in the following table (TAUJnTOC.TOCm = 0).

Table 18-12 Channel output modes

Channel output mode	TAUJnTOE.TOEm	TAUJnTOM.TOMm
By software		
Direct Channel Output Mode	0	x
By timer signals, independently (Independent Channel Output Mode)		
Independent Channel Output Mode 1	1	0
By timer signals, synchronously (Synchronous Channel Output Mode)		
Synchronous Channel Output Mode 1	1	1

- The combinations not listed in this table are forbidden.
- The bit marked with an x can be set to any value.

Note The following bits cannot be changed during count operation (TAUJnTE.TE = 1):

- TAUJnTOM.TOMm
- TAUJnTOC.TOCm

18.9.1 General procedure for specifying a channel output mode

The following steps describe the general procedure for specifying a TAUJnTTOUTm channel output mode. The prerequisite is that timer output operation is disabled (TAUJnTOE.TOEm = 0).

1. Set TAUJnTO.TOm to specify the initial level of the TAUJnTTOUTm output.
2. Set the channel output mode using *Table 18-12 “Channel output modes” on page 1422* and the output logic using the TAUJnTOL.TOLm bit.
3. Start the counter (TAUJnTS.TSm = 1).

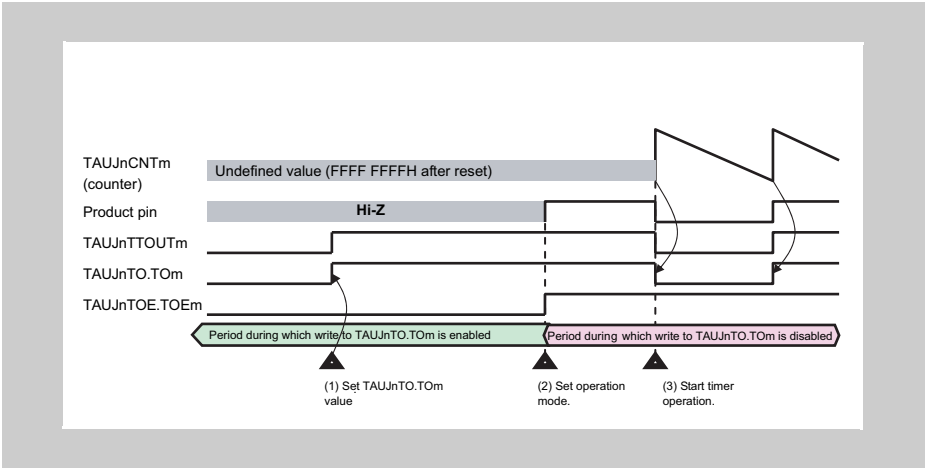


Figure 18-7 General procedure for specifying a TAUJnTTOUTm channel output mode

The following figure shows a general illustration of how the output changes when the counter is enabled:

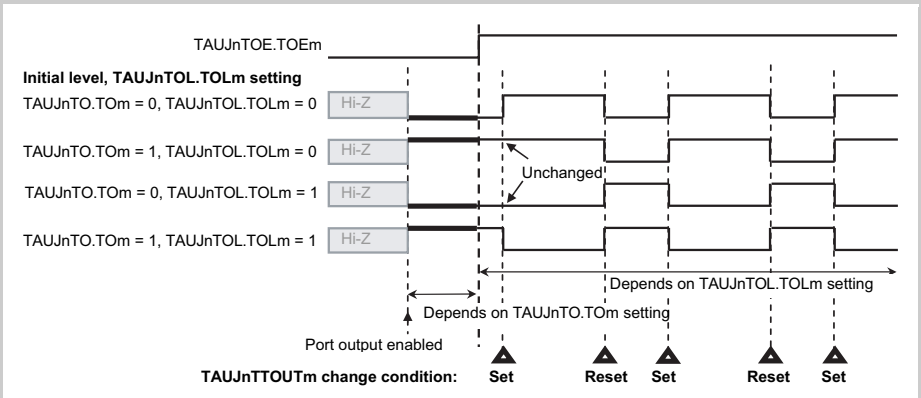


Figure 18-8 General change of the TAUJnTTOUTm output

- TAUJnTO.TOm sets the initial value of TAUJnTTOUTm and can be changed while TAUJnTOE.TOEm = 0.
- TAUJnTOL.TOLm specifies whether the set signal sets TAUJnTO.TOm to high (TAUJnTOL.TOLm = 0) or low (inverted logic, TAUJnTOL.TOLm = 1).

18.9.2 Channel output modes controlled independently by TAUJn signals

This chapter lists the channel output modes that are controlled independently by TAUJn signals. The control bits used to specify a mode are listed in *Table 18-12 “Channel output modes” on page 1422* .

(1) Independent Channel Output Mode 1

Set/reset conditions In this output mode, TAUJnTTOUTm toggles when INTTAUJnIm is detected. The value of TAUJnTOL.TOLm is ignored.

Prerequisites None, other than those in *Table 18-12 “Channel output modes” on page 1422* .

18.9.3 Channel output modes controlled synchronously by TAUJn signals

This chapter lists the channel output modes that are controlled synchronously by TAUJn signals. The control bits used to specify a mode are listed in *Table 18-12 “Channel output modes” on page 1422* .

(1) Synchronous Channel Output Mode 1

Set/reset conditions

In this output mode, INTTAUJnIm of the master channel serves as the set signal and INTTAUJnIm of the slave channel as the reset signal. If INTTAUJnIm of the master channel and INTTAUJnIm of the slave channel are generated at the same time, INTTAUJnIm of the slave channel (reset signal) has priority over INTTAUJnIm (set signal) of the master channel, i.e. the master channel is ignored.

Prerequisites

None, other than those in *Table 18-12 “Channel output modes” on page 1422* .

18.10 Start Timing of Operating Modes

This chapter describes when the counters of the different operating modes start after the TAUJnTS.TSm bit is set to 1.

In all modes, the value of the data register and whether or not an interrupt is issued depends on the individual mode and corresponding register settings.

18.10.1 Interval Timer Mode, Capture Mode

The counter starts at the start of the next count clock cycle after TAUJnTS.TSm is set to 1. The value of data register is also loaded at the point the counter starts.

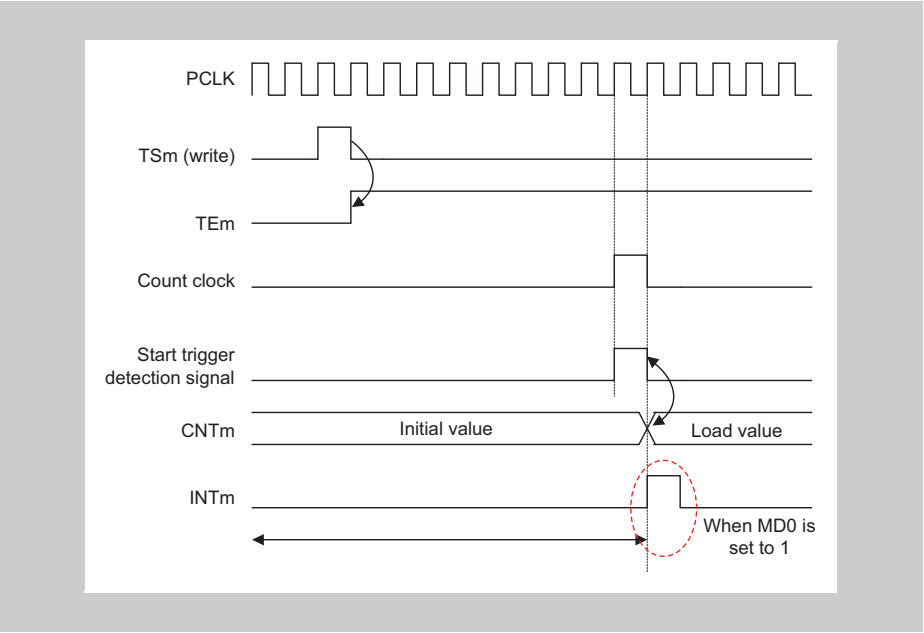
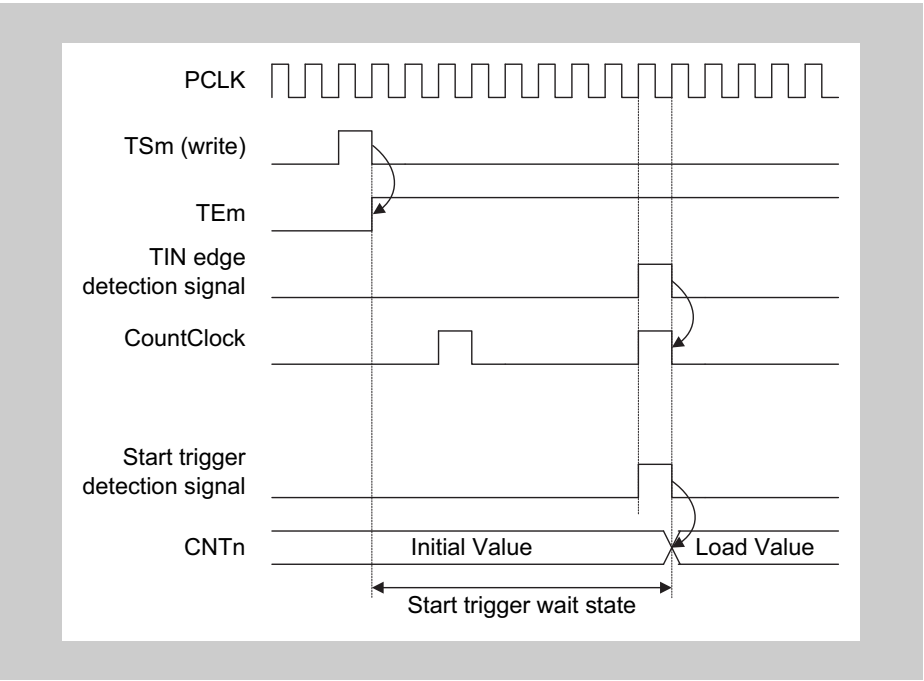


Figure 18-9 Start timing of Interval Timer Mode, Capture Mode

18.10.2 Other operating modes

In all other operating modes, the count clock cycles are ignored with regard to starting the counter. The counter is only triggered by detection of a valid TAUJnTTINm edge. The value of data register is also loaded at the point the counter starts. Nevertheless, the count clock cycles determine the frequency with which all operations take place.



Start timing of all other operating modes

18.11 TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts

When the counter starts, it is possible to specify whether an INTTAUJnIm is generated using the TAUJnCMORm.MD0 bit. The effect of the bit depends on the selected mode, as shown in the following table. The effects of INTTAUJnIm on TAUJnTTOUTm depend on the selected channel operation function.

Table 18-13 Effect of CMORm.MD0 bit on generation of INTTAUJnIm when counter is triggered

Mode	TAUJnCMORm. MD0 bit	INTTAUJnIm generated when counter starts
Interval Timer Mode Capture Mode Count Capture Mode	0	No
	1	Yes
Capture & One Count Mode Capture & Gate Count Mode	0	No
One Count Mode Gate Count Mode	0/1	No, regardless of setting of TAUJnCMORm.MD0 bit.

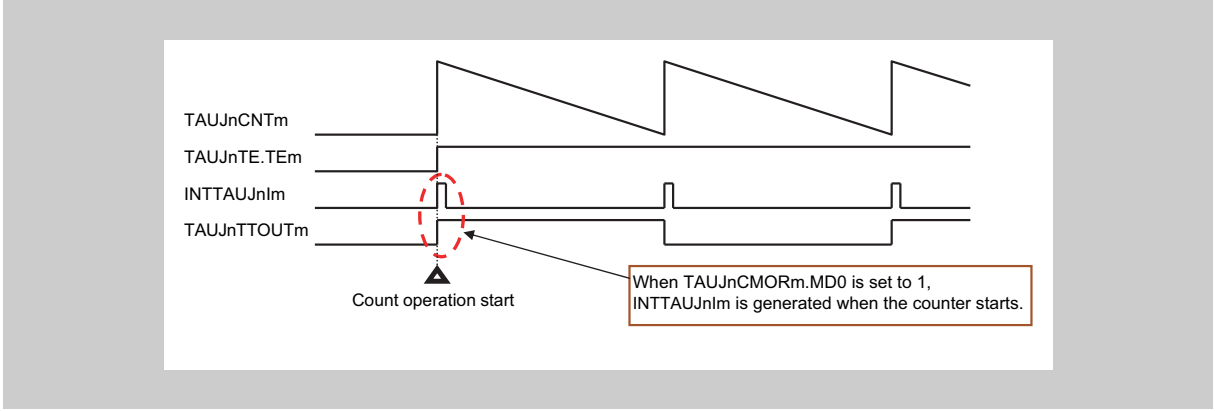


Figure 18-10 INTTAUJnIm generated when counter starts

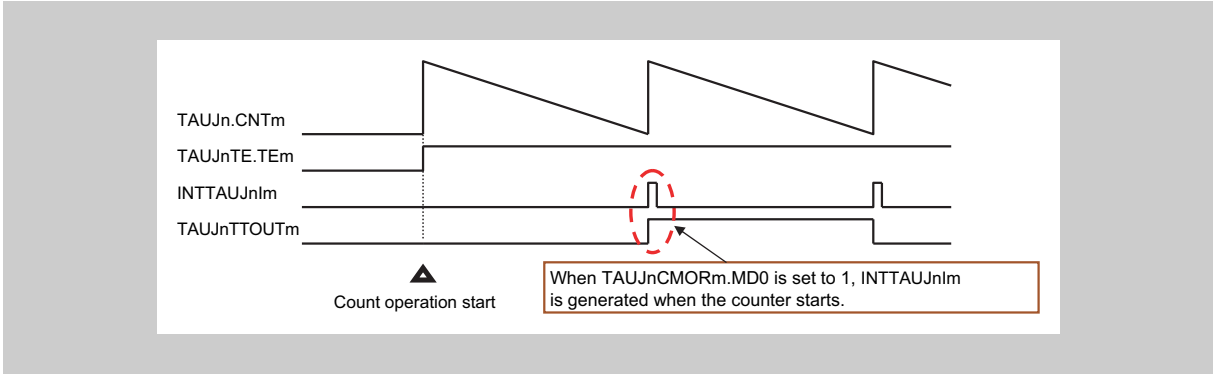


Figure 18-11 INTTAUJnIm not generated when counter starts

18.12 Interrupt Generation upon Overflow

Certain independent functions that count up, overflow without generating an interrupt when they reach FFFF $FFFF_H$. This section describes how it is possible to generate an interrupt, by combining a channel operating in one of these modes with a channel in a different operation mode which counts down.

The appropriate operation mode for the second channel depends on the operation mode of the first channel. Nevertheless, the principle is the same for all combinations:

- Find a operation mode for the second channel that counts down in such a manner, that it reaches 0000 0000_H at the same time as the first channel overflows (TAUJnCNTm = FFFF $FFFF_H$).
- Set TAUJnCDRm of the second channel to FFFF $FFFF_H$
- The two channels must count at the same speed (i.e. they must have the same count clock)
- Both channels are triggered by the same TAUJnTTINm input
- The trigger detection settings (TAUJnCMORm.STS[2:0] and TAUJnCMURm.TIS[1:0]) must be identical for both channels
- This is only possible for TAUJ0 and TAUJ4 by using configurable "input selectors" (on system level)

Result: the down-counter of the second channel reaches 0000 0000_H at exactly the same time as the up-counter of the first channel overflows (TAUJnCNTm = FFFF $FFFF_H$). Thus the second channel generates the desired interrupt.

The following sections list the operating modes that count down that are required to match specific operating modes that count up, as well as example timing diagrams.

18.12.1 Capture Mode

Applies to

- TAUJnTTINm Input Pulse Interval Measurement Function
- Real-Time Output Function Type 2
- Simultaneous Rewrite Trigger Generation Function Type 2

Combine with Interval Timer Mode

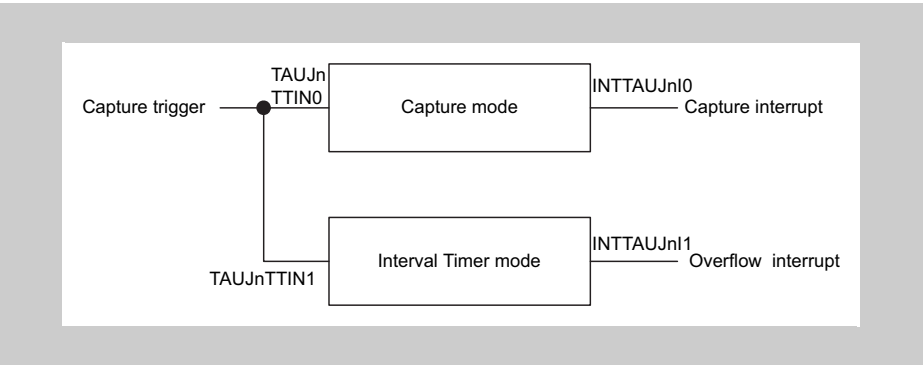


Figure 18-12 Combination of Capture Mode and Interval Timer Mode

Timing diagram

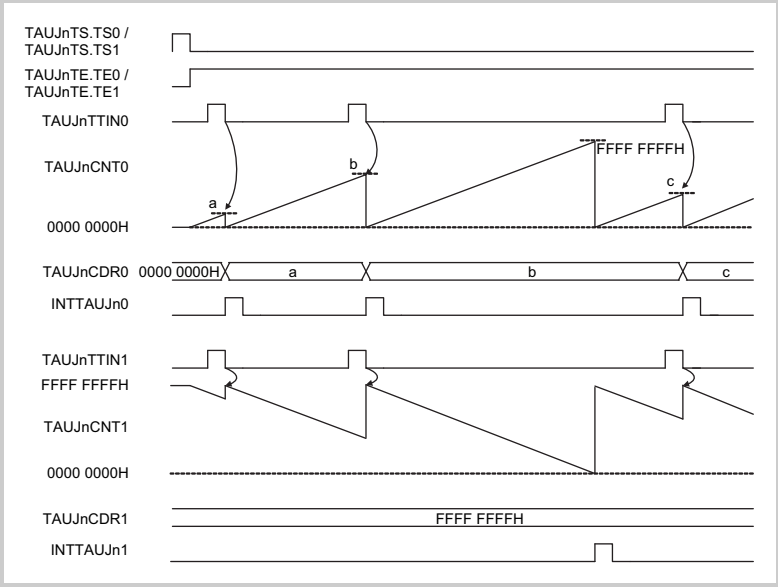


Figure 18-13 Interrupt generation via combination of Capture Mode and Interval Timer Mode

18.12.2 Capture and One Count Mode

Applies to
- TAUJnTTINm Input Signal Width Measurement Function

Combine with
One Count Mode

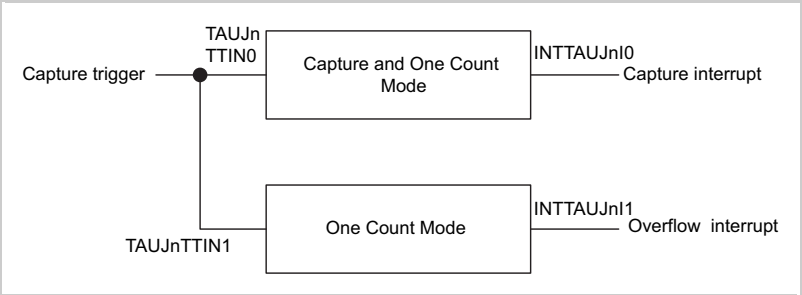


Figure 18-14 Combination of Capture and One Count Mode and One Count Mode

Timing diagram

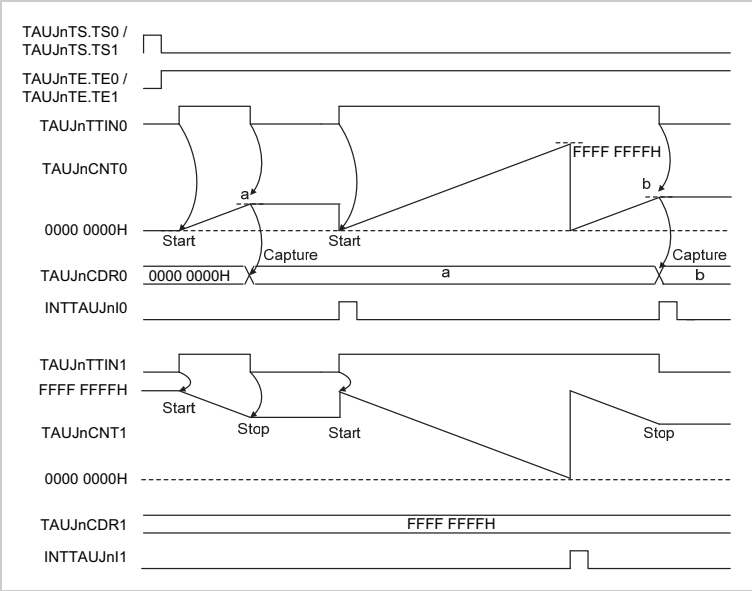


Figure 18-15 Interrupt generation via combination of Capture and One Count Mode and One Count Mode

18.12.3 Count Capture Mode

Applies to • TAUJnTTINm Input Position Detection Function

Combine with Interval Timer Mode

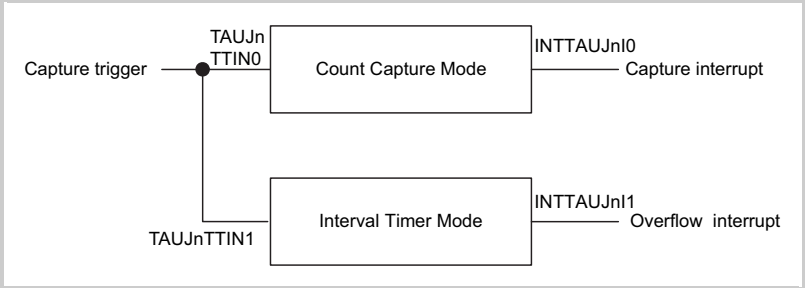


Figure 18-16 Combination of Count Capture Mode and Interval Timer Mode

Timing diagram

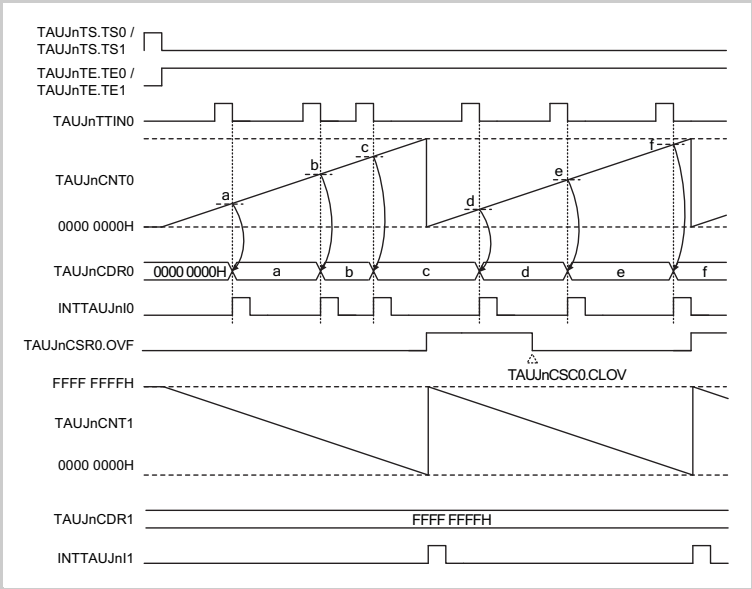


Figure 18-17 Interrupt generation via combination of Count Capture Mode and Interval Timer Mode

In the above timing diagram, TAUJnCSRm.OVF is set to 1 when TAUJnCNTm overflows. It is reset by a software command (TAUJnCSCm.CLOV = 1).

18.12.4 Capture and Gate Count Mode

Applies to • TAUJnTTINm Input Period Count Detection Function

Combine with Gate Count Mode

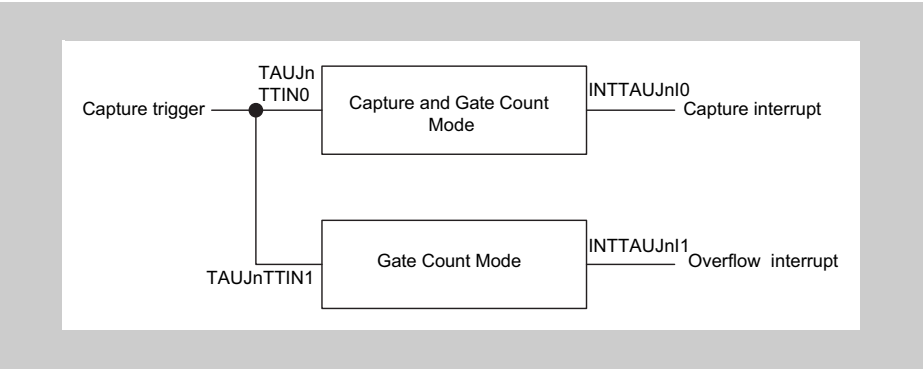


Figure 18-18 Combination of Capture and Gate Count Mode and Gate Count Mode

Timing diagram

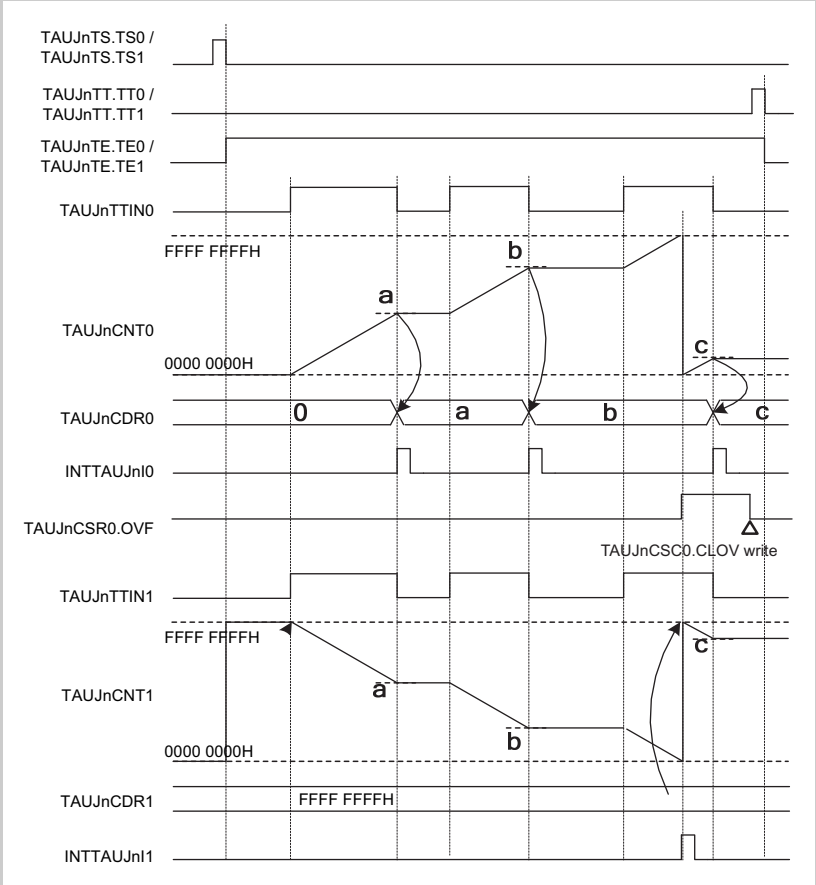


Figure 18-19 Interrupt generation via combination of Capture and Gate Count Mode and Gate Count Mode

In the above timing diagram, TAUJnCSRm.OVF is set to 1 when TAUJnCNTm overflows. It is reset by a software command (TAUJnCSCm.CLOV = 1).

18.13 TAUJnTTINm Edge Detection

Edge detection is based on the operation clock. This means that an edge can only be detected at the next rising edge of the operation clock. This can lead to a maximum delay of one operation clock cycle.

The following figure shows when edge detection takes place.

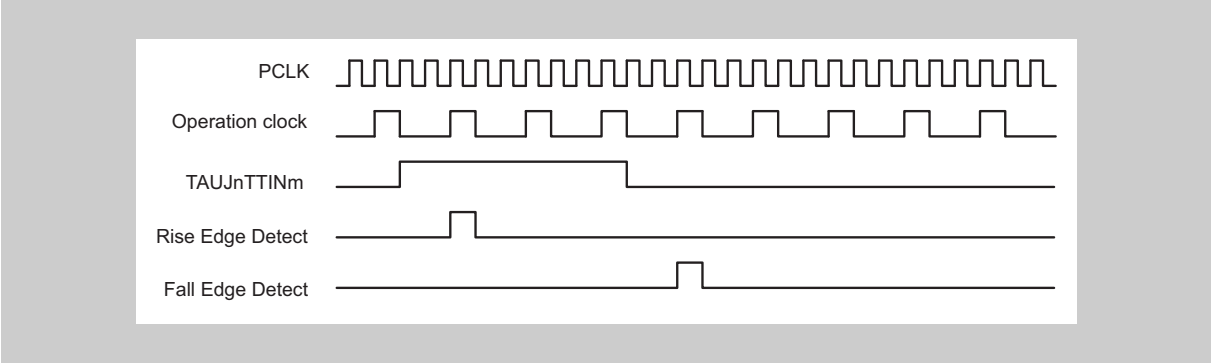


Figure 18-20 Basic edge detection timing

18.14 Independent Channel Operation Functions

The following sections list the independent channel operation functions provided by the Timer Array Unit J. For a general overview of independent channel operation, see *18.4 “Functional Description” on page 1409* .

18.15 Independent Channel Interrupt Functions

This chapter describes functions that generate interrupts at regular intervals.

- *18.15.1 “Interval Timer Function”*
- *18.15.2 “TAUJnTTINm Input Interval Timer Function”*

18.15.1 Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUJnIm) at regular intervals. When an interrupt is generated, the TAUJnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 18-14 "TAUJnCMORm settings for Interval Timer Function" on page 1438*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *18.9 "Channel Output Modes" on page 1421*

Description The counter is started by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. The current value of TAUJnCDRm is written to TAUJnCNTm and the counter starts to count down from this value.

When the counter reaches 0000 0000_H, INTTAUJnIm is generated and the TAUJnTTOUTm signal toggles. TAUJnCNTm then reloads the TAUJnCDRm value and subsequently continues operation.

The value of TAUJnCDRm can be rewritten at any time, and the changed value of TAUJnCDRm is applied the next time the counter starts to count down.

The counter can be stopped by setting TAUJnTT.TTm to 1, which in turn sets TAUJnTE.TEm to 0. TAUJnCNTm and TAUJnTTOUTm stop but retain their values. The counter can be reset by setting TAUJnTS.TSm to 1. The counter can also be forcibly restarted (without stopping it first) by setting TAUJnTS.TSm to 1 during operation.

Conditions If the TAUJnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated, and therefore TAUJnTTOUTm does not toggle. This results in an inverted TAUJnTTOUTm signal compared to when TAUJnCMORm.MD0 is set to 1. For details refer to *18.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts" on page 1428* .

(2) Equations

INTTAUJnIm cycle = count clock cycle x (TAUJnCDRm + 1)

TAUJnTTOUTm square wave cycle = count clock cycle x (TAUJnCDRm + 1) x 2

(3) Block diagram and general timing diagram

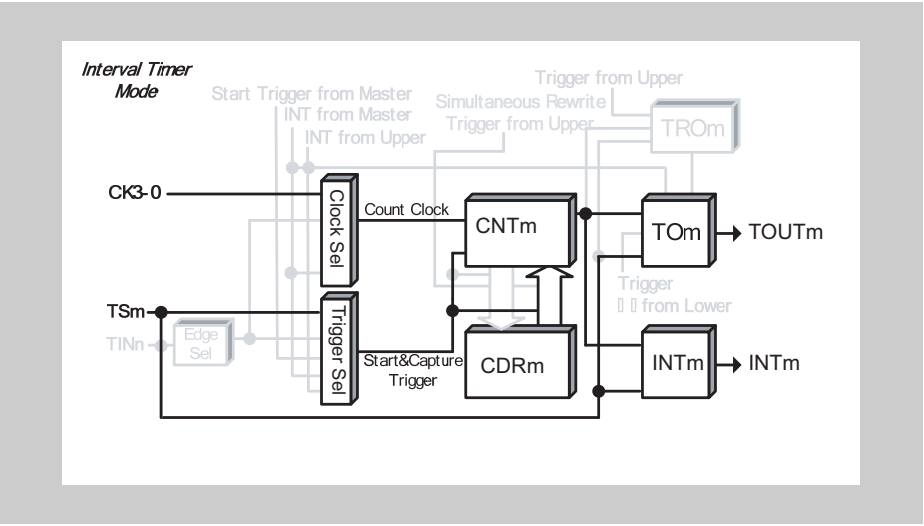


Figure 18-21 Block diagram for Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.MD0 = 0)

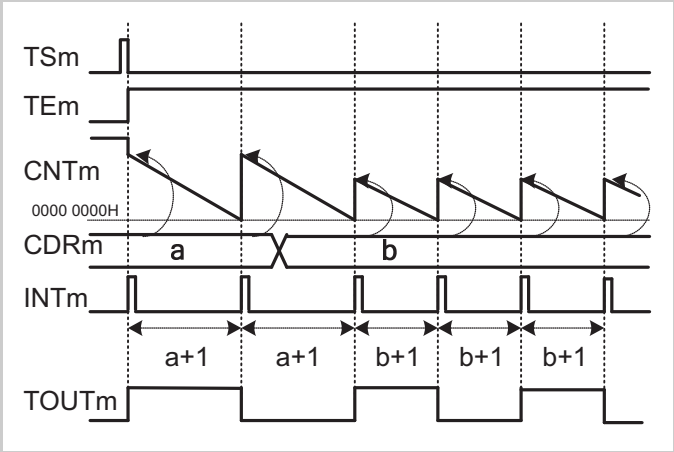


Figure 18-22 General timing diagram for Interval Timer Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-14 TAUJnCMORm settings for Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUJnIm not generated and TAUJnTTOUTm does not toggle at operation start or restart 1: Generates INTTAUJnIm and toggles TAUJnTTOUTm at operation start or restart

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-15 TAUJnCMURm settings for Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Not used, so set to 00

(c) Channel output mode

Table 18-16 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUJnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUJnTOE.TOEm = 0. TAUJnTTOUTm can then be controlled independently of the interrupts. For details refer to *18.9 “Channel Output Modes” on page 1421* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the Interval Timer Function. Therefore, these registers must be set to 0.

Table 18-17 Simultaneous rewrite settings for Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for Interval Timer Function

Table 18-18 Operating procedure for Interval Timer Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-14 "TAUJnCMORm settings for Interval Timer Function" on page 1438* and *Table 18-15 "TAUJnCMURm settings for Interval Timer Function" on page 1438* Set the value of the TAUJnCDRm register Set the channel output mode by setting the control bits as described in *Table 18-16 "Control bit settings for Independent Channel Output Mode 1" on page 1439*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and the counter starts. TAUJnCNTm loads the TAUJnCDRm value. When TAUJnCMORm.MD0 = 1, INTTAUJnIm is generated and TAUJnTTOUTm toggles.
During operation	The TAUJnCDRm register value can be changed at any time. The TAUJnCNTm register can be read at all times.	TAUJnCNTm counts down. When the counter reaches 0000 0000_H: • TAUJnCNTm reloads the TAUJnCDRm value and continues count operation • INTTAUJnIm is generated and TAUJnTTOUTm toggles.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

(a) TAUJnCDRm = 0000 0000$_H$, count clock = PCLK/2

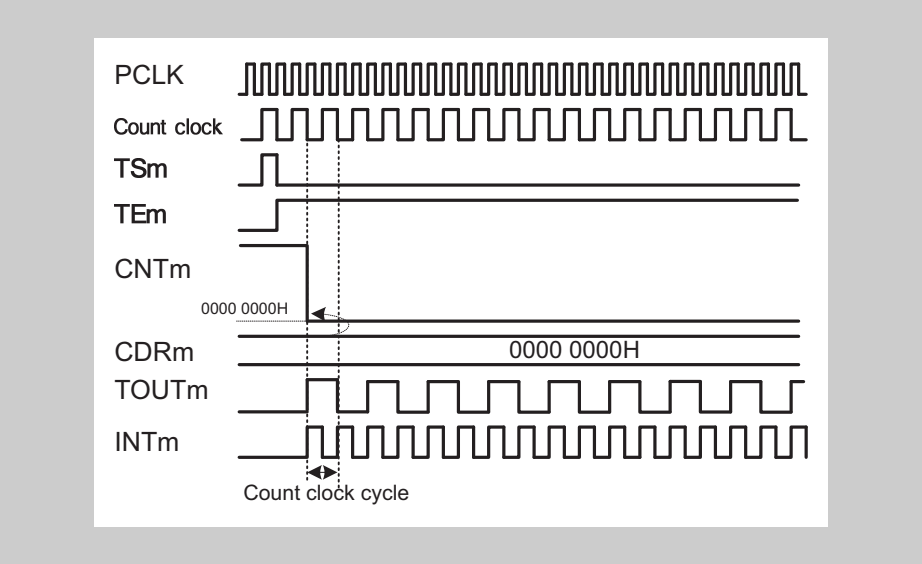


Figure 18-23 TAUJnCDRm = 0000 0000$_H$, count clock = PCLK/2

- If TAUJnCDRm = 0000 0000$_H$ and the count clock = PCLK/2[1], the TAUJnCDRm value is written to TAUJnCNTm every count clock, meaning that TAUJnCNTm is always 0000 0000$_H$.
- INTTAUJnIm is generated every count clock, resulting in TAUJnTTOUTm toggling every count clock.

(b) TAUJnCDRm = 0000 0000$_H$, count clock = PCLK

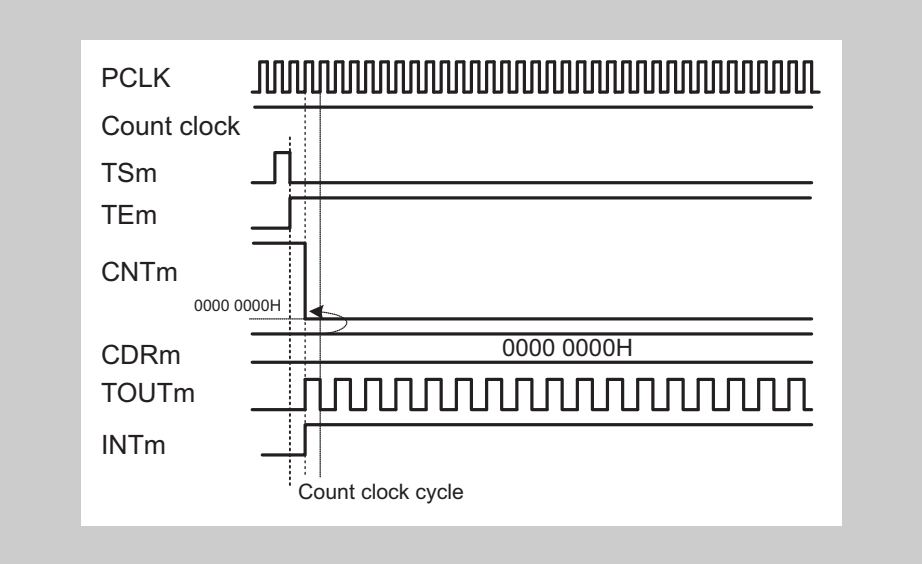


Figure 18-24 TAUJnCDRm = 0000 0000$_H$, count clock = PCLK

- If TAUJnCDRm = 0000 0000$_H$ and the count clock = PCLK, the TAUJnCDRm value is written to TAUJnCNTm every PCLK clock, meaning that TAUJnCNTm is always 0000 0000$_H$.
- INTTAUJnIm is generated continuously, resulting in TAUJnTTOUTm toggling every PCLK clock.

(c) Operation stop and restart

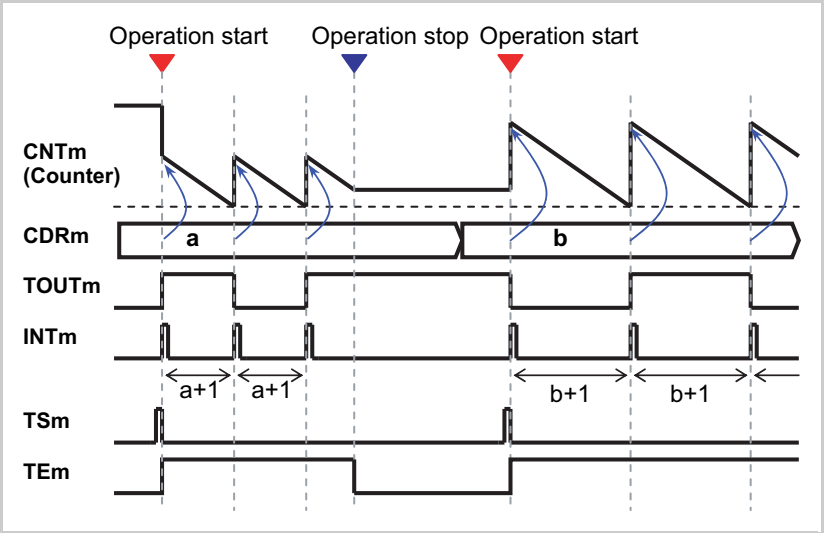


Figure 18-25 Operation stop and restart, TAUJnCMORm.MD0 = 1

- The counter can be stopped by setting TAUJnTT.TTm to 1, which in turn sets TAUJnTE.TEm to 0.
- TAUJnCNTm and TAUJnTTOUTm stop but retain their values.
- The counter can be restarted by setting TAUJnTS.TSm to 1.

(d) Forced restart

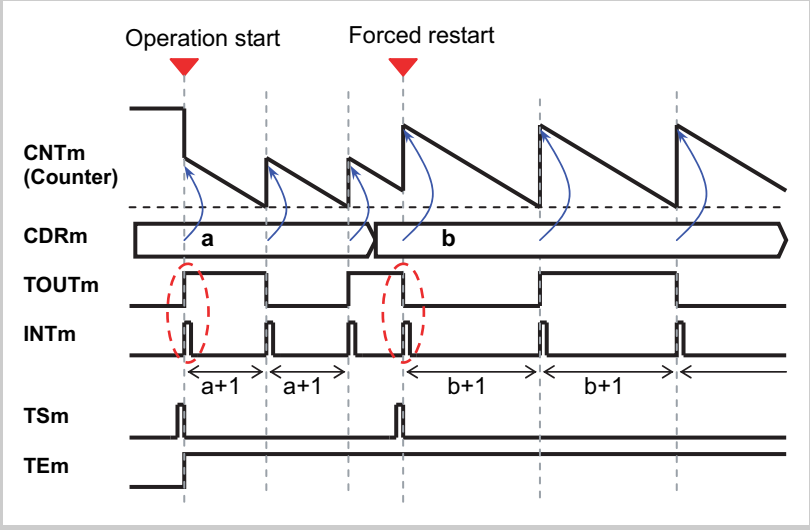


Figure 18-26 Forced restart operation, TAUJnCMORm.MD0 = 1

- The counter can be forcibly restarted (without stopping it first) by setting TAUJnTS.TSm to 1 during operation.
- If the TAUJnCMORm.MD0 bit is set to 1, the first interrupt after a start or restart is generated.

18.15.2 TAUJnTTINm Input Interval Timer Function

(1) Overview

Summary This function is used as a reference timer for generating timer interrupts (INTTAUJnIm) at regular intervals or when a valid TAUJnTTINm input edge is detected. When an interrupt is generated, the TAUJnTTOUTm signal toggles, resulting in a square wave.

Prerequisites

- The operation mode must be set to Interval Timer Mode, refer to *Table 18-19 "TAUJnCMORm settings for TAUJnTTINm Input Interval Timer Function" on page 1445*
- The channel output mode must be set to Independent Channel Output Mode 1, refer to *18.9 "Channel Output Modes" on page 1421*

Description This function operates in an identical manner to the Interval Timer Function (see *18.15.1 "Interval Timer Function" on page 1436*), except that this function is restarted by a valid TAUJnTTINm input edge. The type of edge used as the trigger is specified using the TAUJnCMURm.TIS[1:0] bits. Either rising edge, falling edge, or rising and falling edge can be selected.

(2) Equations

INTTAUJnIm cycle = count clock cycle x (TAUJnCDRm + 1)

TAUJnTTOUTm square wave cycle = count clock cycle x (TAUJnCDRm + 1) x 2

(3) Block diagram and general timing diagram

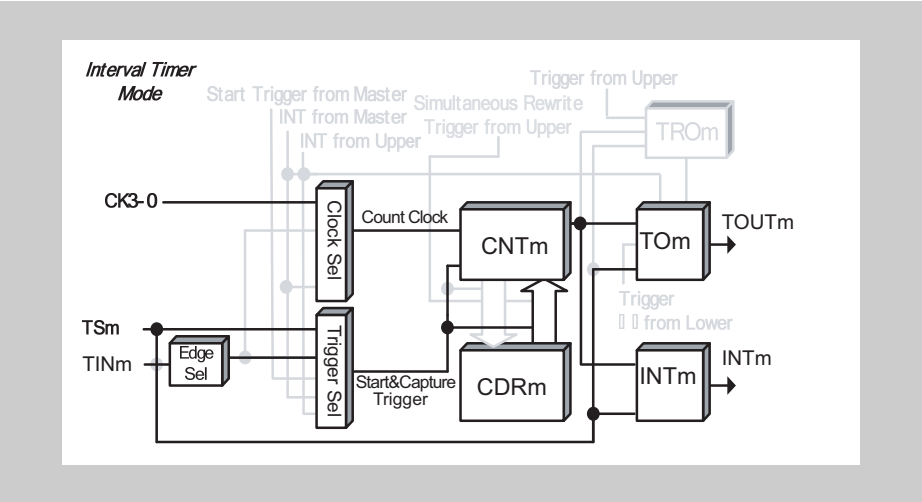


Figure 18-27 Block diagram for TAUJnTTINm Input Interval Timer Function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.MD0 = 0)
- Rising edge detection (TAUJnCMURm.TIS[1:0] = 01_B)

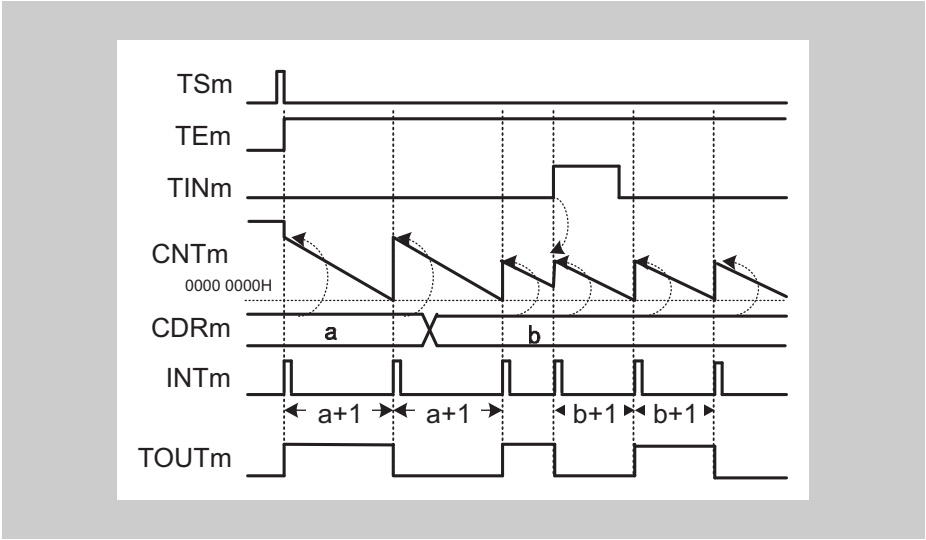


Figure 18-28 General timing diagram for TAUJnTTINm Input Interval Timer Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-19 TAUJnCMORm settings for TAUJnTTINm Input Interval Timer Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUJnTTINm input edge signal is used as the external start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	0: INTTAUJnIm not generated and TAUJnTTOUTm does not toggle at operation start 1: Generates INTTAUJnIm and toggles TAUJnTTOUTm at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-20 TAUJnCMURm settings for TAUJnTTINm Input Interval Timer Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

Table 18-21 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOE.TOEm	1: Disables Direct Channel Output Mode
TOM.TOMm	0: Independent channel output
TOC.TOCm	0: Operation mode 1 (= Toggle mode if TAUJnTOM.TOMm = 0)
TOL.TOLm	0: Positive logic

Note The channel output mode can also be set to Direct Channel Output Mode by setting TAUJnTOE.TOEm = 0. TAUJnTTOUTm can then be controlled independently of the interrupts. For details refer to *18.9 “Channel Output Modes” on page 1421* .

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm Input Interval Timer Function. Therefore, these registers must be set to 0.

Table 18-22 Simultaneous rewrite settings for TAUJnTTINm Input Interval Timer Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm Input Interval Timer Function

Table 18-23 Operating procedure for TAUJnTTINm Input Interval Timer Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-19 "TAUJnCMORm settings for TAUJnTTINm Input Interval Timer Function" on page 1445* and *Table 18-20 "TAUJnCMURm settings for TAUJnTTINm Input Interval Timer Function" on page 1445* Set the value of the TAUJnCDRm register Set the channel output mode by setting the control bits as described in *Table 18-21 "Control bit settings for Independent Channel Output Mode 1" on page 1446*	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and the counter starts. TAUJnCNTm loads the TAUJnCDRm value. When TAUJnCMORm.MD0 = 1, INTTAUJnIm is generated and TAUJnTTOUTm toggles.
During operation	The values of the TAUJnCMURm.TIS[1:0] and TAUJnTO.TOm bits and the TAUJnCDRm register can be changed at any time. The TAUJnCNTm register can be read at all times. Detection of TAUJnTTINm edge	TAUJnCNTm counts down. When the counter reaches 0000 0000_H: • TAUJnCNTm reloads the TAUJnCDRm value and continues count operation • INTTAUJnIm is generated and TAUJnTTOUTm toggles When a TAUJnTTINm input valid edge is detected during count operation, TAUJnCNTm reloads the TAUJnCDRm value and continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values.

Restart

(6) Specific timing diagrams

The timing diagrams in *18.15.1 "Interval Timer Function" on page 1436* also apply, except for this function the counter can also be restarted by a valid TAUJnTTINm input edge.

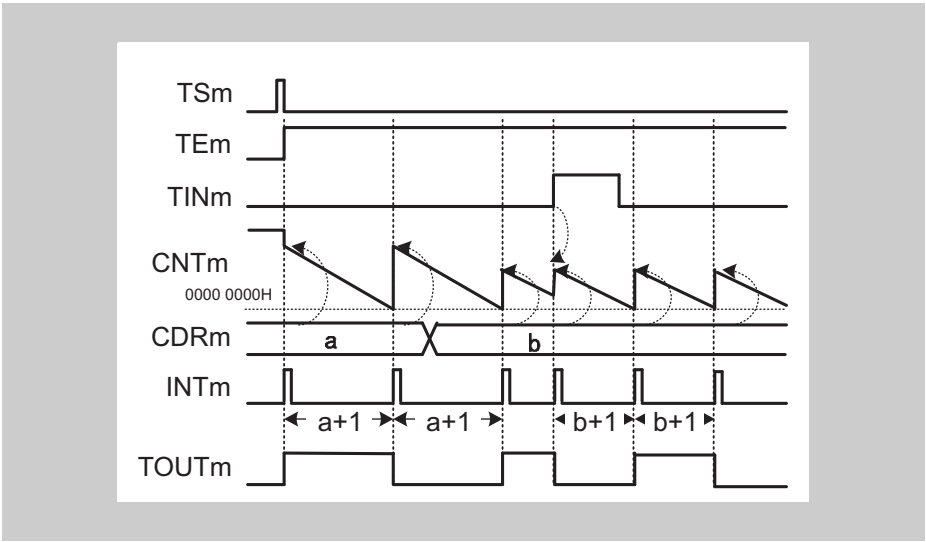


Figure 18-29 Counter triggered by rising TAUJnTTINm input edge (TAUJnCMURm.TIS[1:0] = 01_B), TAUJnCMORm.MD0 = 1

- If a valid TAUJnTTINm input edge is detected, an interrupt is generated which causes TAUJnTTOUTm to toggle. In this example, the valid edge is a rising edge (TAUJnCMURm.TIS[1:0] = 01_B).

18.16 Independent Channel Signal Measurement Fuctions

This chapter describes functions that measure the widths of an individual TAUJnTTINm pulse or the total width of successive TAUJnTTINm pulses. It also describes functions that measure the interval of the signal or that compare the width of a pulse with a reference value.

- *18.16.1 “TAUJnTTINm Input Pulse Interval Measurement Function”*
- *18.16.2 “TAUJnTTINm Input Signal Width Measurement Function”*
- *18.16.3 “Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)”*
- *18.16.4 “TAUJnTTINm Input Period Count Detection Function”*
- *18.16.5 “Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)”*

18.16.1 TAUJnTTINm Input Pulse Interval Measurement Function

(1) Overview

Summary This function captures the count value and uses this value and the overflow bit TAUJnCSRm.OVF to measure the interval of the TAUJnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture Mode, refer to *Table 18-25 "TAUJnCMORm settings for TAUJnTTINm Input Pulse Interval Measurement Function" on page 1452*
- TAUJnTTOUTm is not used for this function

Description The counter is started by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. The counter TAUJnCNTm starts counting up from 0000 0000_H. When a valid TAUJnTTINm edge is detected, the value of TAUJnCNTm is captured, transferred to TAUJnCDRm, and an interrupt INTTAUJnIm is generated. The counter resets to 0000 0000_H and subsequently continues operation.

If the counter reaches FFFF $FFFF_H$ before a valid TAUJnTTINm edge is detected, it overflows to 0000 0000_H. The counter is reset to 0000 0000_H and subsequently continues operation. The values transferred to TAUJnCDRm and TAUJnCSRm.OVF respectively depend on the values of bits TAUJnCMORm.COS[1:0]:

Table 18-24 Effects of an overflow

TAUJnCMORm.COS[1:0]	When overflow occurs		When a valid TAUJnTTINm input is then detected	
	TAUJnCDRm	TAUJnCSRm.OVF	TAUJnCDRm and TAUJnCNTm	TAUJnCSRm.OVF
00	Unchanged	0	TAUJnCNTm written to TAUJnCDRm	1
01		1		
10	Set to FFFF $FFFF_H$	0	TAUJnCNTm set to 0, TAUJnCDRm unchanged	0
11		1		

If an overflow is set (TAUJnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUJnCSCm.CLOV = 1.

The combination of the value of TAUJnCDRm and TAUJnCSRm.OVF can be used to deduce the interval of the TAUJnTTINm signal. However, if an overflow occurs multiple times before a valid TAUJnTTINm input is detected, the overflow bit TAUJnCSRm.OVF cannot indicate this.

The function can be stopped by setting TAUJnTT.TTm = 1, which in turn sets TAUJnTE.TEm = 0. TAUJnCNTm stops but retains its value. While the function is stopped, TAUJnTTINm input valid edge detection and TAUJnCNTm capture are not performed.

The function can be restarted by setting TAUJnTS.TSm = 1. The counter is reset to 0000 0000_H and subsequently continues operation. The counter can also be forcibly restarted (without stopping it first) by setting TAUJnTS.TSm = 1 during operation.

Conditions If the TAUJnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *18.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts" on page 1428* .

Note When TAUJnCMORm.COS[1:0] = 11_B, the value of TAUJnCNTm is *not* written to TAUJnCDRm when the first valid TAUJnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUJnTTINm input pulse interval = count clock cycle x
[(TAUJnCSRm.OVF x (FFFF FFFF$_H$ + 1)) + TAUJnCDRm capture value + 1]

(3) Block diagram and general timing diagram

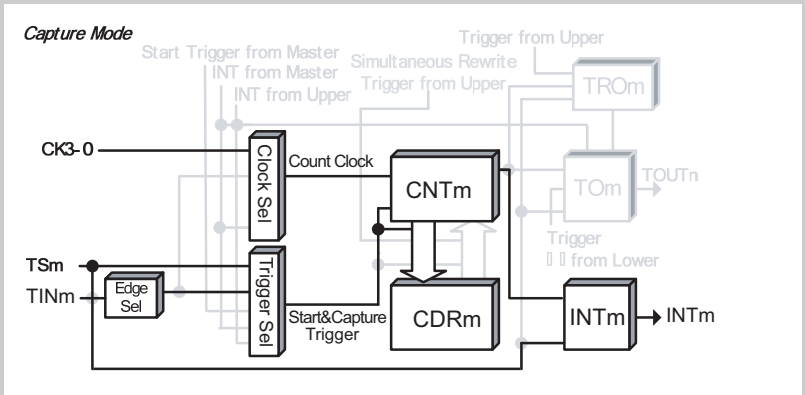


Figure 18-30 Block diagram for TAUJnTTINm Input Pulse Interval Measurement Function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.MD0 = 0)
- Falling edge detection (TAUJnCMURm.TIS[1:0] = 00_B)
- When a valid TAUJnTTINm input is detected after an overflow TAUJnCDRm is changed and TAUJnCSRm.OVF is set to 1 (TAUJnCMORm.COS[1:0] = 00_B)

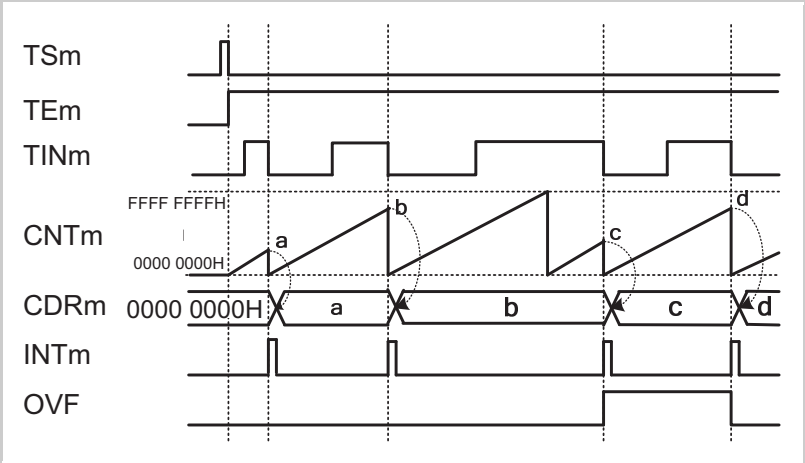


Figure 18-31 General timing diagram for TAUJnTTINm Input Pulse Interval Measurement Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-25 TAUJnCMORm settings for TAUJnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid edge of the TAUJnTTINm input signal is the external capture trigger
COS[1:0]	See *Table 18-24 "Effects of an overflow" on page 1450*
MD[4:1]	0010: Capture Mode
MD0	0: INTTAUJnIm not generated at operation start 1: Generates INTTAUJnIm at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-26 TAUJnCMURm settings for TAUJnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm Input Pulse Interval Measurement Function. Therefore, these registers must be set to 0.

Table 18-27 Simultaneous rewrite settings for TAUJnTTINm Input Pulse Interval Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm Input Pulse Interval Measurement Function

Table 18-28 Operating procedure for TAUJnTTINm Input Pulse Interval Measurement Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-25 "TAUJnCMORm settings for TAUJnTTINm Input Pulse Interval Measurement Function" on page 1452* and *Table 18-26 "TAUJnCMURm settings for TAUJnTTINm Input Pulse Interval Measurement Function" on page 1452* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and the counter starts. TAUJnCNTm is cleared to 0000 0000_H. INTTAUJnIm is generated when TAUJnCMORm.MD0 is set to 1.
During operation	Detection of TAUJnTTINm edges. The TAUJnCMURm.TIS[1:0] bits can be changed at any time. The TAUJnCDRm and TAUJnCSRm registers can be read at any time.	TAUJnCNTm starts to count up from 0000 0000_H. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm, and returns to 0000 0000_H • INTTAUJnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUJnCMORm.COS[1:0] = 00_B

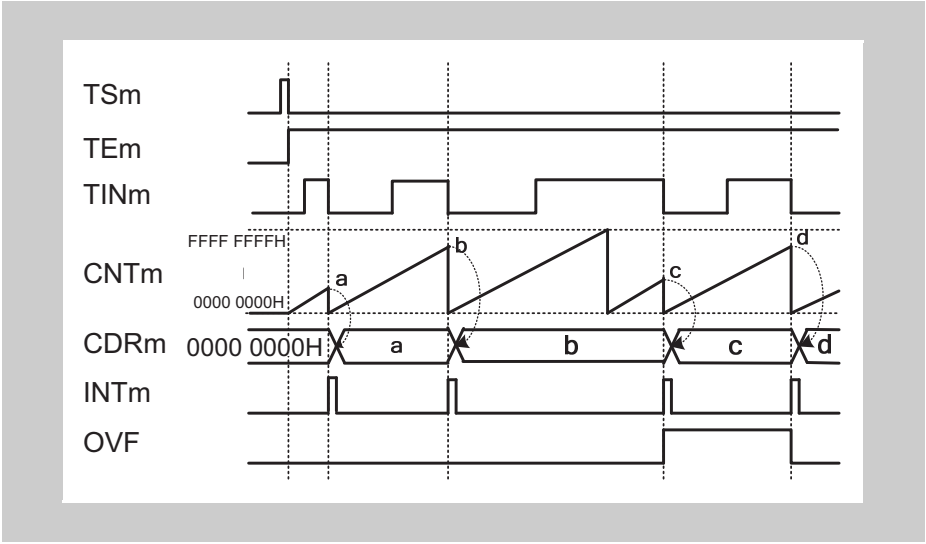


Figure 18-32 TAUJnCMORm.COS[1:0] = 00_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is written to TAUJnCDRm and TAUJnCSRm.OVF is set to 1.

(b) TAUJnCMORm.COS[1:0] = 01_B

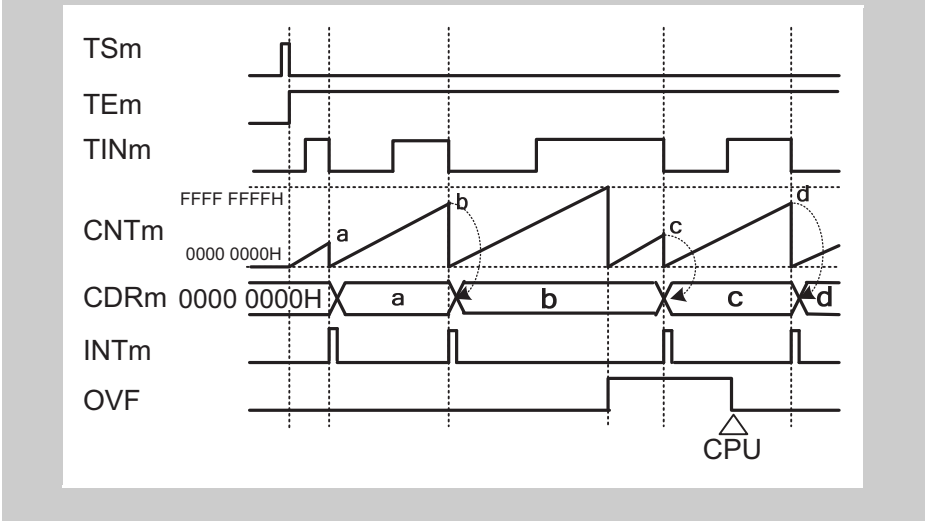


Figure 18-33 TAUJnCMORm.COS[1:0] = 01_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=00_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is written to TAUJnCDRm.
- TAUJnCSRm.OVF is only cleared by a CPU command.

(c) TAUJnCMORm.COS[1:0] = 10_B

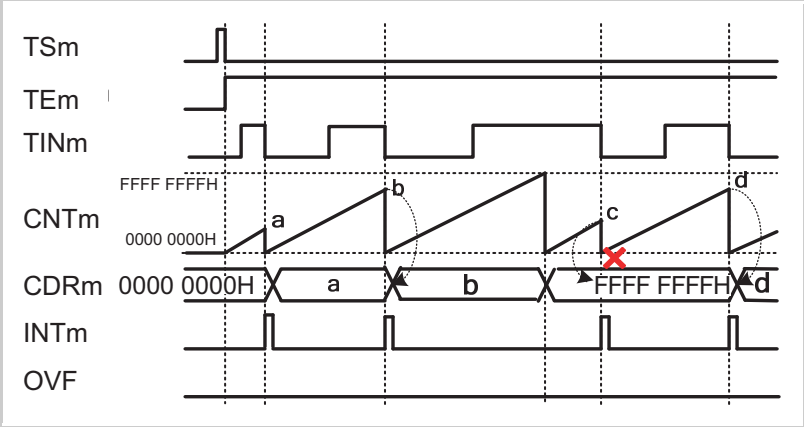


Figure 18-34 TAUJnCMORm.COS[1:0] = 10_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUJnCDRm is set to FFFF $FFFF_H$ and TAUJnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.OVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.

(d) TAUJnCMORm.COS[1:0] = 11_B

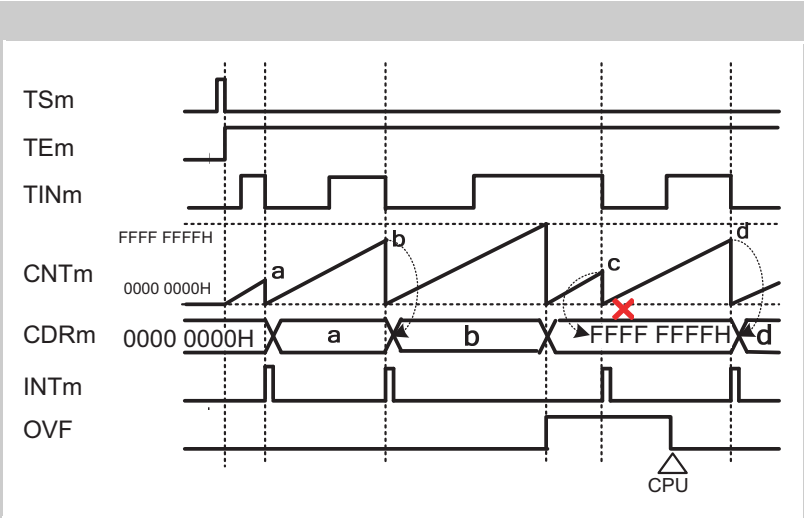


Figure 18-35 TAUJnCMORm.COS[1:0] = 11_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=00_B

- When an overflow occurs, TAUJnCDRm is set to FFFF $FFFF_H$, and TAUJnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.OVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.
- TAUJnCSRm.OVF is cleared by a CPU command.

18.16.2 TAUJnTTINm Input Signal Width Measurement Function

(1) Overview

Summary This function measures the width of a TAUJnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & One Count Mode, refer to *Table 18-30 "TAUJnCMORm settings for TAUJnTTINm Input Signal Width Measurement Function" on page 1459*
- TAUJnTTOUTm is not used for this function
- TAUJnCMORm.MD0 must be set to 0

Description The counter is started by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. When a valid TAUJnTTINm start edge is detected, the counter TAUJnCNTm starts counting up from 0000 0000_H. When a valid TAUJnTTINm stop edge is detected, the value of TAUJnCNTm is captured, transferred to TAUJnCDRm, and an interrupt INTTAUJnIm is generated. The counter retains its value and awaits the next valid TAUJnTTINm input start edge.

If the counter reaches FFFF $FFFF_H$ before a valid TAUJnTTINm stop edge is detected, it overflows. The counter is reset to 0000 0000_H and subsequently continues operation. The values transferred to TAUJnCDRm and TAUJnCSRm.OVF respectively depend on the values of bits TAUJnCMORm.COS[1:0]:

Table 18-29 Effects of an overflow

TAUJnCMORm.COS[1:0]	When overflow occurs		When a valid TAUJnTTINm input stop edge is detected	
	TAUJnCDRm	TAUJnCSRm.OVF	TAUJnCDRm and TAUJnCNTm	TAUJnCSRm.OVF
00	Unchanged	0	TAUJnCNTm written to TAUJnCDRm	1
01		1		
10	Set to FFFF $FFFF_H$	0	TAUJnCNTm stops counting TAUJnCDRm unchanged	0
11		1		

If an overflow is set (TAUJnCSRm.OVF = 1), it can only be cleared by a CPU command that sets TAUJnCSCm.CLOV = 1.

The combination of the value of TAUJnCDRm and TAUJnCSRm.OVF can be used to deduce the width of the TAUJnTTINm signal. However, if an overflow occurs multiple times before a valid TAUJnTTINm input is detected, the overflow bit TAUJnCSRm.OVF cannot indicate this.

This function cannot be forcibly restarted.

Note When TAUJnCMORm.COS[1:0] = 11_B, the value of TAUJnCNTm is *not* written to TAUJnCDRm when the first valid TAUJnTTINm input edge occurs after an overflow. However, an interrupt is generated.

(2) Equations

TAUJnTTINm input signal width = count clock cycle x [(TAUJnCSRm.OVF x (FFFF $FFFF_H$ + 1)) + TAUJnCDRm capture value + 1]

(3) Block diagram and general timing diagram

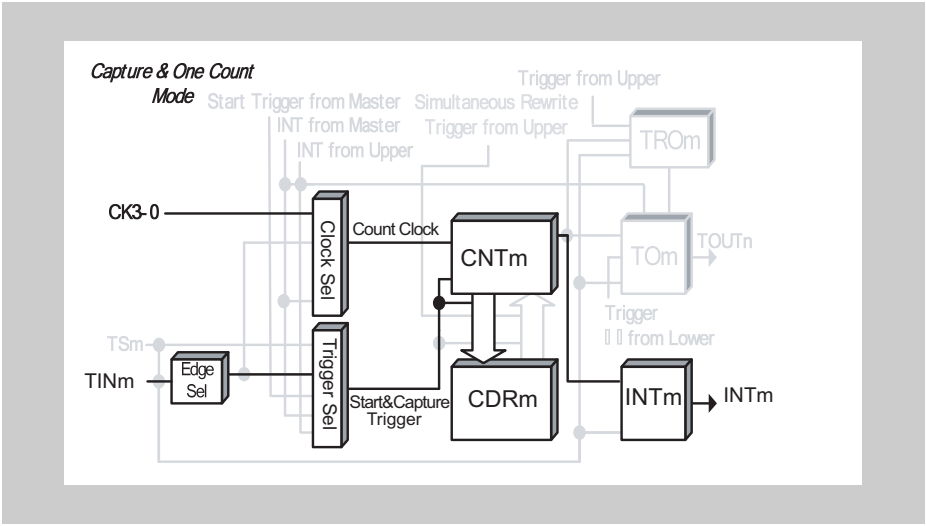


Figure 18-36 Block diagram for TAUJnTTINm Input Signal Width Measurement Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TIS[1:0] = 11_B)
- When a valid TAUJnTTINm input is detected after an overflow TAUJnCDRm is changed and TAUJnCSRm.OVF is set to 1 (TAUJnCMORm.COS[1:0] = 00_B)

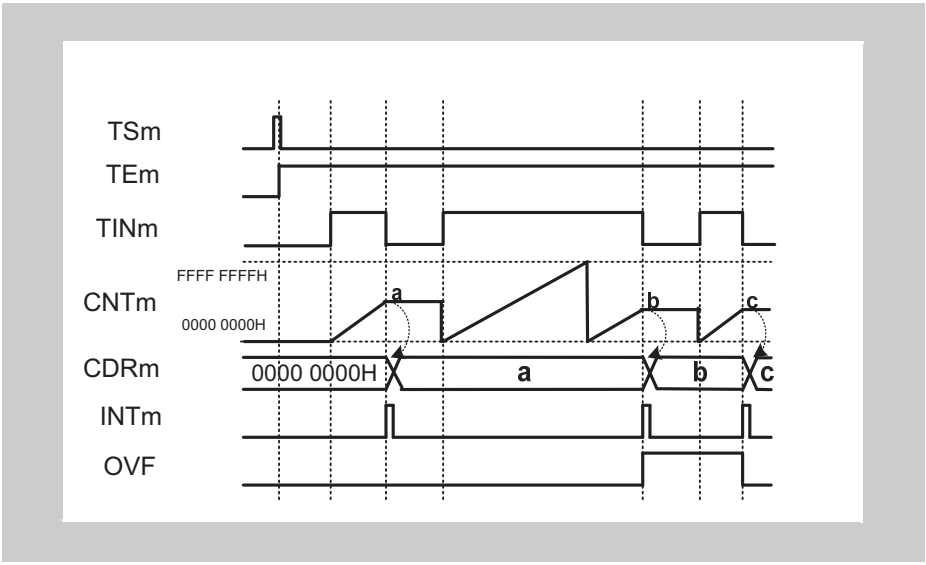


Figure 18-37 General timing diagram for TAUJnTTINm Input Signal Width Measurement Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-30 TAUJnCMORm settings for TAUJnTTINm Input Signal Width Measurement Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	See *Table 18-29 "Effects of an overflow" on page 1457*
MD[4:1]	0110: Capture & One Count Mode
MD0	0: INTTAUJnIm not generated at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-31 TAUJnCMURm settings for TAUJnTTINm Input Signal Width Measurement Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (low width measurement) 11: Rising and falling edge detection (high width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm Input Signal Width Measurement Function. Therefore, these registers must be set to 0.

Table 18-32 Simultaneous rewrite settings for TAUJnTTINm Input Signal Width Measurement Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm Input Signal Width Measurement Function

Table 18-33 Operating procedure for TAUJnTTINm Input Signal Width Measurement Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-25 "TAUJnCMORm settings for TAUJnTTINm Input Pulse Interval Measurement Function" on page 1452* and *Table 18-26 "TAUJnCMURm settings for TAUJnTTINm Input Pulse Interval Measurement Function" on page 1452* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. When a TAUJnTTINm start is detected, TAUJnCNTm starts to count up.
During operation	Detection of TAUJnTTINm edges. The TAUJnCMURm.TIS[1:0] bits can be changed at any time. The TAUJnCDRm and TAUJnCSRm registers can be read at any time.	TAUJnCNTm starts to count up from 0000 0000_H. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm, and retains its value • INTTAUJnIm is then generated. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams: overflow behavior

(a) TAUJnCMORm.COS[1:0] = 00_B

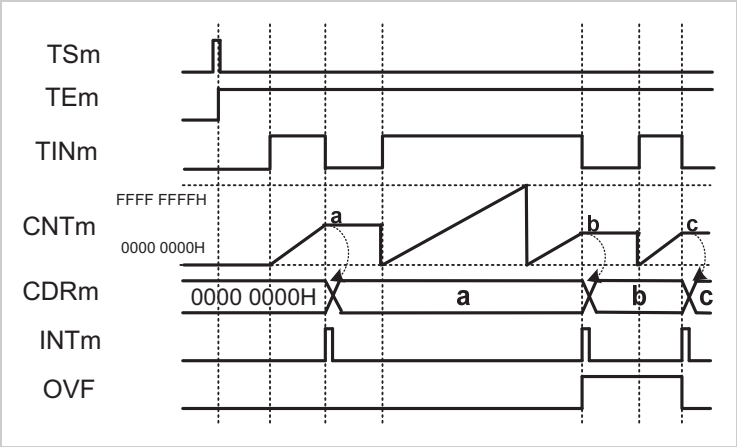


Figure 18-38 TAUJnCMORm.COS[1:0] = 00_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is written to TAUJnCDRm and TAUJnCSRm.OVF is set to 1.

(b) TAUJnCMORm.COS[1:0] = 01_B

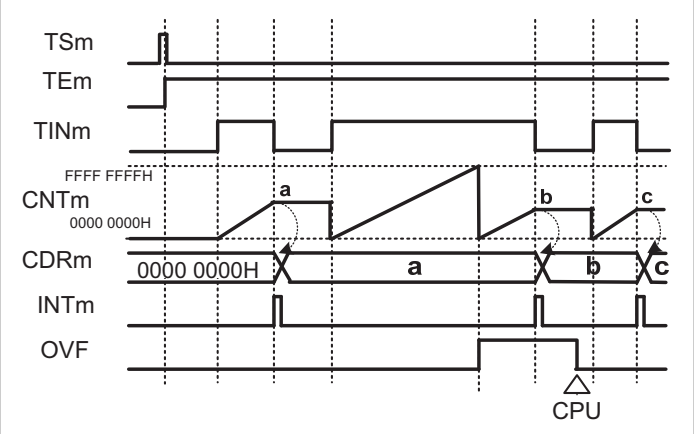


Figure 18-39 TAUJnCMORm.COS[1:0] = 01_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=11_B

- When an overflow occurs, the value of TAUJnCDRm remains unchanged and TAUJnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, the value of TAUJnCNTm is written to TAUJnCDRm.
- TAUJnCSRm.OVF is only cleared by a CPU command.

(c) TAUJnCMORm.COS[1:0] = 10_B

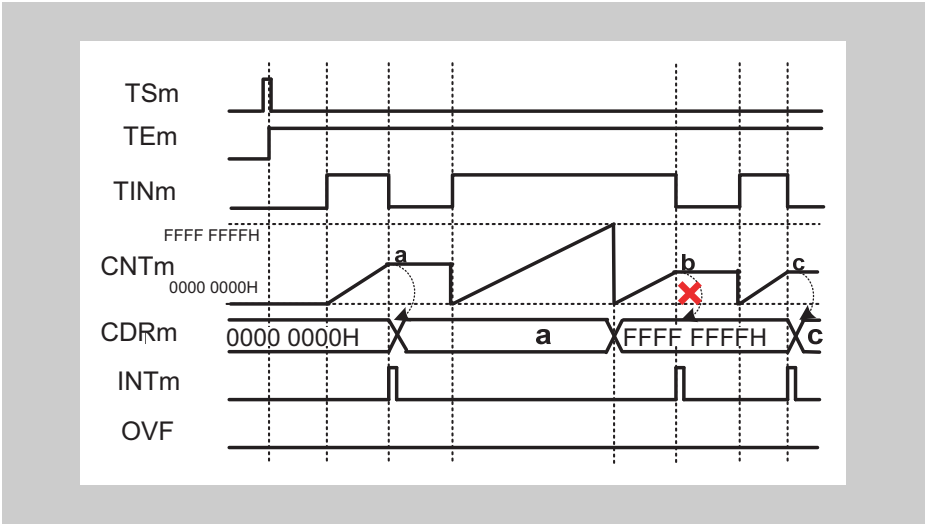


Figure 18-40 TAUJnCMORm.COS[1:0] = 10_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$ and TAUJnCSRm.OVF remains = 0.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.OVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.

(d) TAUJnCMORm.COS[1:0] = 11_B

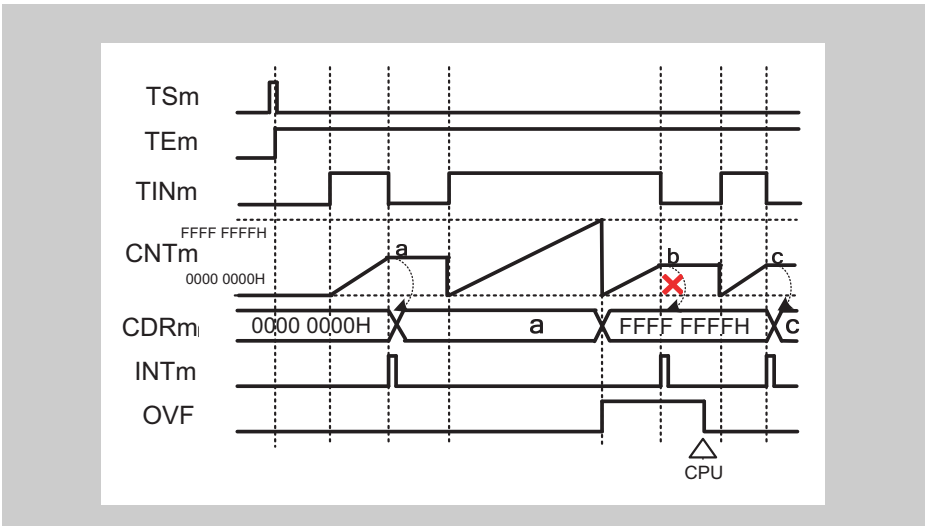


Figure 18-41 TAUJnCMORm.COS[1:0] = 11_B, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0]=11_B

- When an overflow occurs, TAUJnCDRm is set to FFFF FFFF$_H$, and TAUJnCSRm.OVF is set to 1.
- Upon detection of the next valid TAUJnTTINm input edge, TAUJnCNTm is reset to 0, but TAUJnCDRm and TAUJnCSRm.OVF remain unchanged.
- Thus, the next TAUJnTTINm input valid edge after the overflow is ignored.
- TAUJnCSRm.OVF is cleared by a CPU command.

18.16.3 Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

(1) Overview

Summary This function measures the width of an individual TAUJnTTINm input signal. An interrupt is generated if the TAUJnTTINm input width is longer than FFFF $FFFF_H$.

Prerequisites
- The operation mode must be set to One Count Mode, refer to *Table 18-34 "TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)" on page 1466*
- TAUJnTTOUTm is not used for this function
- The value of TAUJnCDRm must be set to FFFF $FFFF_H$.

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUJnTTINm input start edge is detected. FFFF $FFFF_H$ is written to TAUJnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value.

When the next TAUJnTTINm input start edge is detected, TAUJnCNTm reloads FFFF $FFFF_H$ and starts to count down.

If the counter reaches 0000 0000_H before a stop edge is detected, an interrupt is generated.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TIS[1:0] bits:
- If TAUJnCMURm.TIS[1:0] = 10_B, the TAUJnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TIS[1:0] = 11_B, the TAUJnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

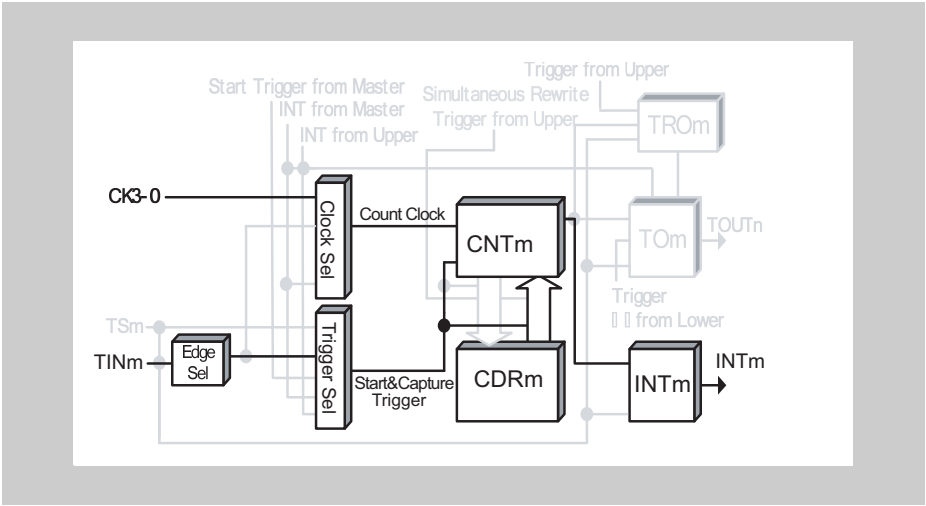


Figure 18-42 Block diagram for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TIS[1:0] = 11_B)

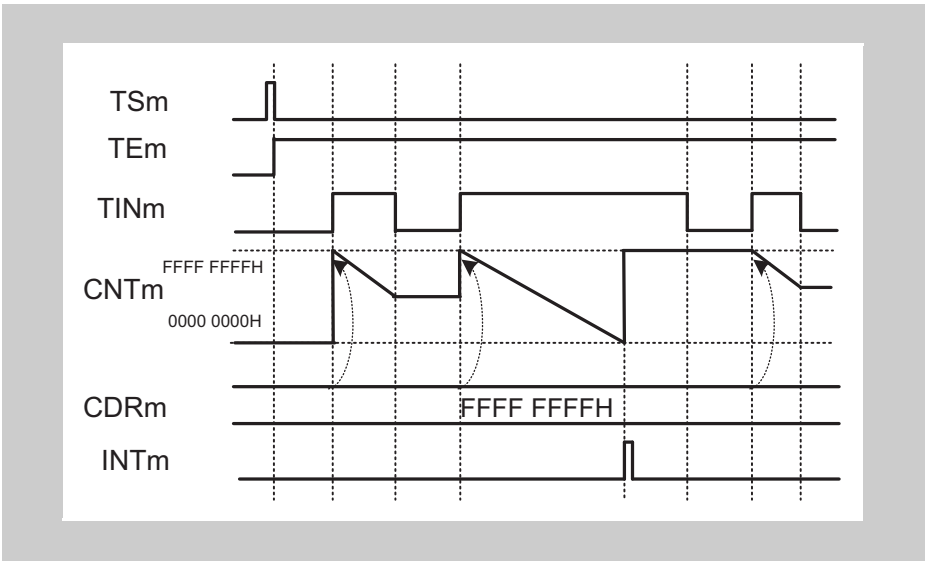


Figure 18-43 General timing diagram for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

(3) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-34 TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	0: INTTAUJnIm not generated at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-35 TAUJnCMURm settings Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement). Therefore, these registers must be set to 0.

Table 18-36 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(4) Operating procedure for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

Table 18-37 Operating procedure for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-34 "TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)" on page 1466* and *Table 18-35 "TAUJnCMURm settings Overflow Interrupt Output Function (During TAUJnTTINm Width Measurement)" on page 1466* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TEm is set to 1 and TAUJnCNTm waits for detection of the start edge. When a start edge is detected, TAUJnCNTm loads the TAUJnCDRm value (FFFF FFFF$_H$).
During operation	The TAUJnCNTm register can be read at any time. Detection of TAUJnTTINm edges.	TAUJnCNTm counts down. When the counter reaches 0000 0000$_H$: • INTTAUJnIm is generated • TAUJnCNTm stops counting at FFFF FFFF$_H$ and waits for a trigger. When a reverse edge of TAUJnTTINm is detected during count operation: • TAUJnCNTm stops counting and waits for a trigger. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and retains its current value.

Restart →

18.16.4 TAUJnTTINm Input Period Count Detection Function

(1) Overview

Summary This function measures the cumulative width of a TAUJnTTINm input signal.

Prerequisites
- The operation mode must be set to Capture & Gate Count Mode, refer to *Table 18-38 "TAUJnCMORm settings for TAUJnTTINm Input Period Count Detection Function" on page 1470*
- TAUJnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. The counter awaits a valid TAUJnTTINm input edge.

When a valid TAUJnTTINm input start edge is detected, the counter starts to count from 0000 0000_H.

When a valid TAUJnTTINm input stop edge is detected, the current TAUJnCNTm value is written to TAUJnCDRm and an interrupt (INTTAUJnIm) is generated. The counter stops and retains its value until the next valid TAUJnTTINm input start edge is detected.

If the counter reaches FFFF $FFFF_H$ the bit TAUJnCSRm.OVF is set to 1 and the counter restarts from 0000 0000_H. The value of TAUJnCSRm.OVF is reset by the CPU by setting TAUJnCSCm.CLOV = 1.

Note The input TAUJnTTINm is sampled at the frequency of the operation clock, specified by the TAUJnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUJnTTOUTm has an error of ± 1 operation clock cycle.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TIS[1:0] bits:
- If TAUJnCMURm.TIS[1:0] = 10_B, the TAUJnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TIS[1:0] = 11_B, the TAUJnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

(2) Equations

Cumulative TAUJnTTINm input width =
count clock cycle × ((FFFF $FFFF_H$ × TAUJnCSRm.OVF) + (TAUJnCDRm capture value + 1))

(3) Block diagram and general timing diagram

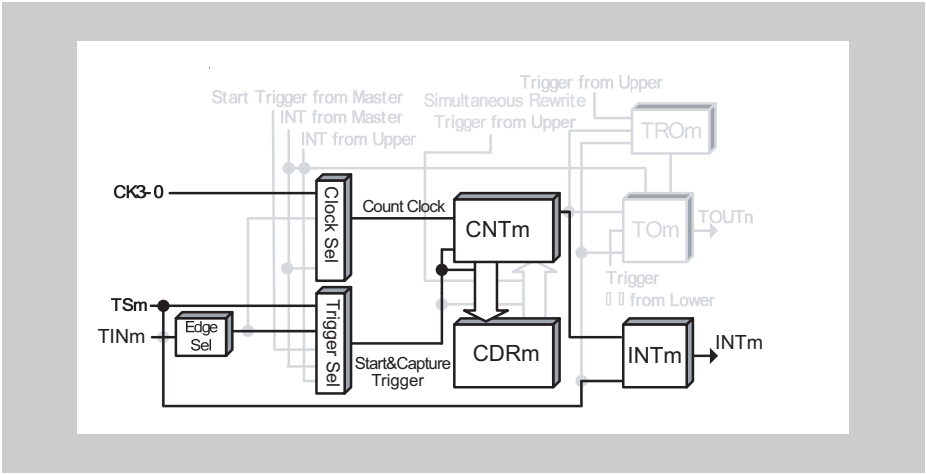


Figure 18-44 Block diagram for TAUJnTTINm Input Period Count Detection Function

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TIS[1:0] = 11_B)

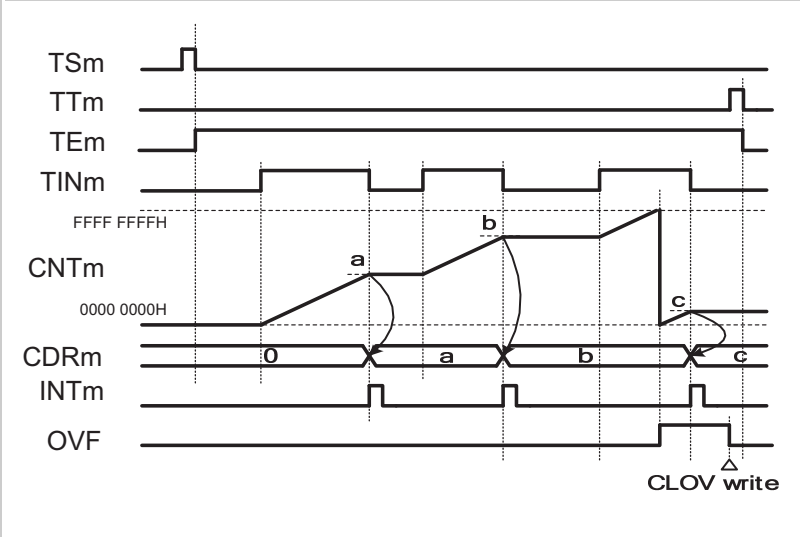


Figure 18-45 General timing diagram for TAUJnTTINm Input Period Count Detection Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-38 TAUJnCMORm settings for TAUJnTTINm Input Period Count Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	01: Overflow (TAUJnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1101: Capture & Gate Count Mode
MD0	0: INTTAUJnIm not generated at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-39 TAUJnCMURm settings for TAUJnTTINm Input Period Count Detection Function

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm Input Period Count Detection Function. Therefore, these registers must be set to 0.

Table 18-40 Simultaneous rewrite settings for TAUJnTTINm Input Period Count Detection Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm Input Period Count Detection Function

Table 18-41 Operating procedure for TAUJnTTINm Input Period Count Detection Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-38 "TAUJnCMORm settings for TAUJnTTINm Input Period Count Detection Function" on page 1470* and *Table 18-39 "TAUJnCMURm settings for TAUJnTTINm Input Period Count Detection Function" on page 1470* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TEm is set to 1 and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. When a start edge is detected, TAUJnCNTm is cleared to 0000 0000_H and TAUJnCNTm starts to count up.
During operation	Detection of TAUJnTTINm edges. The TAUJnCDRm, TAUJnCNTm, and TAUJnCSRm registers can be read at any time. The TAUJnCSC.CLOV bit can be set to 1.	When a TAUJnTTINm start edge (rising edge for high width measurement, falling edge for low width measurement) is detected, TAUJnCNTm starts to count up from the stop value. When TAUJnCNTm detects a capture edge (falling edge for high width measurement, rising edge for low width measurement), it transfers the value to TAUJnCDRm and INTTAUJnIm is generated. Counting stops at the "value transferred to TAUJnCDRm + 1" value and TAUJnCNTm waits for detection of the TAUJnTTINm start edge. If the TAUJnCNTm reaches FFFF $FFFF_H$, the counter overflows and TAUJnCSR.OVF is set to 1. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and it and TAUJnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

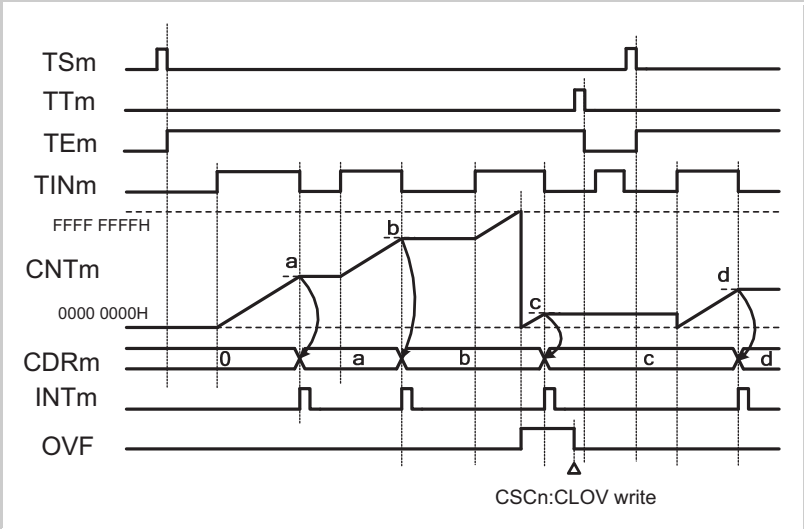


Figure 18-46 Operation stop and restart, TAUJnCMURm.TIS[1:0] = 11_B

- The counter can be stopped by setting TAUJnTT.TTm to 1, which in turn sets TAUJnTE.TEm to 0.
- TAUJnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUJnTTINm input edges are ignored.
- The counter can be restarted by setting TAUJnTS.TSm to 1. TAUJnCNTm restarts to count from 0000 0000_H.

18.16.5 Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

(1) Overview

Summary This function measures the cumulative width of a TAUJnTTINm input signal. An interrupt is generated if the cumulative TAUJnTTINm input width is longer than FFFF FFFF$_H$.

Prerequisites

- The operation mode must be set to Gate Count Mode, refer to *Table 18-42 "TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)" on page 1476*
- TAUJnTTOUTm is not used for this function
- The value of TAUJnCDRm must be set to FFFF FFFF$_H$.

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation.

The counter starts when a valid TAUJnTTINm input start edge is detected. FFFF FFFF$_H$ is written to TAUJnCNTm and the counter starts to count down.

When a valid stop edge is detected, the counter stops and retains the current value. The counter awaits the next TAUJnTTINm input start edge and then continues to count down from the current value.

When the counter reaches 0000 0000$_H$ an interrupt is generated. FFFF FFFF$_H$ is written to TAUJnCNTm and the counter continues to count down until a TAUJnTTINm input stop edge is detected.

Conditions The valid start and stop edges are specified by the TAUJnCMURm.TIS[1:0] bits:

- If TAUJnCMURm.TIS[1:0] = 10$_B$, the TAUJnTTINm input low width is measured. The start trigger is a falling edge and the stop trigger is a rising edge.
- If TAUJnCMURm.TIS[1:0] = 11$_B$, the TAUJnTTINm input high width is measured. The start trigger is a rising edge and the stop trigger is a falling edge.

Note The counter cannot be restarted during operation.

(2) Block diagram and general timing diagram

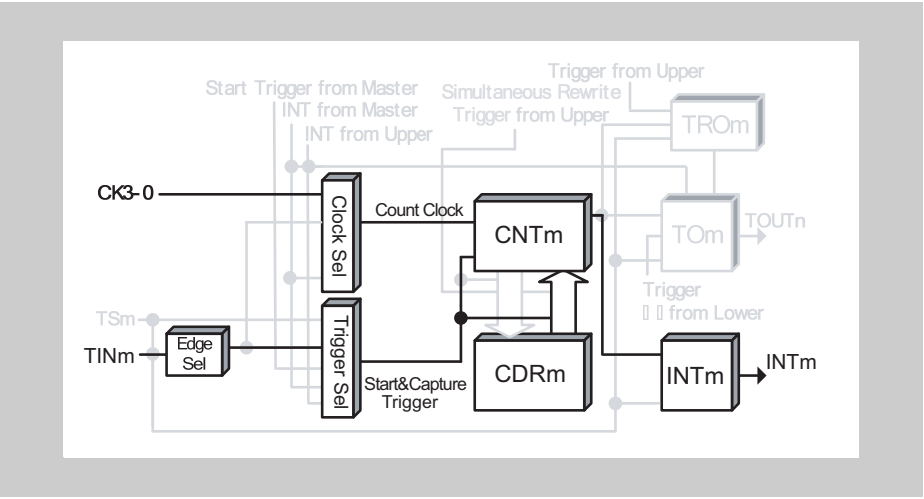


Figure 18-47 Block diagram for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

The following settings apply to the general timing diagram:

- Rising and falling edge detection = high width measurement (TAUJnCMURm.TIS[1:0] = 11_B)

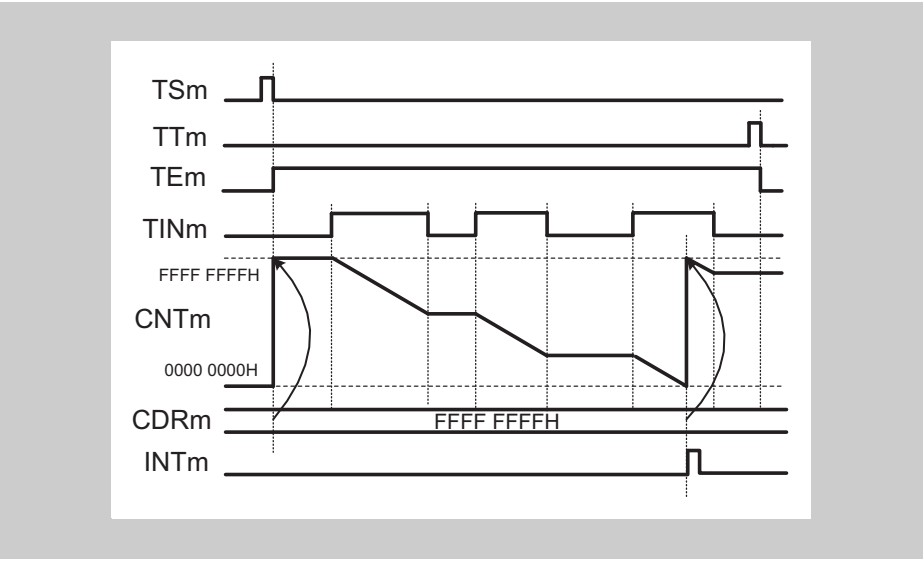


Figure 18-48 General timing diagram for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

(3) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-42 TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	010: Valid edge of the TAUJnTTINm input signal is the external start trigger and the reverse edge is the stop trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	1100: Gate Count Mode
MD0	0: INTTAUJnIm not generated at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-43 TAUJnCMURm settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

Bit name	Setting
TIS[1:0]	10: Rising and falling edge detection (Low width measurement) 11: Rising and falling edge detection (High width measurement)

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection). Therefore, these registers must be set to 0.

Table 18-44 Simultaneous rewrite settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(4) Operating procedure for Overflow Interrupt Output Function (during TAUJnTTINm input period count detection)

Table 18-45 Operating procedure for Overflow Interrupt Output Function (during TAUJnTTINm input period count detection)

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-42 "TAUJnCMORm settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)" on page 1476* and *Table 18-43 "TAUJnCMURm settings for Overflow Interrupt Output Function (During TAUJnTTINm Input Period Count Detection)" on page 1476* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0. Detection of TAUJnTTINm start edge	TAUJnTE.TEm is set to 1 and TAUJnCNTm waits for detection of the start edge. When a start edge is detected, TAUJnCNTm loads the TAUJnCDRm value (FFFF $FFFF_H$).
During operation	The TAUJnCNTm register can be read at all times.	TAUJnCNTm counts down. When the counter reaches 0000 0000_H: • INTTAUJnIm is generated • TAUJnCNTm loads the TAUJnCDRm value (FFFF $FFFF_H$) and continues to count down. When a reverse edge of TAUJnTTINm is detected during count operation: • TAUJnCNTm counts down from the stop value. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and retains its current value.

Restart

18.17 Other Independent Channel Functions

This chapter describes a function that generates an interrupt when a certain number of TAUJnTTINm pulses has occurred, a function that divides the frequency of TAUJnTTINm, and a function that measures the duration between the function start and a TAUJnTTINm input signal:

- *18.17.1 “TAUJnTTINm Input Position Detection Function”*

18.17.1 TAUJnTTINm Input Position Detection Function

(1) Overview

Summary This function measures the duration between the function start and a TAUJnTTINm input signal.

Prerequisites
- The operation mode must be set to Count Capture Mode, refer to *Table 18-46 "TAUJnCMORm settings for TAUJnTTINm Input Position Detection Function" on page 1482*
- TAUJnTTOUTm is not used for this function

Description The counter is enabled by setting the channel trigger bit (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. The counter starts to count from 0000 0000_H. When a valid TAUJnTTINm input stop edge is detected, the current TAUJnCNTm value is written to TAUJnCDRm and an interrupt (INTTAUJnIm) is generated. The counter continues to count from the current value until the next valid TAUJnTTINm input edge is detected.

When the counter reaches FFFF $FFFF_H$, the bit TAUJnCSRm.OVF is set to 1 and the counter restarts from 0000 0000_H. The value of TAUJnCSRm.OVF is reset by the CPU by TAUJnCSCm.CLOV = 1.

Note The input TAUJnTTINm is sampled at the frequency of the operation clock, specified by the TAUJnCMORm.CKS[1:0] bits. As a result, the output cycle of TAUJnTTOUTm has an error of ± 1 operation clock cycle.

Conditions If the TAUJnCMORm.MD0 bit is set to 0, the first interrupt after a start or restart is not generated. For details refer to *18.11 "TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts" on page 1428* .

(2) Equations

Function duration at a TAUJnTTINm input pulse =

count clock cycle × [(FFFF $FFFF_H$+1 × TAUJnCSRm.OVF) + (TAUJnCDRm capture value + 1)]

(3) Block diagram and general timing diagram

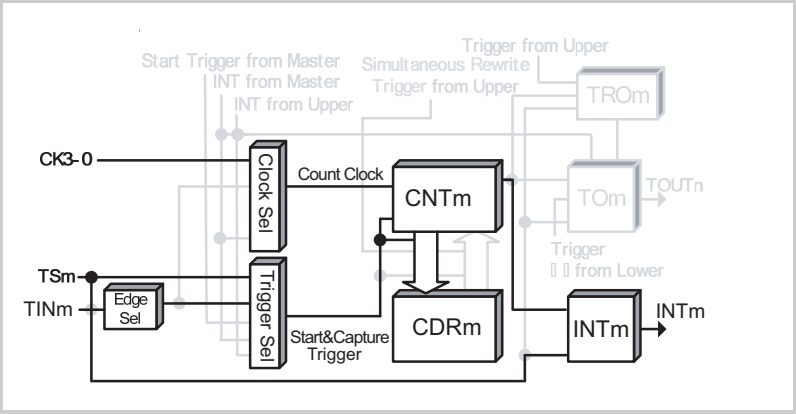


Figure 18-49 Block diagram for TAUJnTTINm Input Position Detection Function

The following settings apply to the general timing diagram:

- INTTAUJnIm not generated at operation start (TAUJnCMORm.MD0 = 0)
- Falling edge detection (TAUJnCMURm.TIS[1:0] = 00_B)

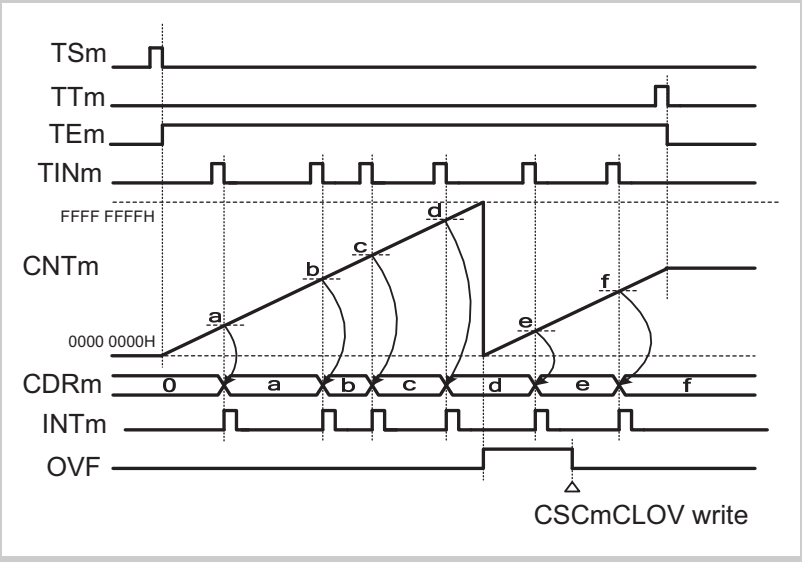


Figure 18-50 General timing diagram for TAUJnTTINm Input Position Detection Function

(4) Register settings

(a) TAUJnCMORm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-46 TAUJnCMORm settings for TAUJnTTINm Input Position Detection Function

Bit name	Setting
CKS[1:0]	00: Operation clock = CK0 01: Operation clock = CK1 10: Operation clock = CK2 11: Operation clock = CK3
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Not used, so set to 0
STS[2:0]	001: Valid TAUJnTTINm input edge signal is used as the external capture trigger
COS[1:0]	01: Overflow (TAUJnCSRm.OVF) set upon counter overflow and cleared by a CPU instruction
MD[4:1]	1011: Count Capture Mode
MD0	0: INTTAUJnIm not generated at operation start 1: Generates INTTAUJnIm at operation start

(b) TAUJnCMURm

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-47 TAUJnCMURm settings for TAUJnTTINm Input Position Detection Function

Bit name	Setting
TIS[1:0]	00: Falling edge detection 01: Rising edge detection 10: Rising and falling edge detection

(c) Channel output mode

The channel output mode is not used by this function. However, it can be used in Direct Channel Output Mode.

(d) Simultaneous rewrite

The simultaneous rewrite registers (TAUJnRDE and TAUJnRDM) cannot be used with the TAUJnTTINm Input Position Detection Function. Therefore, these registers must be set to 0.

Table 18-48 Simultaneous rewrite settings for TAUJnTTINm Input Position Detection Function

Bit name	Setting
RDE.RDEm	0: Disables simultaneous rewrite
RDM.RDMm	0: When simultaneous rewrite is disabled (TAUJnRDE.RDEm = 0), set these bits to 0

(5) Operating procedure for TAUJnTTINm Input Position Detection Function

Table 18-49 Operating procedure for TAUJnTTINm Input Position Detection Function

	Operation	Status of TAUJn
Initial channel setting	Set the TAUJnCMORm register and TAUJnCMURm registers as described in *Table 18-46 "TAUJnCMORm settings for TAUJnTTINm Input Position Detection Function" on page 1482* and *Table 18-47 "TAUJnCMURm settings for TAUJnTTINm Input Position Detection Function" on page 1482* Set the value of the TAUJnCDRm register	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm to 1. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is set to 1 and the counter starts. INTTAUJnIm is generated when TAUJnCMORm.MD0 is set to 1.
During operation	The TAUJnCMURm.TIS[1:0] bits can be changed at any time. The TAUJnCDRm and TAUJnCSRm registers can be read at any time.	TAUJnCNTm starts to count up from 0000 0000_H. When a TAUJnTTINm valid edge is detected: • TAUJnCNTm transfers (captures) its value to TAUJnCDRm • TAUJnTTINm is output. • The counter value is not cleared to 0000 0000_H and TAUJnCNTm continues count operation. Afterwards, this procedure is repeated.
Stop operation	Set TAUJnTT.TTm to 1. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm stops and both it and TAUJnCSRm.OVF retain their current values.

Restart

(6) Specific timing diagrams

(a) Operation stop and restart

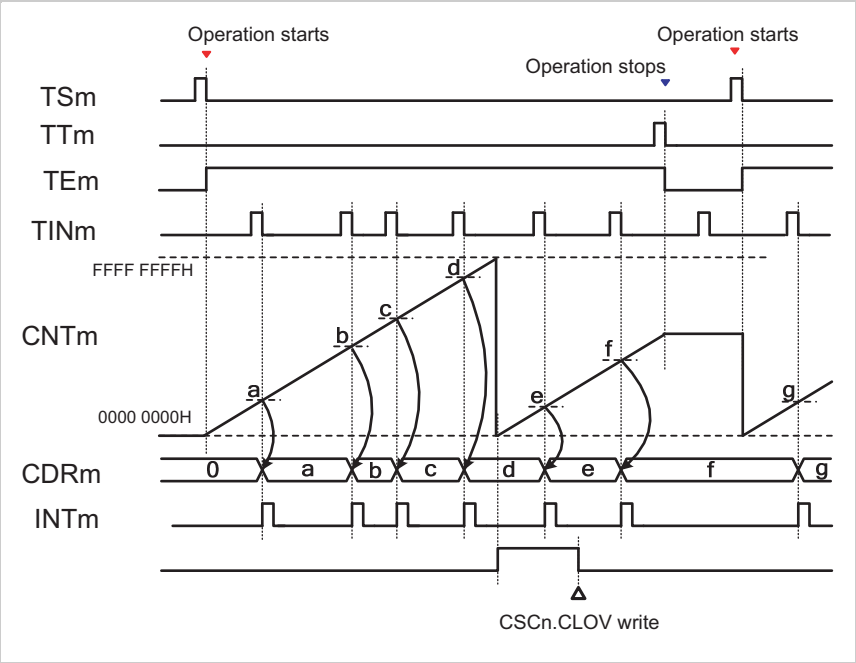


Figure 18-51 Operation stop and restart, TAUJnCMORm.MD0 = 0, TAUJnCMURm.TIS[1:0] = 00

- The counter can be stopped by setting TAUJnTT.TTm to 1, which in turn sets TAUJnTE.TEm to 0.
- TAUJnCNTm stops and the current value is retained.
- If the counter is stopped, valid TAUJnTTINm input edges are ignored.
- The counter can be restarted by setting TAUJnTS.TSm to 1. TAUJnCNTm restarts to count from 0000 0000_H.

18.18 Synchronous PWM Signal Functions Triggered at Regular Intervals

This chapter describes a function that generates PWM signals at regular intervals. For a general overview of synchronous channel operation, see *18.4 "Functional Description" on page 1409* .

- *18.18.1 "PWM Output Function"*

18.18.1 PWM Output Function

(1) Overview

Summary This function generates multiple PWM outputs by using a master and multiple slave channels. It enables the pulse cycle (frequency) and the pulse width (duration) of the TAUJnTTOUTm to be set. The pulse cycle is set in the master channel. The pulse width is set in the slave channel.

Prerequisites

- Two channels
- The operation mode of the master channel must be set to Interval Timer Mode, refer to *Table 18-50 “TAUJnCMORm settings for the master channel of the PWM Output Function” on page 1490*
- The operation mode of the slave channel(s) must be set to One Count Mode, refer to *Table 18-53 “TAUJnCMORm settings for the slave channel of the PWM Output Function” on page 1492*
- TAUJnTTOUTm is not used for the master channel of this function
- The channel output mode of the slave channel(s) must be set to Synchronous Channel Output Mode 1 (*18.9 “Channel Output Modes” on page 1421*)

Description The counters are started by setting the channel trigger bits (TAUJnTS.TSm) to 1. This in turn sets TAUJnTE.TEm = 1, enabling count operation. The current value of TAUJnCDRm is written to TAUJnCNTm and the counters start to count down from these values. INTTAUJnIm is generated on the master channel and TAUJnTTOUTm (slave) toggles.

- Master channel:

  When the counter of the master channel reaches 0000 0000_H, pulse cycle time has elapsed and INTTAUJnIm is generated. The counter reloads the TAUJnCDRm value and counts down.

- Slave channel(s)

  The INTTAUJnIm of the master channel triggers the counter of the slave channel(s). The current value of TAUJnCDRm (slave) is written to TAUJnCNTm (slave) and the counter starts to count down from this value. The TAUJnTTOUTm signal is set.

  When the counter reaches 0000 0000_H, i.e. duty time has elapsed, INTTAUJnIm is generated and the TAUJnTTOUTm signal is reset. The counter returns to FFFF $FFFF_H$ and awaits the next INTTAUJnIm of the master channel, and thus the start of the next pulse cycle.

The counter can be stopped by setting TAUJnTT.TTm to 1 for the master and slave channel(s), which in turn sets TAUJnTE.TEm to 0. TAUJnCNTm and TAUJnTTOUTm of master and slave channel(s) stop but retain their values. The counters can be restarted by setting TAUJnTS.TSm to 1.

Note If a forced restart is executed during operation, TAUJnTTOUTm is not output as a PWM signal.

Conditions Simultaneous rewrite can be used with this function. Please refer to *18.8 “Simultaneous Rewrite” on page 1416*

(2) Equations

Pulse cycle = (TAUJnCDRm (master) + 1) x count clock cycle

Duty cycle [%] = (TAUJnCDRm (slave) / (TAUJnCDRm (master) + 1)) x 100

– Duty cycle = 0 %

  TAUJnCDRm (slave) = 0000 0000_H

– Duty cycle = 100 %

  TAUJnCDRm (slave) ≥ TAUJnCDRm (master) + 1

(3) Block diagram and general timing diagram

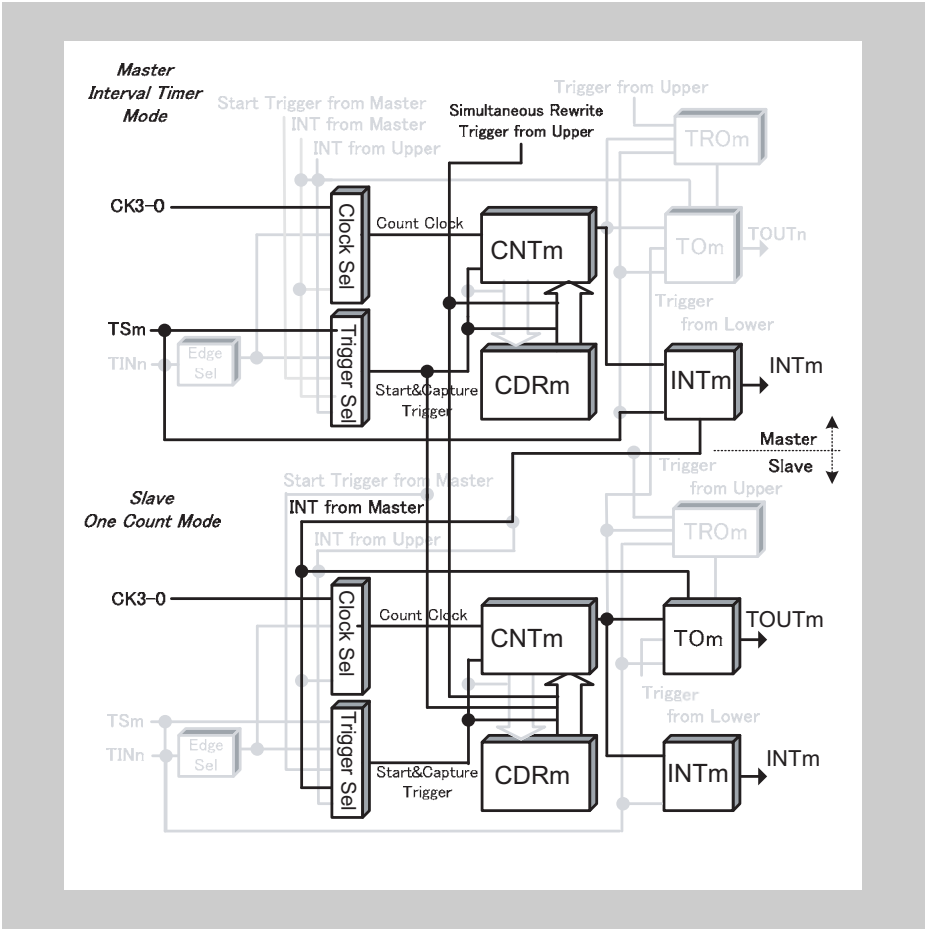


Figure 18-52 Block diagram for PWM Output Function

The following settings apply to the general timing diagram:

- Slave channel: Positive logic (TAUJnTOL.TOLm = 0)

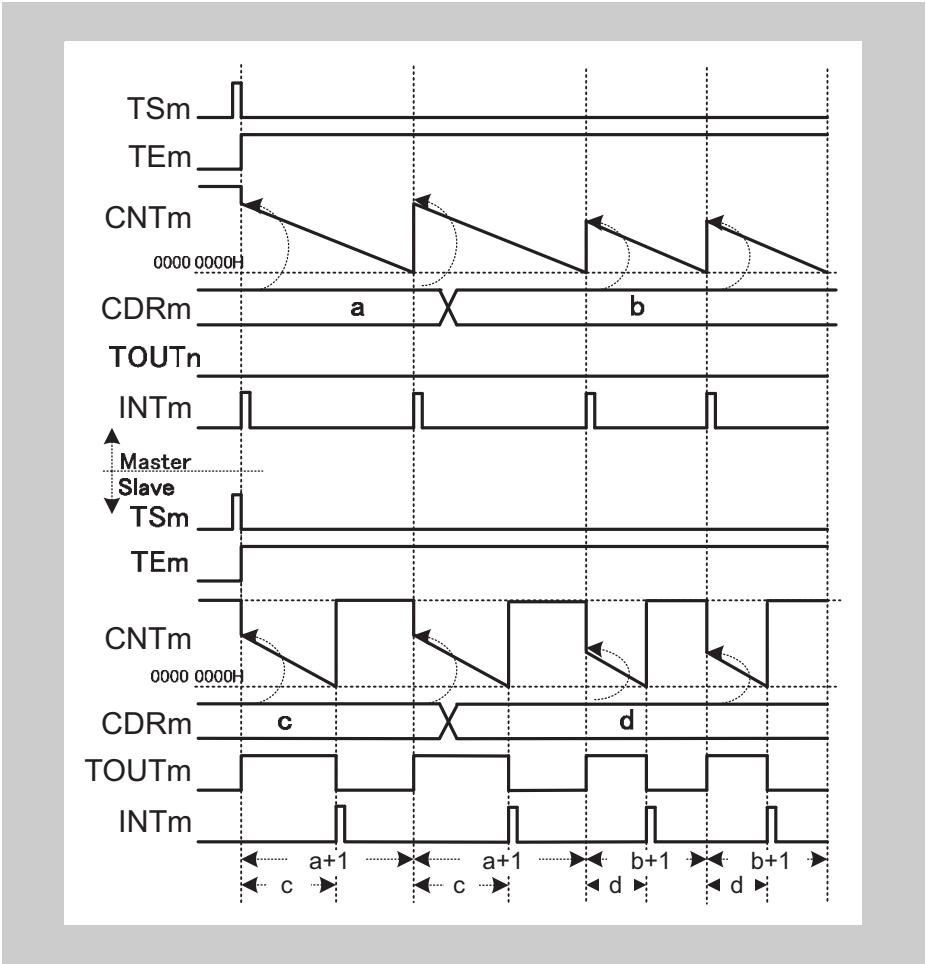


Figure 18-53 General timing diagram for PWM Output Function

Note The interval between the slave channel starting to count and an interrupt being generated is the value of corresponding TAUJnCDRm, whereas for the master channel the interval is the corresponding TAUJnCDRm + 1.

(4) Register settings for the master channel

(a) TAUJnCMORm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-50 TAUJnCMORm settings for the master channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	1: Channel is master channel
STS[2:0]	000: Counter triggered by software trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0000: Interval Timer Mode
MD0	1: Generates INTTAUJnIm at operation start

(b) TAUJnCMURm for the master channel

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-51 TAUJnCMURm settings for the master channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the master channel

The channel output mode is not used by this function. However, it can be used by other functions or in Direct Channel Output Mode.

(d) Simultaneous rewrite for the master channel

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 18-52 Simultaneous rewrite settings for the master channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting

(5) Register settings for the slave channel(s)

(a) TAUJnCMORm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:1]				MD0

Table 18-53 TAUJnCMORm settings for the slave channel of the PWM Output Function

Bit name	Setting
CKS[1:0]	00: Prescaler output CK0 01: Prescaler output CK1 10: Prescaler output CK2 11: Prescaler output CK3 The value of the CKS[1:0] bit of the master and slave channel(s) must be identical.
CCS[1:0]	00: Operation clock is used as the count clock
MAS	0: Channel is a slave channel
STS[2:0]	100: INTTAUJnIm of the master channel is the start trigger
COS[1:0]	00: Not used, so set to 00
MD[4:1]	0100: One Count Mode
MD0	1: Generates INTTAUJnIm and toggles TAUJnTTOUTm at operation start

(b) TAUJnCMURm for the slave channel(s)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-														TIS[1:0]	

Table 18-54 TAUJnCMURm settings for the slave channel of the PWM Output Function

Bit name	Setting
TIS[1:0]	00: Not used so set to 00

(c) Channel output mode for the slave channel(s)

Table 18-55 Control bit settings for Independent Channel Output Mode 1

Bit name	Setting
TOEm	1: Disables Direct Channel Output Mode
TOMm	1: Synchronous channel operation
TOCm	0: Operation mode 1
TOLm	0: Positive logic 1: Inverted logic

(d) Simultaneous rewrite for the slave channel(s)

The simultaneous rewrite settings of the master and slave channel must be identical.

Table 18-56 Simultaneous rewrite settings for the slave channel of the PWM Output Function

Bit name	Setting
RDE.RDEm	1: Enables simultaneous rewrite
RDM.RDMm	0: The simultaneous rewrite trigger signal is generated when the master channel starts counting

(6) Operating procedure for PWM Output Function

Table 18-57 Operating procedure for PWM Output Function

	Operation	Status of TAUJn
Initial channel setting	Master channel: set the TAUJnCMORm and TAUJnCMURm registers and the channel output mode as described in *4 "Register settings for the master channel" on page 1490* Slave channel: set the TAUJnCMORm and TAUJnCMURm registers and the channel output mode as described in *5 "Register settings for the slave channel(s)" on page 1492* Set the values of the TAUJnCDRm registers of all channels	Channel operation is stopped. (A clock is supplied, and a small amount of power is consumed.)
Start operation	Set TAUJnTS.TSm of the master and slave channels to 1 simultaneously. TAUJnTS.TSm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm (master and slave channels) is set to 1 and the counters of the master and slave channels start. INTTAUJnIm is generated on the master channel and TAUJnTTOUTm (slave) is set.
During operation	TAUJnCDRm can be changed at any time. TAUJnCNTm and TAUJnRSF.RSFm can be read at any time. TAUJnRDT.RDTm can be changed during operation.	TAUJnCNTm of the master channel loads TAUJnCDRm and counts down. When the counter reaches 0000 0000_H: • INTTAUJnIm (master) is generated • TAUJnCNTm (master) reloads the TAUJnCDRm value and continues count operation • TAUJnCNTm (slave) reloads the TAUJnCDRm value and counts down • TAUJnTTOUTm (slave) is set When TAUJnCNTm (slave) reaches 0000 0000_H: • INTTAUJnIm (slave) is generated • TAUJnTTOUTm (slave) is reset
Stop operation	Set TAUJnTT.TTm of the master and slave channels to 1 simultaneously. TAUJnTT.TTm is a trigger bit, so it is automatically cleared to 0.	TAUJnTE.TEm is cleared to 0 and the counter stops. TAUJnCNTm and TAUJnTTOUTm stop and retain their current values. When TAUJnTOE.TOEm is 0, TAUJnTTOUTm output is initialized to the value set by TAUJnTO.TOm.

Restart

(7) Specific timing diagrams

(a) Duty cycle = 0 %

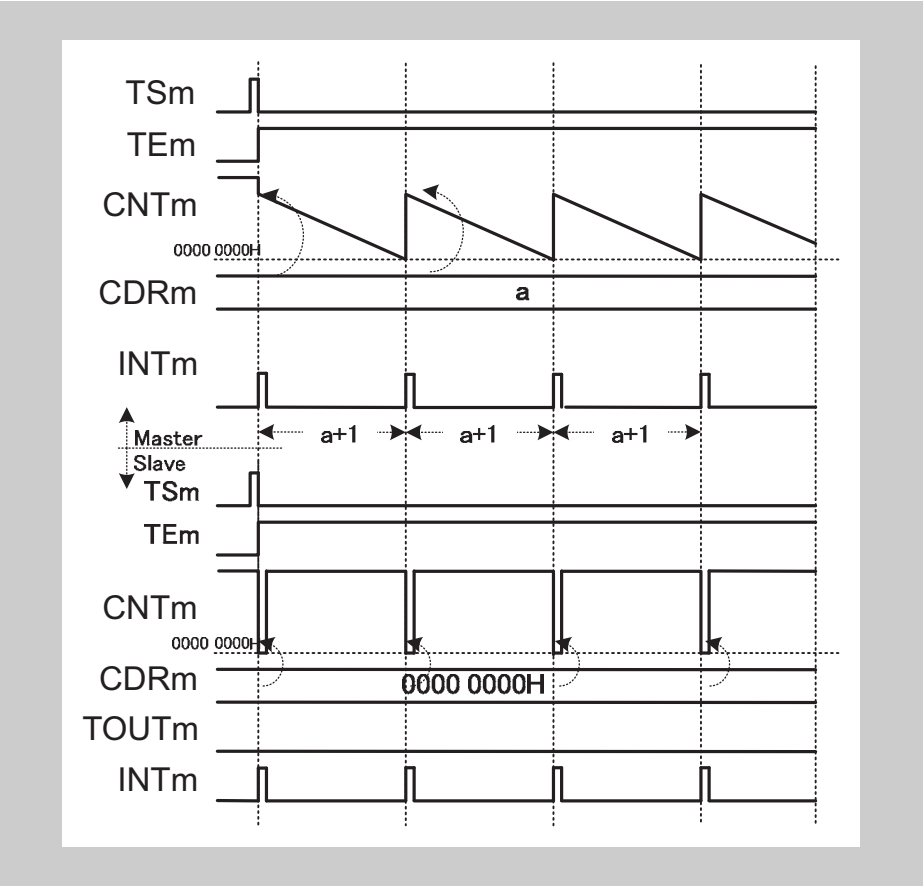


Figure 18-54 TAUJnCDRm (slave) = 0000 0000_H, positive logic (TAUJnTOL.TOLm (slave) = 0)

- Every time the master channel generates an interrupt (INTTAUJnIm), 0000 0000_H is written to TAUJnCNTm (slave). Therefore, TAUJnCNTm (slave) cannot start to count and TAUJnTTOUTm remains at not active state.
- TAUJnCNTm (slave) generates an interrupt every time the value of TAUJnCDRm is reloaded. The slave and the master channel generate interrupts in the same cycle.

(b) Duty cycle = 100 %

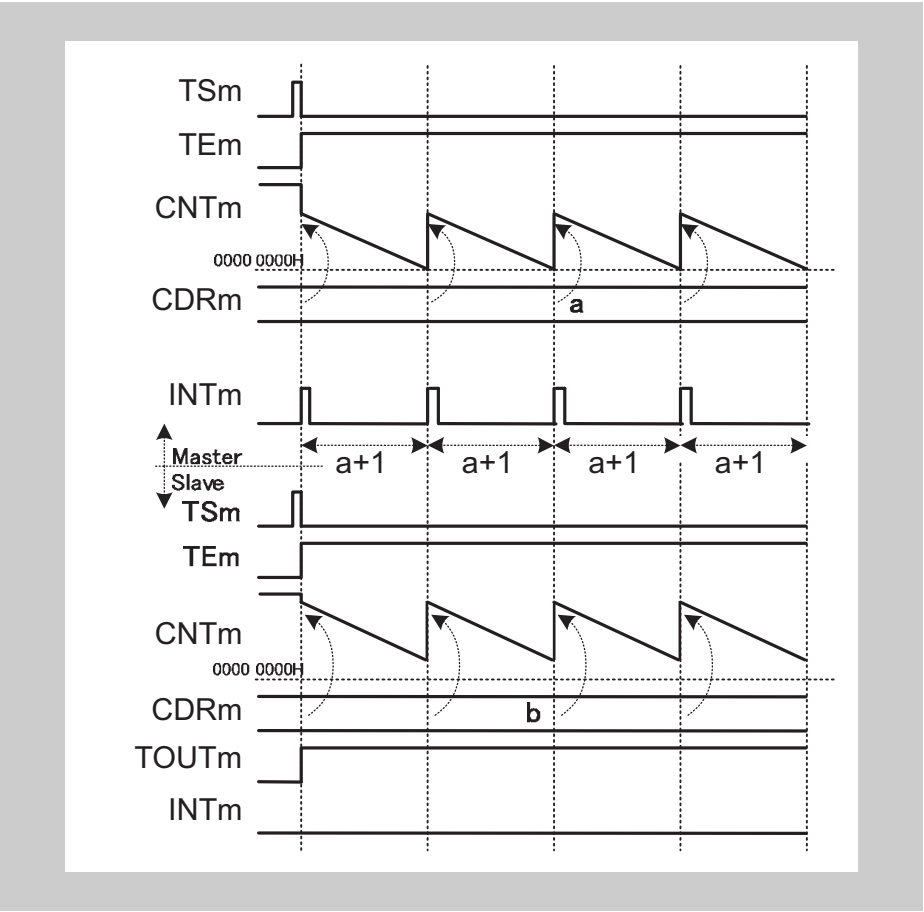


Figure 18-55 TAUJnCDRm (slave) ≥ TAUJnCDRm (master) + 1, positive logic (TAUJnTOL.TOLm (slave) = 0)

- If the value TAUJnCDRm (slave) is higher than the value TAUJnCDRm (master), the counter of the slave channel cannot reach 0000 0000_H and cannot generate interrupts. The TAUJnTTOUTm remains at active state.

(c) Stop and restart operation

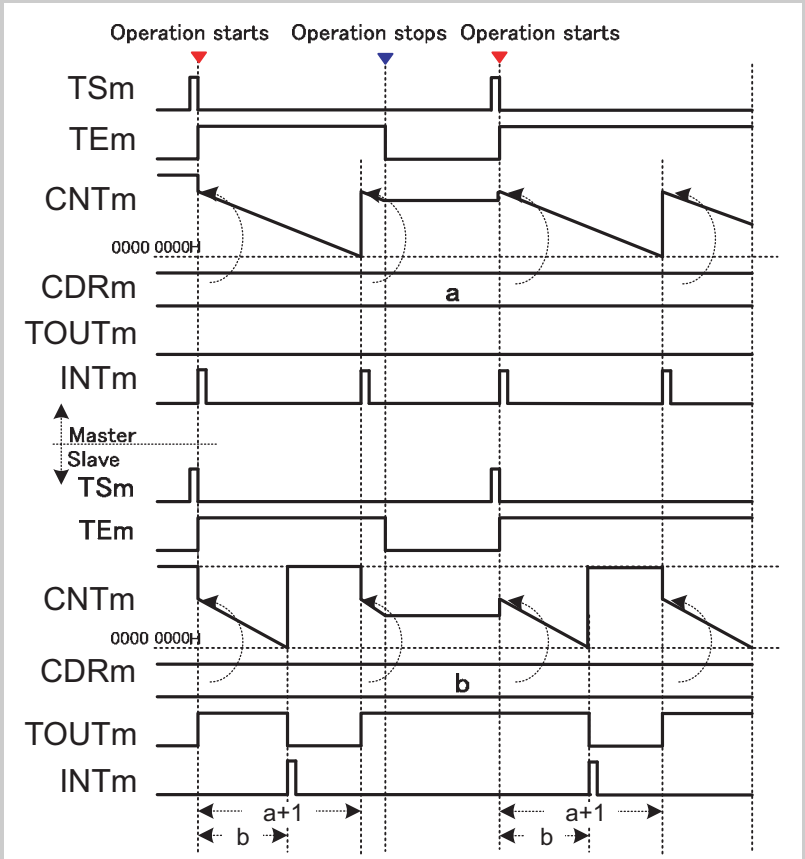


Figure 18-56 Stop and restart operation, positive logic (TAUJnTOL.TOLm (slave) = 0)

- The counter can be stopped by setting TAUJnTT.TTm of the master and slave channel(s) to 1, which in turn sets TAUJnTE.TEm to 0.
- TAUJnCNTm and TAUJnTTOUTm of all channels stop and the current values are retained. No interrupts are generated.
- The counter can be restarted by setting TAUJnTS.TSm of master and slave channel(s) to 1. TAUJnCNTm of master and slave channel reload the current values of TAUJnCDRm and start to count down from these values.

18.19 Registers

This section contains a description of all the registers of the 32-bit Timer Array Unit J.

18.19.1 TAUJn registers overview

The TAUJn is controlled and operated by the registers in the following table. Where there is one register per channel, this is indicated by an “m”, where m runs from 0 to 3.

Table 18-58 TAUJn registers overview

Register name	Shortcut	Address
TAUJn prescaler registers		
TAUJn prescaler clock select register	TAUJnTPS	<TAUJn_base> + 90_H
TAUJn prescaler baud rate setting register	TAUJnBRS	<TAUJn_base> + 94_H
TAUJn control registers		
TAUJn channel data register m	TAUJnCDRm	<TAUJn_base> + m x 4_H
TAUJn channel counter register m	TAUJnCNTm	<TAUJn_base> + 10_H + m x 4_H
TAUJn channel mode OS register m	TAUJnCMORm	<TAUJn_base> + 80_H + m x 4_H
TAUJn channel mode user register m	TAUJnCMURm	<TAUJn_base> + 20_H + m x 4_H
TAUJn channel status register m	TAUJnCSRm	<TAUJn_base> + 30_H + m x 4_H
TAUJn channel status clear trigger register m	TAUJnCSCm	<TAUJn_base> + 40_H + m x 4_H
TAUJn channel start trigger register	TAUJnTS	<TAUJn_base> + 54_H
TAUJn channel enable status register	TAUJnTE	<TAUJn_base> + 50_H
TAUJn channel stop trigger register	TAUJnTT	<TAUJn_base> + 58_H
TAUJn output registers		
TAUJn channel output enable register	TAUJnTOE	<TAUJn_base> + 60_H
TAUJn channel output register	TAUJnTO	<TAUJn_base> + $5C_H$
TAUJn channel output mode register	TAUJnTOM	<TAUJn_base> + 98_H
TAUJn channel output configuration register	TAUJnTOC	<TAUJn_base> + $9C_H$
TAUJn channel output active level register	TAUJnTOL	<TAUJn_base> + 64_H
TAUJn reload data registers		
TAUJn channel reload data enable register	TAUJnRDE	<TAUJn_base> + $A0_H$
TAUJn channel reload data mode register	TAUJnRDM	<TAUJn_base> + $A4_H$
TAUJn channel reload data trigger register	TAUJnRDT	<TAUJn_base> + 68_H
TAUJn channel reload status register	TAUJnRSF	<TAUJn_base> + $6C_H$
TAUJn emulation register		
TAUJn emulation register	TAUJnEMU	<TAUJn_base> + $A8_H$

<TAUJn_base> The <TAUJn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

18.19.2 TAUJn prescaler registers details

(1) TAUJnTPS - TAUJn prescaler clock select register

This register specifies the PCLK prescalers for clocks CK0, CK1, CK2, and CK3_PRE for all channels. CK3 is generated by dividing CK3_PRE by the factor specified in TAUJnBRS.

Access This register can be read/written in 16-bit units.

Address <TAUJn_base> + 90_H

Initial Value $FFFF_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRS3[3:0]				PRS2[3:0]				PRS1[3:0]				PRS0[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-59 TAUJnTPS register contents (1/2)

Bit position	Bit name	Function
15 to 12	PRS3[3:0]	Specifies the CK3_PRE clock. Clock CK3_PRE is the input clock of the BRG unit. The BRG unit supplies the CK3 operation clock for all channels. **PRS3[3:0] / CK3_PRE clock** 0000_B / $PCLK/2^0$ 0001_B / $PCLK/2^1$ 0010_B / $PCLK/2^2$ 0011_B / $PCLK/2^3$ ... / ... 1110_B / $PCLK/2^{14}$ 1111_B / $PCLK/2^{15}$ These bits can only be rewritten when all counters using CK3 are stopped (TAUJnTE.TEm = 0).
11 to 8	PRS2[3:0]	Specifies the CK2 clock. **PRS2[3:0] / CK2 clock** 0000_B / $PCLK/2^0$ 0001_B / $PCLK/2^1$ 0010_B / $PCLK/2^2$ 0011_B / $PCLK/2^3$ ... / ... 1110_B / $PCLK/2^{14}$ 1111_B / $PCLK/2^{15}$ These bits can only be rewritten when all counters using CK2 are stopped (TAUJnTE.TEm = 0).

Table 18-59 TAUJnTPS register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 4</td><td>PRS1[3:0]</td><td>Specifies the CK1 clock.
<table>
<tr><th>PRS1[3:0]</th><th>CK1 clock</th></tr>
<tr><td>0000_B</td><td>$PCLK/2^0$</td></tr>
<tr><td>0001_B</td><td>$PCLK/2^1$</td></tr>
<tr><td>0010_B</td><td>$PCLK/2^2$</td></tr>
<tr><td>0011_B</td><td>$PCLK/2^3$</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>$PCLK/2^{14}$</td></tr>
<tr><td>1111_B</td><td>$PCLK/2^{15}$</td></tr>
</table>
These bits can only be rewritten when all counters using CK1 are stopped (TAUJnTE.TEm = 0).</td></tr>
<tr><td>3 to 0</td><td>PRS0[3:0]</td><td>Specifies the CK0 clock.
<table>
<tr><th>PRS0[3:0]</th><th>CK0 clock</th></tr>
<tr><td>0000_B</td><td>$PCLK/2^0$</td></tr>
<tr><td>0001_B</td><td>$PCLK/2^1$</td></tr>
<tr><td>0010_B</td><td>$PCLK/2^2$</td></tr>
<tr><td>0011_B</td><td>$PCLK/2^3$</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1110_B</td><td>$PCLK/2^{14}$</td></tr>
<tr><td>1111_B</td><td>$PCLK/2^{15}$</td></tr>
</table>
These bits can only be rewritten when all counters using CK0 are stopped (TAUJnTE.TEm = 0).</td></tr>
</table>

Note The TAUJn clock input PCLK is specified in the first section of this chapter under the keyword “Clock supply”.

(2) TAUJnBRS - TAUJn prescaler baud rate setting register

This register specifies the division factor of prescaler clock CK3.

CK3 is generated by dividing CK3_PRE by the factor specified in this register plus one. The PCLK prescaler for CK3_PRE is specified in TAUJnTPS.PRS3[3:0].

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 94_H

Initial Value 00_H

7	6	5	4	3	2	1	0
BRS[07:00]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-60 TAUJnBRS register contents

Bit position	Bit name	Function
7 to 0	BRS[07:00]	Specifies the CK3_PRE clock division factor for generating CK3:

BRS[07:00]	CK3 clock
$0000\ 0000_B$	CK3_PRE / 1
$0000\ 0001_B$	CK3_PRE / 2
$0000\ 0010_B$	CK3_PRE / 3
$0000\ 0011_B$	CK3_PRE / 4
...	...
$1111\ 1110_B$	CK3_PRE / 255
$1111\ 1111_B$	CK3_PRE / 256

18.19.3 TAUJn control registers details

(1) TAUJnCDRm - TAUJn channel data register

This register functions either as a compare register or as a capture register, depending on the operation mode specified in TAUJnCMORm.MD[4:0].

Access This register can be read/written in 32-bit units.

- In capture mode, only reading is possible. Write operation is ignored.
- In compare mode, reading and writing is possible.

Address <TAUJn_base> + 0_H + m x 4_H

Initial Value 0000 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CDR[31:16]															
R/W															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDR[15:0]															
R/W															

Table 18-61 TAUJnCDRm register contents

Bit position	Bit name	Function
31 to 0	CDR[31:0]	Data register for the capture/compare value.

(2) TAUJnCNTm - TAUJn channel counter register

This register is the channel m counter register.

Access This register can be read in 32-bit units.

Address <TAUJn_base> + 10_H + m x 4_H

Initial Value 0000 0000_H or FFFF $FFFF_H$ The initial value depends on the operation mode, see *Table 18-63 "TAUJnCNTm read values after the counter is re-enabled" on page 1503*

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16]															
R															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															
R															

Table 18-62 TAUJnCNTm register contents

Bit position	Bit name	Function
31 to 0	CNT[31:0]	32-bit counter value.

The read value depends on the counter, the operation mode change, and the values of the TAUJnTS.TSm and TAUJnTT.TTm bits.

The *initial* counter read value depends on the operation mode and how the counter was stopped:

- by a reset
- by a counter stop trigger (TAUJnTT.TTm = 1)

The following table lists the initial counter read values after the counter has stopped (TAUJnTE.TEm= 0) and re-enabled (TAUJnTS.TSm = 1).

The table also contains the counter read value one count after the counter is enabled (TAUJnTS.TSm = 1) for modes where the counter waits for a start trigger.

Table 18-63 TAUJnCNTm read values after the counter is re-enabled

Mode name	Count method (up/down)	TAUJnCNTm value		
		After reset	After stop trigger	After one count
Interval Timer mode	Count down	FFFF $FFFF_H$	Stop value	-
Capture mode	Count up	0000 0000_H	Stop value	-
One Count mode	Count down	FFFF $FFFF_H$	Stop value	FFFF $FFFF_H$
Capture & One Count mode	Count up	0000 0000_H	Stop value	Captured value + 1 (TAUJnCDRm)
Count Capture Mode	Count up	0000 0000_H	Stop value	-
Gate Count Mode	Count down	FFFF $FFFF_H$	Stop value	Stop value
Capture & Gate Count Mode	Count up	0000 0000_H	Stop value	Stop value

Note If the operation mode is changed while the counter is stopped, the initial counter value after counter restart is undefined. The operation mode is changed by register TAUJnCMORm.MD[4:0].

(3) TAUJnCMORm - TAUJn channel mode OS register

This register controls channel m operation. It specifies, for example, the operation clock, count clock, and master/slave function.

Access This register can be read/written in 16-bit units. It can only be written when the counter is stopped (TAUJnTE.TEm = 0).

Address <TAUJn_base> + 80_H + m x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS[1:0]		CCS[1:0]		MAS	STS[2:0]			COS[1:0]		-	MD[4:0]				
R/W		R/W		R/W	R/W			R/W		R	R/W				

Table 18-64 TAUJnCMORm register contents (1/3)

Bit position	Bit name	Function
15,14	CKS[1:0]	Selects the operation clock. The operation clock is used for the TAUJnTTINm input edge detection circuit. It can also be used as the count clock depending on bits TAUJnCMORm.CCS[1:0]. **CKS1 / CKS0 / Selected operation clock** 0 / 0 / CK0 0 / 1 / CK1 1 / 0 / CK2 1 / 1 / CK3
13,12	CCS[1:0]	Selects the count clock for TAUJnCNTm counter: **CCS1 / CCS0 / Selected count clock** 0 / 0 / Operation clock as specified by TAUJnCMORm.CKS[1:0]. 0 / 1 / Valid edge of TAUJnTTINm input signal 1 / 0 / Setting prohibited 1 / 1 / Setting prohibited
11	MAS	Specifies the channel as master or slave channel during synchronous channel operation: 0: Slave 1: Master This bit is only valid for even channels (CHm_even). For odd channels (CHm_odd), it is fixed to 0.

Table 18-64 TAUJnCMORm register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>10 to 8</td><td>STS[2:0]</td><td>Selects the external start trigger:
<table>
<tr><th>STS2</th><th>STS1</th><th>STS0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Software trigger.</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Valid edge of the TAUJnTTINm input signal. TAUJnCMURm.TIS[1:0] specifies the valid edge.</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Valid edge of the TAUJnTTINm input signal is the start trigger and the reverse edge is the stop (capture) trigger.</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>INTTAUJnI of the master channel.</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td rowspan="3">Setting prohibited</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table></td></tr>
<tr><td>7, 6</td><td>COS[1:0]</td><td>Specifies when the capture register TAUJnCDRm and the overflow flag TAUJnCSRm.OVF of channel m are updated.
These bits are only valid if channel m is in capture mode.
<table>
<tr><th>COS1</th><th>COS0</th><th>Capture register</th><th>TAUJnCSRm.OVF</th></tr>
<tr><td>0</td><td>0</td><td rowspan="2">Updated upon detection of a TAUJnTTINm input valid edge.</td><td>Updated (cleared or set) upon detection of a TAUJnTTINm input valid edge:
• If a counter overflow has occurred since the last valid edge detection, TAUJnCSRm.OVF is set.
• If no counter overflow has occurred since the last valid edge detection, TAUJnCSRm.OVF is cleared.</td></tr>
<tr><td>0</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
<tr><td>1</td><td>0</td><td rowspan="2">Updated upon detection of a TAUJnTTINm input valid edge and upon counter overflow:
• TAUJnTTINm input valid edge: Counter value is written to TAUJnCDRm.
• Overflow: FFFF FFFF$_H$ is written to TAUJnCDRm. The next TAUJnTTINm input valid edge detection is ignored.</td><td>Not set.</td></tr>
<tr><td>1</td><td>1</td><td>Set upon counter overflow and cleared by a CPU instruction.</td></tr>
</table></td></tr>
</table>

Table 18-64 TAUJnCMORm register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>4 to 0</td><td>MD[4:0]</td><td>Specifies the operation mode.
<table>
<tr><th>MD4</th><th>MD3</th><th>MD2</th><th>MD1</th><th>MD0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>1/0</td><td>Interval Timer mode</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1/0</td><td>Capture mode</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1/0</td><td>One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>1/0</td><td>Setting prohibited</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Capture & One Count mode</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td rowspan="4">1/0</td><td rowspan="4">Setting prohibited</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>1/0</td><td>Count Capture mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>Gate Count mode</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Capture & Gate Count mode</td></tr>
</table>
<table>
<tr><th>Mode</th><th>Role of the MD0 bit</th></tr>
<tr><td>Interval Timer mode
Capture mode
Count Capture mode</td><td>Specifies whether an INTTAUJnIm is generated when the counter is triggered:
0: No INTTAUJnIm generated
1: INTTAUJnIm generated</td></tr>
<tr><td>One Count mode
Gate Count mode</td><td>Enables/disables start trigger detection during counting:
0: Disabled
1: Enabled
INTTAUJnIm and TAUJnTTOUTm are not output when the counter is triggered.</td></tr>
<tr><td>Capture & One Count mode
Capture & Gate Count mode</td><td>This bit must be set to 0:
0: No INTTAUJnIm generated when the counter is triggered. Start trigger is disabled during counting.
1: -</td></tr>
</table>
</td></tr>
</table>

(4) TAUJnCMURm - TAUJn channel mode user register

This register specifies the type of valid edge detection used for the TAUJnTTINm input.

Access This register can be read/written in 8-bit units. It can only be written when the counter is enabled (TAUJnTE.TEm = 1).

Address <TAUJn_base> + 20_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	TIS[1:0]	
R	R	R	R	R	R	R/W	R/W

Table 18-65 TAUJnCMURm register contents

Bit position	Bit name	Function
1, 0	TIS[1:0]	Specifies the valid edge of the TAUJnTTINm input: • To detect rising and falling edges when TAUJnCMORm.STS[2:0] is not set to 010_B, set TAUJnCMURm.TIS[1:0] = 10_B. • Edge detection for TAUJnTTINm input signals is performed based on the operation clock selected by TAUJnCMORm.CKS[1:0].

TIS1	TIS0	Description
0	0	Falling edge
0	1	Rising edge
1	0	Rising and falling edges (low-width measurement selection). Start trigger: falling edge Stop trigger (capture): rising edge
1	1	Rising and falling edges (high-width measurement selection). Start trigger: rising edge Stop trigger (capture): falling edge This setting is only valid if the start trigger selection is set to TAUJnCMORm.STS[2:0] = 010_B

(5) TAUJnCSRm - TAUJn channel status register

This register indicates the overflow status of channel m's counter.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + 30_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	OVF
R	R	R	R	R	R	R	R

Table 18-66 TAUJnCSRm register contents

Bit position	Bit name	Function
0	OVF	Indicates the counter overflow status: 0: No overflow occurred 1: Overflow occurred This bit is only used in the following modes: • Capture mode • Capture & One Count mode • Count Capture mode • Capture & Gate Count mode The function of this bit depends on the setting of control bits TAUJnCMORm.COS[1:0]. OVF is not be set when TAUJnCMORm.COS[1:0] = 10_B.

(6) TAUJnCSCm - TAUJn channel status clear register

This register is a trigger register for clearing the overflow flag TAUJnCSRm.OVF of a channel m.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUJn_base> + 40_H + m x 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	-	-	0	CLOV
R	R	R	R	R	R	R	W

Table 18-67 TAUJnCSCm register contents

Bit position	Bit name	Function
0	CLOV	0: No function 1: Clears the overflow flag TAUJnCSRm.OVF

(7) TAUJnTS - TAUJn channel start trigger register

This register enables the counter for each channel.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUJn_base> + 54_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TS03	TS02	TS01	TS00
W	W	W	W	W	W	W	W

Table 18-68 TAUJnTS register contents

Bit position	Bit name	Function
3 to 0	TSm	Enables the counter for channel m: 0: No function 1: Enables the counter and sets TAUJnTE.TEm = 1. When the counter is enabled, this bit immediately returns to 0. TAUJnTE.TEm = 1 only *enables* counter. Whether the counter *starts* depends on the selected operation mode.

(8) TAUJnTE - TAUJn channel enable status register

This register indicates whether counter is enabled or disabled.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + 50_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TE03	TE02	TE01	TE00
R	R	R	R	R	R	R	R

Table 18-69 TAUJnTE register contents

Bit position	Bit name	Function
3 to 0	TEm	Indicates whether counter for channel m is enabled or disabled: 0: Counter disabled 1: Counter enabled Setting TAUJnTS.TSm to 1 or trigger input detection TAUJnTSSTm = 1 sets this bit to 1. Setting TAUJnTT.TTm to 1 resets this bit to 0.

(9) TAUJnTT - TAUJn channel stop trigger register

This register stops the counter for each channel.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUJn_base> + 58_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TT03	TT02	TT01	TT00
W	W	W	W	W	W	W	W

Table 18-70 TAUJnTT register contents

Bit position	Bit name	Function
3 to 0	TTm	Stops the counter of channel m: 0: No function 1: Stops the counter and sets TAUJnTE.TEm = 0. When the counter has stopped, this bit immediately returns to 0. TAUJnCNTm stops counting and TAUJnCNTm, TAUJnTO.TOm, and TAUJnTTOUTm all retain the values they had before the counter was stopped.

18.19.4 TAUJn output registers details

(1) TAUJnTOE - TAUJn channel output enable register

This register enables and disables Direct Channel Output Mode.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 60_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TOE03	TOE02	TOE01	TOE00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-71 TAUJnTOE register contents

Bit position	Bit name	Function
3 to 0	TOEm	Enables/disables Direct Channel Output Mode: 0: Enables Direct Channel Output Mode (TAUJnTTOUT m output is controlled by the application software) 1: Disables Direct Channel Output Mode (TAUJnTTOUT m output is controlled by the timer)

(2) TAUJnTO - TAUJn channel output register

This register specifies and reads the level of TAUJnTTOUTm.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + $5C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TO03	TO02	TO01	TO00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-72 TAUJnTO register contents

Bit position	Bit name	Function
3 to 0	TOm	Specifies/reads the level of TAUJnTTOUTm: 0: Low 1: High Only TOm bits for which Independent Channel Output function is enabled (TAUJnTOEm = 0) can be written.

(3) TAUJnTOM - TAUJn channel output mode register

This register specifies the output mode of each channel.

Access This register can be read/written in 8-bit units. It can only be written when the counter is stopped (TAUJnTE.TEm = 0).

Address <TAUJn_base> + 98_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TOM 03	TOM 02	TOM 01	TOM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-73 TAUJnTOM register contents

Bit position	Bit name	Function
3 to 0	TOMm	Specifies the timer controlled channel output mode, if Direct Channel Output Mode is disabled (TAUJnTOE.TOEm = 1): 0: Independent Channel Output Mode 1: Synchronous Channel Output Mode The output mode depends on several channel output control bits, as can be seen in *Table 18-12 “Channel output modes” on page 1422*

(4) TAUJnTOC - TAUJn channel output configuration register

This register specifies the output mode of each channel in combination with TAUJnTOMm.

Access This register can be read/written in 8-bit units. It can only be written when the counter is stopped (TAUJnTE.TEm = 0).

Address <TAUJn_base> + $9C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TOC 03	TOC 02	TOC 01	TOC 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-74 TAUJnTOC register contents

Bit position	Bit name	Function
3 to 0	TOCm	Specifies the output mode: 0: Operation mode 1 (= Toggle mode) 1: No function This bit must be set to 0 for all output modes except Direct Channel Output Mode.

(5) TAUJnTOL - TAUJn channel output level register

This register specifies the output logic of the channel output bit (TAUJnTO.TOm).

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + 64_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	TOL 03	TOL 02	TOL 01	TOL 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-75 TAUJnTOL register contents

Bit position	Bit name	Function
3 to 0	TOLm	Specifies the output logic of the channel m output bit (TAUJnTO.TOm): 0: Positive logic (active high) 1: Inverted logic (active low) These bits apply in all channel output modes except Direct Channel Output Mode.

18.19.5 TAUJn simultaneous rewrite register details

(1) TAUJnRDE - TAUJn channel reload data enable register

This register enables and disables simultaneous rewrite of the data register TAUJnCDRm. It also enables simultaneous rewrite of the data register TAUJnTOLm for the PWM output function and the triangle PWM output function.

Access This register can be read/written in 8-bit units. It can only be written when TAUJnTE.TEm = 0.

Address <TAUJn_base> + $A0_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	RDE 03	RDE 02	RDE 01	RDE 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-76 TAUJnRDE register contents

Bit position	Bit name	Function
3 to 0	RDEm	Enables/disables simultaneous rewrite of the data register of channel m: 0: Disables simultaneous rewrite 1: Enabled simultaneous rewrite

(2) TAUJnRDM - TAUJn channel reload data mode register

This register selects when the signal that controls simultaneous rewrite is loaded.

Access This register can be read/written in 8-bit units. It can only be written when TAUJnTE.TEm = 0.

Address <TAUJn_base> + $A4_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
				RDM 03	RDM 02	RDM 01	RDM 00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-77 TAUJnRDM register contents

Bit position	Bit name	Function
3 to 0	RDMm	Specifies when the signal that triggers simultaneous rewrite is generated: 0: When the master channel counter starts counting 1: No function These bits only apply when TAUJnRDE.RDEm = 1.

(3) TAUJnRDT - TAUJn channel reload data trigger register

This register triggers the simultaneous rewrite pending state.

Access This register can be written in 8-bit units. It is always read as 0000_H.

Address <TAUJn_base> + 68_H

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	RDT 03	RDT 02	RDT 01	RDT 00
W	W	W	W	W	W	W	W

Table 18-78 TAUJnRDT register contents

Bit position	Bit name	Function
3 to 0	RDTm	Triggers the simultaneous rewrite pending state: 0: No function 1: Simultaneous rewrite pending state is triggered. The simultaneous rewrite pending flag (TAUJnRSFm) is set to 1. The system waits for the simultaneous rewrite trigger. TAUJnRDT.RDTm immediately returns to 0.

(4) TAUJnRSF - TAUJn channel reload status register

This flag register indicates that simultaneous rewrite is possible.

Access This register can be read in 8-bit units.

Address <TAUJn_base> + $6C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
-	-	-	-	RSF 03	RSF 02	RSF 01	RSF 00
R	R	R	R	R	R	R	R

Table 18-79 TAUJnRSF register contents

Bit position	Bit name	Function
3 to 0	RSFm	Indicates the simultaneous rewrite status: 0: Simultaneous rewrite disabled 1: Simultaneous rewrite enabled

18.19.6 TAUJn emulation register

(1) TAUJnEMU - TAUJ emulation register

This register controls whether the TAUJn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <TAUJn_base> + $A8_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
TAUJn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 18-80 TAUJnEMU register contents

Bit position	Bit name	Function
7	TAUJn SVSDIS	Emulation control 0: TAUJn can be stopped during emulation 1: TAUJn continuous operating during emulation

Chapter 19 Real-Time Clock (RTCA)

This chapter contains a generic description of the Real-Time Clock (RTCA).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/ output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

19.1 V850E2/Fx4-H RTCA Features

Instances

This microcontroller has following number of instances of the Real-Time Clock A.

Table 19-1 Instances of RTCA

Real-Time Clock A	
Instances	1
Names	RTCA0

Instances index n

Throughout this chapter, the individual instances of a Real-Time Clock A are identified by “n” (n = 0), for example, RTCAnCTL0 for the RTCAn control register 0.

Register addresses

All RTCAn register addresses are given as address offsets to the individual base address <RTCAn_base>.
The base address <RTCAn_base> of each RTCAn is listed in the following table:

Table 19-2 Register base addresses <RTCAn_base>

RTCAn instance	<RTCAn_base> address
RTCA0	FF81 4000_H

Clock supply All Real-Time Clocks A provide two clock inputs:

Table 19-3 RTCAn clock supply

RTCAn instance	RTCAn clock	Connected to
RTCA0	RTCATCKI	Clock Controller CKSCLK_A09
	PCLK	Clock Controller CKSCLK_A02

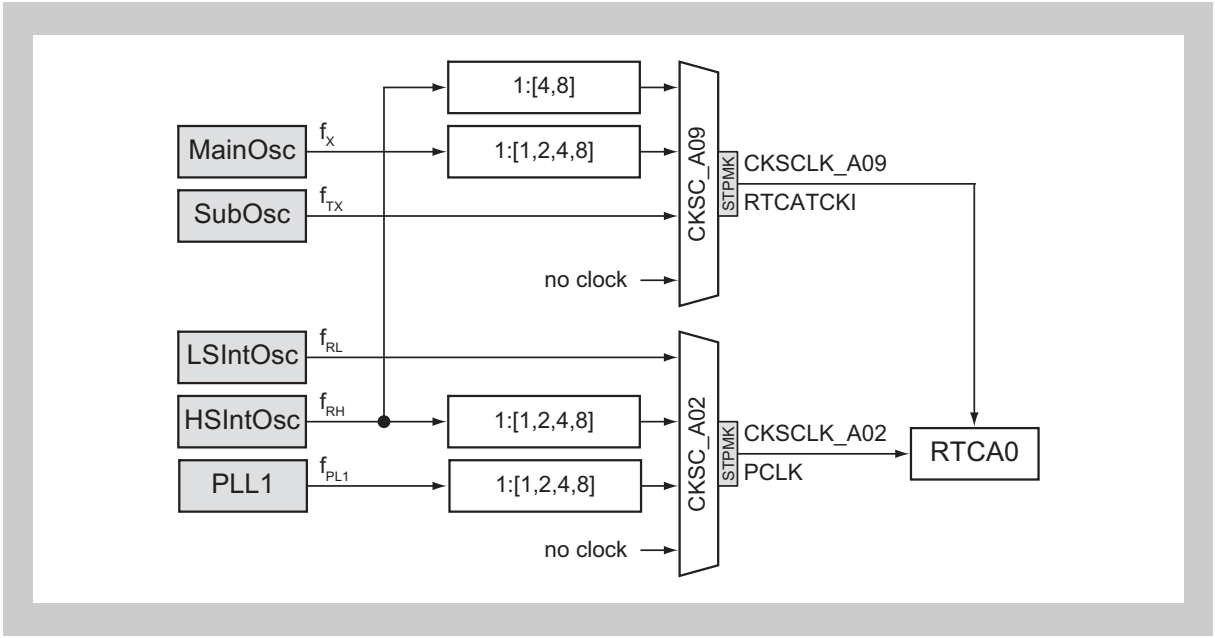


Figure 19-1 RTCA clock supply

Interrupts: The Real-Time Clock A can generate the following interrupt requests:

Table 19-4 RTCAn interrupt requests

RTCAn signals	Function	Connected to
RTCATINT1S	1 second interval interrupt	Interrupt Controller INTRTCA01S[a]
RTCATINTAL	Alarm interrupt	Interrupt Controller INTRTCA0AL[a]
RTCATINTR	Fixed interval interrupt	Interrupt Controller INTRTCA0R[a]

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

RTCA H/W reset The Real-Time Clock A and their registers are initialized by the following reset signal:

Table 19-5 RTCAn reset signal

RTCAn	Reset signal
RTCA0	• Reset Controller: SYSRES

I/O signals

The I/O signals of Real-Time Clock are listed in the following table:

Table 19-6 RTCAn I/O signals

RTCAn signals	Function	Connected to
RTCAT1HZ	1 second interval output	Port RTCA0OUT

Clock correction

For clock correction purposes the 1 second interval output RTCA1HZ can be internally connected to a capture input of the Timer Array Units J. Refer to *18.2 “TAUJ Input Selection” on page 1400* for details.

19.2 Functional Overview

Features summary

The Real-Time Clock RTCA has the following features:

- Count clock selection from 32 KHz to 4.194304 MHz
- Counters for years, months, day of the month, day of the week, hours, minutes, seconds, and a sub-counter

  The calendar covers 99 years. Leap years are handled by hardware automatically.
- One Hz pulse output function
- Fixed interval interrupt function
- Alarm interrupt function
- Clock error correction function if a 32.768 KHz count clock is used

The block diagram shows the main components of the RTCA.

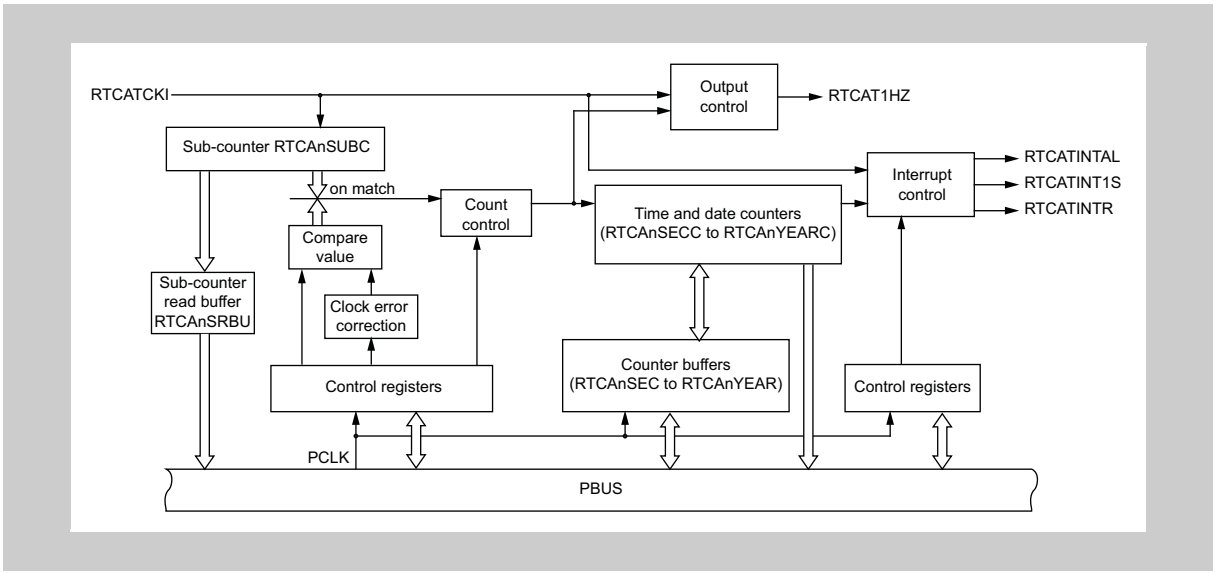


Figure 19-2 Block diagram of the RTCA

19.3 Functional Description

The Real-Time Clock RTCA provides information about the present time and date and can generate wake-up signals (interrupts, alarms). This information is derived from the count clock RTCATCKI.

Sub-counter

RTCATCKI is the input to the sub-counter RTCAnSUBC. The sub-counter counts up from zero until it reaches the compare value. The compare value is always defined as the frequency of RTCATCKI – 1 (in Hz). Thus, the sub-counter overflows after one second. It is then reset to zero and triggers the seconds counter RTCAnSECC (and, if desired, the interrupt RTCATINT1S).

The sub-counter can generate a fixed interval interrupt every 0.25 seconds, 0.5 seconds, or 1 second, and a 1 Hz output pulse.

Time and date counters

The counters for minutes, hours, day of the week, day of the month, months, and years also count up. They have their own overflow limits. One triggers the next one.

The overflow limit of the counter for the day of the month (RTCAnDAYC) depends on the present month (28, 30, or 31 days) and (in February) on the year counter RTCAnYEARC (years 0, 4, 8, 12, etc. are considered leap years).

The hours counter RTCAnHOURC can be switched between 12 and 24 hour format.

The counters for seconds, minutes, hours, day of the month, and month can generate a fixed interval interrupt upon overflow (RTCATINTR).

The counters for minutes, hours, and day of the week can also generate an alarm interrupt, e.g. every Tuesday and Thursday at 10:32 (RTCATINTAL).

Counter buffers

All counters can be read directly at any time. The clock signal used to access the read/write registers and the count clock are usually asychronous. An overflow of the sub-counter during the read operation can make all read values obsolete. Therefore, reading the counters must be performed using a special procedure. For details, refer to *19.5.3 “Reading clock counters” on page 1567* .

For reasons of synchronization, the counters cannot be written directly.

For reading *and* writing, all counters are accompanied by buffer registers. The buffer registers provide a synchronized way for reading the counters and for setting time and date. When they are used, the operation of the sub-counter must first be suspended and then re-activated (see also *19.5.3 “Reading clock counters” on page 1567* and *19.5.2 “Updating clock counters” on page 1566* .

The RTCAnTIMEC and RTCAnCALC registers and their corresponding buffer registers can be used to access the time (hours, minutes and seconds) or the date (day of the week, day of the month, month, and year) with one read/write operation.

19.3.1 Operation modes

The RTCA provides two operation modes:

- Frequency selection mode
- 32.768 KHz mode

The operation mode that can be used depends on the available input clock RTCATCKI. The operation mode specifies the sub-counter compare value that is used to trigger the seconds counter and thus all subsequent counters. Clock error correction is only possible in 32.768 KHz mode.

The following table provides an overview of the properties of the two operation modes.

Table 19-7 RTCA operation mode overview

	Frequency selection mode	32.768 KHz mode	
		Clock correction disabled	Clock correction enabled
Allowed input clock RTCATCKI	Any frequency from 32 KHz to 4.194304 MHz	32.768 KHz	Any frequency from 32.76180000 KHz to 32.77420000 KHz
Sub-counter RTCAnSUBC operation	• Counter overflow at value of RTCAnSCMP • RTCAnSCMP must be set to RTCATCKI - 1 (in Hz)	Counter overflow at $7FFF_H$	Counter overflow at • $7FFF_H$ or • Every 20 or 60 seconds: $7FFF_H \pm$ RTCAnSUBU.RTCAnF[5:0]

The operation mode is selected by control bit RTCAnCTL0.RTCAnSLSB. For details on how to set the operation mode during RTCA initialization, refer to *19.5.1 "Initial setting of the RTCA" on page 1564* .

Cautions

1. The input clock RTCATCKI must not be outside the allowed frequency range.
2. The operation mode must not be changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1).

19.3.2 Clock counter format

The clock counters (RTCAnSECC to RTCAnYEARC) operate on binary coded decimals (BCD): Each digit is represented by its own binary sequence.

Depending on the valid data range, the number of bits for a digit differs. For example, the tens digit of the month of the year counter has only one bit (for 0 and 1) whereas the tens digit of the minutes counter has 3 digits (for 0 to 5).

The following table lists the decimals 0 to 59 in binary and BCD.

Table 19-8 Example of BCD Code – seconds or minutes counter (0 to 59)

Decimal	Binary	BCD
0	000000	000 0000
1	000001	000 0001

Table 19-8 Example of BCD Code – seconds or minutes counter (0 to 59)

Decimal	Binary	BCD
2	000010	000 0010
3	000011	000 0011
4	000100	000 0100
5	000101	000 0101
6	000110	000 0110
7	000111	000 0111
8	001000	000 1000
9	001001	000 1001
10	001010	001 0000
11	001011	001 0001
12	001100	001 0010
:	:	:
58	111010	101 1000
59	111011	101 1001

19.3.3 Fixed interval interrupt function

Interrupt RTCATINTR can be specified to occur after every 0.25 seconds, 0.5 seconds, 1 (full) second, 1 (full) minute, 1 (full) hour, or 1 (full) month.

The fixed interval interrupt function is controlled by bits RTCAnCTL1.RTCAnCT[2:0].

19.3.4 Alarm interrupt function

Interrupt RTCATINTAL can be specified to occur at a certain time on one or several days of the week. This interrupt can be used as a wake-up signal.

The alarm interrupt function is enabled by bit RTCAnCTL1.RTCAnALME.

The alarm setting is specified by the following control registers:

- RTCAnALW selects the weekday(s)

  The allocation of bits to weekdays is defined by the day of the week counter RTCAnWEEKC.

- RTCAnALH and RTCAnALM specify the hour and minute in BCD

Examples The following tables show some examplary settings of the alarm control registers for both 12 hour and 24 hour format.

In this example, Sunday is RTCAnWEEK = 0, Monday is RTCAnWEEK = 1, Tuesday is RTCAnWEEK = 2, …, Saturday is RTCAnWEEK = 6:

Table 19-9 Alarm setting in 12-hour format (RTCAnCTL0.RTCAnAMPM = 0)

Alarm setting time	RTCAnALW	RTCAnALH	RTCAnALM
Sunday 7:00 am	01_H	07_H	00_H
Sunday, Monday 12:15 pm	03_H	32_H	15_H
Monday, Wednesday, Friday 5:30 pm	26_H	25_H	30_H
Daily, 10:45 pm	$7F_H$	30_H	45_H

Table 19-10 Alarm setting in 24-hour format (RTCAnCTL0.RTCAnAMPM = 1)

Alarm setting time	RTCAnALW	RTCAnALH	RTCAnALM
Sunday 7:00	01_H	07_H	00_H
Sunday, Monday 12:15	03_H	12_H	15_H
Monday, Wednesday, Friday 17:30	26_H	17_H	30_H
Daily, 22:45	$7F_H$	22_H	45_H

19.3.5 Clock error correction

Clock error correction compensates for deviations of the oscillator from the nominal clock rate. With clock error correction input clock rates from 32.76180 KHz to 32.77420 KHz are possible.

The clock error correction function is only available in 32.768 KHz operation mode. In this operation mode, a nominal clock rate of 32.768 KHz is expected and the sub-counters overflow value is fixed to $7FFF_H$.

The following figures illustrate the clock error when the input clock rate deviates from the nominal clock.

RTCATCKI = 32.768 KHz

Figure 19-3 "RTCATCKI = 32.768 KHz, no clock error correction required" shows the timing diagram if RTCATCKI matches the nominal clock rate of 32.768 KHz. No clock error correction is required.

Counting from 0 to 32767 (0 to $7FFF_H$) with a 32.768 KHz clock is equal to 1 second exactly.

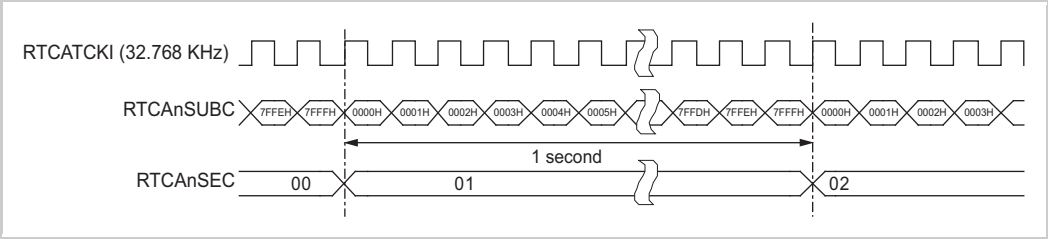


Figure 19-3 RTCATCKI = 32.768 KHz, no clock error correction required

RTCATCKI = 32.769 KHz

Figure 19-4 "RTCATCKI = 32.769 KHz, no clock error correction enabled" shows the timing diagram if RTCATCKI deviates from the nominal clock rate of 32.768 KHz. In this example, RTCATCKI is connected to a 32.769 KHz oscillator. Clock error correction is not enabled.

Counting from 0 to 32767 (0 to $7FFF_H$) with a 32.769 KHz clock is equal to approximately 0.99997 seconds (32768/32769). A "+ error" (faster than 32.768 KHz) occurs. In 1 month, RTCA deviates approximately -79 seconds from the real time.

Error = (32768/ 32769 - 1) x 60 (s) x 60 (min) x 24 (h) x 30 (d)

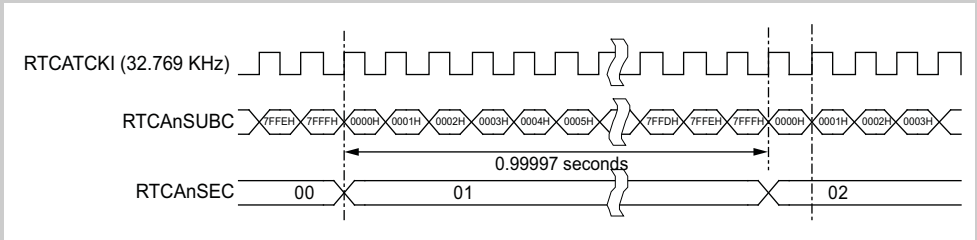


Figure 19-4 RTCATCKI = 32.769 KHz, no clock error correction enabled

Clock error correction is performed by stretching/reducing the 1 second period of the sub-counter in regular intervals. The sub-counter's upper limit of $7FFF_H$ is increased or decreased by setting the following parameters in register RTCAnSUBU:

- A correction value greater than one
- An operator (add/subtract)
- A repetition rate (20 or 60 seconds)

The corrected overflow value becomes effective every 20 or 60 seconds, so that on the average RTCAnSECC is triggered exactly every second.

RTCA1HZ output

The Real-Time Clock module generates the one second interval signal RTCA1HZ with a duty cycle of 50 %. RTCA1HZ can be used as a capture input to a timer, that operates with a precise clock and thus counts the exact number of precise clock cycles within a RTCA1HZ period. The correction value can be derived from the number of measured clock cycles.
Refer to the first section of this chapter and the key word "Clock correction" for details about the timer that can be used for clock correction value measurement.

(1) Setting the correction value and the operator

The correction value and operator are specified by the RTCAnF[6:0] bits of the RTCAnSUBU register:

- RTCAnF[6] specifies whether the overflow value is incremented or decremented
- RTCAnF[5:0] specifies the correction value

The correction values are calculated as follows:

Table 19-11 Correction value settings

RTCAnF[6]	Increment/decrement	Correction value
0	Increment	(Value of RTCAnF[5:0] – 1) x 2
1	Decrement	(Inverted value of RTCAnF[5:0] + 1) x 2

Some examples are given in the following table:

Table 19-12 Correction value examples

RTCAnF[6]	RTCAnF[5:0]	Correction value	Count limit of RTCAnSUBC
0	15_H	$(15_H - 1) \times 2 = 40$	32768 + 40 = 32808
1	15_H	$(\overline{15}_H - 1) \times 2$ $= (2A_H + 1) \times 2$ $= 86$	32768 – 86 = 32682

(2) Impact of the repetition rate

The correction value set by RTCAnF[6:0] does not change the count limit of RTCAnSUBC every second. The repetition rate at which the correction value becomes effective is specified by bit RTCAnDEV.

This bit also influences the size of the correctable frequency range and the correction accuracy.

The following table summarizes the RTCAnDEV settings.

Table 19-13 Setting of bit RTCAnSUBU.RTCAnDEV

RTCAnDEV	Count limit of RTCAnSUBC is changed	Frequency range that can be corrected	Correction accuracy
0	Every 20 seconds when RTCAnSECC = 00, 20, or 40	32.76180000 to 32.77420000 KHz	
1	Every 60 seconds when RTCAnSECC = 00	32.76593333 to 32.77006667 KHz	Three times higher than for RTCAnDEV = 0

(3) Sample settings

The frequencies that can be corrected, as well as the setting values of bits RTCAnDEV and RTCAnF[6:0], are listed in the following table.

Table 19-14 Correctable frequency range when RTCAnDEV = 0 (1/2)

Input clock frequency	RTCAnF6	RTCAnF[5:0]	Correction value
–	0	000000	No correction
–	0	000001	No correction
32.76810000 KHz	0	000010	Once every 20 s, RTCAnSUBC count value + 2
32.76820000 KHz	0	000011	Once every 20 s, RTCAnSUBC count value + 4
32.76830000 KHz	0	000100	Once every 20 s, RTCAnSUBC count value + 6
...	...	...	...
32.77400000 KHz	0	111011	Once every 20 s, RTCAnSUBC count value + 120
32.77410000 KHz	0	111110	Once every 20 s, RTCAnSUBC count value + 122
32.77420000 KHz (upper limit)	0	111111	Once every 20 s, RTCAnSUBC count value + 124
–	1	000000	No correction
–	1	000001	No correction
32.76180000 KHz (lower limit)	1	000010	Once every 20 s, RTCAnSUBC count value – 124
32.76190000 KHz	1	000011	Once every 20 s, RTCAnSUBC count value – 122
32.76200000 KHz	1	000100	Once every 20 s, RTCAnSUBC count value – 120
...	...	...	...

Table 19-14 Correctable frequency range when RTCAnDEV = 0 (2/2)

Input clock frequency	RTCAnF6	RTCAnF[5:0]	Correction value
32.76770000 Hz	1	11011	Once every 20 s, RTCAnSUBC count value – 6
32.76780000 KHz	1	11110	Once every 20 s, RTCAnSUBC count value – 4
32.76790000 KHz	1	11111	Once every 20 s, RTCAnSUBC count value – 2

Table 19-15 Correctable frequency range when RTCAnDEV = 1

Input clock frequency	RTCAnF6	RTCAnF[5:0]	RTCAnSUBC correction value
–	0	000000	No correction
–	0	000001	No correction
32.76803333 KHz	0	000010	Once every 60 s, RTCAnSUBC count value + 2
32.76806667 KHz	0	000011	Once every 60 s, RTCAnSUBC count value + 4
32.76810000 KHz	0	000100	Once every 60 s, RTCAnSUBC count value + 6
...	...	...	...
32.77000000 KHz	0	111011	Once every 60 s, RTCAnSUBC count value + 120
32.77003333 KHz	0	111110	Once every 60 s, RTCAnSUBC count value + 122
32.77006667 KHz (upper limit)	0	111111	Once every 60 s, RTCAnSUBC count value + 124
–	1	000000	No correction
–	1	000001	No correction
32.76593333 KHz (lower limit)	1	000010	Once every 60 s, RTCAnSUBC count value – 124
32.76596667 KHz	1	000011	Once every 60 s, RTCAnSUBC count value – 122
32.76600000 KHz	1	000100	Once every 60 s, RTCAnSUBC count value – 120
...	...	...	...
32.76790000 KHz	1	11011	Once every 60 s, RTCAnSUBC count value – 6
32.76793333 KHz	1	11110	Once every 60 s, RTCAnSUBC count value – 4
32.76796667 KHz	1	11111	Once every 60 s, RTCAnSUBC count value – 2

19.4 Registers

This section contains a description of all registers of the RTCA.

19.4.1 RTCA registers overview

The RTCA is controlled and operated by the following registers:

Table 19-16 RTCA register overview (1/2)

Register Name	Shortcut	Address
Control registers		
Control register 0	RTCAnCTL0	<RTCAn_base> + 00_H
Control register 1	RTCAnCTL1	<RTCAn_base> + 04_H
Control register 2	RTCAnCTL2	<RTCAn_base> + 08_H
Sub-counter registers		
Sub-count register	RTCAnSUBC	<RTCAn_base> + $0C_H$
Sub-count register read buffer	RTCAnSRBU	<RTCAn_base> + 10_H
Sub-counter compare register	RTCAnSCMP	<RTCAn_base> + $3C_H$
Clock error correction register	RTCAnSUBU	<RTCAn_base> + 38_H
Clock counter and buffer registers		
Seconds count register	RTCAnSECC	<RTCAn_base> + $4C_H$
Seconds count buffer register	RTCAnSEC	<RTCAn_base> + 14_H
Minute count register	RTCAnMINC	<RTCAn_base> + 50_H
Minute count buffer register	RTCAnMIN	<RTCAn_base> + 18_H
Hour count register	RTCAnHOURC	<RTCAn_base> + 54_H
Hour count buffer register	RTCAnHOUR	<RTCAn_base> + $1C_H$
Day of the week count register	RTCAnWEEKC	<RTCAn_base> + 58_H
Day of the week count buffer register	RTCAnWEEK	<RTCAn_base> + 20_H
Day count register	RTCAnDAYC	<RTCAn_base> + $5C_H$
Day count buffer register	RTCAnDAY	<RTCAn_base> + 24_H
Month count register	RTCAnMONC	<RTCAn_base> + 60_H
Month count buffer register	RTCAnMONTH	<RTCAn_base> + 28_H
Year count register	RTCAnYEARC	<RTCAn_base> + 64_H
Year count buffer register	RTCAnYEAR	<RTCAn_base> + $2C_H$
Special counter and buffer registers		
Time count register	RTCAnTIMEC	<RTCAn_base> + 68_H
Time count buffer register	RTCAnTIME	<RTCAn_base> + 30_H
Calendar count register	RTCAnCALC	<RTCAn_base> + $6C_H$
Calendar count buffer register	RTCAnCAL	<RTCAn_base> + 34_H

Table 19-16 RTCA register overview (2/2)

Register Name	Shortcut	Address
Alarm setting registers		
Alarm minute setting register	RTCAnALM	<RTCAn_base> + 40_H
Alarm hour setting register	RTCAnALH	<RTCAn_base> + 44_H
Alarm day of the week setting register	RTCAnALW	<RTCAn_base> + 48_H
Emulation register		
Emulation register	RTCAnEMU	<RTCAn_base> + 74_H

<RTCAn_base> The base addresses <RTCAn_base> of the RTCAn is defined in the first section of this chapter under the key word “Register addresses”.

19.4.2 RTCA control registers details

(1) RTCAnCTL0 – RTCA control register 0

This register controls the count operation of the sub-counter RTCAnSUBC, the format of the hours counter RTCAnHOURC and the alarm hour setting register RTCAnALH, and the operation mode.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base>

Initial Value 00_H

7	6	5	4	3	2	1	0
RTCAnCE	RTCAnCEST	RTCAnAMPM	RTCAnSLSB	0	0	0	0
R/W	R	R/W	R/W	R	R	R	R

Table 19-17 RTCAnCTL0 register contents

Bit position	Bit name	Function
7	RTCAnCE	Starts/stops the sub-counter RTCAnSUBC operation. 0: Stops the sub-counter operation All output pins and all status flags in control register RTCAnCTL2 are cleared. 1: Starts the sub-counter operation The sub-counter counts up.
6	RTCAnCEST	Indicates whether the sub-counter is enabled: 0: Sub-counter operation disabled 1: Sub-counter operation enabled For details on how to use this status flag, refer to *19.5.1 "Initial setting of the RTCA" on page 1564* .
5	RTCAnAMPM	Selects the format of the hours counter RTCAnHOURC and the alarm hour setting register RTCAnALH: 0: 12 hour format (1 to 12, am/pm) 1: 24 hour format (0 to 23, military time) For details on the format, refer to the description of the hours counter *6 "RTCAnHOUR – RTCA hour count buffer register"*.
4	RTCAnSLSB	Selects the operation mode: 0: 32.768 KHz mode 1: Frequency selection mode For details on the operation modes, refer to *19.3.1 "Operation modes" on page 1524* . The operation mode must not be changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). For details on the initialization of RTCAn, refer to *19.5.1 "Initial setting of the RTCA" on page 1564* .

(2) RTCAnCTL1 – RTCA control register 1

This register controls the interrupt request generation and the 1 Hz pulse output.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 04_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	RTCAn1HZE	RTCAnALME	RTCAn1SE	RTCAnCT2	RTCAnCT1	RTCAnCT0
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-18 RTCAnCTL1 register contents

Bit position	Bit name	Function
5	RTCAn1HZE	Enables/stops 1 Hz pulse output (RTCAT1HZ): 0: RTCAT1HZ disabled (RTCAT1HZ is fixed to 0) 1: RTCAT1HZ enabled
4	RTCAnALME	Enables/disables alarm interrupt request generation (RTCATINTAL): 0: RTCATINTAL disabled 1: RTCATINTAL enabled
3	RTCAn1SE	Enables/disables 1 second interrupt request generation (RTCATINT1S): 0: RTCATINT1S disabled 1: RTCATINT1S enabled
2 to 0	RTCAnCT[2:0]	Specifies the fixed interval interrupt request (RTCATINTR) setting:

RTCAnCT[2:0]	RTCATINTR interrupt request generation	
	Interval	Timing
000	No interrupt request generation	
001	Every 0.25 seconds	Every 0.25, 0.5, 0.75 and 1 second
010	Every 0.5 seconds	Every 0.5 and 1 second
011	Every second	Every 1 second
100	Every minute	Every 1 minute 00 seconds
101	Every hour	Every 1 hour 0 minutes 0 seconds
110	Every day	Every 1 day 0 hours 0 minutes 0 seconds (i.e. every midnight)
111	Every month	Every 1 month first day 0 hours 0 minutes 0 seconds (i.e. every first midnight of a month)

If the settings of RTCAnCT[2:0] are changed while sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTR. Implement appropriate interrupt mask processing procedures.

(3) RTCAnCTL2 – RTCA control register 2

This register contains status information and controls the data transfer from the sub-counter RTCAnSUBC to the dedicated sub-counter read buffer RTCAnSRBU and the operation setting of the clock counters (RTCAnSECC to RTCAnYEARC).

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 08_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	RTCAnWUST	RTCAnWSST	RTCAnRSST	RTCAnRSUB	RTCAnWST	RTCAnWAIT
R	R	R	R	R	R/W	R	R/W

Table 19-19 RTCAnCTL2 register contents (1/2)

Bit position	Bit name	Function
5	RTCAnWUST	Indicates whether RTCAnSUBU write operation has been completed: 0: RTCAnSUBU write completed 1: RTCAnSUBU write in progress The write operation ends with the next sub-counter overflow. This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). Refer to *19.5.5 "Writing to RTCAnSUBU" on page 1571* for details.
4	RTCAnWSST	Indicates whether RTCAnSCMP write operation has been completed: 0: RTCAnSCMP write completed 1: RTCAnSCMP write in progress The write operation ends with the next sub-counter overflow. This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). Refer to *19.5.6 "Writing to RTCAnSCMP" on page 1572* for details.
3	RTCAnRSST	Indicates whether the value of the sub-counter (RTCAnSUBC) has been transferred to the sub-count register read buffer (RTCAnSRBU): 0: Transfer in progress 1: Transfer completed This bit is cleared (transfer is triggered) by RTCAnRSUB = 1. This bit is automatically set when the transfer is completed. Refer to *19.5.4 "Reading RTCAnSRBU" on page 1570* for details.

Table 19-19 RTCAnCTL2 register contents (2/2)

Bit position	Bit name	Function
2	RTCAnRSUB	Enables/disables the transfer of the value of the sub-counter (RTCAnSUBC) to the dedicated read buffer (RTCAnSRBU): 0: Disables the transfer 1: Enables the transfer This bit is only valid when sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1). For details refer to *19.6.3 "Timing of sub-counter buffer read while counter is enabled" on page 1575* .
1	RTCAnWST	Indicates the status of all clock counters (RTCAnSECC to RTCAnYEARC): 0: All clock counters are running 1: All clock counters are stopped The sub-counter is still running. The clock counters must be stopped before reading or writing clock counter values during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1). To stop the clock counters, set RTCAnWAIT = 1.
0	RTCAnWAIT	Restarts/stops all clock counters (RTCAnSECC to RTCAnYEARC): 0: Restarts all clock counters either immediatelly or immediatelly after the clock counter write operation finishes. 1: Stops all clock counters temporarily The sub-counter is still running. The clock counters must be stopped before reading or writing counter buffers during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1).

19.4.3 RTCA sub-counter registers details

(1) RTCAnSUBC – RTCA sub-count register

This counter counts the 1 second reference time. It operates using the count clock RTCATCKI.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + $0C_H$

Initial Value 0000_H.

This register is initialized

- By an RTCA reset
- Upon write to the seconds count buffer (RTCAnSEC) or to the clock time setting register (RTCAnTIME).

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSUBC[21:16]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSUBC[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 19-20 RTCAnSUBC register contents

Bit position	Bit name	Function
21 to 0	RTCAnSUBC [21:0]	Sub-counter value. The sub-counter only operates while RTCAnCTL0.RTCAnCEST = 1.

Notes

1. This sub-counter operates with RTCATCKI while the read operation is clocked by PCLK. Reading this sub-counter during operation (RTCAnCTL0.RTCAnCEST = 1) is asynchronous to RTCATCKI and can lead to wrong results.

   Use the sub-count register read buffer (RTCAnSRBU) to read the sub-counter value during operation.

   For details refer to *19.5.4 “Reading RTCAnSRBU” on page 1570* .

2. The count-operation of this sub-counter depends on the selected operation mode. Refer to *19.3.1 “Operation modes” on page 1524* for details.

(2) RTCAnSRBU – RTCA sub-count register read buffer

This register is the read buffer for the sub-counter RTCAnSUBC.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + 10_H

Initial Value 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSRBU[21:16]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSRBU[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 19-21 RTCAnSRBU register contents

Bit position	Bit name	Function
21 to 0	RTCAnSRBU [21:0]	Sub-counter value at the time of the last RTCAnSUBC read. The sub-counter value is only read when control bit RTCAnCTL2.RTCAnRSUB is set to 1. The read operation is synchronized with RTCATCKI.

Note Perform RTCAnSRBU read according to the flow described in *19.5.4 “Reading RTCAnSRBU” on page 1570* .

(3) RTCAnSUBU – RTCA clock error correction register

This register enables and specifies clock error correction. This register only applies in 32.768 KHz mode (RTCAnCTL0.RTCAnSLSB = 0).

For details on clock error correction refer to *19.3.5 “Clock error correction” on page 1527* .

Access This register can be read/written in 8-bit units.

Note the following when writing this register during sub-counter operation:

- Previous RTCAnSUBU write must be completed (RTCAnCTL2.RTCAnWUST = 0).
- The write operation ends with the next sub-counter overflow.

Address <RTCAn_base> + 38_H

Initial Value 00_H

7	6	5	4	3	2	1	0
RTCAnDEV	RTCAnF6	RTCAnF[5:0]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-22 RTCAnSUBU register contents

Bit position	Bit name	Function
7	RTCAnDEV	Specifies how often clock error correction is performed per minute: 0: Three times every minute (when RTCAnSECC equals 00, 20, and 40) 1: Once every minute (when RTCAnSECC equals 00)
6	RTCAnF6	Specifies whether the sub-counter value is incremented or decremented: 0: Incremented (+ correction) Incrementation value = (RTCAnF[5:0] value – 1) x 2 1: Decremented (– correction) Decrementation value = (inverted data of RTCAnF[5:0] value + 1) x 2
5 to 0	RTCAnF[5:0]	Error correction value.

Notes

1. When RTCAnF[5:0] = 00 0000_B, clock error correction is not performed.
2. Perform RTCAnSUBU write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* and
   - *19.5.5 “Writing to RTCAnSUBU” on page 1571* .

(4) RTCAnSCMP – RTCA sub-counter compare register

This register sets the compare value of the sub-counter RTCAnSUBC in frequency selection mode (RTCAnCTL0.RTCAnSLSB = 1).

When the sub-counter values matches the value of this register an overflow signal is output to the seconds counter RTCAnSECC and the sub-counter is cleared.

Set the value for this register according to the frequency of the input clock RTCATCKI.

Access This register can be read/written in 32-bit units.

Note the following when writing this register during sub-counter operation:

- Previous RTCAnSCMP write must be completed (RTCAnCTL2.RTCAnWSST = 0)
- The write operation ends with the next sub-counter overflow.

Address <RTCAn_base> + $3C_H$

Initial Value 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	RTCAnSCMP[21:16]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnSCMP[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-23 RTCAnSCMP register contents

Bit position	Bit name	Function
21 to 0	RTCAnSCMP [21:0]	Sub-counter compare value in frequency selection mode.

Example The following example illustrates the correct setting of RTCAnSCMP:

- RTCATCKI = 4 MHz = 4,000,000 Hz
- Set RTCAnSCMP = 4,000,000 – 1 = 3,999,999 (decimal code) = $3D08FF_H$
- The seconds counter RTCAnSECC is triggered when the sub-counter value changes from $3D08FF_H$ to 0_H.

Notes

1. The operation of the RTCA cannot be guaranteed if a value of 3198 (decimal code) or lower is set to this register.
2. Perform RTCAnSCMP write as described in *19.5.1 “Initial setting of the RTCA” on page 1564* and *19.5.6 “Writing to RTCAnSCMP” on page 1572* .

19.4.4 RTCA clock counter and buffer registers details

(1) RTCAnSECC – RTCA seconds count register

This register is the seconds counter. It counts seconds from 00 to 59 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the sub-counter RTCAnSUBC.

  If the sub-counter overflows while the seconds counter is stopped (RTCAnCTL2.RTCAnWST = 1), the seconds counter behaves as follows:

  – If *one* sub-counter overflow occurs while the seconds counter is stopped, the overflow is held internally.
  The seconds counter is incremented by one when it is restarted.

  – If *two or more* overflows occurs while the seconds counter is stopped, the overflow count cannot be held internally.
  The seconds counter is incremented by one when it is restarted.

  – If the seconds counter was updated while the seconds counter is stopped, the sub-counter overflow(s) are ignored.

- It outputs an overflow signal when the value changes from 59 to 00. The overflow signal triggers the minutes counter (RTCAnMINC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + $4C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
0	RTCAnSECC[6:0]						
R	R	R	R	R	R	R	R

Table 19-24 RTCAnSECC register contents

Bit position	Bit name	Function
6 to 0	RTCAnSECC [6:0]	Seconds in BCD.

Notes

1. Perform RTCAnSECC read according to the flow described in *19.5.3 “Reading clock counters” on page 1567* .
2. A start value can be assigned to this register by writing to the seconds count buffer register RTCAnSEC or to the clock time setting register RTCAnTIME. Refer to
   - *19.5.1 “Initial setting of the RTCA” on page 1564* and
   - *19.5.2 “Updating clock counters” on page 1566* .

(2) RTCAnSEC – RTCA seconds count buffer register

This register is a buffer register to read/write the seconds counter RTCAnSECC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 14_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	RTCAnSEC[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-25 RTCAnSEC register contents

Bit position	Bit name	Function
6 to 0	RTCAnSEC [6:0]	Seconds in BCD.

Notes

1. When writing this register, only decimal values between 00 and 59 in BCD are allowed.
2. Perform RTCAnSEC read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* ,
   - *19.5.2 “Updating clock counters” on page 1566* , and
   - *19.5.3 “Reading clock counters” on page 1567* .

(3) RTCAnMINC – RTCA minute count register

This register is the minutes counter. It counts minutes from 00 to 59 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the seconds counter RTCAnSECC.
- It outputs an overflow signal when the value changes from 59 to 00. The overflow signal triggers the hours counter (RTCAnHOURC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 50_H

Initial Value 00_H

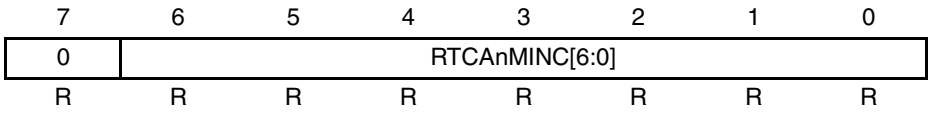


Table 19-26 RTCAnMINC register contents

Bit position	Bit name	Function
6 to 0	RTCAnMINC [6:0]	Minutes in BCD.

Notes

1. Perform RTCAnMINC read according to the flow described in *19.5.3 "Reading clock counters" on page 1567* .
2. A start value can be assigned to this register by writing to the minutes count buffer register RTCAnMIN or to the clock time setting register RTCAnTIME. Refer to
   - *19.5.1 "Initial setting of the RTCA" on page 1564* and
   - *19.5.2 "Updating clock counters" on page 1566* .

(4) RTCAnMIN – RTCA minute count buffer register

This register is a buffer register to read/write the minutes counter RTCAnMINC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 18_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	RTCAnMIN[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-27 RTCAnMIN register contents

Bit position	Bit name	Function
6 to 0	RTCAnMIN [6:0]	Minutes in BCD.

Notes

1. When writing this register, only decimal values between 00 and 59 in BCD are allowed.
2. Perform RTCAnMIN read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* ,
   - *19.5.2 “Updating clock counters” on page 1566* , and
   - *19.5.3 “Reading clock counters” on page 1567* .

(5) RTCAnHOURC – RTCA hour count register

This register is the hours counter. It counts the hours in BCD. The count range depends on the selected hour format. Refer to *Table 19-29 "12 and 24 hour format" on page 1546* .

This register has the following trigger properties:

- It is triggered by every overflow of the minutes counter RTCAnMINC.
- It outputs an overflow signal when the value changes from 23 to 00 (in 24 hour format) or from 32 to 01 (in 12 hour format). The overflow signal triggers two counters:
  - Day of the week counter (RTCAnWEEKC)
  - Day of the month counter (RTCAnDAYC)

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 54_H

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0
00_H when RTCAnCTL0.RTCAnAMPM = 1.

7	6	5	4	3	2	1	0
0	0	RTCAnHOURC[5:0]					
R	R	R	R	R	R	R	R

Table 19-28 RTCAnHOURC register contents

Bit position	Bit name	Function
5 to 0	RTCAnHOURC [5:0]	Hours in BCD. Refer to *Table 19-29 "12 and 24 hour format" on page 1546* for details.

Notes

1. Perform RTCAnHOURC read according to the flow described in *19.5.3 "Reading clock counters" on page 1567* .
2. A start value can be assigned to this register by writing to the hours count buffer register RTCAnHOUR or to the clock time setting register RTCAnTIME. Refer to
   - *19.5.1 "Initial setting of the RTCA" on page 1564* and
   - *19.5.2 "Updating clock counters" on page 1566* .

12 or 24 hour format

The count values of RTCAnHOURC depend on the selected hour format.

If 12 hour format is selected (RTCAnCTL0.RTCAnAMPM = 0), bit RTCAnHOURC5 is the am/ pm indicator:

- RTCAnHOURC5 = 0: am
- RTCAnHOURC5 = 1: pm

The following table shows the count range of RTCAnHOURC in both 12 and 24 hour format.

Table 19-29 12 and 24 hour format

12 hour format (RTCAnAMPM = 0)			24 hour format (RTCAnAMPM = 1)	
Time	**RTCAnHOUR**		**Time**	**RTCAnHOUR**
0 am	12_H		0	00_H
1 am	01_H		1	01_H
2 am	02_H		2	02_H
3 am	03_H		3	03_H
4 am	04_H		4	04_H
5 am	05_H		5	05_H
6 am	06_H		6	06_H
7 am	07_H		7	07_H
8 am	08_H		8	08_H
9 am	09_H		9	09_H
10 am	10_H		10	10_H
11 am	11_H		11	11_H
0 pm	32_H	⇓	12	12_H
1 pm	21_H		13	13_H
2 pm	22_H	pm indicator in 12 hour format: RTCAnHOUR5 = 1	14	14_H
3 pm	23_H		15	15_H
4 pm	24_H		16	16_H
5 pm	25_H		17	17_H
6 pm	26_H		18	18_H
7 pm	27_H		19	19_H
8 pm	28_H		20	20_H
9 pm	29_H		21	21_H
10 pm	30_H		22	22_H
11 pm	31_H		23	23_H

(6) RTCAnHOUR – RTCA hour count buffer register

This register is a buffer register to read/write the hours counter RTCAnHOURC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + $1C_H$

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0
00_H when RTCAnCTL0.RTCAnAMPM = 1.

7	6	5	4	3	2	1	0
0	0	RTCAnHOUR[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-30 RTCAnHOUR register contents

Bit position	Bit name	Function
5 to 0	RTCAnHOUR [5:0]	Hours in BCD. Refer to *Table 19-29 "12 and 24 hour format" on page 1546* for details.

Notes

1. When writing this register, only the following decimal values in BCD are allowed:
   - 12 hour format (RTCAnCTL0.RTCAnAMPM = 0):
     01 to 12 or 21 to 32
   - 24 hour format (RTCAnCTL0.RTCAnAMPM = 1):
     00 to 23
2. Perform RTCAnHOUR read/write as described in
   - *19.5.1 "Initial setting of the RTCA" on page 1564* ,
   - *19.5.2 "Updating clock counters" on page 1566* , and
   - *19.5.3 "Reading clock counters" on page 1567* .

(7) RTCAnWEEKC – RTCA day of the week count register

This register is the day of the week counter. It counts from 0 to 6.

It has the following trigger properties:

- It is triggered by every overflow of the hours counter RTCAnHOURC.
- It does not trigger any other counter.

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 58_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	RTCAnWEEKC[2:0]		
R	R	R	R	R	R	R	R

Table 19-31 RTCAnWEEKC register contents

Bit position	Bit name	Function
2 to 0	RTCAnWEEKC [2:0]	Day of the week.

Notes

1. Perform RTCAnWEEKC read according to the flow described in *19.5.3 “Reading clock counters” on page 1567* .
2. A start value can be assigned to this register by writing to the week count buffer register RTCAnWEEK or to the calender setting register RTCAnCAL. Refer to
   - *19.5.1 “Initial setting of the RTCA” on page 1564* and
   - *19.5.2 “Updating clock counters” on page 1566* .

(8) RTCAnWEEK – RTCA day of the week count buffer register

This register is a buffer register to read/write the day of the week counter RTCAnWEEKC.

There is no particular correspondence between the value of RTCAnWEEK and the day of the week. Set the correspondence according to the application to be used.

Example: 0 = Sunday, 1 = Monday, …, 6 = Saturday

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 20_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	RTCAnWEEK[2:0]		
R	R	R	R	R	R/W	R/W	R/W

Table 19-32 RTCAnWEEK register contents

Bit position	Bit name	Function
2 to 0	RTCAnWEEK [2:0]	Day of the week.

Notes

1. When writing this register, only decimal values between 0 and 6 in BCD are allowed.
2. Perform RTCAnWEEK read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* ,
   - *19.5.2 “Updating clock counters” on page 1566* , and
   - *19.5.3 “Reading clock counters” on page 1567* .

(9) RTCAnDAYC – RTCA day of the month count register

This register is the day of the month counter. It counts from 01 to a maximum of 31 in BCD, depending on the value of the month counter (RTCAnMONC) and the year counter (RTCAnYEARC):

- 01 to 31 (January, March, May, July, August, October, December)
- 01 to 30 (April, June, September, November)
- 01 to 29 (February, leap year)
- 01 to 28 (February, non-leap year)

Years 0, 4, 8, 12, etc. are considered leap years.

It has the following trigger properties:

- It is triggered by every overflow of the hours of the day counter RTCAnHOURC.
- It outputs an overflow signal when the value changes from 28, 29, 30, or 31 to 01, depending on the current month and year. The overflow signal triggers the month counter (RTCAnMONC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + $5C_H$

Initial Value 01_H

7	6	5	4	3	2	1	0
0	0	RTCAnDAYC[5:0]					
R	R	R	R	R	R	R	R

Table 19-33 RTCAnDAYC register contents

Bit position	Bit name	Function
5 to 0	RTCAnDAYC [5:0]	Day of the month in BCD

Notes

1. Perform RTCAnDAYC read according to the flow described in *19.5.3 “Reading clock counters” on page 1567* .
2. A start value can be assigned to this register by writing to the day count buffer register RTCAnDAY or to the calender setting register RTCAnCAL. Refer to
   - *19.5.1 “Initial setting of the RTCA” on page 1564* and
   - *19.5.2 “Updating clock counters” on page 1566* .

(10) RTCAnDAY – RTCA day of the month count buffer register

This register is a buffer register to read/write the day of the month counter RTCAnDAYC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 24_H

Initial Value 01_H

7	6	5	4	3	2	1	0
0	0	RTCAnDAY[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-34 RTCAnDAY register contents

Bit position	Bit name	Function
5 to 0	RTCAnDAY [5:0]	Day of the month in BCD.

Notes

1. When writing this register, only decimal values between 01 and 31 in BCD are allowed:
   - 01 to 31 (January, March, May, July, August, October, December)
   - 01 to 30 (April, June, September, November)
   - 01 to 29 (February, leap year)
   - 01 to 28 (February, non-leap year)
2. Perform RTCAnDAY read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564*
   - *19.5.2 “Updating clock counters” on page 1566*
   - *19.5.3 “Reading clock counters” on page 1567* .

(11) RTCAnMONC – RTCA month count register

This register is the month counter. It counts the month of the year, starting from 01 to 12 in BCD.

It has the following trigger properties:

- It is triggered by every overflow of the sub-counter RTCAnDAYC.
- It outputs an overflow signal when the value changes from 12 to 01. The overflow signal triggers the year counter (RTCAnYEARC).

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 60_H

Initial Value 01_H

7	6	5	4	3	2	1	0
0	0	0	RTCAnMONC[4:0]				
R	R	R	R	R	R	R	R

Table 19-35 RTCAnMONC register contents

Bit position	Bit name	Function
4 to 0	RTCAnMONC [4:0]	Month of the year in BCD.

Notes

1. Perform RTCAnMONC read according to the flow described in *19.5.3 “Reading clock counters” on page 1567* .
2. A start value can be assigned to this register by writing to the month count buffer register RTCAnMONTH or to the calender setting register RTCAnCAL. Refer to
   - *19.5.1 “Initial setting of the RTCA” on page 1564*
   - *19.5.2 “Updating clock counters” on page 1566* .

(12) RTCAnMONTH – RTCA month count buffer register

This register is a buffer register to read/write the month counter RTCAnMONC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 28_H

Initial Value 01_H

7	6	5	4	3	2	1	0
0	0	0	RTCAnMONTH[4:0]				
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 19-36 RTCAnMONTH register contents

Bit position	Bit name	Function
4 to 0	RTCAnMONTH [4:0]	Month of the year in BCD.

Notes

1. When writing this register, only decimal values between 01 and 12 in BCD are allowed.
2. Perform RTCAnMONTH read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* ,
   - *19.5.2 “Updating clock counters” on page 1566* , and
   - *19.5.3 “Reading clock counters” on page 1567* .

(13) RTCAnYEARC – RTCA year count register

This register is the year counter. It counts years from 00 to a maximum of 99 in BCD.

Years 00, 04, 08, …, 92, and 96 (every four years) are considered leap years.

It has the following trigger properties:

- It is triggered by every overflow of the month counter RTCAnMONC.
- It does not trigger any other counter.

Access This register can be read in 8-bit units.

Address <RTCAn_base> + 64_H

Initial Value 00_H

7	6	5	4	3	2	1	0
RTCAnYEAR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-37 RTCAnYEARC register contents

Bit position	Bit name	Function
7 to 0	RTCAnYEARC [7:0]	Year in BCD.

Notes

1. Perform RTCAnYEARC read according to the flow described in *19.5.3 “Reading clock counters” on page 1567* .
2. A start value can be assigned to this register by writing to the year count buffer register RTCAnYEAR or to the calender setting register RTCAnCAL. Refer to
   - *19.5.1 “Initial setting of the RTCA” on page 1564* and
   - *19.5.2 “Updating clock counters” on page 1566* .

(14) RTCAnYEAR – RTCA year count buffer register

This register is a buffer register to read/write the year counter RTCAnYEARC.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + $2C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
RTCAnYEAR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-38 RTCAnYEAR register contents

Bit position	Bit name	Function
7 to 0	RTCAnYEAR [7:0]	Year in BCD.

Notes

1. When writing this register, only decimal values between 00 and 99 in BCD are allowed.
2. Perform RTCAnYEAR read/write as described in
   - *19.5.1 “Initial setting of the RTCA” on page 1564* ,
   - *19.5.2 “Updating clock counters” on page 1566* , and
   - *19.5.3 “Reading clock counters” on page 1567* .

19.4.5 RTCA special counter and buffer registers details

(1) RTCAnTIMEC – RTCA time count register

This register enables the RTCAnHOURC, RTCAnMINC, and RTCAnSECC counters to be read simultaneously.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + 68_H

Initial Value 0012 0000_H when RTCAnCTL0.RTCAnAMPM = 0,
0000 0000_H when RTCAnCTL0.RTCAnAMPM = 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	RTCAnHOURC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnMINC[7:0]								RTCAnSECC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 19-39 RTCAnTIMEC register contents

Bit position	Bit name	Function
23 to 16	RTCAnHOURC [7:0]	Hours in BCD. Refer to *Table 19-29 "12 and 24 hour format" on page 1546* for details.
15 to 8	RTCAnMINC [7:0]	Minutes in BCD.
7 to 0	RTCAnSECC [7:0]	Seconds in BCD.

Notes

1. Perform RTCAnTIMEC read according to the flow described in *19.5.3 "Reading clock counters" on page 1567* .
2. A start value can be assigned to this register by writing to the clock time setting register RTCAnTIME. Refer to
   - *19.5.1 "Initial setting of the RTCA" on page 1564* and
   - *19.5.2 "Updating clock counters" on page 1566* .

(2) RTCAnTIME – RTCA time count buffer register

This register enables the RTCAnHOUR, RTCAnMIN, and RTCAnSEC buffer registers to be read/written simultaneously.

Access This register can be read/written in 32-bit units.

Address <RTCAn_base> + 30_H

Initial Value $0012\ 0000_H$ when RTCAnCTL0.RTCAnAMPM = 0,
$0000\ 0000_H$ when RTCAnCTL0.RTCAnAMPM = 1.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	RTCAnHOUR[7:0]							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnMIN[7:0]								RTCAnSEC[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-40 RTCAnTIME register contents

Bit position	Bit name	Function
23 to 16	RTCAnHOUR [7:0]	Hours in BCD. Refer to *Table 19-29 "12 and 24 hour format" on page 1546* for details.
15 to 8	RTCAnMIN [7:0]	Minutes in BCD.
7 to 0	RTCAnSEC [7:0]	Seconds in BCD.

Note Perform RTCAnTIME read/write as described in

- *19.5.1 "Initial setting of the RTCA" on page 1564* ,
- *19.5.2 "Updating clock counters" on page 1566* , and
- *19.5.3 "Reading clock counters" on page 1567* .

(3) RTCAnCALC – RTCA calendar count register

This register enables the RTCAnYEARC, RTCAnMONC, RTCAnDAYC, and RTCAnWEEKC counters to be read simultaneously.

Access This register can be read in 32-bit units.

Address <RTCAn_base> + $6C_H$

Initial Value $0001\ 0100_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCTAnYEARC[7:0]								RTCAnMONC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnDAYC[7:0]								RTCAnWEEKC[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 19-41 RTCAnCALC register contents

Bit position	Bit name	Function
31 to 24	RTCAnYEARC [7:0]	Year in BCD.
23 to 16	RTCAnMONC [7:0]	Month of the year in BCD.
15 to 8	RTCAnDAYC [7:0]	Day of the month in BCD.
7 to 0	RTCAnWEEKC [7:0]	Day of the week in BCD.

Notes

1. Perform RTCAnCALC read according to the flow described in *19.5.3 "Reading clock counters" on page 1567* .
2. A start value can be assigned to this register by writing to the clock time setting register RTCAnCAL. Refer to
   - *19.5.1 "Initial setting of the RTCA" on page 1564* and
   - *19.5.2 "Updating clock counters" on page 1566* .

(4) RTCAnCAL – RTCA calendar count buffer register

This register enables the RTCAnYEAR, RTCAnMONTH, RTCAnDAY, and RTCAnWEEK buffer registers to be read/written simultaneously.

Access This register can be read/written in 32-bit units.

Address <RTCAn_base> + 34_H

Initial Value $0001\ 0100_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RCTAnYEAR[7:0]								RTCAnMONTH[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCAnDAY[7:0]								RTCAnWEEK[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-42 RTCAnCAL register contents

Bit position	Bit name	Function
31 to 24	RTCAnYEAR [7:0]	Year in BCD.
23 to 16	RTCAnMONTH [7:0]	Month of the year in BCD.
15 to 8	RTCAnDAY [7:0]	Day of the month in BCD.
7 to 0	RTCAnWEEK [7:0]	Day of the week in BCD.

Note Perform RTCAnCAL read/write as described in

- *19.5.1 “Initial setting of the RTCA” on page 1564* ,
- *19.5.2 “Updating clock counters” on page 1566* , and
- *19.5.3 “Reading clock counters” on page 1567* .

19.4.6 RTCA alarm setting registers details

(1) RTCAnALM – RTCA alarm minute setting register

This register specifies the minute of the alarm interrupt.

For details and example settings refer to *19.3.4 “Alarm interrupt function” on page 1526* .

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 40_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	RTCAnALM[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-43 RTCAnALM register contents

Bit position	Bit name	Function
6 to 0	RTCAnALM [6:0]	Minute of the alarm interrupt in BCD.

Notes

1. If decimal values outside the range of 00 to 59 in BCD are set, no alarm interrupt request will be generated.
2. When the setting of RTCAnALM is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

(2) RTCAnALH – RTCA alarm hour setting register

This register specifies the hour of the alarm interrupt.

For details and example settings refer to *19.3.4 “Alarm interrupt function” on page 1526* .

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 44_H

Initial Value 12_H when RTCAnCTL0.RTCAnAMPM = 0,
00_H when RTCAnCTL0.RTCAnAMPM = 1.

7	6	5	4	3	2	1	0
0	0	RTCAnALH[5:0]					
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-44 RTCAnALH register contents

Bit position	Bit name	Function
5 to 0	RTCAnALH [5:0]	Hour of the alarm interrupt in BCD.

Notes

1. If decimal values outside the following range are set, no alarm interrupt request will be generated:
   - 12 hour format (RTCAnCTL0.RTCAnAMPM = 0):
     01 to 12 or 21 to 32
   - 24 hour format (RTCAnCTL0.RTCAnAMPM = 1):
     00 to 23
2. When the setting of RTCAnALH is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

(3) RTCAnALW – RTCA alarm day of the week setting register

This register specifies the day(s) of the week of the alarm interrupt.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 48_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	RTCAnALW[6:0]						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-45 RTCAnALW register contents

Bit position	Bit name	Function
6 to 0	RTCAnALW [6:0]	Specifies day of the week m (m = 0 to 6) as a day, where an alarm interrupt request is generated: 0: No alarm interrupt request is generated on day m 1: Alarm interrupt request is generated on day m at the time set using RTCAnALM and RTCAnALH The bits of this register correspond to the count value of the day of the week counter (RTCAnWEEKC).

Note When the setting of RTCAnALW is changed during sub-counter operation (RTCAnCTL0.RTCAnCEST = 1), a glitch may be output to RTCATINTAL. Implement appropriate interrupt mask processing procedures.

Example If Sunday is RTCAnWEEK = 0, Monday is RTCAnWEEK = 1, Tuesday is RTCAnWEEK = 2, …, Saturday is RTCAnWEEK = 6:

- To set the alarm to Sunday, set RTCAnALW = 0000 0001.
- To set the alarm to Monday and Wednesday, set RTCAnALW = 0000 1010.
- To set the alarm to Tuesday, Thursday, and Saturday, set RTCAnALW = 0101 0100

For more examples, refer to *19.3.4 “Alarm interrupt function” on page 1526* .

19.4.7 RTCA emulation register details

(1) RTCAnEMU - RTCAn emulation register

This register controls whether the RTCA can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <RTCAn_base> + 74_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
RTCAn SVS DIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 19-46 RTCAnEMU register contents

Bit position	Bit name	Function
7	RTCAn SVSDIS	Emulation control 0: RTCA can be stopped during emulation 1: RTCA continuous operating during emulation

19.5 Procedures for Setup, Writing and Reading

The following subsections provide flow charts that illustrate the procedures for RTCA setup and for reading and writing the RTCA clock counters.

19.5.1 Initial setting of the RTCA

The RTCA must be stopped before setting initial counter values.

(1) RTCA stop procedure

Stop the RTCA according to the following flow.

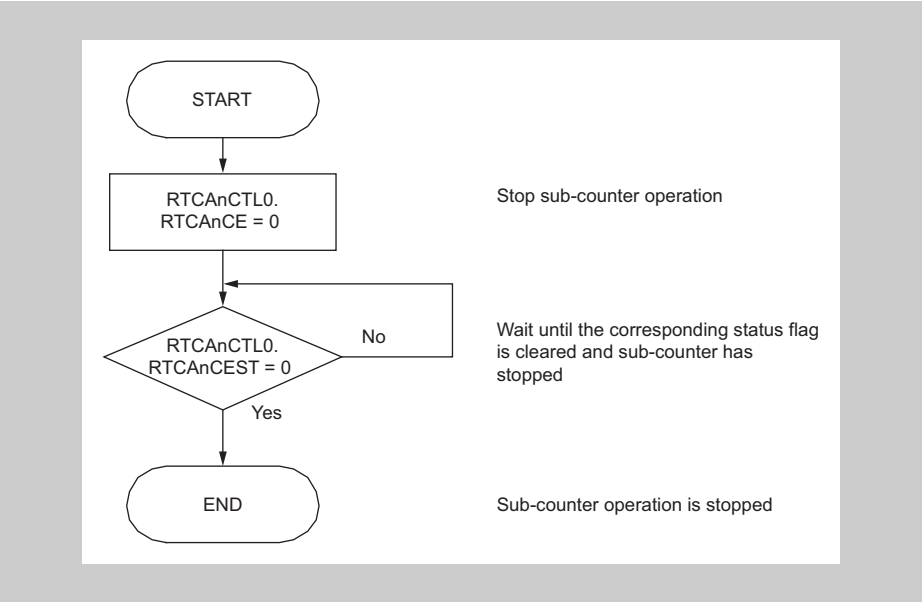


Figure 19-5 RTCA stop procedure

(2) RTCA initialization procedure

Perform the initial setting of the RTCA according to the following flow:

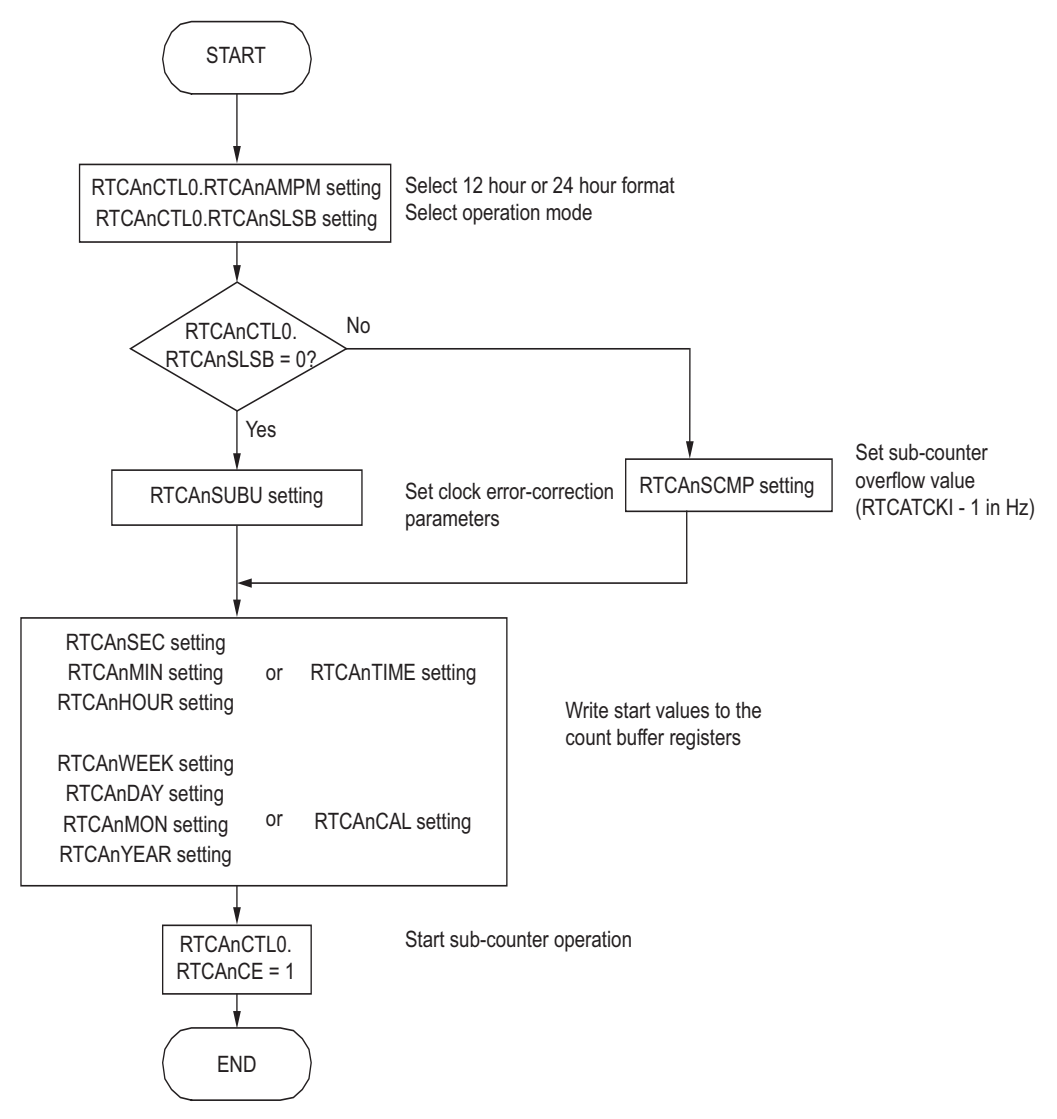


Figure 19-6 RTCA setup procedure

Caution The register settings are becoming effective in synchronization with the RTCACKI clock. Consequently rewriting a register within a single RTCACKI period overwrites the previous register setting, which is lost in such case.

Thus do not rewrite a register within the flow of the above procedure.

19.5.2 Updating clock counters

The clock counters RTCAnSECC to RTCAnYEARC can be stopped and updated while the sub-counter is running.

The update procedure is illustrated in the following flow.

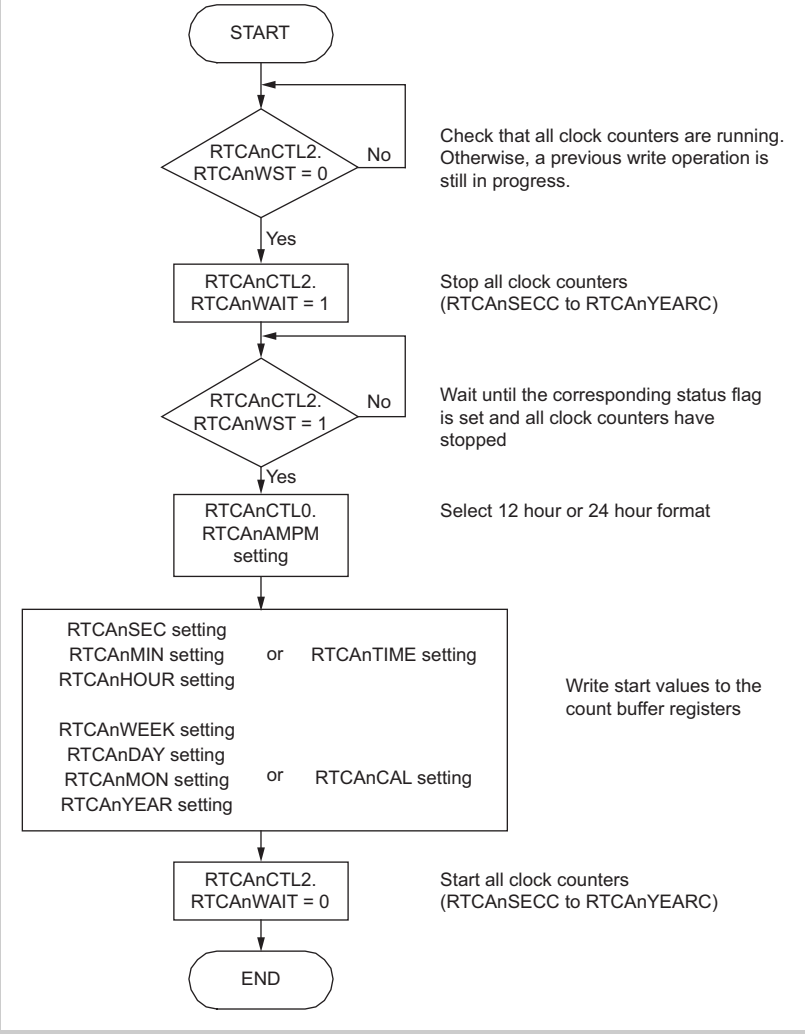


Figure 19-7 Updating clock counter values

Caution The register settings are becoming effective in synchronization with the RTCACKI clock. Consequently rewriting a register within a single RTCACKI period overwrites the previous register setting, which is lost in such case.

Thus do not rewrite a register within the flow of the above procedure.

Note The update procedure must be completed within 1 second. Otherwise the Real-Time Clock will not count correctly any more:

- Only *one* sub-counter overflow can be held internally and increment the seconds counter after restarting the clock counters.

- If the sub-counter overflows more than once during clock counter stop, the overflow count cannot be held internally. Thus the seconds counter is incremented by one instead of by two when it is restarted.

19.5.3 Reading clock counters

There are two methods to read the clock counters while sub-counter operation is enabled:

- Reading count buffer registers
- Reading counter registers

The advantages and disadvantages of the two methods are summarized in the following table.

Table 19-47 Comparison of the two read methods

	Advantage	Disadvantage
Reading count buffer registers	It is unnecessary to read clock counters several times because the clock counters are read synchronously.	A program wait state occurs between setting RTCAnCTL2.RTCAnWAIT = 1 and completion of data transfer.
Reading count registers	Program wait state does not occur.	The clock counters must be read several times because they are read asynchronously to RTCATCKI.

(1) Procedure for reading count buffer registers

The following operations are necessary:

1. Stop all clock counters (RTCAnCTL2.RTCAnWAIT = 1)
   The value of the clock counters is transferred to the count buffer registers.
2. Read the count buffer registers

A program wait state occurs between setting RTCAnCTL2.RTCAnWAIT = 1 and completion of data transfer.

The maximum delay is three PCLK periods plus two RTCATCKI periods. For example, if the RTCA operates with PCLK = 80 MHz and RTCATCKI = 32.768 KHz, the delay is about 61 µsec.

For a timing example, refer to *19.6.2 “Timing of RTCA while counter is enabled” on page 1574* .

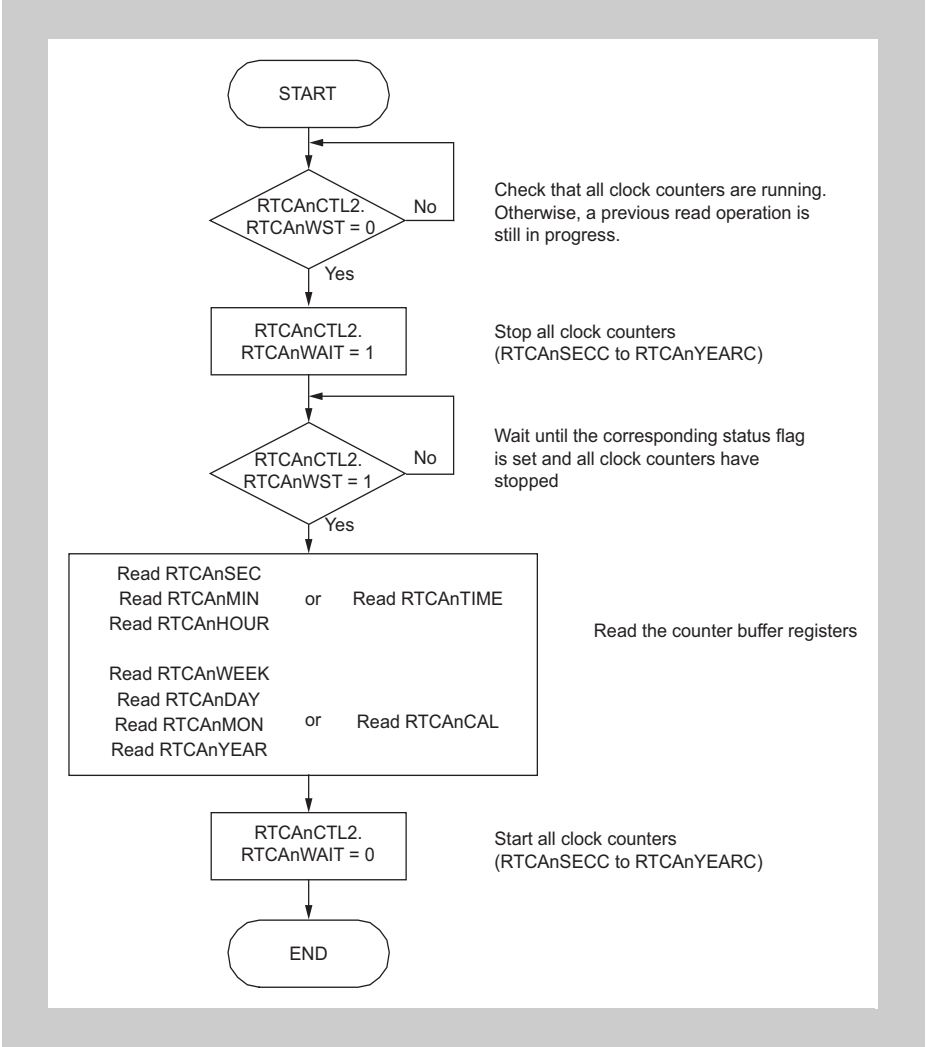


Figure 19-8 Reading clock count buffer registers

Note The reading procedure must be completed within 1 second. Otherwise the Real-Time Clock will not count correctly any more:

- Only *one* sub-counter overflow can be held internally and increment the seconds counter after restarting the clock counters.
- If the sub-counter overflows more than once during clock counter stop, the overflow count cannot be held internally. Thus the seconds counter is incremented by one instead of by two when it is restarted.

(2) Procedure for reading counter registers directly

To ensure that the sub-counter did not overflow while reading the counters, the seconds counter RTCAnSECC must be read twice in the beginning and at the end of the procedure. The first read value is compared with the second read value.

- First read value = second read value:

  No overflow of sub-counter occurred during counter read operation.

- First read value ≠ second read value:

  Overflow of the sub-counter occurred during counter read operation. The counters must be read again to get the current counter values.

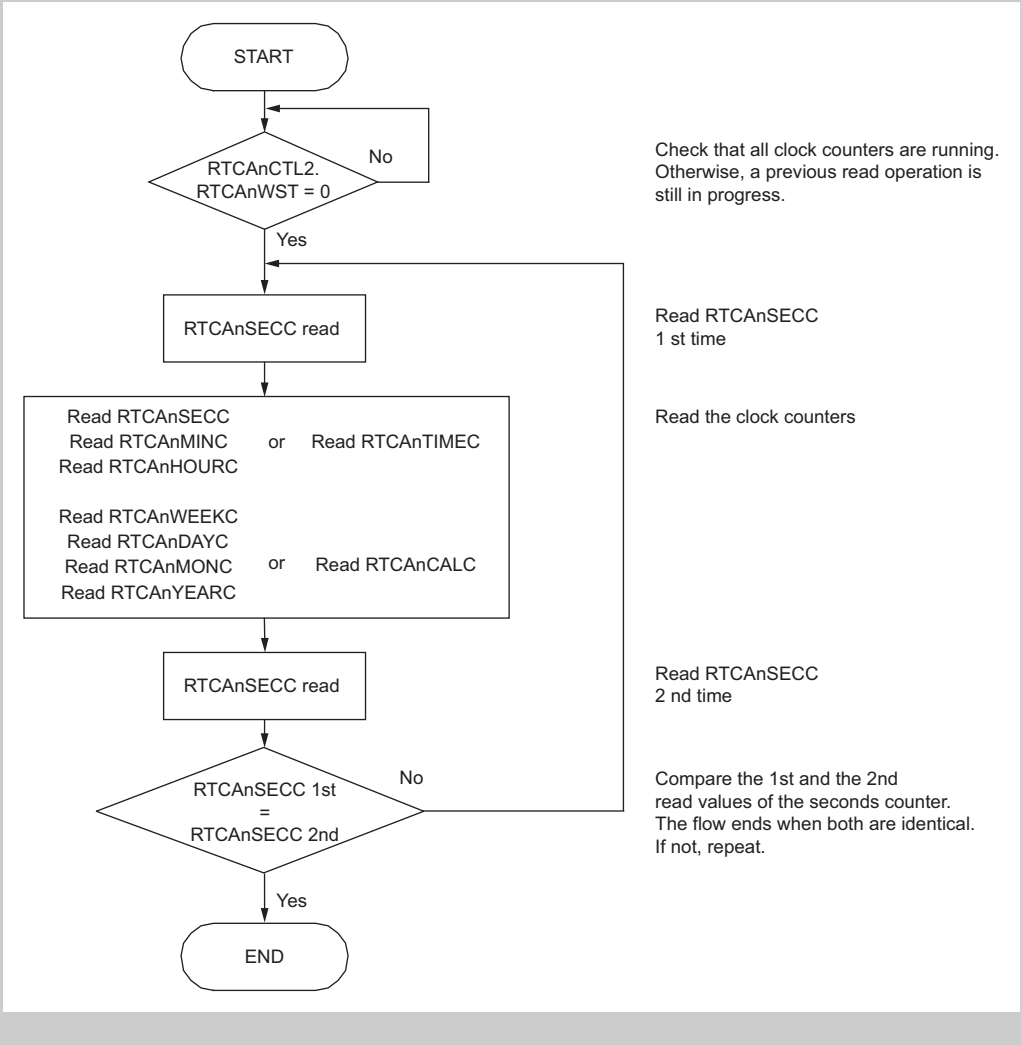


Figure 19-9 Reading clock counter registers

Note The procedure must be completed within 1 second.

19.5.4 Reading RTCAnSRBU

RTCAnSRBU is the read buffer register for the sub-counter.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), read RTCAnSRBU according to the following flow.

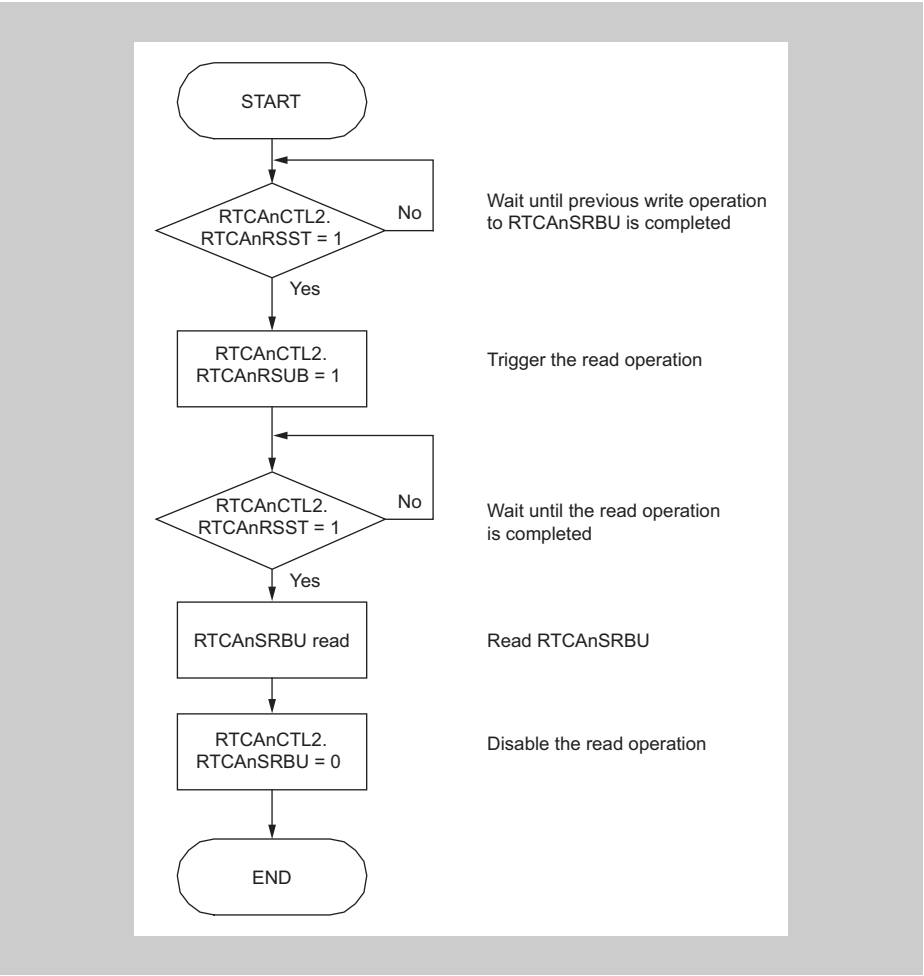


Figure 19-10 Reading the RTCAnSRBU register

19.5.5 Writing to RTCAnSUBU

RTCAnSUBU is the clock error correction register for the sub-counter.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), write to RTCAnSUBU according to the flow described below.

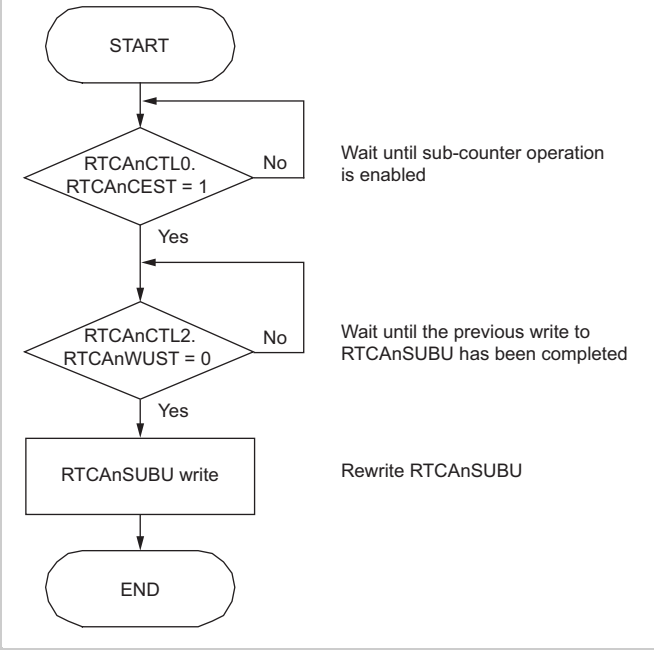


Figure 19-11 Writing to the RTCAnSUBU register

Note While the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), the status flag RTCAnCTL2.RTCAnWUST is set when RTCAnSUBU is written to. It is cleared when the write operation to RTCAnSUBU is completed. This is synchronous with the next RTCAnSUBC overflow.

RTCAnCTL2.RTCAnWUST can be set for up to 1 second.

19.5.6 Writing to RTCAnSCMP

RTCAnSCMP is the sub-counter compare register.

When the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), write to RTCAnSCMP according to the flow described below.

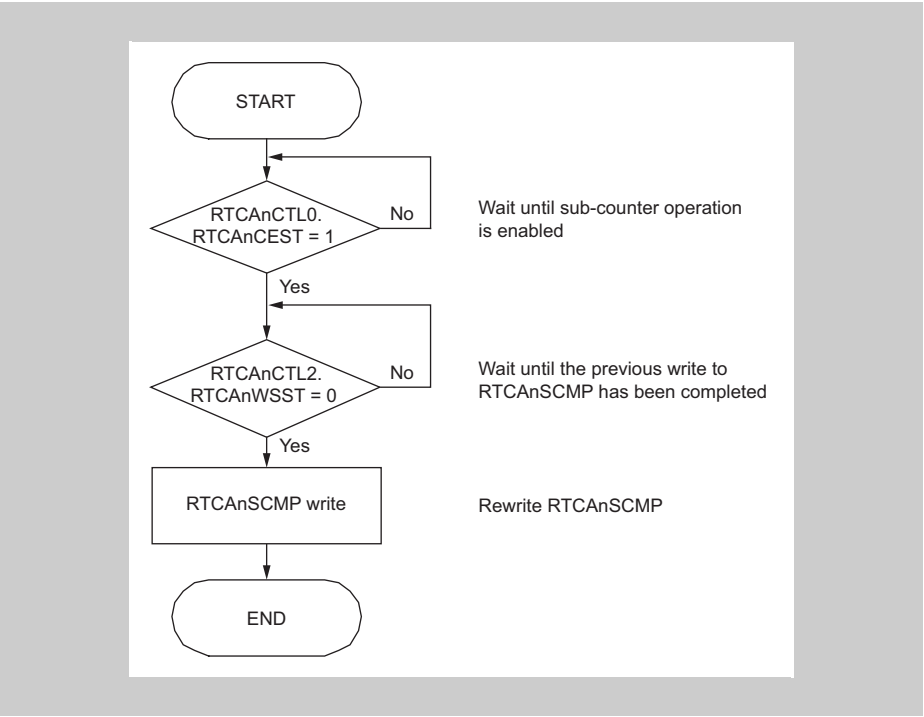


Figure 19-12 Writing the RTCAnSCMP register

Note While the sub-counter operation is enabled (RTCAnCTL0.RTCAnCEST = 1), the status flag RTCAnCTL2.RTCAnWSST is set when RTCAnSCMP is written to. It is cleared when the write operation to RTCAnSCMP is completed. This is synchronous with the next RTCAnSUBC overflow.

RTCAnCTL2.RTCAnWSST can be set for up to 1 second.

19.6 Timing diagrams

19.6.1 Timing of RTCA counter start

The following diagram illustrates the counter start after setting the time in the buffer registers.

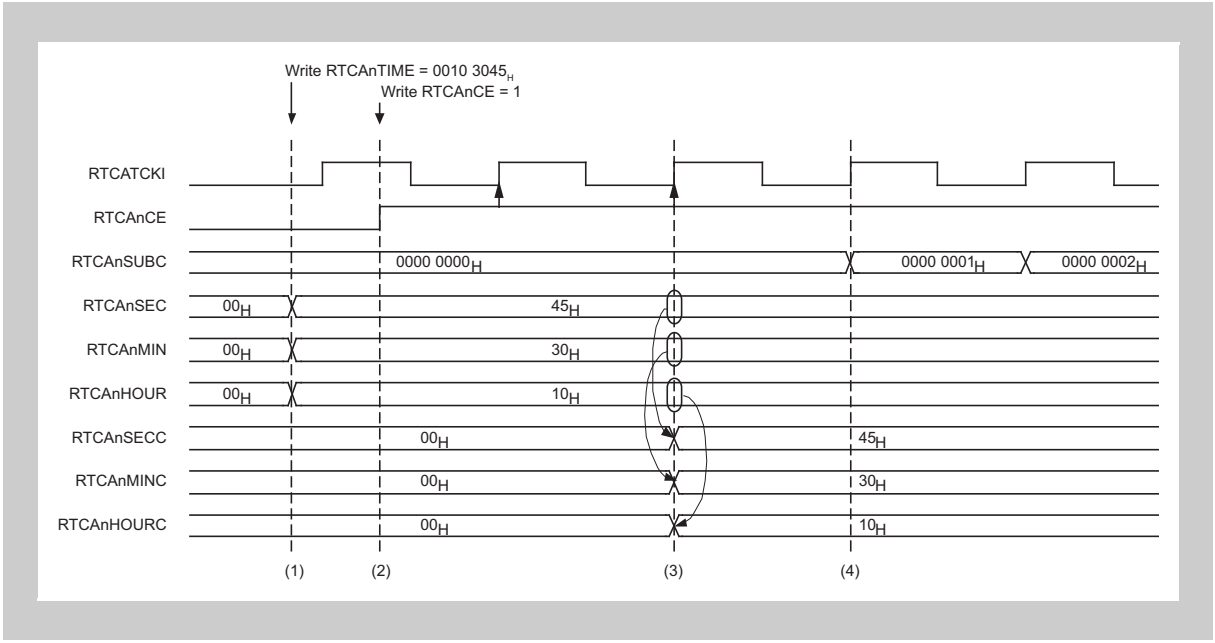


Figure 19-13 RTCA counter start timing

The timing diagram above shows the following:

1. The initial value of the time counters is set to 10:30:45 by writing RTCAnTIME= 0010 3045_H.
   Count buffer registers RTCAnSEC, RTCAnMIN, and RTCAnHOUR are automatically written.
2. Sub-counter operation is started by setting RTCAnCTL0.RTCAnCE = 1.
3. When the second rising edge of RTCATCKI occurs, both internal synchronization signals are set and the buffer register values are loaded to the count registers.
4. When the next rising edge of RTCATCKI occurs, count up of the sub-counter starts.

19.6.2 Timing of RTCA while counter is enabled

The following diagram illustrates the counter continuation after setting the time in the buffer registers.

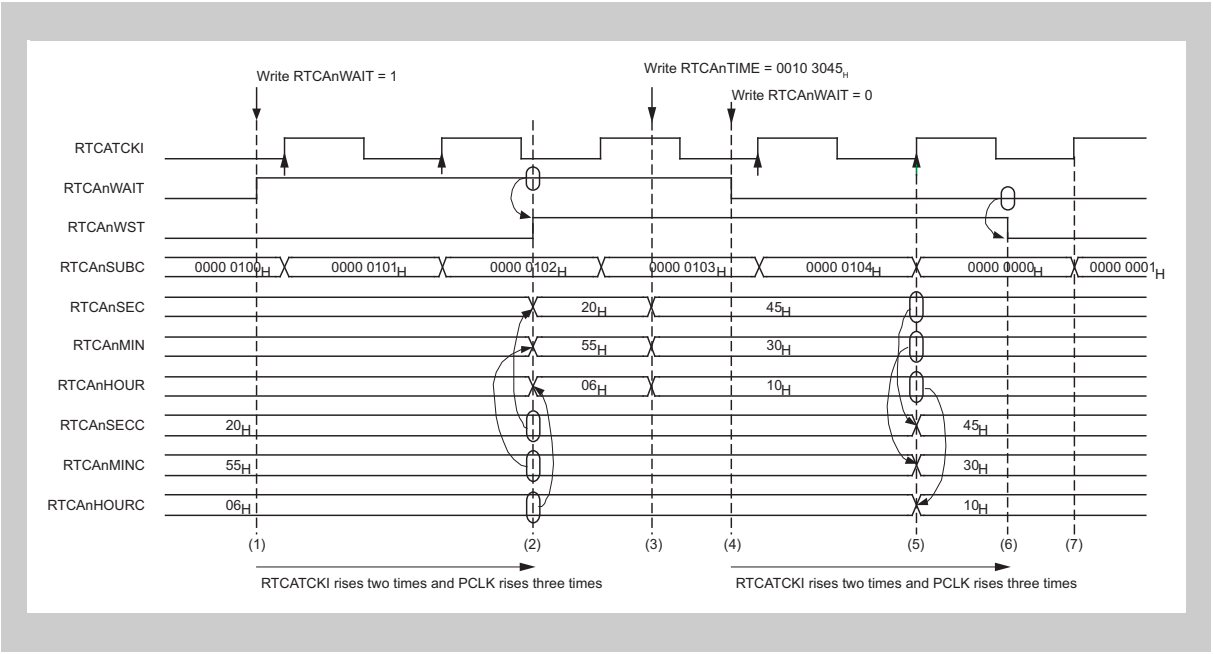


Figure 19-14 RTCA counter continuation timing

The timing diagram above shows the following:

1. Trigger the clock counters stop (RTCAnCTL2.RTCAnWAIT = 1).
2. It takes at least one RTCATCKI period to stop the counters. Stop of the counters is reflected by RTCAnCTL2.RTCAnWST = 1.
   The sub-counter continues counting.
3. The initial value of the time counters is set to 10:30:45 by writing RTCAnTIME= 0010 3045_H.
   Count buffer registers RTCAnSEC, RTCAnMIN, and RTCAnHOUR are automatically written.
4. Trigger the clock counters restart (RTCAnCTL2.RTCAnWAIT = 0).
5. After at least one RTCATCKI period RTCAnSECC is written and RTCAnSUBC is cleared.
6. The clock counters are ready for counter re-start (RTCAnCTL2.RTCAnWST = 0).
7. Clock counter operation is resumed.

19.6.3 Timing of sub-counter buffer read while counter is enabled

The following diagram illustrates the timing when reading the sub-counter read buffer RTCAnSRBU.

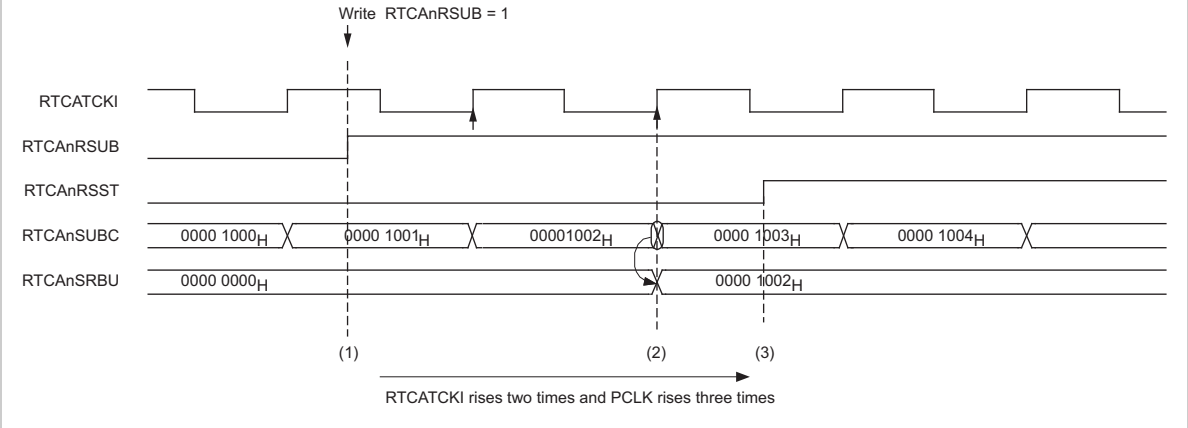


Figure 19-15 Timing when reading the sub-counter buffer register value

The timing diagram above shows the following:

1. Transfer of RTCAnSRBU value to the sub-counter is triggered (RTCAnCTL2.RTCAnRSUB = 1).
2. When the second rising edge of RTCATCKI occurs, the value of RTCAnSUBC is loaded to RTCAnSRBU.
3. Completion of the transfer is indicated by RTCAnCTL2.RTCAnRSST = 1. RTCAnSRBU can now be read.

Chapter 20 Motor Control

20.1 Basic structure of motor control

Implementation of motor control functions involve several modules, as shown in the basic block diagram below.

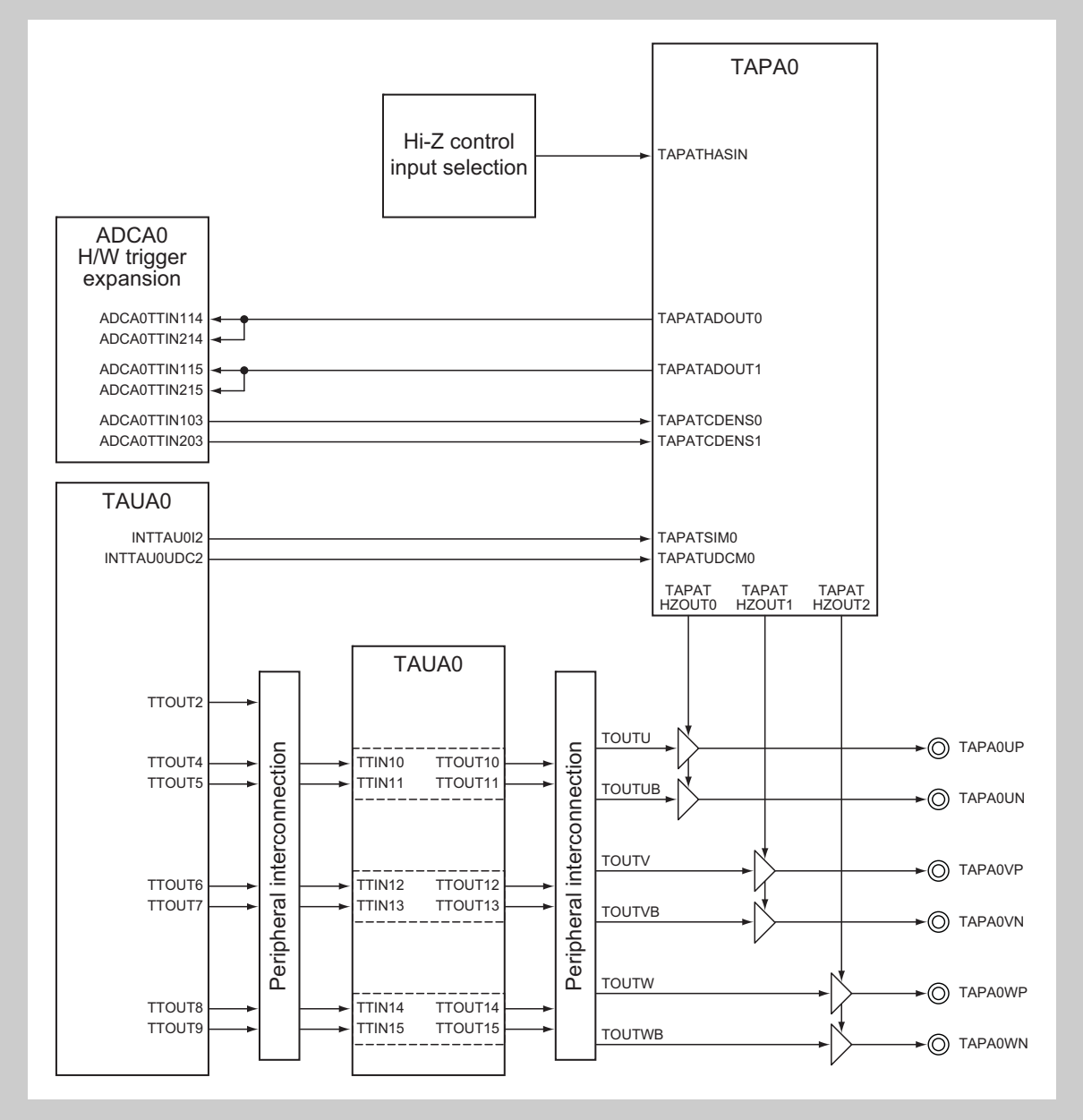


Figure 20-1 Basic structure of motor control

For the generation of the required motor control PWM signals for three-phase motors (TAPA0UP/UN/VP/VN/WP/WN) the Timer Array Unit TAUA0 and the Timer Motor Control Function (TAPA) are used.

Peripheral interconnection circuits are used to set up the required connections between the TAUA0 channels and the motor control outputs.

The Timer Motor Control Function (TAPA) module controls the motor control signals output buffers.

The following sections

- *V850E2/Fx4-H Timer Motor Control Function (TAPA) Features*
- *TAPA0 Hi-Z control input selection*
- *TAPA Functional Overview*
- *TAPA Registers*
- *TAPA Basic Functions*

describe the function of the Timer Motor Control Function (TAPA) module.

Afterwards the sections

- *Three-Phase PWM Output with Dead Time*
- *High-accuracy Triangle PWM Output with Dead Time*
- *Delay Pulse Output with Dead Time*

describe how to set up the involved modules for a dedicated motor control application.

20.2 V850E2/Fx4-H Timer Motor Control Function (TAPA) Features

This chapter contains a generic description of the Timer Motor Control Function (TAPA).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

Instances

This microcontroller has following number of instances of the Timer Motor Control Function.

Table 20-1 Instances of TAPA

Timer Motor Control Function	
Instance	1
Name	TAPA0

Instances index n

Throughout this chapter, the individual instances of a Timer Motor Control Function is identified by the index "n" (n = 0), for example, TAPAnCTL0 for the TAPAn control register 0.

Register addresses

All TAPAn register addresses are given as address offsets to the individual base address <TAPAn_base>.
The base address <TAPAn_base> of each TAPAn is listed in the following table:

Table 20-2 Register base addresses <TAPAn_base>

TAPAn instance	<TAPAn_base> address
TAPA0	FF81 5000_H

Clock supply All Timer Motor Control Functions provide one clock input:

Table 20-3 TAPAn clock supply

TAPAn instance	TAPAn clock	Connected to
TAPA0	PCLK	Clock Controller CKSCLK_006

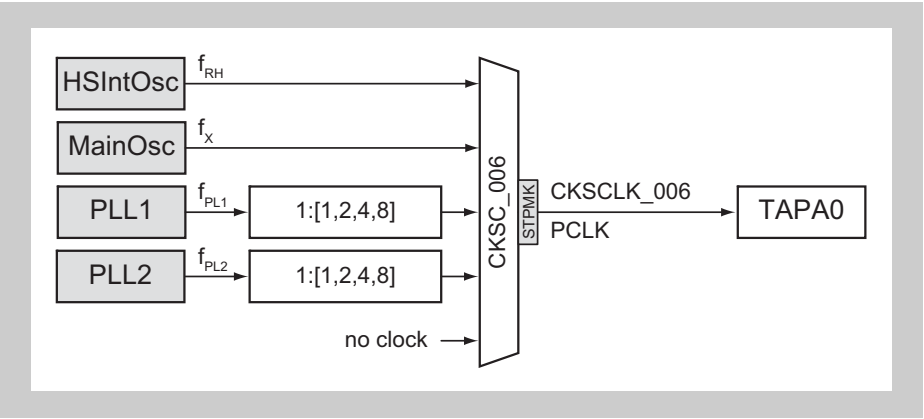


Figure 20-2 TAPA clock supply

Interrupts The Timer Motor Control Function can generate the following interrupt requests:

Table 20-4 TAPAn interrupt requests

TAPAn signals	Function	Connected to
TAPA0:		
TAPATIVLY0	TAPA0 valley interrupt 0	Interrupt Controller INTTAPA0IVLY0
TAPATIVLY1	TAPA0 valley interrupt 1	no connected
TAPATIPEK0	TAPA0 peak interrupt 0	Interrupt Controller INTTAPA0IPEK0
TAPATIPEK1	TAPA0 peak interrupt 1	no connected

TAPA H/W reset The Timer Motor Control Function and their registers are initialized by the following reset signal:

Table 20-5 TAPAn reset signal

TAUAn	Reset signal
TAPA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

Internal signals The internal signal connections of the Timer Motor Control Function are listed in the following table.

Table 20-6 TAPAn internal signal connections

TAPAn signal	Function	Connected to
TAPA0:		
TAPATHSIN	TAPA0 Hi-Z input signal	Hi-Z control signal TAPA0HASIN[ab]
$\overline{\text{TAPATHZOUT0}}$	TAPA0 Hi-Z period output 0 signal	Motor control signals TAPA0UP/ TAPA0UN output buffers Hi-Z control signal TAPA0HZOUT0[a]
$\overline{\text{TAPATHZOUT1}}$	TAPA0 Hi-Z period output 1 signal	Motor control signals TAPA0VP/ TAPA0VN output buffers Hi-Z control signal TAPA0HZOUT1[a]
$\overline{\text{TAPATHZOUT2}}$	TAPA0 Hi-Z period output 2 signal	Motor control signals TAPA0WP/ TAPA0WN output buffers Hi-Z control signal TAPA0HZOUT2[a]
TAPATSIM0	TAUA master channel interrupt signal 0	TAUA0 INTTAU0I2
TAPATUDCM0	TAUA master up/down signal output	TAUA0 TAUA0UDC2
TAPATSIM1	TAUA master channel interrupt signal 1	not connected
TAPATUDCM1	TAUA master up/down signal 1	not connected
TAPATTOEM0	TAUA master timer enable signal 0	not connected
TAPATTOEM1	TAUA master timer enable signal 1	not connected
TAPATCDENM0	TAUA master cycle detection signal 0	not connected
TAPATCDENM1	TAUA master cycle detection signal 1	not connected
TAPATCDENS0	TAUA slave 0 match detection input	ADCA0 H/W trigger expansion ADCA0TTIN103[c]
TAPATCDENS1	TAUA slave 1 match detection input	ADCA0 H/W trigger expansion ADCA0TTIN203[c]
TAPATADOUT0	A/D Converter trigger signal 0	ADCAn H/W trigger expansion[d]
TAPATADOUT1	A/D Converter trigger signal 1	ADCAn H/W trigger expansion[d]

a) Refer to *20.3 "TAPA0 Hi-Z control input selection" on page 1581* for details.

b) This input signal is passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

c) These signals are selected by the A/D Converter ADCA0 H/W trigger selection. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

d) These signals can be used as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

20.3 TAPA0 Hi-Z control input selection

In order to stop output of the motor control signals in case of errors several error events can be enabled to switch the motor control signals output buffers into a high impedance state, as show in the diagram below.

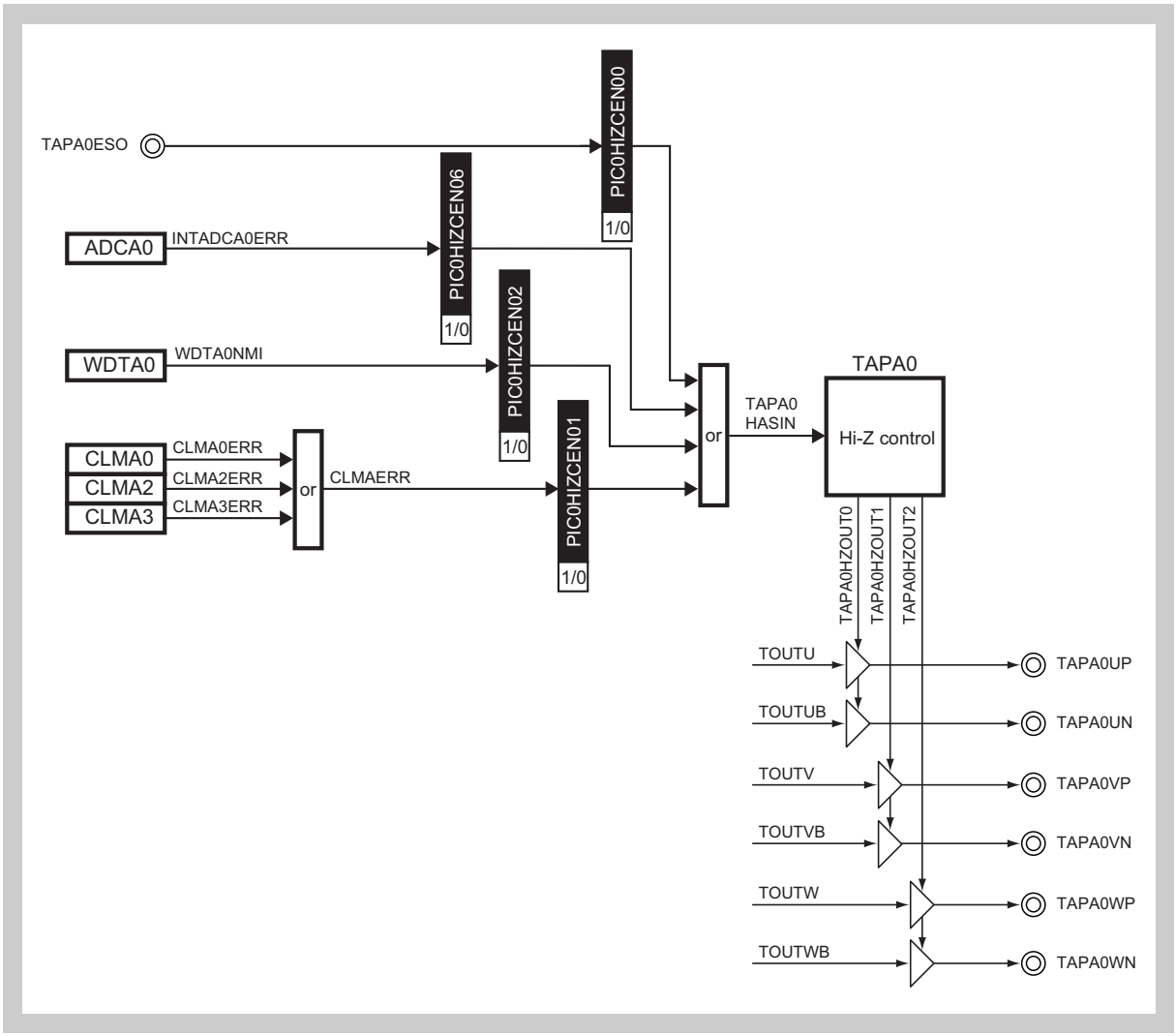


Figure 20-3 Hi-Z CONTROL Block diagram

Hi-Z control can be enabled for the

- external signal TAPA0ESO
- A/D COnverter ADCA0 error signal ADCA0ERR
- Window Watchdog Timer WDTA0 non maskable interrupt WDTA0NMI
- Clock Monitors CLMA0, CLMA2 and CLMA3 error signals CLMAnERR, combined to the CLMAERR Hi-Z control signal

For details about these signals refer to the respective module description.

(1) PIC0HIZCEN0 - Hi-Z output control register

The PIC0HIZCEN0 register selects the Hi-Z output control input signal of TAPA0.

Access This register can be read or written in 8-bit units.

Address FF81 C0B4$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	PIC0HIZCEN06	0	0	0	PIC0HIZCEN02	PIC0HIZCEN01	PIC0HIZCEN00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-7 PIC0HIZCENn register contents

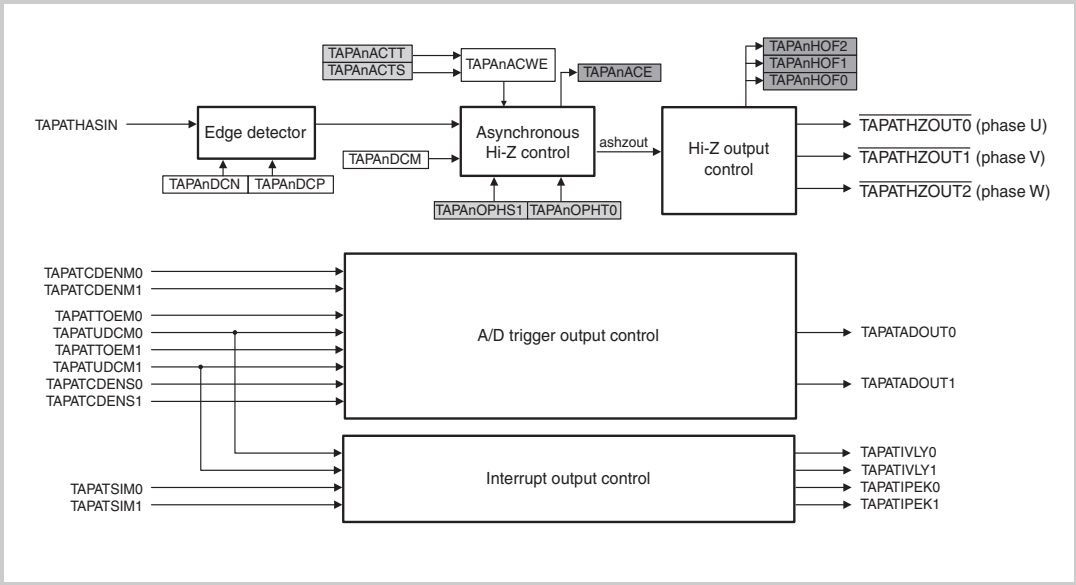
Bit position	Bit name	Function
6	PIC0HIZCEN06	Hi-Z output control by INTADCA0ERR interrupt signal enable: 0: Disable 1: Enable
2	PIC0HIZCEN02	Hi-Z output control by the WDTA0TNMI interrupt signal enable: 0: Disable 1: Enable
1	PIC0HIZCEN01	Hi-Z output control by the CLMAERR error signal enable: 0: Disable 1: Enable
0	PIC0HIZCEN00	Hi-Z output control by the TAPA0ESO external input enable: 0: Disable 1: Enable

20.4 Functional Overview

Function overview The timer motor control function (TAPA) is to be connected to Timer Array Unit A (TAUA) to implement motor systems.

- Asynchronous Hi-Z control functions
  Timer output for motor control can be forcibly stopped by asynchronously setting its level to high-impedance.
- INT signal output selection function
  Two types of interrupts, peak interrupts and valley interrupts, can be output based on the INT signal output by the TAUA.
- A/D conversion trigger selection function
  Two A/D conversion trigger signals can be output based on the INT signal output by the TAUA.

20.4.1 Block diagram



20.4.2 Peak and valley interrupts - Peak and valley of timer counter

In this document, the period from a TAUA up status (counting-up status) to generation of INT from the master channel is defined as a peak period, and this INT is defined as a peak interrupt.
In contrast, the period from a TAUA down status (counting-down status) to generation of INT from the master channel is defined as a valley period, and this INT is defined as a valley interrupt.

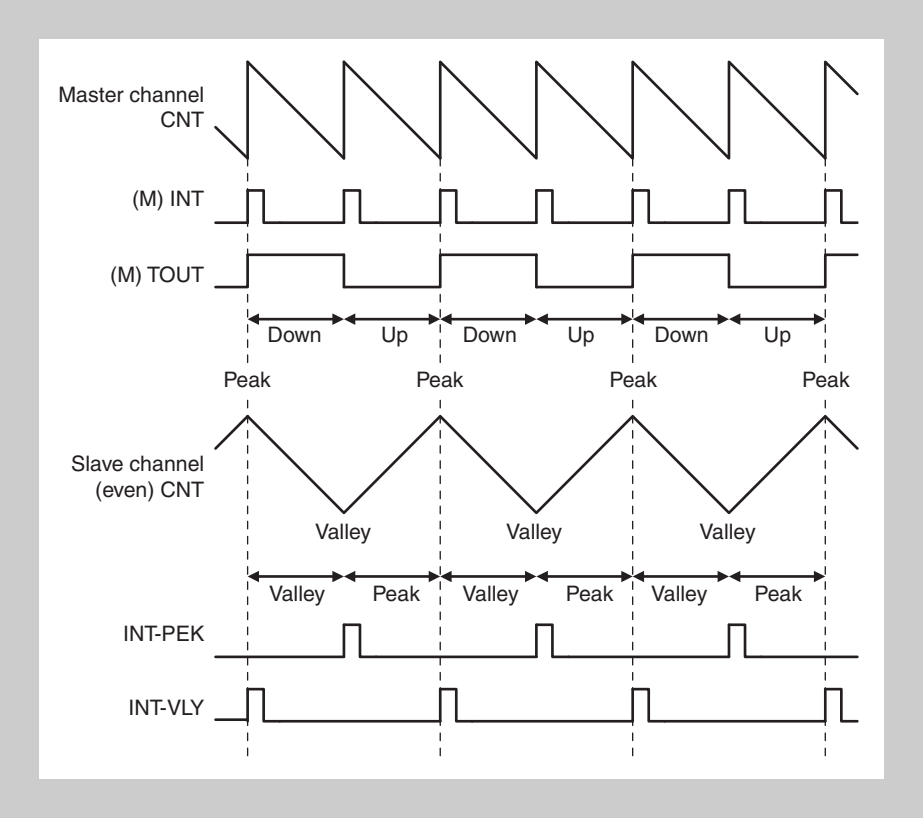


Figure 20-4 Peak and valley interrupts

20.5 Registers

The timer motor control function is controlled and operated by means of the following registers.

20.5.1 Registers overview

Table 20-8 Control registers overview

Register name	Shortcut	Address
Control register 0	TAPAnCTL0	<TAPAn_base> + 20_H
Control register 1	TAPAnCTL1	<TAPAn_base> + 24_H
Flag register	TAPAnFLG	<TAPAn_base> + 00_H
Asynchronous Hi-Z control write enable register	TAPAnACWE	<TAPAn_base> + 04_H
Asynchronous Hi-Z control start trigger register	TAPAnACTS	<TAPAn_base> + 08_H
Asynchronous Hi-Z control stop trigger register	TAPAnACTT	<TAPAn_base> + $0C_H$
Hi-Z start trigger register	TAPAnOPHS	<TAPAn_base> + 14_H
Hi-Z stop trigger register	TAPAnOPHT	<TAPAn_base> + 18_H

<TAPAn_base> The base addresses <TAPAn_base> of the TAPAn is defined in the first section of this chapter under the key word “Register addresses”.

20.5.2 Registers details

(1) TAPAnCTL0 - Control register 0

This register is used to set up the asynchronous Hi-Z control function.

The values of this register can be rewritten only when TAPAnFLG.TAPAnASCE is 0 and the TAUAnTE bit for the corresponding TAUA's master channel is 0.

Access This register can be read or written in 16-bit units.

Address <TAPAn_base> + 20_H

Initial value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	TAPAnDCM	TAPAnDCN	TAPAnDCP	0	0
R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R	R

Table 20-9 TAPAnCTL0 register contents

Bit position	Bit name	Function
4	TAPAnDCM	Clear condition configuration bit This control bit specifies the clear conditions for Hi-Z control output. 0: Enables manipulation of TAPAnOPHT0, regardless of the TAPATHASIN signal input level. 1: Enables manipulation of TAPAnOPHT0 *only* if the TAPATHASIN signal input is inactive.
3 to 2	TAPAnDCN, TAPAnDCP	Hi-Z input edge selection bits These are control bits that specify the valid edge of TAPATHASIN.

TAPAnDCN	TAPAnDCP	Description
0	0	Does not detect valid edges.
0	1	Detects a rising edge as the valid edge (active level = high).
1	0	Detects a falling edge as the valid edge (active level = low).
1	1	Setting prohibited

(2) TAPAnCTL1 - Control register 1

This register is used to specify the A/D conversion trigger.

Access This register can be read or written in 8-bit units.

Address <TAPAn_base> + 24_H

Initial value 00_H

7		5	4	3	2	1	0
0	0	0	0	TAPAn ATS3	TAPAn ATS2	TAPAn ATS1	TAPAn ATS0
R	R	R	R	R/W	R/W	R/W	R/W

Table 20-10 TAPAnCTL1 register contents

Bit position	Bit name	Function
3 to 2	TAPAn ATS[3:2]	A/D conversion trigger 1 selection bits These are control bits that specify the A/D conversion trigger output 1 (TAPATADOUT1). TAPAnATS3 / TAPAnATS2 / Description 0 / 0 / INT signal output while the triangle wave is falling (counting down) 0 / 1 / INT signal output while the triangle wave is rising (counting up) 1 / 0 / INT signal output while the triangle wave is rising (counting up) or falling (counting down) 1 / 1 / INT signal and valley interrupt TAPATIVLY1 output while the triangle wave is rising (counting up) or falling (counting down)
1 to 0	TAPAn ATS[1:0]	A/D conversion trigger 0 selection bits These are control bits that specify the A/D conversion trigger output 0 (TAPATADOUT0). TAPAnATS1 / TAPAnATS0 / Description 0 / 0 / INT signal output while the triangle wave is falling (counting down) 0 / 1 / INT signal output while the triangle wave is rising (counting up) 1 / 0 / INT signal output while the triangle wave is rising (counting up) or falling (counting down) 1 / 1 / INT signal and valley interrupt TAPATIVLY0 output while the triangle wave is rising (counting up) or falling (counting down)

(3) TAPAnFLG - Flag register

This flag register is for asynchronous Hi-Z control.

Access This register can be read in 16-bit units.

Address <TAPAn_base> + 00_H

Initial value 0700_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	TAPAn HOF2	TAPAn HOF1	TAPAn HOF0	0	0	0	0	0	0	0	TAPAn ACE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 20-11 TAPAnFLG register contents

Bit position	Bit name	Function
10 to 8	TAPAnHOFm	Hi-Z control monitor bits (m = 0, 1, 2) These bits are used to monitor the Hi-Z control status. 0: Hi-Z is not being controlled. 1: Hi-Z is being controlled.
0	TAPAnACE	Asynchronous Hi-Z control enable bit This bit indicates the status of the asynchronous Hi-Z control signal TAPATHASIN. 0: Indicates that the asynchronous Hi-Z control is stopped. 1: Indicates that the asynchronous Hi-Z control is enabled. The conditions for setting or clearing this bit are as follows: Clear condition: Writing 1 to TAPAnACTT while TAPAnACWE = 1 Set condition: Writing 1 to TAPAnACTS while TAPAnACWE = 1

(4) TAPAnACWE – Asynchronous Hi-Z control write enable register

This register is used to enable writing for asynchronous Hi-Z control.

Access This register can be read or written in 8-bit units.

Address <TAPAn_base> + 04_H

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TAPAn ACWE
R	R	R	R	R	R	R	R/W

Table 20-12 TAPAnACWE register contents

Bit position	Bit name	Function
0	TAPAnACWE	Asynchronous control write enable bit This is a write enable bit for asynchronous Hi-Z control. After 1 is written, this bit is automatically cleared to 0 by writing 1 to TAPAnACTS and TAPAnACTT. 0: Disables writing to TAPAnACTS and TAPAnACTT. 1: Enables writing to TAPAnACTS and TAPAnACTT.

(5) TAPAnACTS - Asynchronous Hi-Z control start trigger register

This register is used to enable the start trigger for asynchronous Hi-Z control.

Access This register is write-only, in 8-bit units. This register is always read as 00_H.

Address <TAPAn_base> + 08_H

Initial value Reading this register returns always 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TAPAn ACTS
W	W	W	W	W	W	W	W

Table 20-13 TAPAnACTS register contents

Bit position	Bit name	Function
0	TAPAnACTS	Asynchronous Hi-Z control start trigger bit This bit enables the start trigger for asynchronous Hi-Z control. The setting of this bit is valid only when TAPAnACWE = 1. 0: Writing 0 to this bit is ignored (no function). 1: Enables asynchronous Hi-Z control and then sets TAPAnACE to 1.

(6) TAPAnACTT - Asynchronous Hi-Z control stop trigger register

This bit enables the stop trigger for asynchronous Hi-Z control.

Access This register is write-only, in 8-bit units.

Address <TAPAn_base> + $0C_H$

Initial value Reading this register returns always 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TAPAn ACTT
W	W	W	W	W	W	W	W

Table 20-14 TAPAnACTT register contents

Bit position	Bit name	Function
0	TAPAn ACTT	Asynchronous Hi-Z control stop trigger bit This bit enables the stop trigger for asynchronous Hi-Z control. The setting of this bit is valid only when TAPAnACWE = 1. 0: Writing 0 to this bit is ignored (no function). 1: Disables asynchronous Hi-Z control and then clears TAPAnACE to 0.

(7) TAPAnOPHS - Hi-Z start trigger register

This software trigger register is used to start Hi-Z control for timer output pins.

Access This register is write-only, in 8-bit units.

Address <TAPAn_base> + 14_H

Initial value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TAPAn OPHS0
W	W	W	W	W	W	W	W

Table 20-15 TAPAnOPHS register contents

Bit position	Bit name	Function
0	TAPAn OPHS0	Hi-Z control start trigger bit This bit starts Hi-Z control for timer output pins. 0: Writing 0 to this bit is ignored (no function). 1: Starts Hi-Z control.

(8) TAPAnOPHT - Hi-Z stop trigger register

This software trigger register is used to stop Hi-Z control for timer output pins.

Access This register is write-only, in 8-bit units.

Address <TAPAn_base> + 18_H

Initial value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	TAPAn OPHT0
W	W	W	W	W	W	W	W

Table 20-16 TAPAnOPHT register contents

Bit position	Bit name	Function
0	TAPAn OPHT0	Hi-Z control stop trigger bit This bit stops Hi-Z control for timer output pins. 0: Writing 0 to this bit is ignored (no function). 1: Stops Hi-Z control. Whether the setting of this bit is valid or invalid depends on the setting of TAPAnCTL0.TAPAnDCM.

20.6 Basic Functions

20.6.1 Asynchronous Hi-Z control function

(1) Purpose

If the operation of the timer motor control function controlled by the CPU becomes abnormal, the rotation of the external motor also becomes abnormal. This function can forcibly set the motor control output to the Hi-Z state upon detection of abnormal motor operation, independent of the CPU control.

(2) Overview

This function forcibly stops TAPAn output through asynchronous Hi-Z control.

When the TAPATHASIN signal becomes active, the levels of the timer output pins are set to Hi-Z, and timer output is forcibly stopped.

If TAPAnCTL0.TAPAnDCM is 0, setting the TAPAnACTT.TAPAnOPHT0 bit stops Hi-Z output control regardless of the TAPATHASIN signal input level.

If TAPAnCTL0.TAPAnDCM is 1, setting the TAPAnACTT.TAPAnOPHT0 bit while the TAPATHASIN signal input level is inactive stops Hi-Z output control.

(3) System configuration example

A system configuration example is shown below, where an external error detection signal (the TAPAnESO0 signal) is used for the Hi-Z control of the TAUA0O10 to TAUA0O15 outputs signals.

When an external error detection signal is received, an interrupt is generated and the level of the motor control timer outputs (TAUA0 channels 10 to 15) is simultaneously set to Hi-Z.

Because the microcontroller might freeze when an error occurs, external error detection signals are continuously processed so that the motor control timer outputs can be set to Hi-Z even if no clock is supplied.

Note that an error is detected only when the valid edge of the error detection signal is detected. Therefore, no error is detected if the output level is fixed and the signal level does not change.

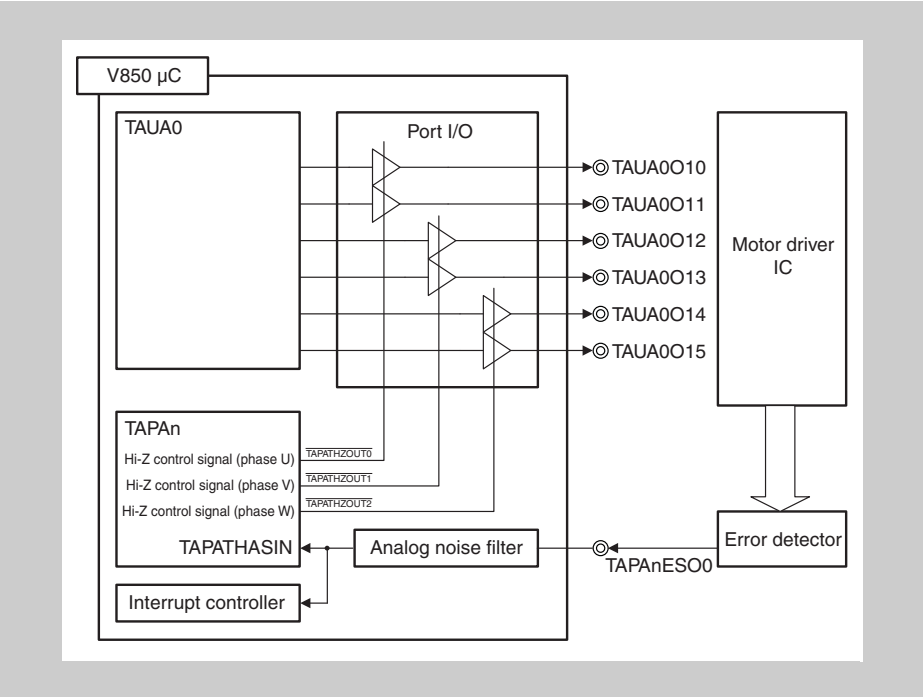


Figure 20-5 Hi-Z control function overbiew

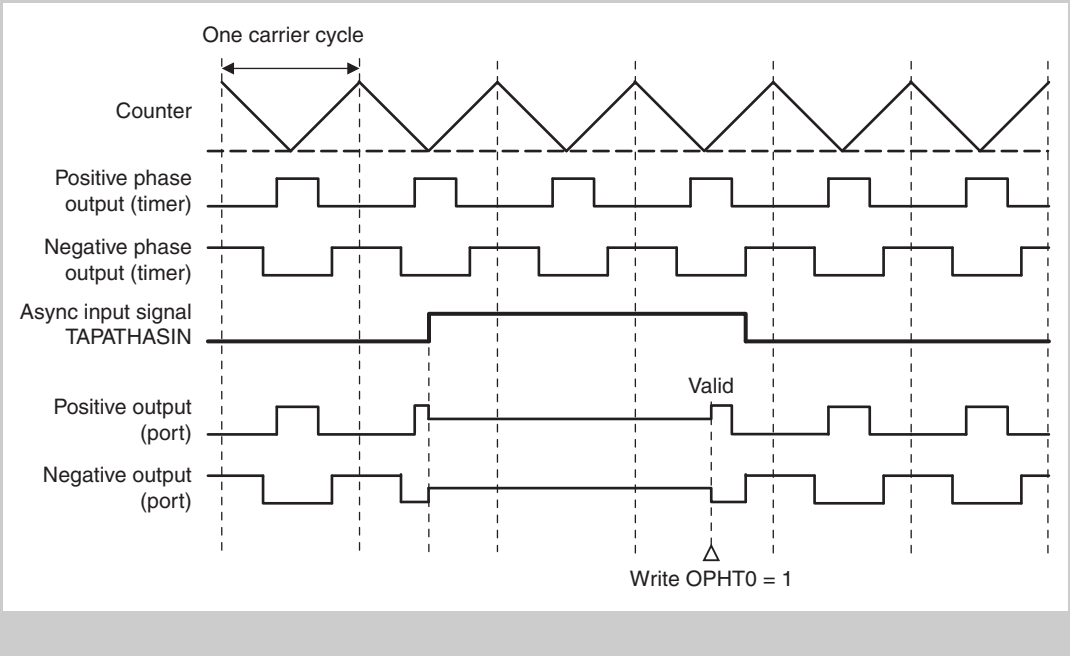
(4) Basic operation

Hi-Z control for timer output pins can be started as follows:

- Detecting the valid edge of TAPATHASIN
- Setting the start trigger bit TAPAnOPHS0 of the Hi-Z control signal

The levels of the timer output pins are set to Hi-Z until the stop trigger bit TAPAnOPHT0 of the Hi-Z control signal is set. Note that whether the setting of TAPAnOPTH0 is valid or invalid depends on the setting of TAPAnCTL0.TAPAnDCM.

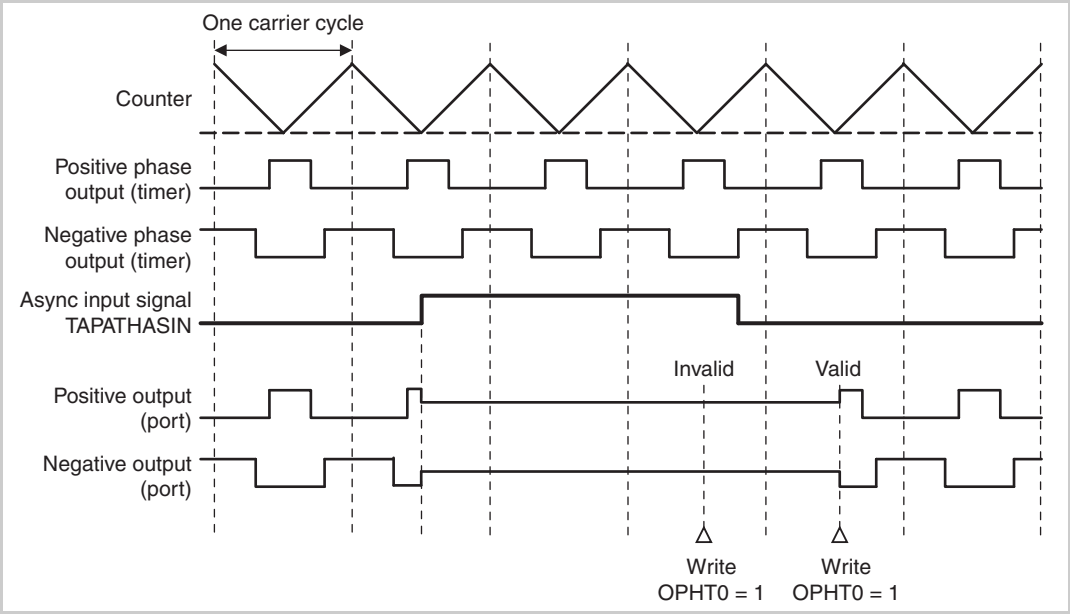
(a) Operation when TAPAnCTL0.TAPAnDCM = 0, TAPAnDCP = 0, and TAPAnDCN = 0



The timer outputs are forcibly stopped (Hi-Z output by port control) when the valid edge of the asynchronous input signal TAPATHASIN is detected.

The timer outputs restart when 1 is written to the Hi-Z stop trigger bit TAPAnOPHT.TAPAnOPHT0, regardless of the level of TAPATHASIN.

(b) Operation when TAPAnCTL0.TAPAnDCM = 1, TAPAnDCP = 1, and TAPAnDCN = 0



The timer outputs are forcibly stopped (Hi-Z output by port control) when the valid edge of asynchronous input signal TAPATHASIN is detected.

Writing 1 to the Hi-Z stop trigger bit TAPAnOPHT0 is ignored while the asynchronous input signal TAPATHASIN is active (high level because TAPAnDCP is 1).

The timer outputs restart when 1 is written to the Hi-Z stop trigger bit TAPAnOPHT0 after TAPATHASIN becomes inactive (low level because TAPAnDCP is 1).

(5) Asynchronous Hi-Z control using software trigger

TAPA supports Hi-Z control using a software trigger.

Hi-Z control for timer outputs is possible by using the Hi-Z control start trigger bit TAPAnOPHS.TAPAnOPHS0 and Hi-Z control stop trigger bit TAPAnOPHT.TAPAnOPHT0.

(a) Function of Hi-Z control start trigger bit TAPAnOPHS.TAPAnOPHS0

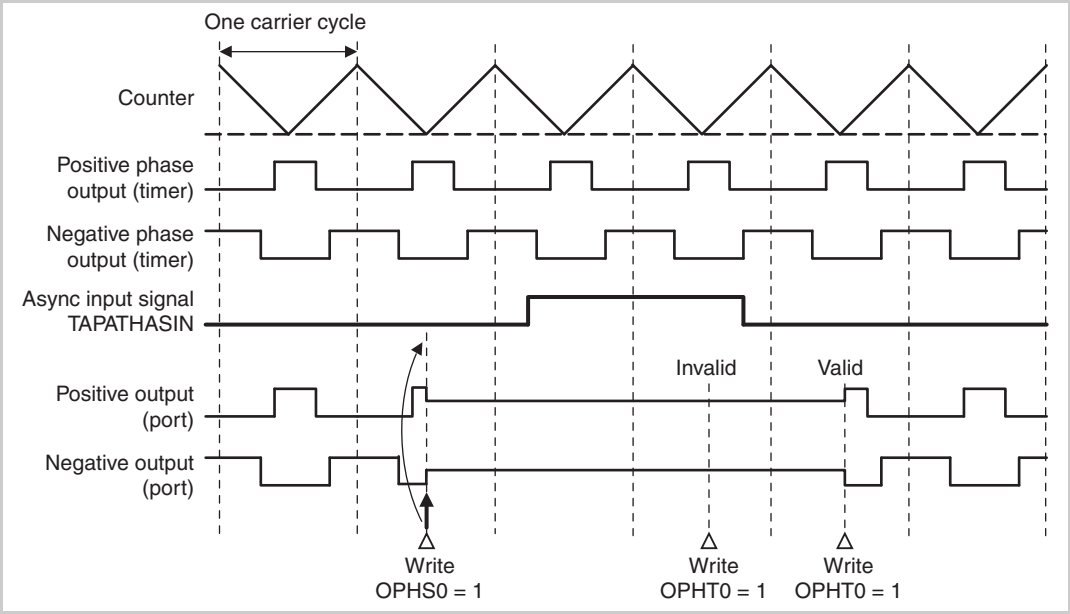
TAPAnDCM	Function
0/1	Writing 1 to TAPAnOPHS0 starts Hi-Z control and forcibly stops the timer outputs (Hi-Z output by port control).

(b) Function of Hi-Z control stop trigger bit TAPAnOPHT.TAPAnOPHT0

Whether the Hi-Z control stop trigger is valid or invalid depends on the conditions below:

TAPAnDCM	Function
0	Writing 1 to TAPAnOPHT0 stops Hi-Z control and restarts timer outputs.
1	If TAPATHASIN is inactive, writing 1 to TAPAnOPHT0 stops Hi-Z control and restarts timer outputs. If TAPATHASIN is active, writing 1 to TAPAnOPHT0 is ignored.

(c) Operation when TAPAnCTL0.TAPAnDCM = 1, TAPAnDCP = 1, and TAPAnDCN = 0



The timer outputs are forcibly stopped (Hi-Z output by port control) when 1 is written to TAPAnOPHS0.

After that, the levels of the timer outputs remain Hi-Z even if a rising edge of TAPATHASIN is detected.

Writing to TAPAnOPHT0 is ignored while TAPATHASIN is active (high level because TAPAnDCN is 0 and TAPAnDCP is 1).

After detection of a falling edge of TAPATHASIN, the timer outputs restart when 1 is written to TAPAnOPHT0 while TAPATHASIN is inactive (low level because TAPAnDCN is 0 and TAPAnDCP is 1).

(6) Operating procedure

The operating procedure for the asynchronous input Hi-Z control function is shown below:

	Operation	Status of TAPA
Initial setup	Set up the TAPAnCTL0 register. Specify TAPAnDCP and TAPAnDCN to select the input edge. Specify TAPAnDCM to select the clear mode.	Asynchronous Hi-Z control stopped (TAPAnFLG.TAPAnACE = 0)
Start operation	Set up the TAPAnACWE register. Set TAPAnACWE to 1. Set up the TAPAnACTS register. Set TAPAnACTS to 1.	Writing to TAPAnACTS is enabled. Asynchronous Hi-Z control enabled (TAPAnFLG.TAPAnACE = 1)
During operation	Hi-Z control for the timer function outputs can be started by controlling the following: • TAPAnOPHS register • Hi-Z control asynchronous input signal TAPATHASIN Hi-Z control for the timer function outputs can be stopped by controlling the following: • TAPAnOPHT register (If TAPAnDCM is 1, control by the TAPAnOPHT register is enabled only while TAPATHASIN is inactive.) The TAPA operating status can always be read using the TAPAnFLG register.	Hi-Z control for the timer output pins is started by detecting the valid edge of the Hi-Z control asynchronous input signal (TAPATHASIN) or by setting the start trigger bit TAPAnOPHS0. Hi-Z control for the timer output pins is stopped by setting the stop trigger bit TAPAnOPHT0 according to the operation mode specified by TAPAnCTL0.TAPAnDCM.
Stop operation	Set up the TAPAnACWE register. Set TAPAnACWE to 1. Set up the TAPAnACTT register. Set TAPAnACTT to 1.	Writing to TAPAnACTT is enabled. Asynchronous Hi-Z control stopped (TAPAnFLG.TAPAnACE = 0)

Restart

20.6.2 INT signal output selection function

(1) Configuration of the INT signal output selection function

This function generates the peak interrupt TAPATIPEKm and valley interrupt TAPATIVLYm by using the TAPATSIMm signal, which is connected to the INT signal on the TAUA's triangular carrier cycle generation channel (master).

For the connection destination of TAPATSIMm, see the table TAPAn internal signals in the first section of this chapter.

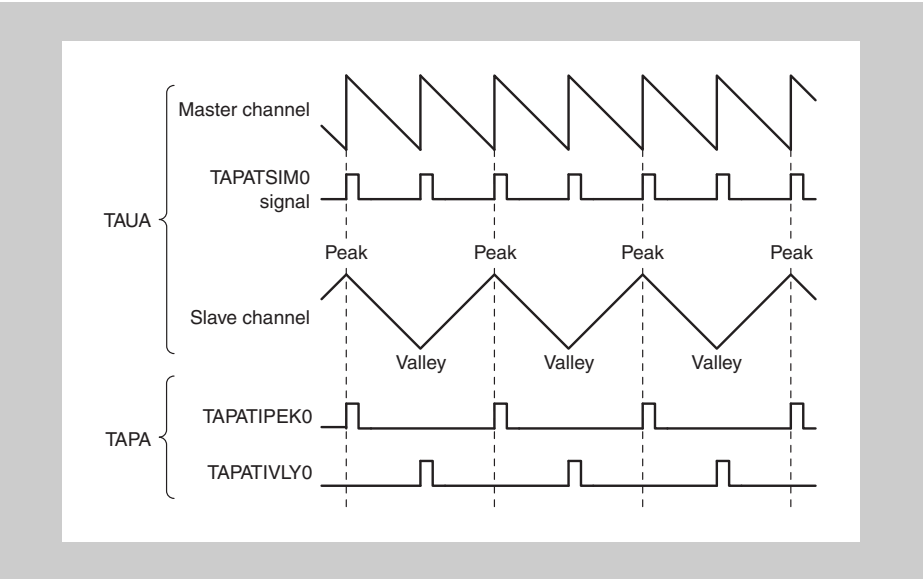


Figure 20-6 Basic timing chart of signals for the INT signal output selection function

Triangular carrier cycles are generated on the master channel.

An interrupt generated on the master channel in each half triangular carrier cycle is connected to TAPATSIMm, and then input to TAPAn. From TAPATSIMm, the peak interrupt TAPATIPEKm is generated at the peak of the triangle wave, and the valley interrupt TAPATIVLYm is generated at the valley of the triangle wave.

Caution TAPATIPEKm and TAPATIVLYm are generated regardless of the TAUA mode.

When not using these peak and valley interrupts, mask them by using the ICTAPAnIPEKm and ICTAPAnIVLYm registers, respectively.

20.6.3 A/D conversion trigger selection function

This function outputs the A/D conversion trigger signals TAPATADOUT0 and TAPATADOUT1 from the signals TAPATCDENS0 and TAPATCDENS1, which are connected to a compare match interrupt with TAUA's triangular carrier cycle.

(1) Configuration of A/D conversion trigger selection function

Table 20-17 Signals used for TAPATADOUT generation

Output signal	Slave match detection signal	Valley interrupt signal
TAPATADOUT0	TAPATCDENS0	TAPATIVLY0
TAPATADOUT1	TAPATCDENS1	TAPATIVLY0

Table 20-18 Operation of TAPATADOUT0 according to the setting of TAPAnCTL1.TAPAnATS[3:0]

TAPAnATS3	TAPAnATS2	Description
0	0	Outputs the INT signal while the triangle wave is falling (counting down).
0	1	Outputs the INT signal while the triangle wave is rising (counting up).
1	0	Outputs the INT signal while the triangle wave is rising (counting up) or falling (counting down).
1	1	Outputs the INT signal and valley interrupt TAPATIVLY1 while the triangle wave is rising (counting up) or falling (counting down).

TAPAnATS1	TAPAnATS0	Description
0	0	Outputs the INT signal while the triangle wave is falling (counting down).
0	1	Outputs the INT signal while the triangle wave is rising (counting up).
1	0	Outputs the INT signal while the triangle wave is rising (counting up) or falling (counting down).
1	1	Outputs the INT signal and valley interrupt TAPATIVLY0 while the triangle wave is rising (counting up) or falling (counting down).

(2) Waveforms of A/D conversion trigger output control operation in triangle PWM mode

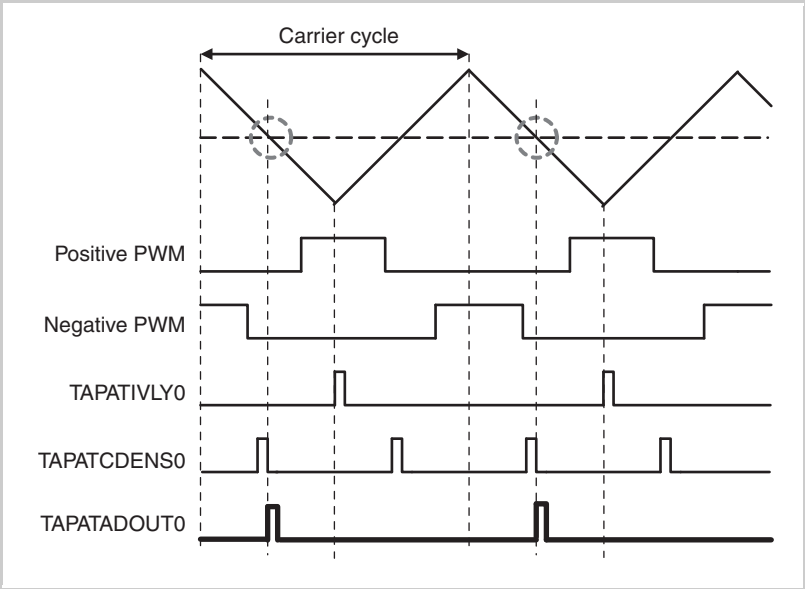


Figure 20-7 TAPAnATS[1:0] = 00_B: Output of INT signal while the triangle wave is falling (counting down)

While the triangle wave is falling (counting down), the signals TAPATCDNS0 and TAPATCDNS1 are output as the A/D conversion trigger signals TAPATADOUT0 and TAPATADOUT1.

In this case, no A/D conversion trigger signal is output while the triangle wave is rising (counting up).

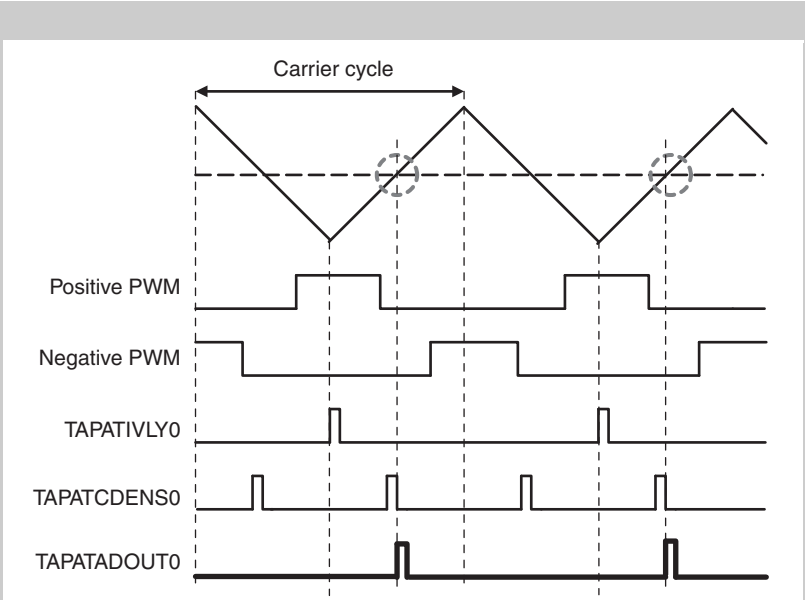


Figure 20-8 TAPAnATS[1:0] = 01_B: Output of INT signal while the triangle wave is rising (counting up)

While the triangle wave is rising (counting up), the TAPATCDNS0 and TAPATCDNS1 signals are output as A/D conversion trigger signals TAPATADOUT0 and TAPATADOUT1.

In this case, no A/D conversion trigger signal is output while the triangle wave is falling (counting down).

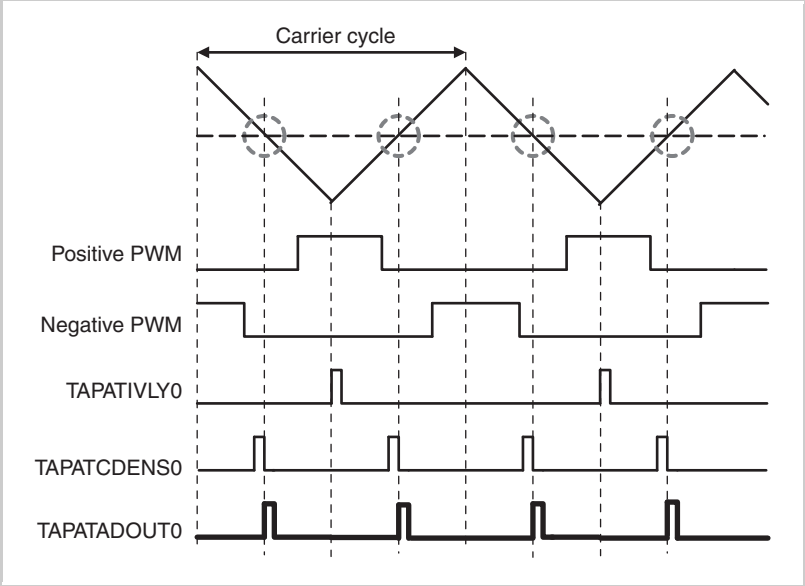


Figure 20-9 TAPAnATS[1:0] = 10_B: Output of INT signal while the triangle wave is rising (counting up) or falling (counting down)

The signals TAPATCDNS0 and TAPATCDNS1 are output as the A/D conversion trigger signals TAPATADOUT0 and TAPATADOUT1.

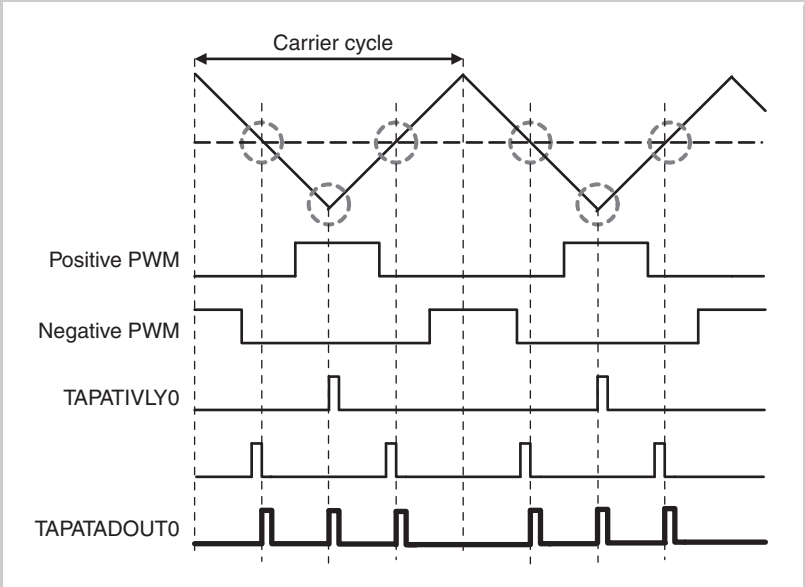


Figure 20-10 TAPAnATS[1:0] = 11_B: Output of INT signal and valley interrupt while the triangle wave is rising (counting up) or falling (counting down)

The signals TAPATCDNS0 and TAPATCDNS1 and valley interrupt TAPATIVLY0 are output as the A/D conversion trigger signals TAPATADOUT0 and TAPATADOUT1.

(3) Operating procedure for A/D conversion trigger selection function

The operating procedure for the A/D conversion trigger selection function is shown below.

	Operation	Status of TAUA and TAPA
Initial setup	Initialize TAUA. Specify the timer operation mode. Set up the TAPAnCTL1 register. Specify TAPAnATS[1:0] (TAPATADOUT0 setting). Specify TAPAnATS[3:2] (TAPATADOUT1 setting).	TAUA and TAPA stop the operation.
Start operation	Start the TAUA operation.	TAUA starts count operation.
During operation	TAUA operates according to the setting of each function.	The A/D conversion trigger selection function outputs either TAPATADOUT0 according to the setting of TAPAnATS[1:0] or TAPATADOUT1 according to the setting of TAPAnATS[3:2], based on the interrupt TAPATCDENS1 or TAPATCDENS0, which is input from TAUA, and the valley interrupt TAPATIVLY1 or TAPATIVLY0, which is generated by TAPA.
Stop operation	Stop the TAUA operation.	TAUA stops the count operation.

Restart

20.7 Three-Phase PWM Output with Dead Time

20.7.1 Functional overview

This feature generates each of the set signals (assert timing signals) and clear signals (deassert timing signals) once or less per cycle and then uses the results to output a three-phase PWM waveform with dead time.

For the PWM output feature of TAUA, only the clear timing used during each cycle is specified by specifying the duty value, but, for the feature described here, the set timing can also be specified, which allows more flexible PWM output with dead time possible.

20.7.2 Configuration

The unit and channel configuration for this feature are shown below.

Table 20-19 Configuration of three-phase PWM output with dead time

Timer	Timer motor control function
TAUA0 CH2, CH4 to CH15 (used channels fixed)	TAPA0

Note The signal names used in the descriptions below are abbreviations. The actual signal names corresponding to each abbreviation are as follows:

- INTm → INTTAUAnIm (TAUAn channel m interrupt)
- TINm → TAUAnTTINm (TAUAn channel m input)
- TOUTm → TAUAnTOUTm (TAUAn channel m output)
- CDRm → TAUAnCDRm (TAUAn channel m data register)
- CNTm → TAUAnCNTm (TAUAn channel m counter register)

(1) TAUAn configuration

Because CH10, CH12, and CH14 are only used for TOUTm, these channels can be used for features that do not use TOUTm (m = 10, 12, 14).

Table 20-20 TAUA configuration

CH	Function name	M/S[a]	CDR setting	Description
2	PWM output (CH2 is the master channel for CH4 to CH9.)	M	Cycle	
4		S	Duty (U-phase signal setting)	
5		S	Duty (U-phase signal clearing)	
6		S	Duty (V-phase signal setting)	
7		S	Duty (V-phase signal clearing)	
8		S	Duty (W-phase signal setting)	
9		S	Duty (W-phase signal clearing)	
10	Any feature that does not use TOUT10	Any		TOUT10: U-phase output
11	One-phase PWM output	-	Dead time (U phase)	TOUT11: UB-phase output
12	Any feature that does not use TOUT12	Any		TOUT12: V-phase output
13	One-phase PWM output	-	Dead time (V phase)	TOUT13: VB-phase output
14	Any feature that does not use TOUT14	Any		TOUT14: W-phase output
15	One-phase PWM output	-	Dead time (W phase)	TOUT15: WB-phase output

a) M: Master channel, S: Slave channel, -: Standalone

(2) Block diagram

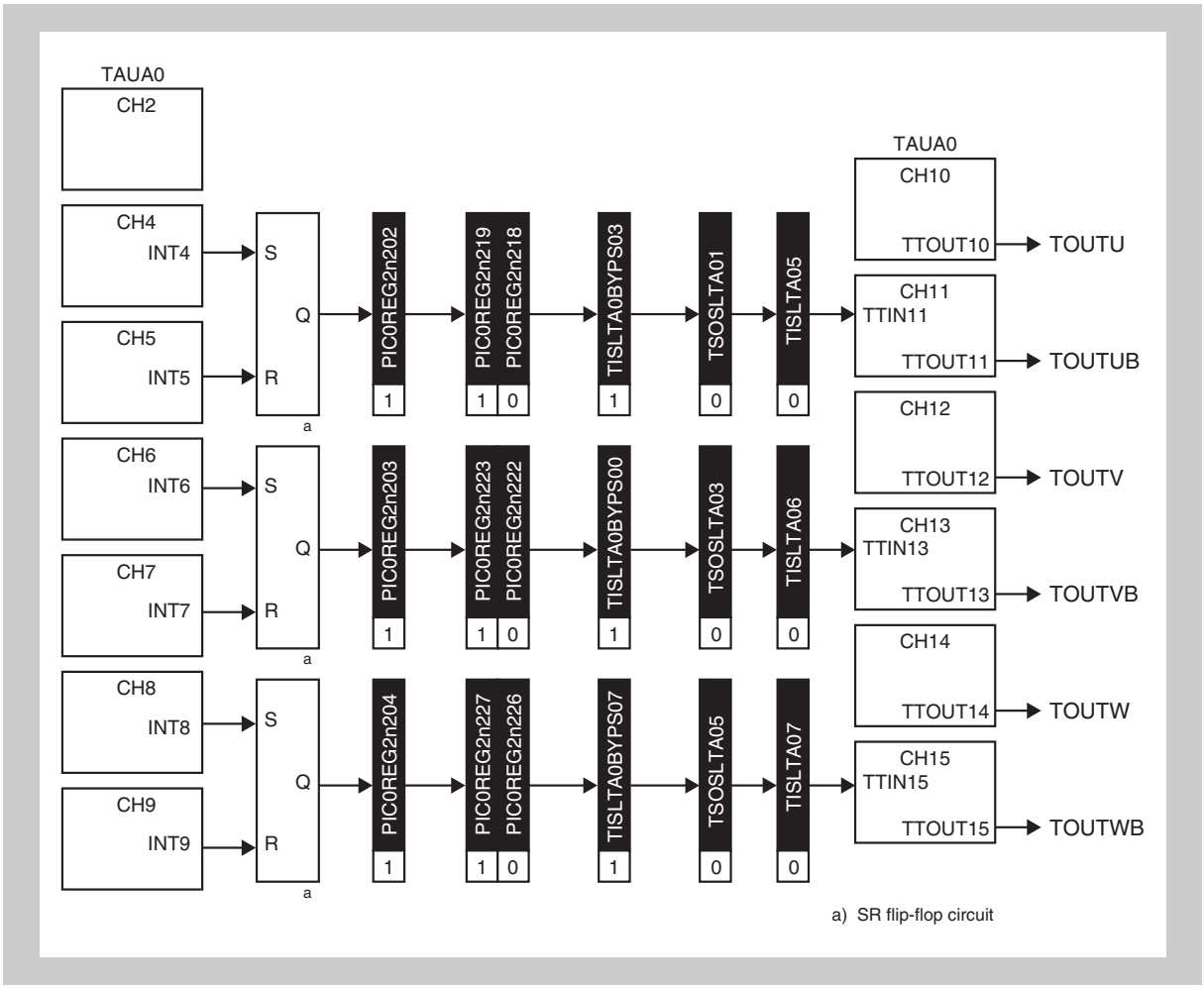


Figure 20-11 Block diagram: Three-phase PWM output with dead time

20.7.3 Operation example

This example shows how to generate each of the set signals and clear signals once or less per cycle and then use the results to output a three-phase PWM waveform with dead time.

This is achieved by combining the following TAUA features:

- PWM output
- One-phase PWM output

In addition, the following peripheral interconnections are used to create the PWM waveform supplied from the set and clear signals generated during PWM output to the input TINm signal (m = 11, 13, or 15) of one-phase PWM output:

- SR flip-flop circuit

Three-phase PWM output is achieved by assigning the one-phase PWM output with dead time achieved using the above features to the U, V, and W phases. Therefore, the set and clear signals of PWM output can be freely specified for each PWM phase. Because the only difference between phases is the assigned channel, only one phase (the U phase) is described below.

(1) PWM output

PWM output uses a combination of CH2, CH4, and CH5.

By specifying the cycle for CDR02, the U-phase set value for CDR04, and the U-phase clear value for CDR05, a set/clear signal is generated for the SR flip-flop circuit that generates the input TIN11 signal of one-phase PWM output from INT04 and INT05.

Instead of CH4 and CH5, which are used for the above described U-phase set/clear signal generation, the V phase uses CH6 and CH7, and the W phase uses CH8 and CH9.

(2) One-phase PWM output

One-phase PWM output is generated from TOUT10 and TOUT11 by using a combination of CH10 and CH11.

By specifying the dead time value for CDR11, a one-phase PWM signal with dead time is output for the TIN11 input.

Similarly, the V phase uses CH12 and CH13 to output a one-phase PWM signal with dead time, while the W phase uses CH14 and CH15.

Caution Specify the same clock for each TAUAn channel that uses the PWM output and one-phase PWM output features.

For details about the TAUA functions, see the chapter *"Timer Array Unit A (TAUA)"*.

(3) SR flip-flop circuit

The PWM waveform supplied to the input TIN11 signal of one-phase PWM output is generated by using the U-phase set signal generated by CH4 of TAUA and the U-phase clear signal generated by CH5.

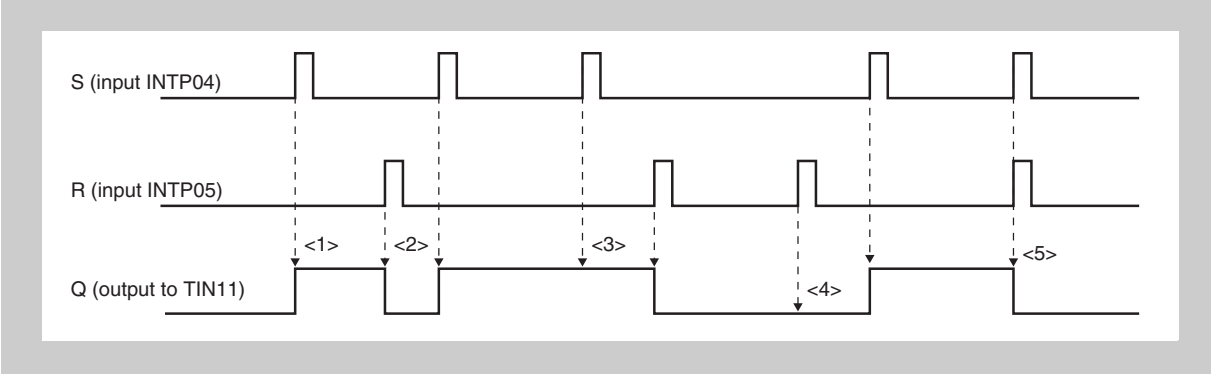


Figure 20-12 SR flip-flop circuit operation timing chart (U-phase example)

- When a signal is input to input S, output Q goes to the high level at the rising edge of S (point <1> in the figure).
- When a signal is input to input R, output Q goes to the low level at the rising edge of R (point <2> in the figure).
- If a signal is input to input S while output Q is at the high level, output Q is not affected (point <3> in the figure).
- If a signal is input to input R while output Q is at the low level, output Q is not affected (point <4> in the figure).
- If a signal is input to input S and input R at the same time, input R is prioritized and output Q goes to the low level at the rising edge of R (point <5> in the figure).

The V phase uses INT06 and INT07 as input to supply a PWM waveform to TIN13, and the W phase uses INT08 and INT09 as input to supply a PWM waveform to TIN15.

The output change timing of the PWM waveform generated during one-phase PWM output is based on PWM output.

The active level output timing set signal and inactive level output timing clear signal of PWM are generated during PWM output. By inputting these signals to the SR flip-flop circuit, a PWM signal that can be changed at any time is generated.

A one-phase PWM signal is output by generating a positive or negative PWM waveform and then adding dead time to it according to changes in the generated PWM signal.

The peripheral interconnections are used to set/clear signal generated during PWM output as the TIN input for one-phase PWM output through the SR flip-flop circuit.

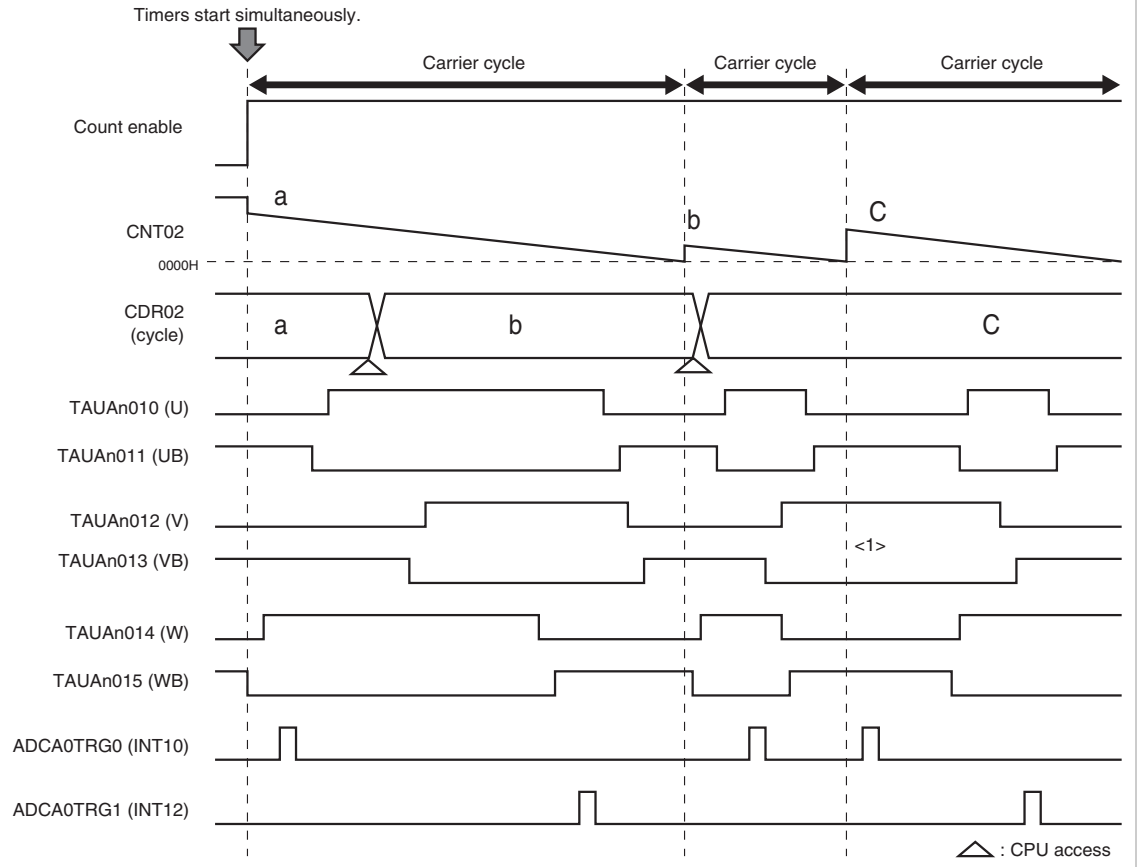


Figure 20-13 Example of three-phase PWM (U/UB, V/VB, W/WB) output with dead time

Figure 20-13 shows a typical example of three-phase PWM output with dead time.

By appropriately setting up the set/clear signal output timing, PWM output that extends across carrier cycles (point <1>) and other types of output are also possible.

In this example, ADCA0TRG0 and ADCA0TRG1 (which are at the bottom) use the CNT and INT signals of CH10 and CH12, which are not used for one-phase PWM output, and the A/D trigger signal is output by performing type-1 A/D trigger output.

In this way, because only the TOUTm signal that performs signal output for the channel performing positive phase output is used during one-phase PWM output, any feature that uses CNTm, CDRm, or INTm can be specified. For details, see the chapter *"Timer Array Unit A (TAUA)"* (m = 10, 12, or 14).

The following figures show timing charts for outputting a three-phase PWM signal with dead time.

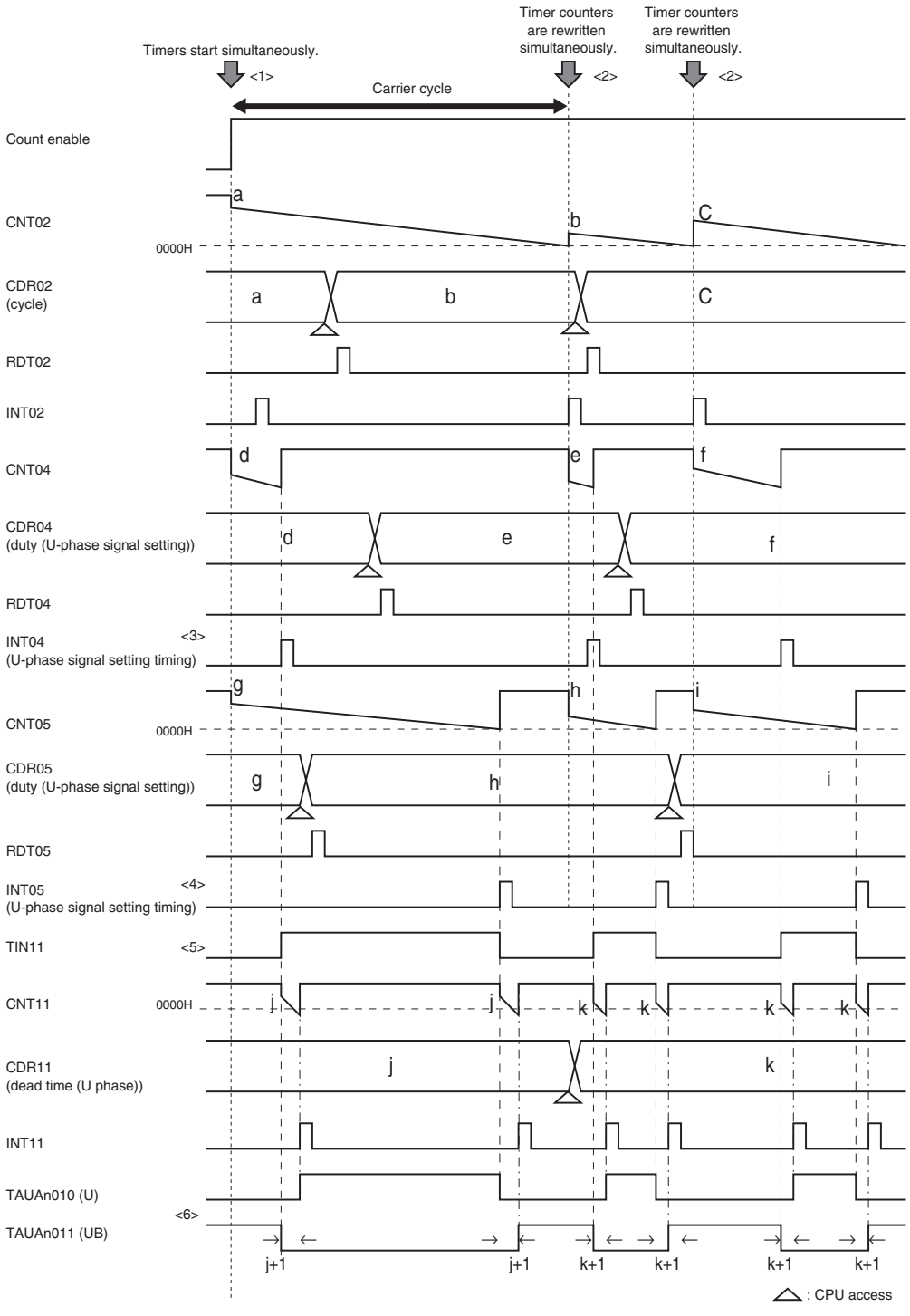


Figure 20-14 Example of one-phase PWM (U phase, UB phase) output with dead time

An operation example of the timer configuration for performing the U-phase PWM output in *Figure 20-14* is provided below.

- By simultaneously starting timers, CH2 (the carrier cycle timer), CH4 (the U-phase set signal output timing timer), and CH5 (the U-phase clear signal output timing timer) are started simultaneously (point <1> in the figure). The CH11 timer is also enabled, but, until a TIN11 edge is detected, which is the count start timing, counting is not performed.
- For CH4 and CH5, when there is a CH2 underflow, the settings from CDR04 and CDR05 are reloaded to CNT04 and CNT05 (point <2> in the figure).
- When there is a CH4 underflow, the U-phase set timing signal (INT04) is generated (point <3> in the figure).
- When there is a CH5 underflow, the U-phase clear timing signal (INT05) is generated (point <4> in the figure).
- The peripheral interconnections supply the output of the SR flip-flop circuit that uses INT04 (the set timing signal) and INT05 (the clear timing signal) as input to the input TIN11 signal of one-phase PWM output (point <5> in the figure).
- During one-phase PWM output, a PWM waveform with dead time is output by detecting a TIN11 edge (point <6> in the figure).

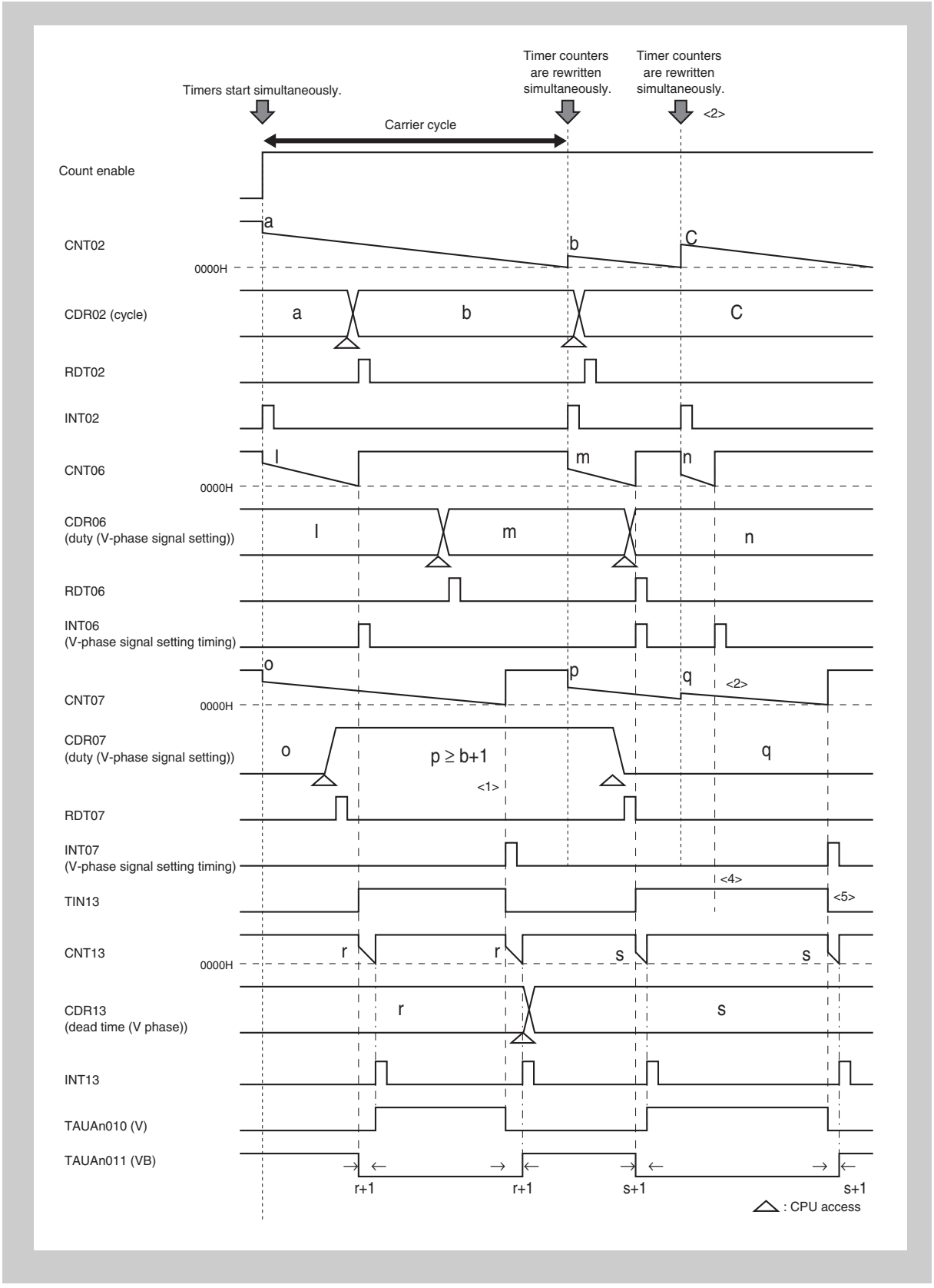


Figure 20-15 Example of one-phase PWM (V phase, VB phase) output with dead time

An operation example of the timer configuration for performing the V-phase PWM output in *Figure 20-15* is provided below.

For details about the operations from when timers are simultaneously started until a one-phase PWM signal is output, see the U-phase operation example.

- If the setting of CH7 (the V-phase clear signal output timing timer), which generates the V-phase clear timing signal (INT07), is greater than the CH2 (the carrier cycle timer) setting (point <1> in the figure)
- Before a V-phase clear timing signal (INT07) is generated by a CH7 underflow, a CH2 (carrier cycle timer) underflow occurs, and the CH7 setting is reloaded (point <2> in the figure).
- This results in consecutive V-phase set timing signals (INT06) being generated instead of the V-phase clear timing signal (INT07) that is supposed to be generated (point <3> in the figure).
- In this case, because the V-phase set timing signal (INT06) is ignored by the RS flip-flop circuit, there is no effect on the PWM output waveform (point <4> in the figure). Therefore, a PWM waveform that extends across carrier cycles is output.
- The PWM output is changed at the timing of the next V-phase clear timing signal (INT07) (point <5> in the figure).

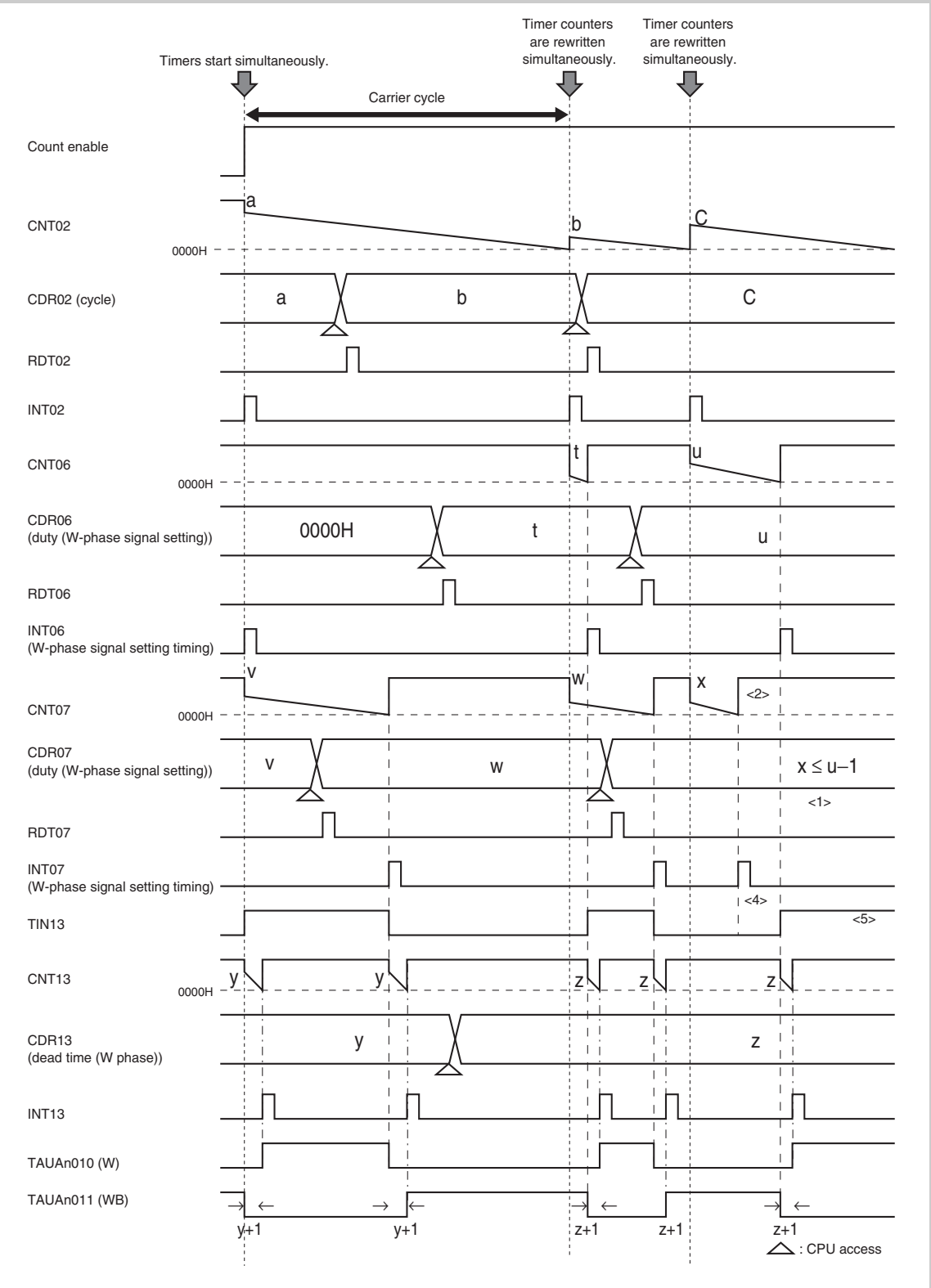


Figure 20-16 Example of one-phase PWM (W phase, WB phase) output with dead time

An operation example of the timer configuration for performing the W-phase PWM output in *Figure 20-16* is provided below.

For details about the operations from when timers are simultaneously started until a one-phase PWM signal is output, see the U-phase operation example.

- If the setting of CH9 (the W-phase clear signal output timing timer), which generates the W-phase clear timing signal (INT09), is less than the CH8 (the W-phase set signal output timing timer) setting (point <1> in the figure)
- Before a W-phase set timing signal (INT08) is generated by a CH8 underflow, a CH9 (W-phase clear signal output timing timer) underflow occurs, and the W-phase set timing signal (INT08) is generated (point <2> in the figure).
- This results in consecutive W-phase clear timing signals (INT09) being generated (point <3> in the figure).
- In this case, because the consecutively generated W-phase clear timing signals (INT09) are ignored by the RS flip-flop circuit, there is no effect on the PWM output waveform (point <4> in the figure).
- The PWM output is changed at the timing of the next W-phase set timing signal (INT08) (point <5> in the figure).

20.7.4 Setup flow

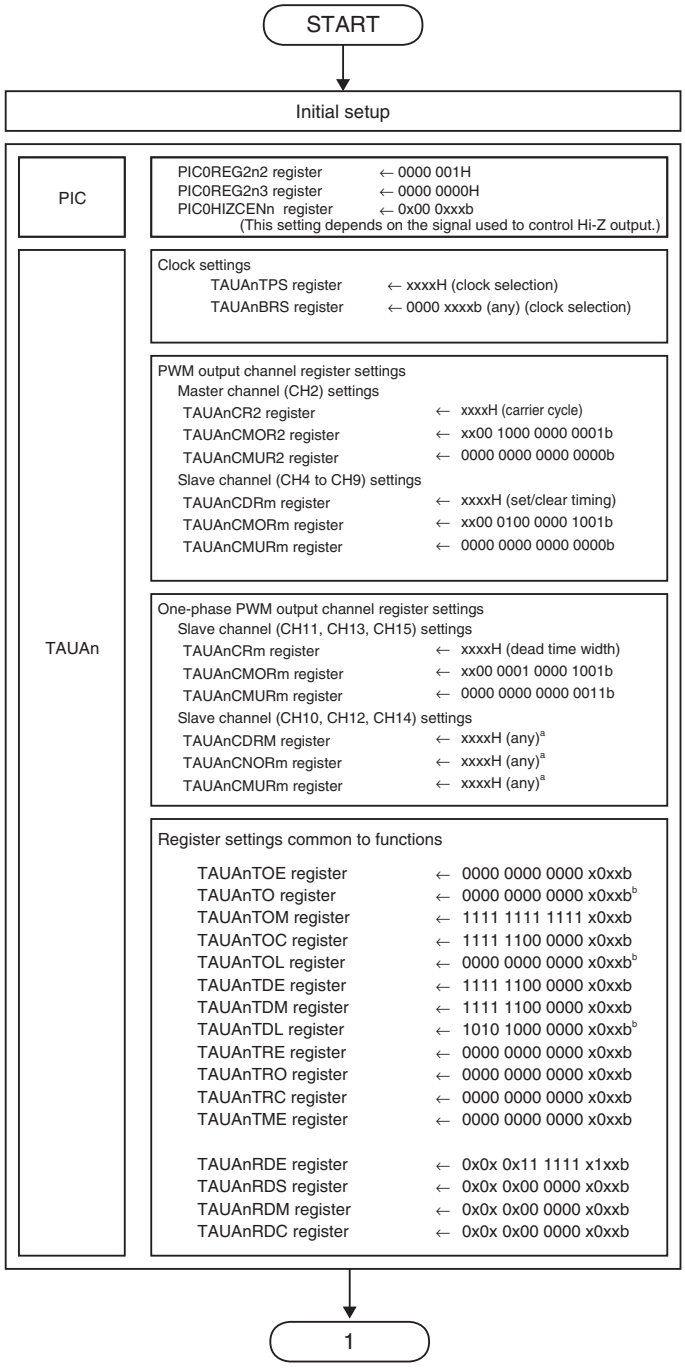


Figure 20-17 Setup flow (active high example)

a) Specify a feature that does not use TOUTm.

b) Change settings according to the active level of the PWM signal to be output.

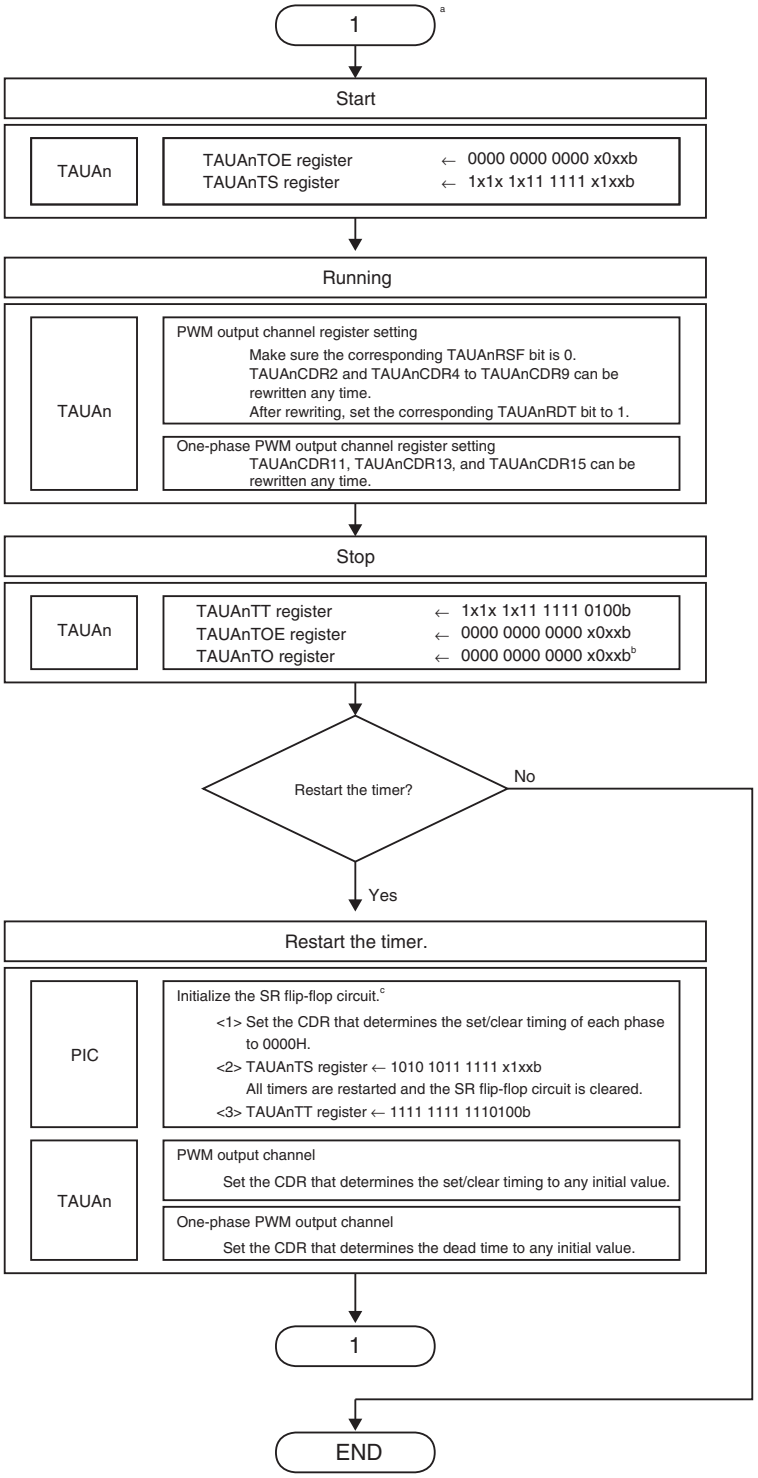


Figure 20-18 Setup flow (active high example) (continued)

a) Specify the selection register (such as TISLTA0BYPS1) and output port to use after specifying the initial settings for the peripheral interconnections and timers.

b) Change settings according to the active level of the PWM signal to be output.

c) If initialization processing is not performed, the SR flip-flop circuit enters the timer-stopped state, and the timer restart output pulse might be output at an unintended level.

20.7.5 Example of setting up operation functions

This section provides example settings for each register.

(1) TAUAn settings (active high example)

Table 20-21 TAUAn: CH2-related (PWM output master channel[a])

Register	Bit position	Bit name	Setting	Remark
TAUAnCMOR2	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	1	
	10 to 8	TAUAnSTS[2:0]	000	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0000	
	0	TAUAnMD0	1	
TAUAnCMUR2	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA PWM output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the master channel and slave channel.

For the TAUAnCMORm register used during PWM output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)*".

Table 20-22 TAUAn: CH4 to CH9-related (PWM output slave channel[a]) (m = 4 to 9)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	100	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA PWM output. For details, see the *"Timer Array Unit A (TAUA)*".

b) The same operation clock must be specified for the master channel and slave channel.

Note For the TAUAnCMORm register used during PWM output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)*".

Table 20-23 TAUAn: CH11, CH13, and CH15-related (one-phase PWM output) (m = 11, 13, or 15)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[a]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	001	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	11	Both rising and falling TINm edges are detected as valid. (High width)

a) Specify the same operation clock settings as for the PWM output master channel (CH2).

Note For the TAUAnCMORm register used during one-phase PWM output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. CH10, CH12, and CH14 can be used with any feature that does not use TOUTm output (such as A/D trigger output). For details, see *"Timer Array Unit A (TAUA)"*.

Table 20-24 Common TAUAn channel settings (1/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTOE	15 to 10	TAUAnTOE15 to TAUAnTOE10	0 1	Disable the timer. Enable the timer.
	9 to 4	TAUAnTOE09 to TAUAnTOE04	0	These are fixed to 0 because TOUT09 to TOUT04 are not used.
	3	TAUAnTOE03	Any	
	2	TAUAnTOE02	0	This is fixed to 0 because TOUT02 is not used.
	1, 0	TAUAnTOE01 TAUAnTOE00	Any	
TAUAnTO	15 to 10	TAUAnTO15 to TAUAnTO10	0[a]	Output a low-level signal to TOUT15 to TOUT10.
	9 to 4	TAUAnTO09 to TAUAnTO04	0	Output a low-level signal to TOUT09 to TOUT04.
	3	TAUAnTO03	Any	
	2	TAUAnTO02	0	Output a low-level signal to TOUT02.
	1, 0	TAUAnTO01 TAUAnTO00	Any	
TAUAnTOM	15 to 4	TAUAnTOM15 to TAUAnTOM04	1	Synchronous channel operation mode
	3	TAUAnTOM03	Any	
	2	TAUAnTOM02	0	Independent channel operation mode
	1, 0	TAUAnTOM01 TAUAnTOM00	Any	
TAUAnTOC	15 to 10	TAUAnTOC15 to TAUAnTOC10	1	Synchronous channel operation mode 2
	9 to 4	TAUAnTOC09 to TAUAnTOC04	0	Synchronous channel operation mode 1
	3	TAUAnTOC03	Any	
	2	TAUAnTOC02	0	Toggle mode
	1, 0	TAUAnTOC01 TAUAnTOC00	Any	
TAUAnTOL	15 to 4	TAUAnTOL15 to TAUAnTOL04	0[a]	Positive logic output (active high)
	3	TAUAnTOL03	Any	
	2	TAUAnTOL02	0	Positive logic output (active high)
	1, 0	TAUAnTOL01 TAUAnTOL00	Any	
TAUAnTDE	15 to 10	TAUAnTDE15 to TAUAnTDE10	1	Enable dead time control.[b]
	9 to 4	TAUAnTDE09 to TAUAnTDE04	0	Disable dead time control.
	3	TAUAnTDE03	Any	
	2	TAUAnTDE02	0	Disable dead time control.
	1, 0	TAUAnTDE01 TAUAnTDE00	Any	

Table 20-24 Common TAUAn channel settings (2/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTDM	15 to 10	TAUAnTDM15 to TAUAnTDM10	1	Output dead time upon detecting a TINm input edge at a lower odd channel.
	9 to 4	TAUAnTDM09 to TAUAnTDM04	0	Invalid because dead time control is disabled.
	3	TAUAnTDM03	Any	
	2	TAUAnTDM02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDM01 TAUAnTDM00	Any	
TAUAnTDL	15	TAUAnTDL15	1[a]	Add dead time to the negative W phase period.
	14	TAUAnTDL14	0[a]	Add dead time to the positive W phase period.
	13	TAUAnTDL13	1[a]	Add dead time to the negative V phase period.
	12	TAUAnTDL12	0[a]	Add dead time to the positive V phase period.
	11	TAUAnTDL11	1[a]	Add dead time to the negative U phase period.
	10	TAUAnTDL10	0[a]	Add dead time to the positive U phase period.
	9 to 4	TAUAnTDL09 to TAUAnTDL04	0	Invalid because dead time control is disabled.
	3	TAUAnTDL03	Any	
	2	TAUAnTDL02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDL01 TAUAnTDL00	Any	
TAUAnTRE	15 to 4	TAUAnTRE15 to TAUAnTRE04	0	Disable real-time output.
	3	TAUAnTRE03	Any	
	2	TAUAnTRE02	0	Disable real-time output.
	1, 0	TAUAnTRE01 TAUAnTRE00	Any	
TAUAnTRO	15 to 4	TAUAnTRO15 to TAUAnTRO04	0	Invalid because real-time output is disabled.
	3	TAUAnTRO03	Any	
	2	TAUAnTRO02	0	Invalid because real-time output is disabled.
	1, 0	TAUAnTRO01 TAUAnTRO00	Any	
TAUAnTRC	15 to 4	TAUAnTRC15 to TAUAnTRC04	0	Do not use this channel to generate the real-time output trigger.
	3	TAUAnTRC03	Any	
	2	TAUAnTRC02	0	Do not use this channel to generate the real-time output trigger.
	1, 0	TAUAnTRC01 TAUAnTRC00	Any	

Table 20-24 Common TAUAn channel settings (3/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTME	15 to 4	TAUAnTME15 to TAUAnTME04	0	Disable modulation output for timer output and real-time output.
	3	TAUAnTME03	Any	
	2	TAUAnTME02	0	Disable modulation output for timer output and real-time output.
	1, 0	TAUAnTME01 TAUAnTME00	Any	
TAUAnRDE	15	TAUAnRDE15	0	Disable simultaneous rewriting.
	14	TAUAnRDE14	Any	
	13	TAUAnRDE13	0	Disable simultaneous rewriting.
	12	TAUAnRDE12	Any	
	11	TAUAnRDE11	0	Disable simultaneous rewriting.
	10	TAUAnRDE10	Any	
	9 to 4	TAUAnRDE09 to TAUAnRDE04	1	Enable simultaneous rewriting.
	3	TAUAnRDE03	Any	
	2	TAUAnRDE02	1	Enable simultaneous rewriting.
	1, 0	TAUAnRDE01 TAUAnRDE00	Any	
TAUAnRDS	15	TAUAnRDS15	0	Do not enable simultaneous rewriting by using another upper channel.
	14	TAUAnRDS14	Any	
	13	TAUAnRDS13	0	Do not enable simultaneous rewriting by using another upper channel.
	12	TAUAnRDS12	Any	
	11	TAUAnRDS11	0	Do not enable simultaneous rewriting by using another upper channel.
	10	TAUAnRDS10	Any	
	9 to 4	TAUAnRDS09 to TAUAnRDS04	0	Enable simultaneous rewriting by using a master channel.
	3	TAUAnRDS03	Any	
	2	TAUAnRDS02	0	Enable simultaneous rewriting by using a master channel.
	1, 0	TAUAnRDS01 TAUAnRDS00	Any	

Table 20-24 Common TAUAn channel settings (4/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnRDM	15	TAUAnRDM15	0	Invalid because simultaneous rewriting is not enabled.
	14	TAUAnRDM14	Any	
	13	TAUAnRDM13	0	Invalid because simultaneous rewriting is not enabled.
	12	TAUAnRDM12	Any	
	11	TAUAnRDM11	0	Invalid because simultaneous rewriting is not enabled.
	10	TAUAnRDM10	Any	
	9 to 4	TAUAnRDM09 to TAUAnRDM04	0	Load the signal when the master channel starts counting.
	3	TAUAnRDM03	Any	
	2	TAUAnRDM02	0	Load the signal when the master channel starts counting.
	1, 0	TAUAnRDM01 TAUAnRDM00	Any	
TAUAnRDC	15	TAUAnRDC15	0	Invalid because simultaneous rewriting is not enabled.
	14	TAUAnRDC14	Any	
	13	TAUAnRDC13	0	Invalid because simultaneous rewriting is not enabled.
	12	TAUAnRDC12	Any	
	11	TAUAnRDC11	0	Invalid because simultaneous rewriting is not enabled.
	10	TAUAnRDC10	Any	
	9 to 4	TAUAnRDC09 to TAUAnRDC04	0	Do not use this channel to generate the simultaneous rewrite trigger.
	3	TAUAnRDC03	Any	
	2	TAUAnRDC02	1	Do not use this channel to generate the simultaneous rewrite trigger.
	1, 0	TAUAnRDC01 TAUAnRDC00	Any	

a) Change the setting according to the used system.

b) These are used to control positive/negative phase waveform output for which even channels are paired with odd channels to perform dead time control. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

(2) Peripheral interconnections settings

Table 20-25 Peripheral interconnections settings

Register	Bit position	Bit name	Setting	Remark
PIC0REG2n2	27, 26	PIC0REG2n227 PIC0REG2n226	1 0	Select the input selected by the PIC0REG2n204 bit.
	23, 22	PIC0REG2n223 PIC0REG2n222	1 0	Select the input selected by the PIC0REG2n203 bit.
	19, 18	PIC0REG2n219 PIC0REG2n218	1 0	Select the input selected by the PIC0REG2n202 bit.
	4	PIC0REG2n204	1	Select the set/clear output according to INTTAUAnI8 and INTTAUAnI9.
	3	PIC0REG2n203	1	Select the set/clear output according to INTTAUAnI6 and INTTAUAnI7.
	2	PIC0REG2n202	1	Select the set/clear output according to INTTAUAnI4 and INTTAUAnI5.

20.7.6 Registers

(1) PIC0REG2n2 - Timer I/O control register 2n2

Access This register can be read or written in 32-bit units.

Address PIC0REG202: FF81 C094$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	PIC0REG2n227	PIC0REG2n226	PIC0REG2n225	PIC0REG2n224
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
PIC0REG2n223	PIC0REG2n222	PIC0REG2n221	PIC0REG2n220	PIC0REG2n219	PIC0REG2n218	PIC0REG2n217	PIC0REG2n216
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	PIC0REG2n204	PIC0REG2n203	PIC0REG2n202	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-26 PIC0REG2n2 register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>27
26</td><td>PIC0REG2n227
PIC0REG2n226</td><td>Select the signal input to TAUAnTTIN15.
<table>
<tr><th>PIC0REG2n227</th><th>PIC0REG2n226</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n204 bit.</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>25
24</td><td>PIC0REG2n225
PIC0REG2n224</td><td>Select the signal input to TAUAnTTIN14.[a]</td></tr>
<tr><td>23
22</td><td>PIC0REG2n223
PIC0REG2n222</td><td>Select the signal input to TAUAnTTIN13.
<table>
<tr><th>PIC0REG2n223</th><th>PIC0REG2n222</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n203 bit</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>21
20</td><td>PIC0REG2n221
PIC0REG2n220</td><td>Select the signal input to TAUAnTTIN12[a].</td></tr>
</table>

Table 20-26 PIC0REG2n2 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>19
18</td><td>PIC0REG2n219
PIC0REG2n218</td><td>Select the signal input to TAUAnTTIN11.
<table>
<tr><th>PIC0REG2n219</th><th>PIC0REG2n218</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n202 bit</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>17
16</td><td>PIC0REG2n217
PIC0REG2n216</td><td>Select the signal input to TAUAnTTIN10[a].
See chapter “High-accuracy Triangle PWM Output with Dead Time” for details.</td></tr>
<tr><td>4</td><td>PIC0REG2n204</td><td>Select the signal input to TAUAnTTIN15[b].
0: Select TAUAnTTOUT9.
1: Select the set/clear output according to INTTAUAnI8 and INTTAUAnI9.</td></tr>
<tr><td>3</td><td>PIC0REG2n203</td><td>Select the signal input to TAUAnTTIN13[b].
0: Select TAUAnTTOUT7
1: Select the set/clear output according to INTTAUAnI6 and INTTAUAnI7.</td></tr>
<tr><td>2</td><td>PIC0REG2n202</td><td>Select the signal input to TAUAnTTIN11[b].
0: Select TAUAnTTOUT5
1: Select the set/clear output according to INTTAUAnI4 and INTTAUAnI5.</td></tr>
</table>

a) See chapter “High-accuracy Triangle PWM Output with Dead Time” for details
b) See chapter “Delay Pulse Output with Dead Time” for details.

Caution Some of the bits defined as 0 in the PIC0REG2n2 register are defined for the other timers. For such bits, use the bit definition of those timers.

(2) TISLTA0BYPS1 - TAUA input selection register

The TISLTA0BYPS1 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 100C$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
TISLTA0BYPS17	TISLTA0BYPS16	TISLTA0BYPS15	TISLTA0BYPS14	TISLTA0BYPS13	TISLTA0BYPS12	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-27 TISLTA0BYPS1 register contents

Bit position	Bit name	Function
7	TISLTA0BYPS17	Select the signal input to TAUA0TTIN15. (This is valid when TISLTA07 = 0 and TSOSLTA05 = 0.) 0: Input signal from TAUA0I15 1: Select the signal selected by PIC0REG20204, PIC0REG20226, and PIC0REG20227.
6	TISLTA0BYPS16	Select the signal input to TAUA0TTIN14. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.
5	TISLTA0BYPS15	Select the signal input to TAUA0TTIN13. (This is valid when TISLTA06 = 0 and TSOSLTA03 = 0.) 0: Input signal from TAUA0I13 1: Select the signal selected by PIC0REG20203, PIC0REG20222, and PIC0REG20223.
4	TISLTA0BYPS14	Select the signal input to TAUA0TTIN12. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.
3	TISLTA0BYPS13	Select the signal input to TAUA0TTIN11. (This is valid when TISLTA05 = 0 and TSOSLTA01 = 0.) 0: Input signal from TAUA0I11 1: Select the signal selected by PIC0REG20202, PIC0REG20218, and PIC0REG20219.
2	TISLTA0BYPS12	Select the signal input to TAUA0TTIN10. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.

(3) TSOSLTA0 - TAUA input selection register

The TSOSLTA0 register selects TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 2014$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	TSOSLTA05	TSOSLTA04	TSOSLTA03	TSOSLTA02	TSOSLTA01	TSOSLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-28 TSOSLTA0 register contents

Bit position	Bit name	Function
5	TSOSLTA05	Select the signal input to TAUA0TTIN15. 0: Select the signal selected by TISLTA0BYPS17. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TSOSLTA04	Select the signal input to TAUA0TTIN14. 0: Select the signal selected by TISLTA0BYPS16. 1: Setting prohibited.
3	TSOSLTA03	Select the signal input to TAUA0TTIN13. 0: Select the signal selected by TISLTA0BYPS15. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TSOSLTA02	Select the signal input to TAUA0TTIN12. 0: Select the signal selected by TISLTA0BYPS14. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TSOSLTA01	Select the signal input to TAUA0TTIN11. 0: Select the signal selected by TISLTA0BYPS13. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TSOSLTA00	Select the signal input to TAUA0TTIN10. 0: Select the signal selected by TISLTA0BYPS12. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(4) TISLTA0 - TAUA input selection register

The TISLTA0 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 1000_H

Initial value 00_H

7	6	5	4	3	2	1	0
TISLTA07	TISLTA06	TISLTA05	TISLTA04	TISLTA03	TISLTA02	TISLTA01	TISLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-29 TISLTA0 register contents

Bit position	Bit name	Function
7	TISLTA07	Select the signal input to TAUA0TTIN15. 0: Select the signal selected by TSOSLTA05. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
6	TISLTA06	Select the signal input to TAUA0TTIN13. 0: Select the signal selected by TSOSLTA03. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
5	TISLTA05	Select the signal input to TAUA0TTIN11. 0: Select the signal selected by TSOSLTA01. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TISLTA04	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
3	TISLTA03	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TISLTA02	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TISLTA01	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TISLTA00	1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(5) PIC0HIZCENn - Hi-Z output control register n

The PIC0HIZCENn register selects the Hi-Z output control input signal of TAPAn.

Access This register can be read or written in 8-bit units.

Address PIC0HIZCEN0: FF81 C0B4$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	PIC0HIZCENn6	0	0	0	PIC0HIZCENn2	PIC0HIZCENn1	PIC0HIZCENn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-30 PIC0HIZCENn register contents

Bit position	Bit name	Function
6	PIC0HIZCENn6	Select whether to enable or disable Hi-Z output control by the INTADCA0ERR interrupt signal. 0: Disable 1: Enable
2	PIC0HIZCENn2	Select whether to enable or disable Hi-Z output control by the WDTA0TNMI interrupt signal. 0: Disable 1: Enable
1	PIC0HIZCENn1	Select whether to enable or disable Hi-Z output control by the CLMATERR error signal from CLMA0, CLMA2, and CLMA3. 0: Disable 1: Enable
0	PIC0HIZCENn0	Select whether to enable or disable Hi-Z output control by the TAPAnESO pin input. 0: Disable 1: Enable

20.8 High-accuracy Triangle PWM Output with Dead Time

20.8.1 Functional overview

Compared to the triangle PWM output with dead time feature of TAUA, this feature makes it possible to control the variable dead time areas near duties of 100% and 0%. This makes more accurate triangle PWM output possible.

For the triangle PWM output with dead time feature of TAUA, it is not possible to output a UB-phase dead time pulse, such as when transitioning to U-phase 0% triangular wave output. (See *Figure 20-19*.)

For this feature, a pulse is generated in combination with the TAUA timer output, and a pseudo dead time pulse is added.

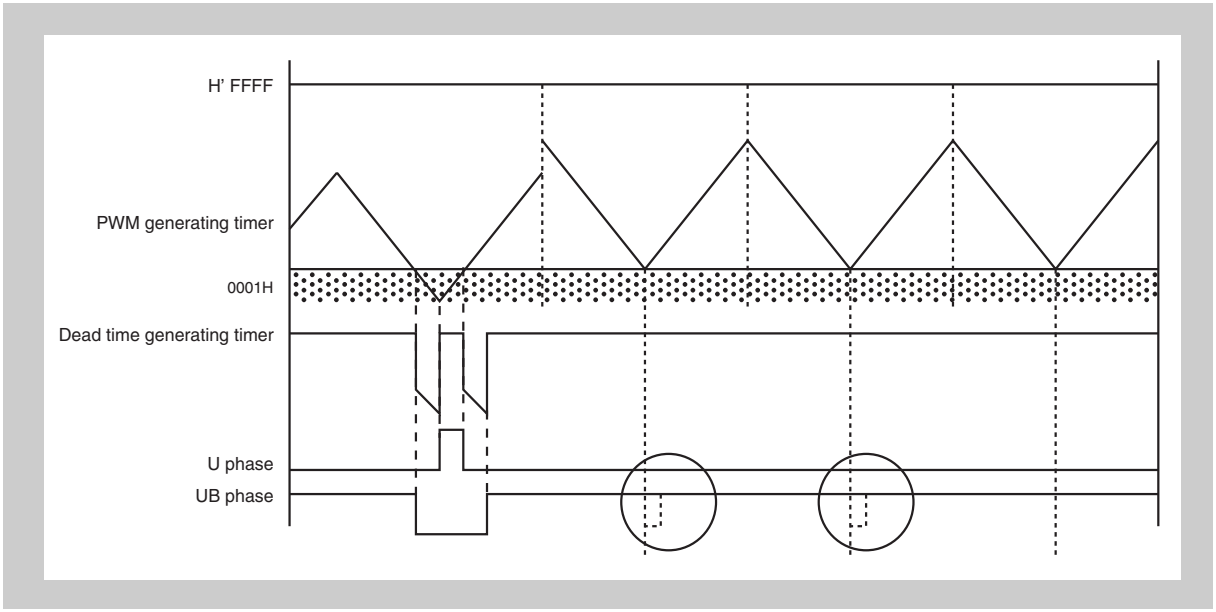


Figure 20-19 Timing of dead time output by the TAUA feature for outputting a triangle PWM signal with dead time

20.8.2 Configuration

The unit and channel configuration for this feature are shown below.

Table 20-31 Configuration of delay pulse output with dead time

Timer	Timer motor control function
TAUA0 CH2, CH4 to CH15 (used channels fixed)	TAPA0

The signal names used in the descriptions below are abbreviations. The actual signal names corresponding to each abbreviation are as follows:

- INTm → INTTAUAnIm (TAUAn channel m interrupt)
- TINm → TAUAnTTINm (TAUAn channel m input)
- TOUTm → TAUAnTOUTm (TAUAn channel m output)
- CDRm → TAUAnCDRm (TAUAn channel m data register)
- CNTm → TAUAnCNTm (TAUAn channel m counter register)

(1) TAUAn configuration

Table 20-32 TAUA configuration

CH	Function name	M/S[a]	CDR setting	Description
2	Triangle PWM output with dead time (CH2 is the master channel for CH4 to CH9.)	M	Cycle	
4		S	Duty (U phase)	
5		S	Dead time (U phase)	
6		S	Duty (V phase)	
7		S	Dead time (V phase)	
8		S	Duty (W phase)	
9		S	Dead time (W phase)	
10	One-shot pulse output	M	Delay	Generate the pulse to be inserted into the variable dead time area for U-phase PWM.
11		S	Pulse width	
12	One-shot pulse output	M	Delay	Generate the pulse to be inserted into the variable dead time area for V-phase PWM.
13		S	Pulse width	
14	One-shot pulse output	M	Delay	Generate the pulse to be inserted into the variable dead time area for W-phase PWM.
15		S	Pulse width	

a) M: Master channel, S: Slave channel, -: Standalone

(2) Block diagram

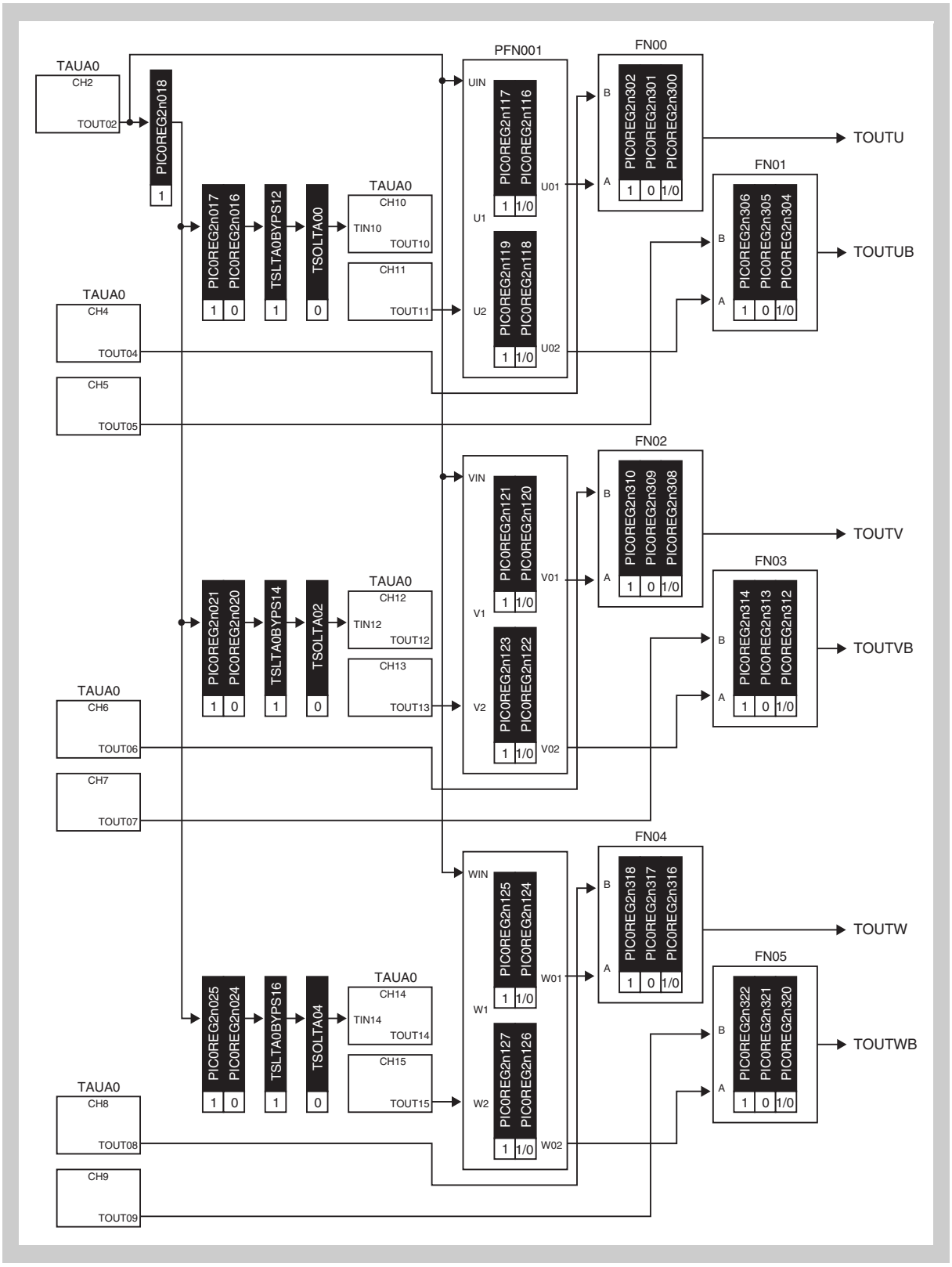


Figure 20-20 Block diagram: High-accuracy triangle PWM output with dead time

20.8.3 Operation example

This is achieved by combining the following TAUA features:

- Triangle PWM output with dead time
- One-shot pulse output

In addition, the following peripheral interconnections are also used because the pulse to be inserted into the variable dead time area is generated for the positive or negative phase:

- Combination circuit (PFN001, PFN023, and PFN045)

In addition, the following peripheral interconnections are also used because the pulse to be inserted into the variable dead time area is combined with the triangle PWM output waveform:

- Logical operation circuit (FN0i) (i = 0 to 5)

A high-accuracy triangle PWM signal with dead time is output by assigning the PWM output achieved using the above features to the U, V, and W phases. Therefore, the PWM output dead time can be freely specified for the PWM signal of each phase. Because the only difference between phases is the assigned channel, only one phase (the U phase) is described below.

(1) Triangle PWM output with dead time

A triangle PWM signal with dead time is output from TOUT04 and TOUT05 by using CH2, CH4, and CH5 in combination.

(2) One-shot pulse output

A CDR11 pulse for which the width is delayed by the delay time (CDR10) from the valid edge of the TIN10 (TOUT02) signal of CH10 is output as TOUT11 by using CH10 and CH11 in combination.

This pulse is used as the variable dead time area pulse used near duties of 100% and 0%.

Caution Specify each CDR setting for one-shot pulse output such that the following condition is satisfied: CDR05 ≥ (CDR10 + CDR11)

If a value that does not satisfy the above condition is specified, the output waveform might be affected. To minimize this effect, in addition to satisfying the above setting condition, leave CDR11 set to 0000H until the variable dead time area pulse is required.
Detect both rising and falling edges as the valid TIN10 (TOUT02) edge, and set TAUAnTOL11 to 1 (active low).
Specify the same operation clock for each TAUAn channel used for outputting a triangle PWM signal with dead time or a one-shot pulse.

For details about the TAUA functions, see the chapter *"Timer Array Unit A (TAUA)"*.

(3) U-phase combination circuit (PFN001)

This circuit generates a variable dead time area pulse (FN00 A, FN01 A) for adding a generated one-shot pulse to a generated triangle PWM signal with dead time.

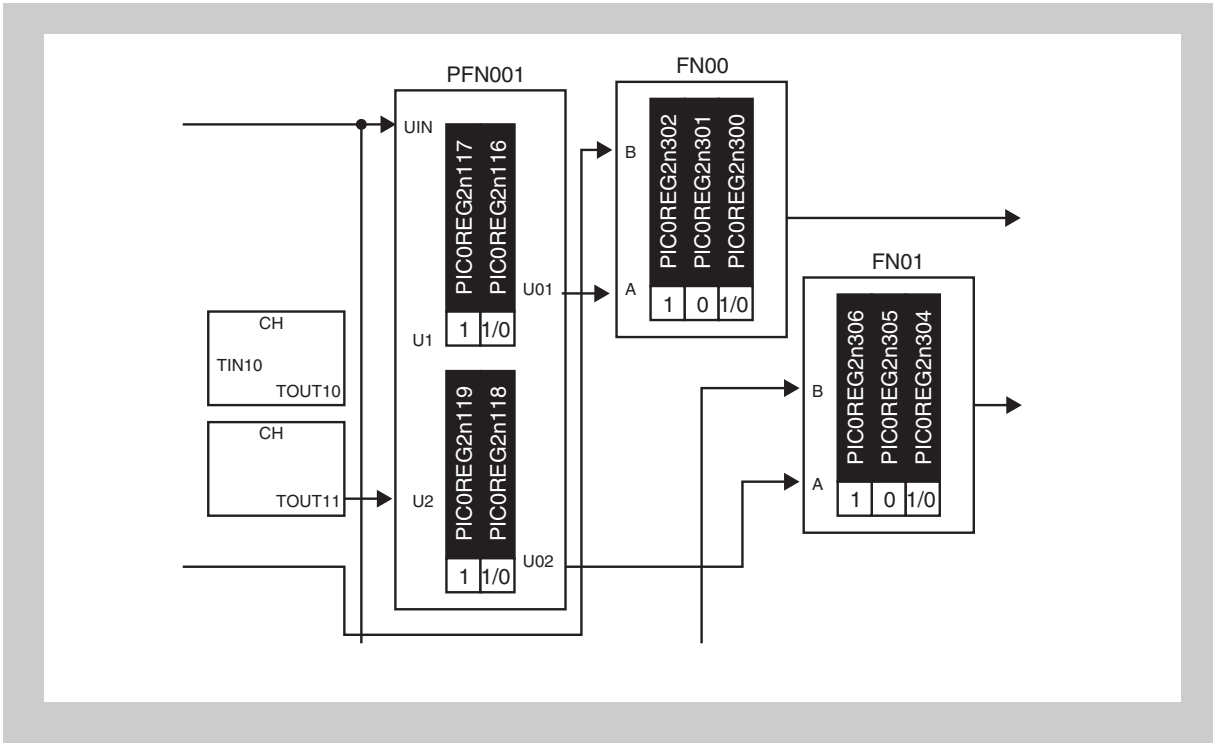


Figure 20-21 Block diagram excerpt (PFN001, FN00, and FN01)

The table below shows the relationships between combination circuit input (UIN, U2) and output (UC0, UO1).

Table 20-33 U, UB phase combination circuit (PFN001) I/O table

- UO0 (U-phase variable dead time area pulse) output

UIN (TOUT02)	U2 (TOUT11)	UO0	
		PIC0REG2n117, 16 = 10B U-phase output active high (TAUAnTOL04 = 0)	PIC0REG2n117, 16 = 11B U-phase output active low (TAUAnTOL04 = 1)
0	0	1	0
0	1	1	0
1	0	0	1
1	1	1	0

- UO1 (UB-phase variable dead time area pulse) output

UIN (TOUT02)	U2 (TOUT11)	UO1	
		PIC0REG2n119, 18 = 10B UB-phase output active high (TAUAnTOL05 = 0)	PIC0REG2n119, 18 = 11B UB-phase output active low (TAUAnTOL05 = 1)
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Note The PIC0REG2n116, PIC0REG2n117, PIC0REG2n118, and PIC0REG2n119 settings change depending on the active U-phase and UB-phase levels of the generated triangle PWM signal with dead time.

(4) Logical operation circuit (FN0i) (i = 0 or 1)

This circuit combines an output triangle PWM signal with dead time (TOUT04, TOUT05) with combination circuit output (UO0 and UO1 of PFN001) and generates a PWM signal to which a variable dead time area pulse has been added.

The combination logic for the logical operation circuit is switched according to the PIC0REG2n3 register setting. (Bits 0 to 2 are specified for U-phase output, and bits 4 to 6 are specified for UB-phase output.)

Set up the logical operation circuit as shown in the table below. The combined signal is output from the TAPAnUP and TAPAnUM pins according to the specified combination logic.

Table 20-34 Logical operation circuit (FN0i) (i = 0 or 1) settings and TAPAnUP and TAPAnUM pin output

- U-phase output (TOUT04)

Active level	PIC0REG2n302 to 00	TAPAnUP pin output waveform
Active high (TAUAnTOL04 = 0)	100B	AND of FN00 B (TOUT04) and FN00 A (UO0)
Active low (TAUAnTOL04 = 1)	101B	OR of FN00 B (TOUT04) and FN00 A (UO0)

- UB-phase output (TOUT05)

Active level	PIC0REG2n306 to 04	TAPAnUM pin output
Active high (TAUAnTOL05 = 0)	100B	AND of FN01 B (TOUT05) and FN01 A (UO1)
Active low (TAUAnTOL05 = 1)	101B	OR of FN01 B (TOUT05) and FN01 A (UO1)

Because the above makes variable dead time control possible to ensure output accuracy near duties of 0% and 100% even for TAUA, a more accurate triangle PWM signal can be output than that output using the TAUA feature for outputting a triangle PWM signal with dead time.

For the V/VB phase and W/WB phase, the used channels and register bits differ, but the settings are the same, as shown in *Figure 20-20 “Block diagram: High-accuracy triangle PWM output with dead time” on page 1633* .

The peripheral interconnections provide a connection for adding the pulse generated during one-shot pulse output to the PWM signal generated during output of a triangle PWM signal with dead time by using the combination circuit and logical operation circuit of the peripheral interconnections.

The following figures show timing charts for outputting a high-accuracy triangle PWM signal with dead time.

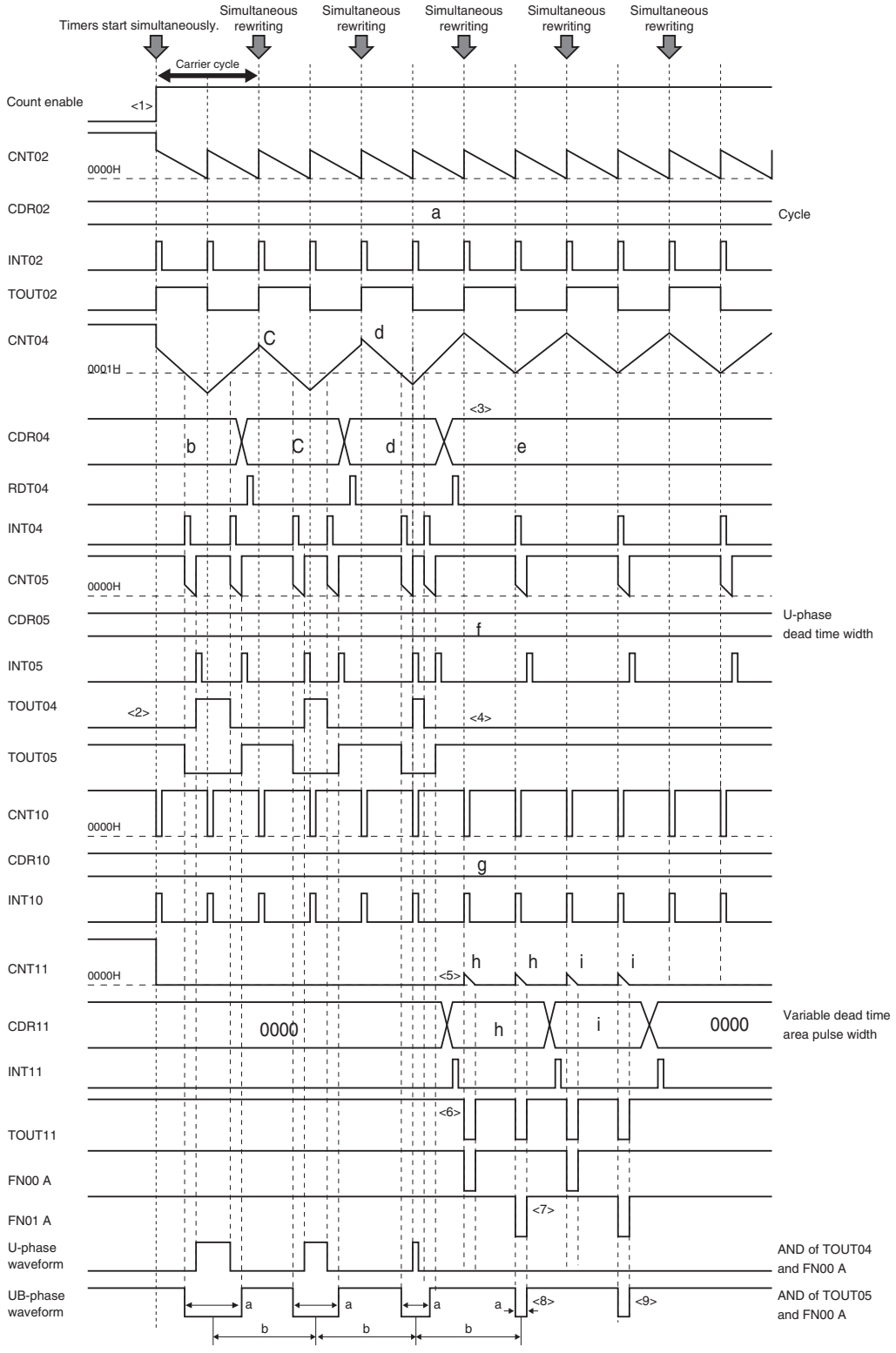


Figure 20-22 Example of a high-accuracy PWM signal output with dead time (U-phase: 0%, UB-phase: 100%) (when TAUAnTOL04 = 0 (active high) and TAUAnTOL05 = 0 (active high))

a) The variable dead time area pulse uses a sawtooth wave and is therefore expanded on one side, unlike a pulse that uses a triangle wave, which is expanded on both sides.

b) Because the variable dead time area pulse is expanded on one side, the length of the one-phase PWM signal output cycle for the variable dead time area increases by 1/2 the added variable dead time area pulse width.

An operation example in which the system transitions to a U-phase of 0% and UB-phase of 100% in the timer configuration for performing the U-phase PWM output shown in *Figure 20-22* is provided below. Output of a triangle PWM signal with dead time is active high.

- When timer operation is started, output of a triangle PWM with dead time is started by the CH2, CH4, and CH5 channels of TAUAn (point <1> in the figure).
- A triangle PWM waveform with dead time is generated from TOUT04 and TOUT05 (point <2> in the figure).
- A U-phase duty output value of 0% is specified for CDR04 (point <3> in the figure).
- Due to the setting in <3>, TOUT04 output is set to the inactive level, and TOUT05 output is set to the active level (point <4> in the figure). However, no variable dead time area pulse is output during this operation.
- To create a variable dead time area pulse, the value to be used as the pulse width is specified for CDR11 when specifying the 0% U-phase duty in <3> (point <5> in the figure).
  For this example, the CDR11 setting is fixed to 0000H until the system enters the variable dead time area to prevent adverse effects on the output PWM signal.
- The variable dead time area pulse is output as a pulse that has the width specified for CDR11 after the delay time specified for CDR10 elapses, starting at the TOUT02 edge (point <6> in the figure).
- The pulse output in <6> is converted to a variable dead time area pulse for the U phase (FN00 A) and UB phase (FN01 A) by the combination circuit (PFN001) (point <7> in the figure).
- The pulse generated in <7> is combined with the TOUT04 and TOUT05 output waveforms by using the logical operation circuits (FN00, FN01), and the result is output from TAPAnUP (U-phase output) and TAPAnUM (UB-phase output) (point <8> in the figure).
- By later changing the CDR11 setting, which specifies the width of the variable dead time area pulse, the desired variable dead time area pulse can be added.

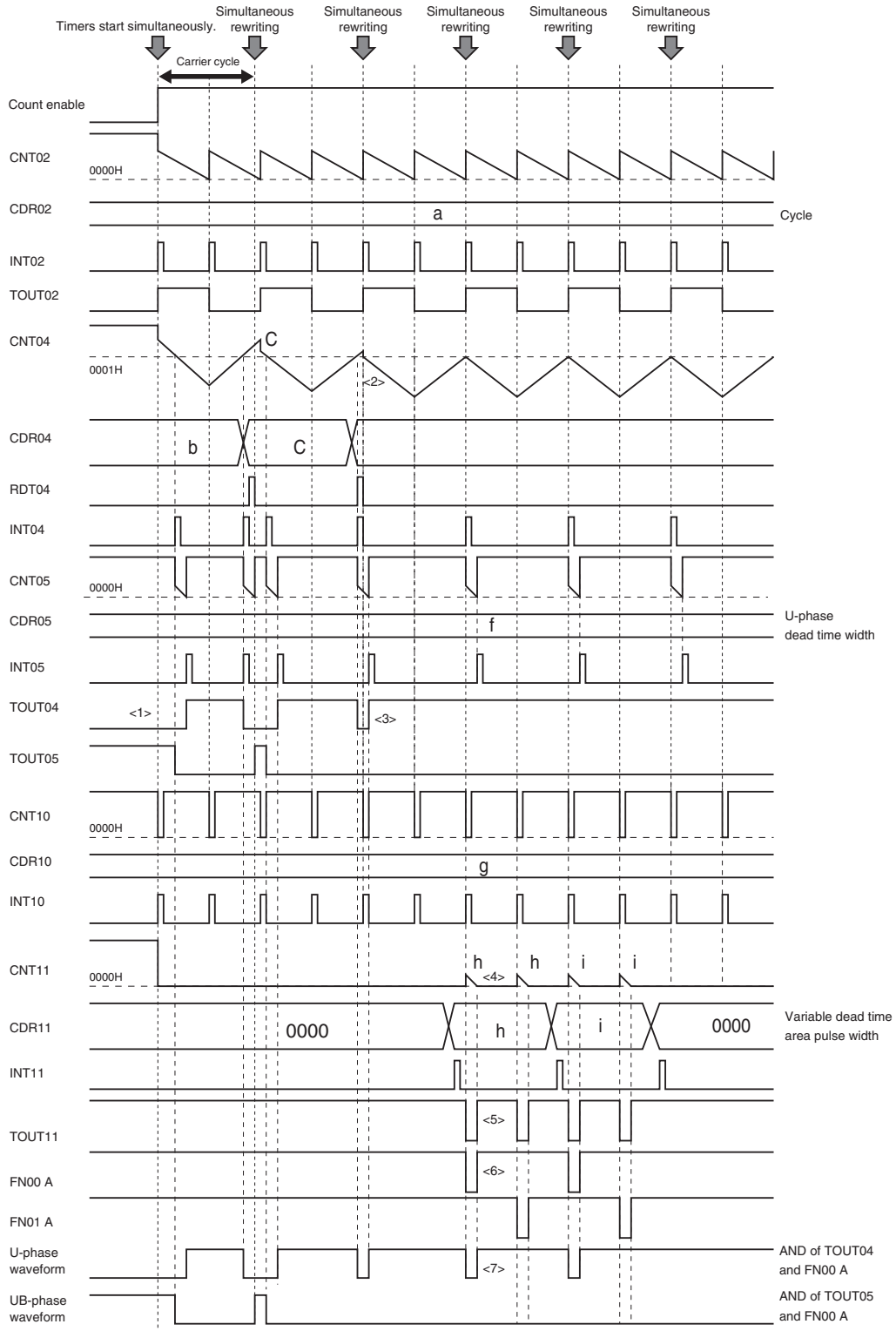


Figure 20-23 Example of a high-accuracy PWM signal output with dead time (U-phase: 100%, UB-phase: 0%) (when TAUAnTOL04 = 0 (active high) and TAUAnTOL05 = 0 (active high))

An operation example in which the system transitions to a U-phase of 100% and UB-phase of 0% in the timer configuration for performing the U-phase PWM output shown in *Figure 20-23* is provided below. Output of a triangle PWM signal with dead time is active high.

- The timer operation from the start of timer operation until the output of a triangle PWM signal with dead time is the same (point <1> in the figure).
- A U-phase duty output value of 100% (CDR04 = 0000H) is specified for CDR04 (point <2> in the figure).
- Due to the setting in <2>, TOUT04 output is set to the active level, and TOUT05 output is set to the inactive level (point <3> in the figure). However, no variable dead time area pulse is output during this operation.
- To create a variable dead time area pulse, the value to be used as the pulse width is specified for CDR11 one cycle after specifying the 100% U-phase duty setting in <2> (point <4> in the figure).
  For this example, the CDR11 setting is fixed to 0000H until the system enters the variable dead time area to prevent adverse effects on the output PWM signal.
- The variable dead time area pulse is output as a pulse that has the width specified for CDR11 after the delay time specified for CDR10 elapses, starting at the TOUT02 edge (point <5> in the figure).
- The pulse output in <5> is converted to a variable dead time area pulse for the U phase (FN00 A) and UB phase (FN01 A) by the combination circuit (PFN001) (point <6> in the figure).
- The pulse generated in <6> is combined with the TOUT04 and TOUT05 output waveforms by using the logical operation circuits (FN00, FN01), and the result is output from TAPAnUP (U-phase output) and TAPAnUM (UB-phase output) (point <7> in the figure).

Caution If the 100% U-phase duty setting for CDR04 and the variable dead time area pulse width for CDR11 are specified at the same time, the variable dead time area pulse is affected by the amount shown by <2> for the last PWM signal output from TOUT04 and shown by feature specification <1>, as shown in *Figure 20-24*.
To cancel this effect, the CDR11 setting is delayed one cycle.

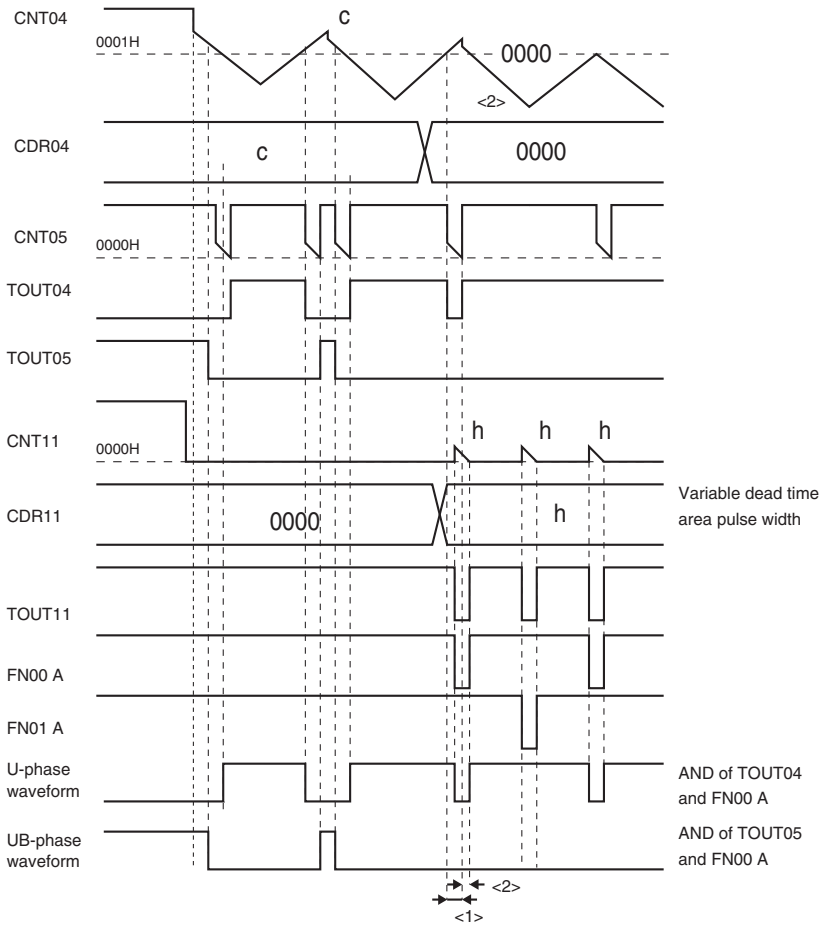


Figure 20-24 Effect on the output triangle PWM wave with dead time by the variable dead time area pulse

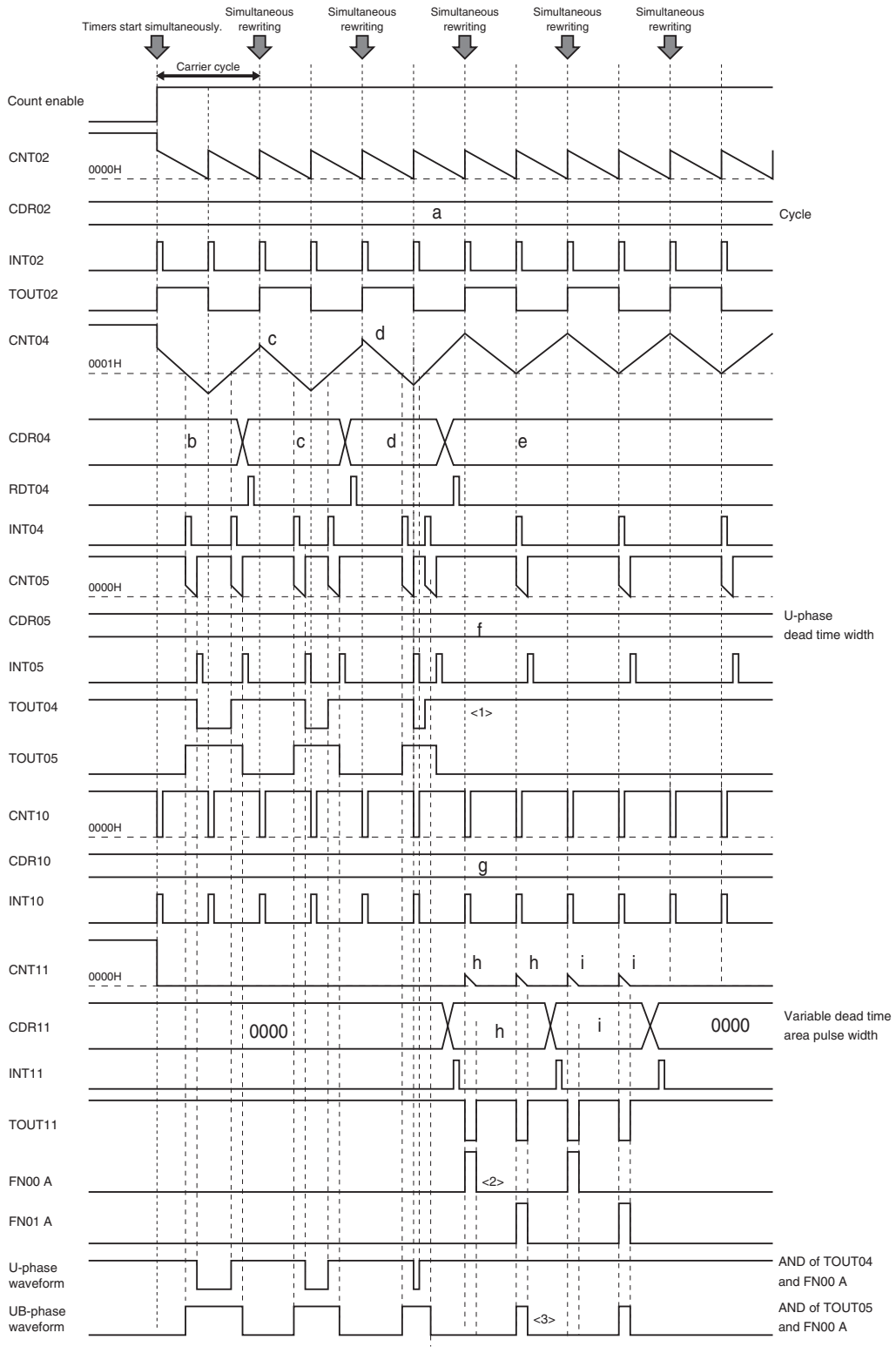


Figure 20-25 Example of a high-accuracy PWM signal output with dead time (U-phase: 100%, UB-phase: 0%) (TAUAnTOL04 = 1 (active low), TAUAnTOL05 = 1 (active low))

An operation example in which the system transitions to a U-phase of 100% and UB-phase of 0% in the timer configuration for performing the U-phase PWM output shown in *Figure 20-25* is provided below. Output of a triangle PWM signal with dead time is active low.

- The timer operation from the start of timer operation until the output of a triangle PWM signal with dead time is the same as in *Figure 20-22 "Example of a high-accuracy PWM signal output with dead time (U-phase: 0%, UB-phase: 100%) (when TAUAnTOL04 = 0 (active high) and TAUAnTOL05 = 0 (active high))"* (point <1> in the figure). However, an active low PWM signal is output from TOUT04 and TOUT05.
- Therefore, active low output that corresponds with PWM output is specified as the combination circuit setting (PIC0REG2n116 and PIC0REG2n117, and PIC0REG2n118 and PIC0REG2n119) (point <2> in the figure). This results in the output of an active low variable dead time area pulse for the U phase (FN00 A) and UB phase (FN01 A).
- In addition, active low output that corresponds with PWM output is also specified as the logical operation circuit setting (PIC0REG2n302 to PIC0REG2n300 and PIC0REG2n306 to PIC0REG2n304) (point <3> in the figure). The pulse generated in <2> is combined with the TOUT04 and TOUT05 output waveforms, and the result is output from TAPAnUP (U-phase output) and TAPAnUM (UB-phase output) as an active low PWM signal.

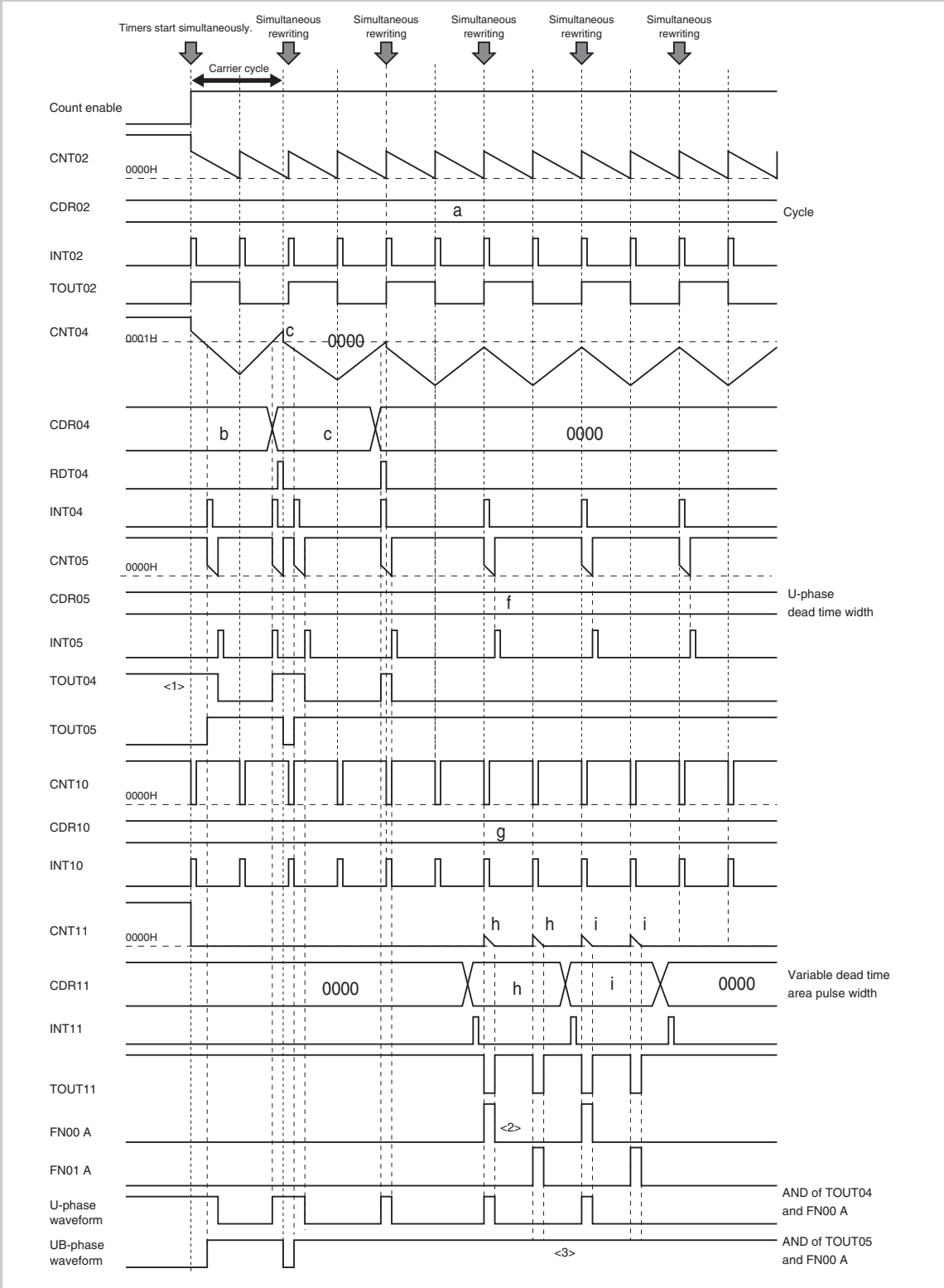


Figure 20-26 Example of a high-accuracy PWM signal output with dead time (U-phase: 0%, UB-phase: 100%) (when TAUAnTOL04 = 0 (active low) and TAUAnTOL05 = 0 (active low))

An operation example in which the system transitions to a U-phase of 0% and UB-phase of 100% in the timer configuration for performing the U-phase PWM output shown in *Figure 20-26* is provided below. Output of a triangle PWM signal with dead time is active low.

- The timer operation from the start of timer operation until the output of a triangle PWM signal with dead time is the same as in *Figure 20-23 "Example of a high-accuracy PWM signal output with dead time (U-phase: 100%, UB-phase: 0%) (when TAUAnTOL04 = 0 (active high) and TAUAnTOL05 = 0 (active high))"* (point <1> in the figure). However, an active low PWM signal is output.
- Therefore, active low output that corresponds with PWM output is specified as the combination circuit setting (PIC0REG2n116 and PIC0REG2n117, and PIC0REG2n118 and PIC0REG2n119) (point <2> in the figure). This results in the output of an active low variable dead time area pulse for the U phase (FN00 A) and UB phase (FN01 A).
- In addition, active low output that corresponds with PWM output is also specified as the logical operation circuit setting (PIC0REG2n302 to PIC0REG2n300 and PIC0REG2n306 to PIC0REG2n304) (point <3> in the figure). The pulse generated in <2> is combined with the TOUT04 and TOUT05 output waveforms, and the result is output from TAPAnUP (U-phase output) and TAPAnUM (UB-phase output) as an active low PWM signal.

Caution If the 100% U-phase duty setting for CDR04 and the variable dead time area pulse width for CDR11 are specified at the same time, the last PWM signal output from TOUT04 is adversely affected due to the feature specifications. To cancel this effect, the CDR11 setting is delayed one cycle.
For details, see *Figure 20-24 "Effect on the output triangle PWM wave with dead time by the variable dead time area pulse"*.

20.8.4 Setup flow

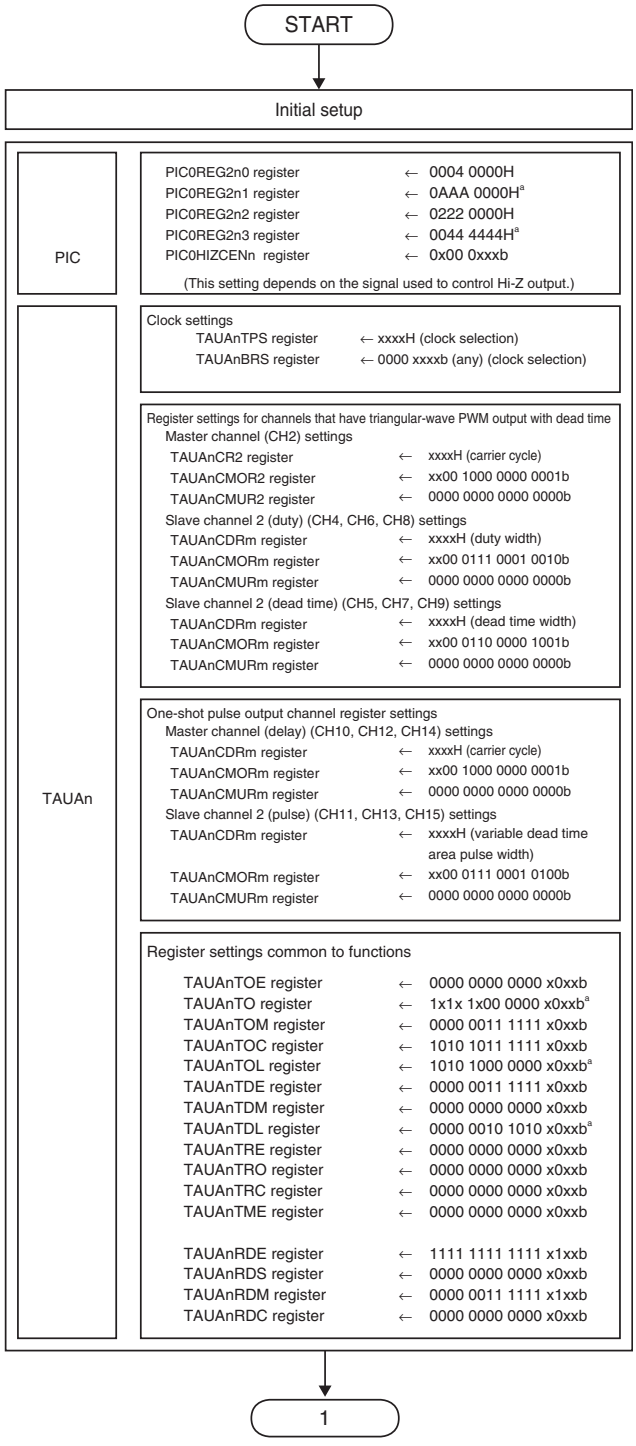


Figure 20-27 Setup flow (active high example)

a) Change settings according to the active level of the PWM signal to be output.

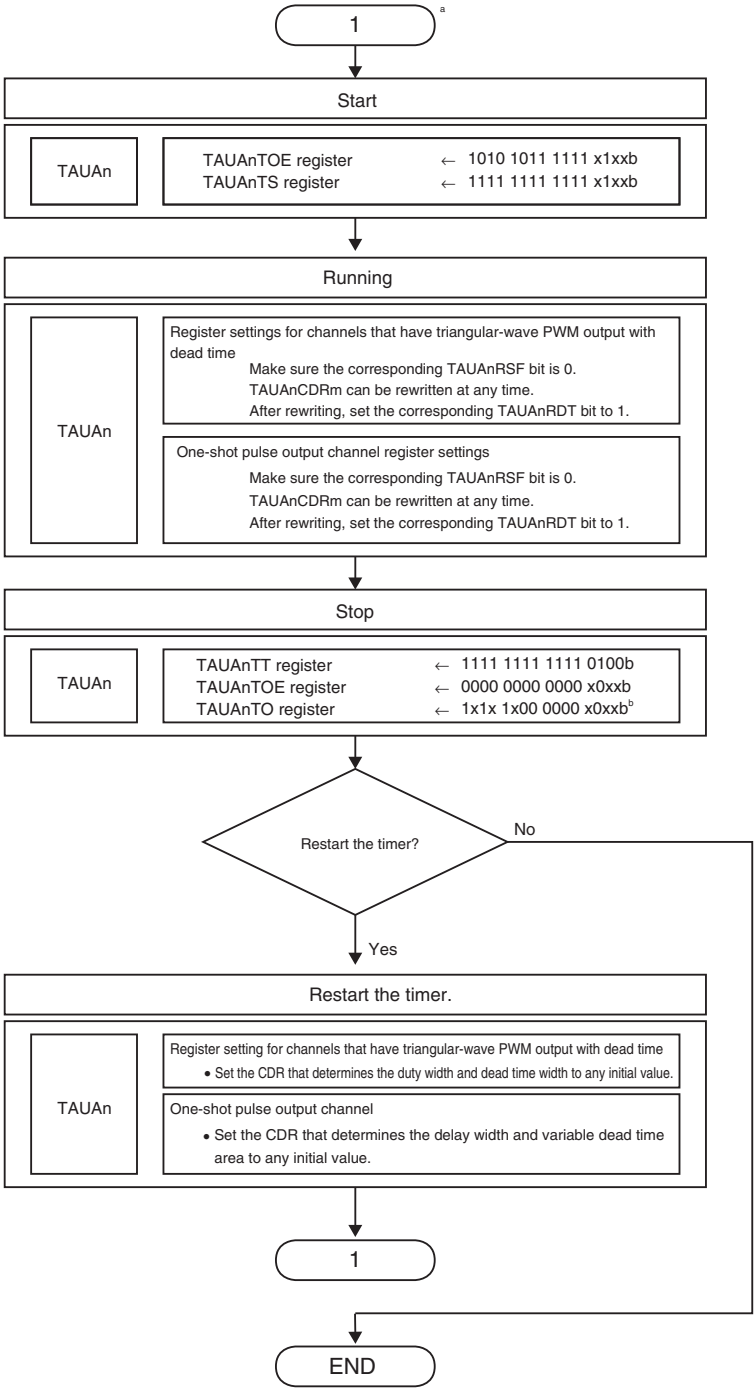


Figure 20-28 Setup flow (active high example) (continued)

a) Specify the selection register (such as TISLTA0BYPS1) and output port to use after specifying the initial settings for the peripheral interconnections and timers.

b) Change settings according to the active level of the PWM signal to be output.

20.8.5 Example of setting up operation functions

This section provides example settings for each register.

(1) TAUAn settings (active high example)

Table 20-35 TAUAn: CH2-related (master channel used to output a triangle PWM signal with dead time[a])

Register	Bit position	Bit name	Setting	Remark
TAUAnCMOR2	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	1	
	10 to 8	TAUAnSTS[2:0]	000	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	
	4 to 1	TAUAnMD[4:1]	0000	
	0	TAUAnMD0	1	At the start of operation, output INTm and toggle TOUTm.
TAUAnCMUR2	1, 0	TAUAnTIS[1:0]	00	Fixed

a) The master channel and slave channel names are defined for TAUA triangle PWM output with dead time. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the master channel and slave channel.

Note For the TAUAnCMORm register of the master channel used when outputting a triangle PWM signal with dead time, TAUAnCKS[1:0] (which selects the operation clock) and TAUAnMD0 can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*. For this peripheral interconnections feature, set TAUAnMD0 to 1.

Table 20-36 TAUAn: CH4, CH6, and CH8-related (slave channel 2 used to output a triangle PWM signal with dead time[a]) (m = 4, 6, or 8)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	111	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	
	4 to 1	TAUAnMD[4:1]	1001	
	0	TAUAnMD0	0	
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA triangle PWM output with dead time. For details, see the chapter *"Timer Array Unit A (TAUA)*".

b) The same operation clock must be specified for the master channel and slave channel.

For the TAUAnCMORm register of slave channels 2 and 3, which is used when outputting a triangle PWM signal with dead time, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

Table 20-37 TAUAn: CH5, CH7, and CH9-related (slave channel 3 used to output a triangle PWM signal with dead time[a]) (m = 5, 7, or 9)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	110	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA triangle PWM output with dead time. For details, see the chapter *"Timer Array Unit A (TAUA)*".

b) The same operation clock must be specified for the master channel and slave channel.

Note For the TAUAnCMORm register of slave channels 2 and 3, which is used when outputting a triangle PWM signal with dead time, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)*".

Table 20-38 TAUAn: CH10, CH12, and CH14-related (master channel used to output a one-shot pulse[a]) (m = 10, 12, or 14)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	1	
	10 to 8	TAUAnSTS[2:0]	001	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	0	Disable start triggers during counting.
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	10	Detect both rising and falling edges as valid.

a) The master channel and slave channel names are defined for TAUA one-shot pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the master channel and slave channel.

Table 20-39 TAUAn: CH11, CH13, and CH15-related (slave channel used to output a one-shot pulse[a]) (m = 11, 13, or 15)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	100	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	
	4 to 1	TAUAnMD[4:1]	1010	
	0	TAUAnMD0	0	Disable start triggers during counting.
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA one-shot pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the master channel and slave channel. Specify the same clock setting as for the master channel (CH2) used to output a triangle PWM signal with dead time.

Note For the TAUAnCMORm register used during one-shot pulse output, TAUAnCKS[1:0] (which selects the operation clock) and TAUAnMD0 can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*.
For this feature clear TAUAnMD0 to 0.

Table 20-40 Common TAUAn channel settings (1/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTOE	15	TAUAnTOE15	0 1	Disable the timer. Enable the timer.
	14	TAUAnTOE14	0	
	13	TAUAnTOE13	0 1	Disable the timer. Enable the timer.
	12	TAUAnTOE12	0	
	11	TAUAnTOE11	0 1	Disable the timer. Enable the timer.
	10	TAUAnTOE10	0	
	9 to 4	TAUAnTOE09 to TAUAnTOE04	0 1	Disable the timer. Enable the timer.
	3	TAUAnTOE03	Any	
	2	TAUAnTOE02	0 1	Disable the timer. Enable the timer.
	1, 0	TAUAnTOE01 TAUAnTOE00	Any	
TAUAnTO	15	TAUAnTO15	1[a]	Output a high-level signal to TOUT15.
	14	TAUAnTO14	Any	
	13	TAUAnTO13	1[a]	Output a high-level signal to TOUT13.
	12	TAUAnTO12	Any	
	11	TAUAnTO11	1[a]	Output a high-level signal to TOUT11.
	10	TAUAnTO10	Any	
	9 to 4	TAUAnTO09 to TAUAnTO04	0[a]	Output a low-level signal to TOUT09 to TOUT04.
	3	TAUAnTO03	Any	
	2	TAUAnTO02	0	Output a low-level signal to TOUT02.
	1, 0	TAUAnTO01 TAUAnTO00	Any	
TAUAnTOM	15 to 10	TAUAnTOM15 to TAUAnTOM10	0	Independent channel operation mode
	9 to 4	TAUAnTOM09 to TAUAnTOM04	1	Synchronous channel operation mode
	3	TAUAnTOM03	Any	
	2	TAUAnTOM02	0	Independent channel operation mode
	1, 0	TAUAnTOM01 TAUAnTOM00	Any	

Table 20-40 Common TAUAn channel settings (2/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTOC	15	TAUAnTOC15	1	Set/reset mode
	14	TAUAnTOC14	0	
	13	TAUAnTOC13	1	Set/reset mode
	12	TAUAnTOC12	0	
	11	TAUAnTOC11	1	Set/reset mode
	10	TAUAnTOC10	0	
	9 to 4	TAUAnTOC09 to TAUAnTOC04	1	Synchronous channel operation mode 2
	3	TAUAnTOC03	Any	
	2	TAUAnTOC02	0	Toggle mode
	1, 0	TAUAnTOC01 TAUAnTOC00	Any	
TAUAnTOL	15	TAUAnTOL15	1[a]	Inverted logic output (active low)
	14	TAUAnTOL14	Any	
	13	TAUAnTOL13	1[a]	Inverted logic output (active low)
	12	TAUAnTOL12	Any	
	11	TAUAnTOL11	1[a]	Inverted logic output (active low)
	10	TAUAnTOL10	Any	
	9 to 4	TAUAnTOL09 to TAUAnTOL04	0[a]	Positive logic output (active high)
	3	TAUAnTOL03	Any	
	2	TAUAnTOL02	0	Positive logic output (active high)
	1, 0	TAUAnTOL01 TAUAnTOL00	Any	
TAUAnTDE	15 to 10	TAUAnTDE15 to TAUAnTDE10	0	Disable dead time control.
	9 to 4	TAUAnTDE09 to TAUAnTDE04	1	Enable dead time control.[b]
	3	TAUAnTDE03	Any	
	2	TAUAnTDE02	0	Disable dead time control.
	1, 0	TAUAnTDE01 TAUAnTDE00	Any	
TAUAnTDM	15 to 9	TAUAnTDM15 to TAUAnTDM09	0	
	3	TAUAnTDM03	Any	
	2	TAUAnTDM02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDM01 TAUAnTDM00	Any	

Table 20-40 Common TAUAn channel settings (3/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTDL	15 to 10	TAUAnTDL15 to TAUAnTDL10	0	Invalid because dead time control is disabled.
	9	TAUAnTDL09	1[a]	Add dead time to the negative W phase period.
	8	TAUAnTDL08	0[a]	Add dead time to the positive W phase period.
	7	TAUAnTDL07	1[a]	Add dead time to the negative V phase period.
	6	TAUAnTDL06	0[a]	Add dead time to the positive V phase period.
	5	TAUAnTDL05	1[a]	Add dead time to the negative U phase period.
	4	TAUAnTDL04	0[a]	Add dead time to the positive U phase period.
	3	TAUAnTDL03	Any	
	2	TAUAnTDL02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDL01 TAUAnTDL00	Any	
TAUAnTRE	15 to 4	TAUAnTRE15 to TAUAnTRE04	0	Disable real-time output.
	3	TAUAnTRE03	Any	
	2	TAUAnTER02	0	Disable real-time output.
	1, 0	TAUAnTRE01 TAUAnTRE00	Any	
TAUAnTRO	15 to 4	TAUAnTRO15 to TAUAnTRO04	0	Invalid because real-time output is disabled.
	3	TAUAnTRO03	Any	
	2	TAUAnTRO02	0	Invalid because real-time output is disabled.
	1, 0	TAUAnTRO01 TAUAnTRO00	Any	
TAUAnTRC	15 to 4	TAUAnTRC15 to TAUAnTRC04	0	Do not use this channel to generate the real-time output trigger.
	3	TAUAnTRC03	Any	
	2	TAUAnTRC02	0	Do not use this channel to generate the real-time output trigger.
	1, 0	TAUAnTRC01 TAUAnTRC00	Any	
TAUAnTME	15 to 4	TAUAnTME15 to TAUAnTME04	0	Disable modulation output for timer output and real-time output.
	3	TAUAnTME03	Any	
	2	TAUAnTME02	0	Disable modulation output for timer output and real-time output.
	1, 0	TAUAnTME01 TAUAnTME00	Any	

Table 20-40 Common TAUAn channel settings (4/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnRDE	15 to 4	TAUAnRDE15 to TAUAnRDE04	1	Enable simultaneous rewriting.
	3	TAUAnRDE03	Any	
	2	TAUAnRDE02	1	Enable simultaneous rewriting.
	1, 0	TAUAnRDE01 TAUAnRDE00	Any	
TAUAnRDS	15 to 4	TAUAnRDS15 to TAUAnRDS04	0	Do not enable simultaneous rewriting by using another upper channel.
	3	TAUAnRDS03	Any	
	2	TAUAnRDS02	0	Do not enable simultaneous rewriting by using another upper channel.
	1, 0	TAUAnRDS01 TAUAnRDS00	Any	
TAUAnRDM	15 to 10	TAUAnRDM15 to TAUAnRDM10	0	Perform simultaneous rewriting when the master channel starts counting.
	9 to 4	TAUAnRDM09 to TAUAnRDM04	1	Perform simultaneous rewriting after the master channel starts counting when there is a peak in the triangle wave on the corresponding slave channel.
	3	TAUAnRDM03	Any	
	2	TAUAnRDM02	1	Perform simultaneous rewriting after the master channel starts counting when there is a peak in the triangle wave on the corresponding slave channel.
	1, 0	TAUAnRDM01 TAUAnRDM00	Any	
TAUAnRDC	15 to 4	TAUAnRDC15 to TAUAnRDC04	0	Do not use this channel to generate the simultaneous rewrite trigger.
	3	TAUAnRDC03	Any	
	2	TAUAnRDC02	0	Do not use this channel to generate the simultaneous rewrite trigger.
	1, 0	TAUAnRDC01 TAUAnRDC00	Any	

a) Change the setting according to the used system.

b) These are used to control positive/negative phase waveform output for which even channels are paired with odd channels to perform dead time control. For details, see the chapter *"Timer Array Unit A (TAUA)*".

(2) Peripheral interconnections settings (active high example)

Table 20-41 Peripheral interconnections settings

Register	Bit position	Bit name	Setting	Remark
PIC0REG2n0	18	PIC0REG2n018	1	Select the TOUT signal of CH2 of TAUAn.
PIC0REG2n1	27, 26	PIC0REG2n127 PIC0REG2n126	1 0	Negative W-phase active high combination circuit output
	25, 24	PIC0REG2n125 PIC0REG2n124	1 0	Positive W-phase active high combination circuit output
	23, 22	PIC0REG2n123 PIC0REG2n122	1 0	Negative V-phase active high combination circuit output
	21, 20	PIC0REG2n121 PIC0REG2n120	1 0	Positive V-phase active high combination circuit output
	19, 18	PIC0REG2n119 PIC0REG2n118	1 0	Negative U-phase active high combination circuit output
	17, 16	PIC0REG2n117 PIC0REG2n116	1 0	Positive U-phase active high combination circuit output
PIC0REG2n2	25, 24	PIC0REG2n225 PIC0REG2n224	1 0	Select the input selected by the PIC0REG2n018 bit.
	21, 20	PIC0REG2n221 PIC0REG2n220	1 0	Select the input selected by the PIC0REG2n018 bit.
	17, 16	PIC0REG2n217 PIC0REG2n216	1 0	Select the input selected by the PIC0REG2n018 bit.
PIC0REG2n3	22, 21, 20	PIC0REG2n322 PIC0REG2n321 PIC0REG2n320	1 0 0	Negative W-phase active high logical operation circuit output
	18, 17, 16	PIC0REG2n318 PIC0REG2n317 PIC0REG2n316	1 0 0	Positive W-phase active high logical operation circuit output
	14, 13, 12	PIC0REG2n314 PIC0REG2n313 PIC0REG2n312	1 0 0	Negative V-phase active high logical operation circuit output
	10, 9, 8	PIC0REG2n310 PIC0REG2n309 PIC0REG2n308	1 0 0	Positive V-phase active high logical operation circuit output
	6, 5, 4	PIC0REG2n306 PIC0REG2n305 PIC0REG2n304	1 0 0	Negative U-phase active high logical operation circuit output
	2, 1, 0	PIC0REG2n302 PIC0REG2n301 PIC0REG2n300	1 0 0	Positive U-phase active high logical operation circuit output

20.8.6 Registers

(1) PIC0REG2n0 - Timer I/O control register 2n0

Access This register can be read or written in 32-bit units.

Address PIC0REG200: FF81 C08C$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	PIC0REG 2n018	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-42 PIC0REG2n0 register contents

Bit position	Bit name	Function
18	PIC0REG2n018	Select the signal input to TAUAnTTIN10, TAUAnTTIN12, and TAUAnTTIN14. 0: Setting prohibited 1: Select TAUAnTTOUT2.

Caution Some of the bits defined as 0 in the PIC0REG2n0 register are defined for the other timers. For such bits, use the bit definition of those timers.

(2) PIC0REG2n1 - Timer I/O control register 2n1

Access This register can be read or written in 32-bit units.

Address PIC0REG201: FF81 C090$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	PIC0REG 2n127	PIC0REG 2n126	PIC0REG 2n125	PIC0REG 2n124
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
PIC0REG 2n123	PIC0REG 2n123	PIC0REG 2n121	PIC0REG 2n120	PIC0REG 2n119	PIC0REG 2n118	PIC0REG 2n117	PIC0REG 2n116
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-43 PIC0REG2n1 register contents (1/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>27
26</td><td>PIC0REG2n127
PIC0REG2n126</td><td>Select the FN05 A input signal according to the output logic specified for CH9 of TAUAn.

<table>
<tr><th>PIC0REG2n127</th><th>PIC0REG2n126</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL09 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL09 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
</table>

Table 20-43 PIC0REG2n1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>25
24</td><td>PIC0REG2n125
PIC0REG2n124</td><td>Select the FN04 A input signal according to the output logic specified for CH8 of TAUAn.

<table>
<tr><th>PIC0REG2n125</th><th>PIC0REG2n124</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL08 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL08 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>23
22</td><td>PIC0REG2n123
PIC0REG2n122</td><td>Select the FN03 A input signal according to the output logic specified for CH7 of TAUAn.

<table>
<tr><th>PIC0REG2n123</th><th>PIC0REG2n122</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL07 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL07 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>21
20</td><td>PIC0REG2n121
PIC0REG2n120</td><td>Select the FN02 A input signal according to the output logic specified for CH6 of TAUAn.

<table>
<tr><th>PIC0REG2n121</th><th>PIC0REG2n120</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL06 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL06 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
</table>

Table 20-43 PIC0REG2n1 register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>19
18</td><td>PIC0REG2n119
PIC0REG2n118</td><td>Select the FN01 A input signal according to the output logic specified for CH5 of TAUAn.

<table>
<tr><th>PIC0REG2n119</th><th>PIC0REG2n118</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL05 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL05 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
<tr><td>17
16</td><td>PIC0REG2n117
PIC0REG2n116</td><td>Select the FN00 A input signal according to the output logic specified for CH4 of TAUAn.

<table>
<tr><th>PIC0REG2n117</th><th>PIC0REG2n116</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Combination circuit output
(Select this when the active high setting is specified (TAUAnTOL04 = 0).)</td></tr>
<tr><td>1</td><td>1</td><td>Inverted combination circuit output
(Select this when the active low setting is specified (TAUAnTOL04 = 1).)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
</table>

Caution Some of the bits defined as 0 in the PIC0REG2n1 register are defined for the other timers. For such bits, use the bit definition of those timers.

(3) PIC0REG2n2 - Timer I/O control register 2n2

Access This register can be read or written in 32-bit units.

Address PIC0REG202: FF81 C094$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	PIC0REG 2n227	PIC0REG 2n226	PIC0REG 2n225	PIC0REG 2n224
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
PIC0REG 2n223	PIC0REG 2n222	PIC0REG 2n221	PIC0REG 2n220	PIC0REG 2n219	PIC0REG 2n218	PIC0REG 2n217	PIC0REG 2n216
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	PIC0REG 2n204	PIC0REG 2n203	PIC0REG 2n202	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-44 PIC0REG2n2 register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>27
26</td><td>PIC0REG2n227
PIC0REG2n226</td><td>Select the signal input to TAUAnTTIN15.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
<tr><td>25
24</td><td>PIC0REG2n225
PIC0REG2n224</td><td>Select the signal input to TAUAnTTIN14.
<table>
<tr><th>PIC0REG2n225</th><th>PIC0REG2n224</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n018 bit (TOUT of CH2 of TAUAn)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>23
22</td><td>PIC0REG2n223
PIC0REG2n222</td><td>Select the signal input to TAUAnTTIN13.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
<tr><td>21
20</td><td>PIC0REG2n221
PIC0REG2n220</td><td>Select the signal input to TAUAnTTIN12.
<table>
<tr><th>PIC0REG2n221</th><th>PIC0REG2n220</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n018 bit (TOUT of CH2 of TAUAn)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
</table>

Table 20-44 PIC0REG2n2 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>19
18</td><td>PIC0REG2n219
PIC0REG2n218</td><td>Select the signal input to TAUAnTTIN11.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
<tr><td>17
16</td><td>PIC0REG2n217
PIC0REG2n216</td><td>Select the signal input to TAUAnTTIN10.
<table>
<tr><th>PIC0REG2n217</th><th>PIC0REG2n216</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n018 bit (TOUT of CH2 of TAUAn)</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>4</td><td>PIC0REG2n204</td><td>Select the signal supplied to TAUAnTTIN15.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
<tr><td>3</td><td>PIC0REG2n203</td><td>Select the signal supplied to TAUAnTTIN13.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
<tr><td>2</td><td>PIC0REG2n202</td><td>Select the signal supplied to TAUAnTTIN11.
See chapter “Delay Pulse Output with Dead Time” and “Three-Phase PWM Output with Dead Time” for details.</td></tr>
</table>

Caution Some of the bits defined as 0 in the PIC0REG2n2 register are defined for the other timers. For such bits, use the bit definition of those timers.

(4) PIC0REG2n3 - Timer I/O control register 2n3

The PIC0REG2n3 register selects logical operations.

Access This register can be read or written in 32-bit units.

Address PIC0REG203: FF81 C098$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	PIC0REG 2n322	PIC0REG 2n321	PIC0REG 2n320	0	PIC0REG 2n318	PIC0REG 2n317	PIC0REG 2n316
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	PIC0REG 2n314	PIC0REG 2n313	PIC0REG 2n312	0	PIC0REG 2n310	PIC0REG 2n309	PIC0REG 2n308
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	PIC0REG 2n306	PIC0REG 2n305	PIC0REG 2n304	0	PIC0REG 2n302	PIC0REG 2n301	PIC0REG 2n300
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-45 PIC0REG2n3 register contents (1/3)

Bit position	Bit name	Function
22 21 20	PIC0REG2n322 PIC0REG2n321 PIC0REG2n320	Select the logical operation to perform on input signals A and B according to the output logic specified for CH9 of TAUAn.

PIC0REG 2n322	PIC0REG 2n321	PIC0REG 2n320	Input signal
1	0	0	A and B (Select this when the active high setting is specified (TAUAnTOL09 = 0).)
1	0	1	A or B (Select this when the active low setting is specified (TAUAnTOL09 = 1).)
Other than the above			Setting prohibited

Table 20-45 PIC0REG2n3 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>18
17
16</td><td>PIC0REG2n318
PIC0REG2n317
PIC0REG2n316</td><td>Select the logical operation to perform on input signals A and B according to the output logic specified for CH8 of TAUAn.
<table>
<tr><th>PIC0REG 2n318</th><th>PIC0REG 2n317</th><th>PIC0REG 2n316</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>0</td><td>A and B
(Select this when the active high setting is specified (TAUAnTOL08 = 0).)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>A or B
(Select this when the active low setting is specified (TAUAnTOL08 = 1).)</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>14
13
12</td><td>PIC0REG2n314
PIC0REG2n313
PIC0REG2n312</td><td>Select the logical operation to perform on input signals A and B according to the output logic specified for CH7 of TAUAn.
<table>
<tr><th>PIC0REG 2n314</th><th>PIC0REG 2n313</th><th>PIC0REG 2n312</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>0</td><td>A and B
(Select this when the active high setting is specified (TAUAnTOL07 = 0).)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>A or B
(Select this when the active low setting is specified (TAUAnTOL07 = 1).)</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>10
9
8</td><td>PIC0REG2n310
PIC0REG2n309
PIC0REG2n308</td><td>Select the logical operation to perform on input signals A and B according to the output logic specified for CH6 of TAUAn.
<table>
<tr><th>PIC0REG 2n310</th><th>PIC0REG 2n309</th><th>PIC0REG 2n308</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>0</td><td>A and B
(Select this when the active high setting is specified (TAUAnTOL06 = 0).)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>A or B
(Select this when the active low setting is specified (TAUAnTOL06 = 1).)</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
</table>

Table 20-45 PIC0REG2n3 register contents (3/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>6
5
4</td><td>PIC0REG2n306
PIC0REG2n305
PIC0REG2n304</td><td>Select the logical operation to perform on input signals A and B according to the output logic specified for CH5 of TAUAn.
<table>
<tr><th>PIC0REG 2n306</th><th>PIC0REG 2n305</th><th>PIC0REG 2n304</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>0</td><td>A and B
(Select this when the active high setting is specified (TAUAnTOL05 = 0).)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>A or B
(Select this when the active low setting is specified (TAUAnTOL05 = 1).)</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>2
1
0</td><td>PIC0REG2n302
PIC0REG2n301
PIC0REG2n300</td><td>Select the logical operation to perform on input signals A and B according to the output logic specified for CH4 of TAUAn.
<table>
<tr><th>PIC0REG 2n302</th><th>PIC0REG 2n301</th><th>PIC0REG 2n300</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>0</td><td>A and B
(Select this when the active high setting is specified (TAUAnTOL04 = 0).)</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>A or B
(Select this when the active low setting is specified (TAUAnTOL04 = 1).)</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
</table>

Caution Some of the bits defined as 0 in the PIC0REG2n3 register are defined for the other timers. For such bits, use the bit definition of those timers.

(5) TISLTA0BYPS1 - TAUA input selection register

The TISLTA0BYPS1 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 100C$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
TISLTA0BYPS17	TISLTA0BYPS16	TISLTA0BYPS15	TISLTA0BYPS14	TISLTA0BYPS13	TISLTA0BYPS12	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-46 TISLTA0BYPS1 register contents

Bit position	Bit name	Function
7	TISLTA0BYPS17	Select the signal input to TAUA0TTIN15. See chapters *"Delay Pulse Output with Dead Time"* and *"Three-Phase PWM Output with Dead Time"* for details.
6	TISLTA0BYPS16	Select the signal input to TAUA0TTIN14. (This is valid when TSOSLTA05 = 0.) 0: Input signal from TAUA0I15 1: Select the signal selected by PIC0REG20018, PIC0REG20224, and PIC0REG20225.
5	TISLTA0BYPS15	Select the signal input to TAUA0TTIN13. See chapters *"Delay Pulse Output with Dead Time"* and *"Three-Phase PWM Output with Dead Time"* for details.
4	TISLTA0BYPS14	Select the signal input to TAUA0TTIN12. (This is valid when TSOSLTA03 = 0.) 0: Input signal from TAUA0I13 1: Select the signal selected by PIC0REG20018, PIC0REG20220, and PIC0REG20221.
3	TISLTA0BYPS13	Select the signal input to TAUA0TTIN11. See chapters *"Delay Pulse Output with Dead Time"* and *"Three-Phase PWM Output with Dead Time"* for details.
2	TISLTA0BYPS12	Select the signal input to TAUA0TTIN10. (This is valid when TSOSLTA01 = 0.) 0: Input signal from TAUA0I11 1: Select the signal selected by PIC0REG20018, PIC0REG20216, and PIC0REG20217.

(6) TSOSLTA0 - TAUA input selection register

The TSOSLTA0 register selects TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 2014$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	TSOSLTA05	TSOSLTA04	TSOSLTA03	TSOSLTA02	TSOSLTA01	TSOSLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-47 TSOSLTA0 register contents

Bit position	Bit name	Function
5	TSOSLTA05	Select the signal input to TAUA0TTIN15. 0: Select the signal selected by TISLTA0BYPS17. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TSOSLTA04	Select the signal input to TAUA0TTIN14. 0: Select the signal selected by TISLTA0BYPS16. 1: Setting prohibited
3	TSOSLTA03	Select the signal input to TAUA0TTIN13. 0: Select the signal selected by TISLTA0BYPS15. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TSOSLTA02	Select the signal input to TAUA0TTIN12. 0: Select the signal selected by TISLTA0BYPS14. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TSOSLTA01	Select the signal input to TAUA0TTIN11. 0: Select the signal selected by TISLTA0BYPS13. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TSOSLTA00	Select the signal input to TAUA0TTIN10. 0: Select the signal selected by TISLTA0BYPS12. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(7) PIC0HIZCENn - Hi-Z output control register n

The PIC0HIZCENn register selects the Hi-Z output control input signal of TAPAn.

Access This register can be read or written in 8-bit units.

Address PIC0HIZCEN0: FF81 C0B4$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	PIC0HIZCENn6	0	0	0	PIC0HIZCENn2	PIC0HIZCENn1	PIC0HIZCENn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-48 PIC0HIZCENn register contents

Bit position	Bit name	Function
6	PIC0HIZCENn6	Select whether to enable or disable Hi-Z output control by the INTADCA0ERR interrupt signal. 0: Disable 1: Enable
2	PIC0HIZCENn2	Select whether to enable or disable Hi-Z output control by the WDTA0TNMI interrupt signal. 0: Disable 1: Enable
1	PIC0HIZCENn1	Select whether to enable or disable Hi-Z output control by the CLMATERR error signal from CLMA0, CLMA2, and CLMA3. 0: Disable 1: Enable
0	PIC0HIZCENn0	Select whether to enable or disable Hi-Z output control by the TAPAnESO pin input. 0: Disable 1: Enable

20.9 Delay Pulse Output with Dead Time

20.9.1 Functional overview

This feature outputs a three-phase PWM signal with dead time that is later than the cycle timing by an amount equal to the delay amount.

Unlike in chapter*"Three Phase PWM output with Dead Time"*, a PWM signal that has a reset can be output during the next cycle.

20.9.2 Configuration

The unit and channel configuration for this feature are shown below.

Table 20-49 Configuration of delay pulse output with dead time

Timer	Timer motor control function
TAUA0 CH2 to CH15 (used channels fixed)	TAPA0

Note The signal names used in the descriptions below are abbreviations. The actual signal names corresponding to each abbreviation are as follows:

- INTm → INTTAUAnIm (TAUAn channel m interrupt)
- TINm → TAUAnTTINm (TAUAn channel m input)
- TOUTm → TAUAnTOUTm (TAUAn channel m output)
- CDRm → TAUAnCDRm (TAUAn channel m data register)
- CNTm → TAUAnCNTm (TAUAn channel m counter register)

(1) TAUAn configuration

Because the CDRm value of CH3 does not affect TOUT0 to TOUT15, the INTm signal of CH3 can also be used for other purposes such as A/D conversion trigger generation.

Table 20-50 TAUA configuration

CH	Function name	M/S[a]	CDR setting	Description
2	Delay pulse output function (CH2 is the master channel for CH3 to CH9.)	M	Cycle	
3		S		Reserved
4		S	Delay (U phase)	
5		S	Pulse width (U phase)	
6		S	Delay (V phase)	
7		S	Pulse width (V phase)	
8		S	Delay (W phase)	
9		S	Pulse width (W phase)	
10	Any feature that does not use TOUTm	Any		TOUT: U-phase output
11	One-phase PWM output	-	Dead time (U phase)	TOUT: UB-phase output
12	Any feature that does not use TOUTm	Any		TOUT: V-phase output
13	One-phase PWM output	-	Dead time (V phase)	TOUT: VB-phase output
14	Any feature that does not use TOUTm	Any		TOUT: W-phase output
15	One-phase PWM output	-	Dead time (W phase)	TOUT: WB-phase output

a) M: Master channel, S: Slave channel, -: Standalone

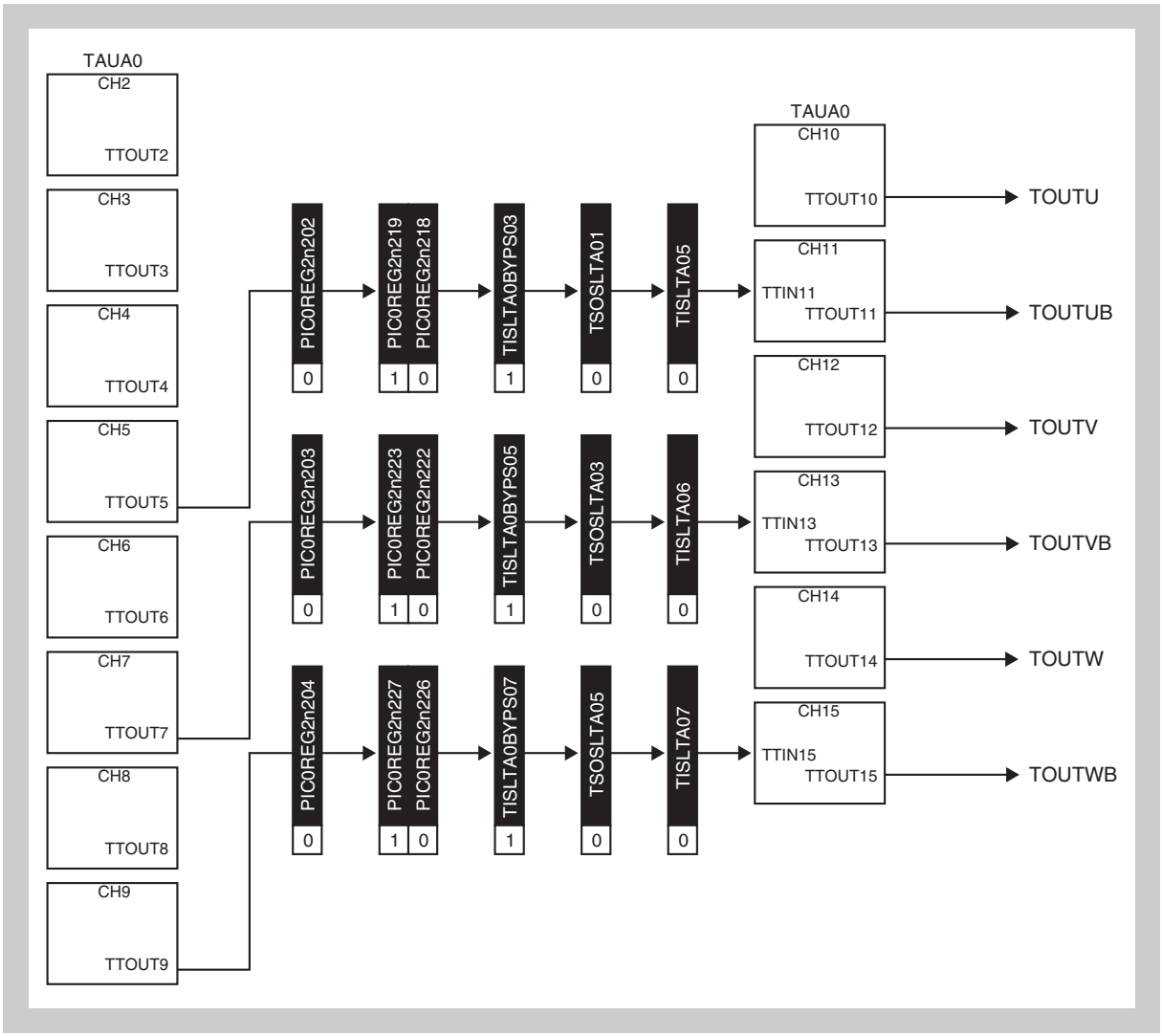


Figure 20-29 Block diagram: Delay pulse output with dead time

20.9.3 Operation example

This is achieved by combining the following TAUA features:

- Delay pulse output function
- One-phase PWM output

The delay pulse output feature generates a PWM signal that is later than the cycle timing by an amount equal to the delay amount. Next, a one-phase PWM signal to which dead time has been added is output for the delayed PWM signal.

A delay pulse with dead time is output by assigning the PWM output achieved using the above features to the U, V, and W phases. Therefore, the PWM output dead time can be freely specified for the PWM signal of each phase. Because the only difference between phases is the assigned channel, only one phase (the U phase) is described below.

(1) Delay pulse output function

A basic PWM signal for the one-phase PWM output delayed by the amount specified on CH4 for the cycle specified by TOUT05 on CH2 is output by using CH2, CH4, and CH5 in combination.

Note that CH3 is a reserved timer for achieving this feature, so do not use it for other features.

Caution Do not specify a delay amount that exceeds the cycle.

(2) One-phase PWM output

One-phase PWM output is generated from TOUT10 and TOUT11 by using a combination of CH10 and CH11.

By specifying the dead time value for CDR11, a one-phase PWM signal with dead time is output for the TIN11 input.

Similarly, the V phase uses CH12 and CH13 to output a one-phase PWM signal with dead time, while the W phase uses CH14 and CH15.

Caution Specify the same clock for each TAUAn channel that uses the delay pulse output and one-phase PWM output features.

For details about the TAUA functions, see the chapter *“Timer Array Unit A (TAUA)”*.

The differences between outputting a delay pulse with dead time and outputting a three-phase PWM signal with dead time are described below.

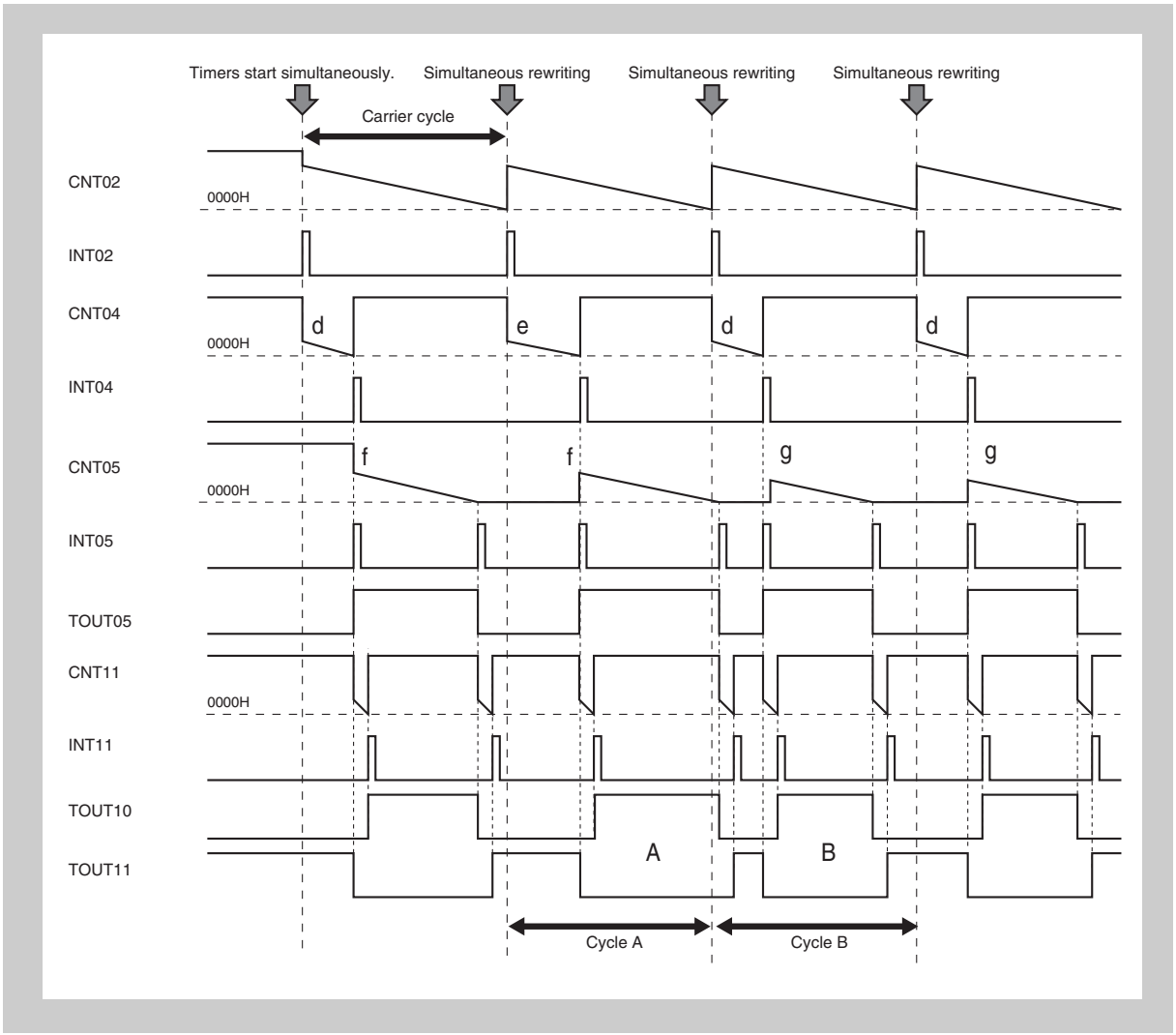


Figure 20-30 PWM output by outputting a delay pulse with dead time

In *Figure 20-30*, PWM waveform A is supposed to be output before cycle A ends, but, because the delay timing is too long, the PWM clear position is after the end of cycle A. Next, PWM waveform B, which is for cycle B, is output.

The operations shown below occur when an attempt is made to achieve the operations shown in *Figure 20-30 "PWM output by outputting a delay pulse with dead time"* by outputting a three-phase PWM signal with dead time.

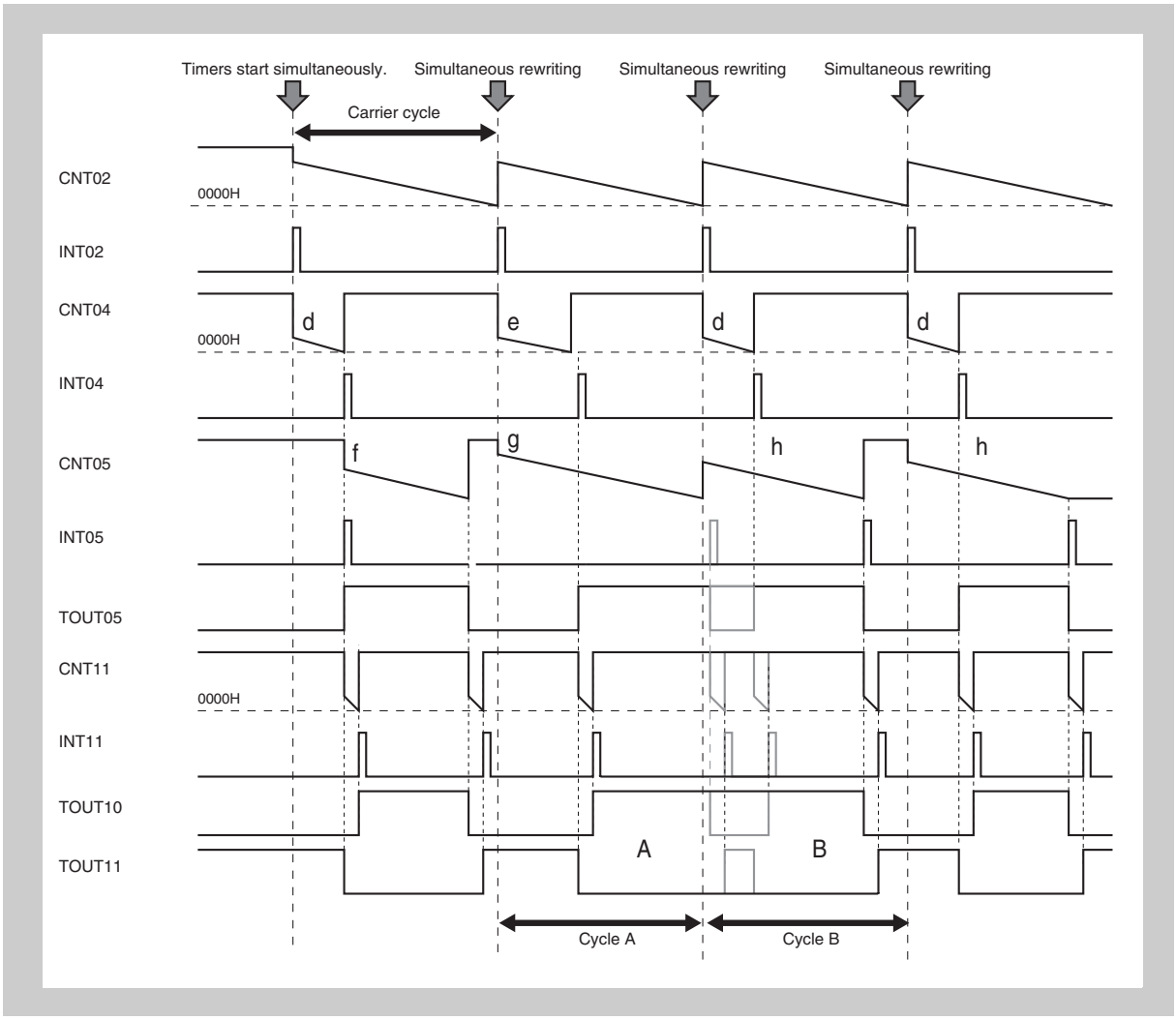


Figure 20-31 Output of a three-phase PWM signal with dead time (1)

Figure 20-31 shows an example in which the output PWM signal does not end before carrier cycle A because the set timing for outputting a three-phase PWM signal with dead time is delayed and the clear timing is after the end of the carrier cycle.

For cycle A, the set timing of PWM waveform A is the same as that in the figure on the previous page, but, because the clear timing is after the end of cycle A, a reload operation occurs in cycle A before PWM waveform A is cleared, and the clear timing for PWM waveform A does not occur.

In addition, the set timing of PWM waveform B for cycle B is ignored because a PWM waveform is already set. The result is that there is no PWM waveform change until the clear timing of cycle B, and a waveform that combines PWM waveform A and PWM waveform B is output.

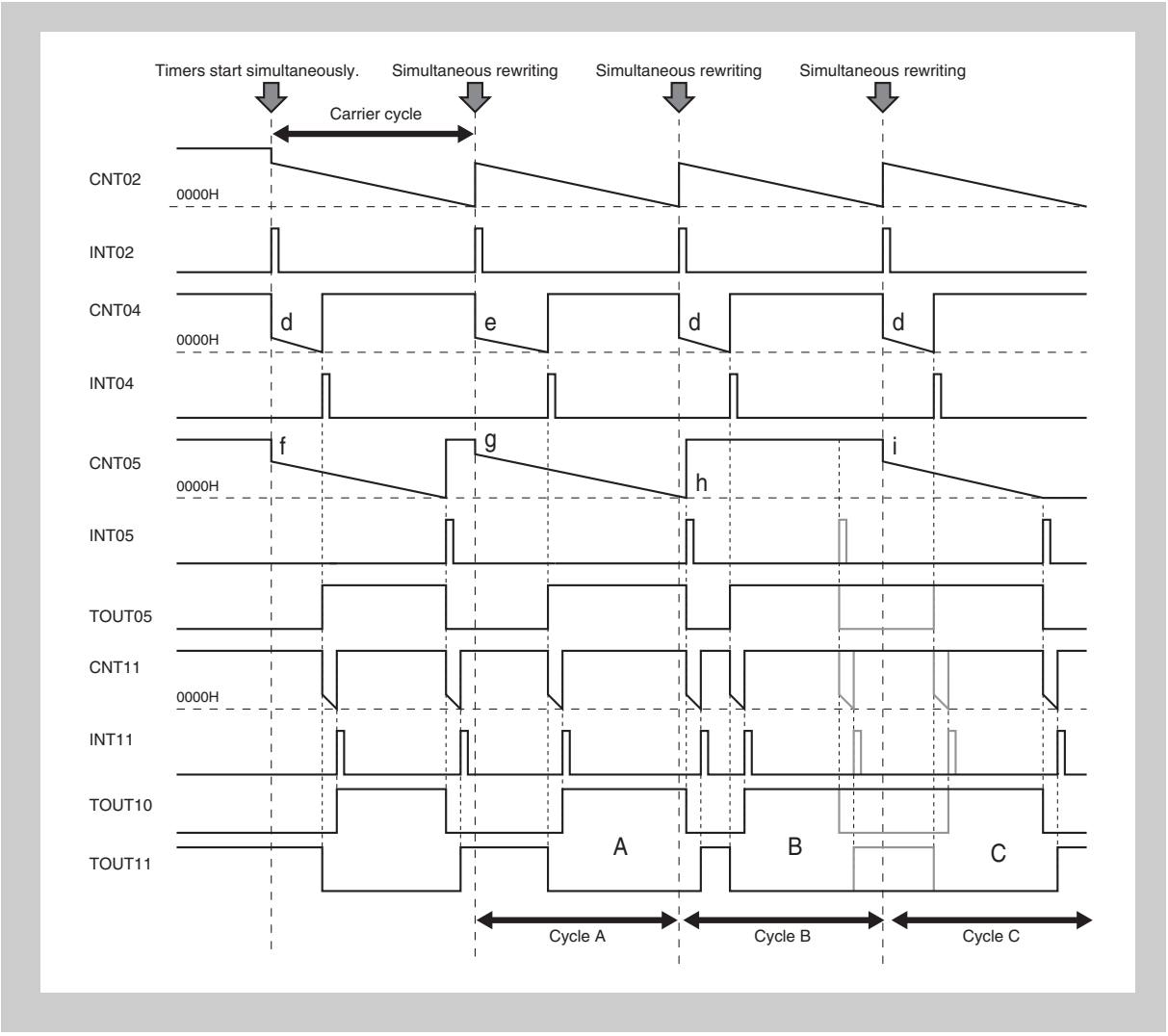


Figure 20-32 Output of a three-phase PWM signal with dead time (2)

Figure 20-32 shows an example of outputting a three-phase PWM signal with dead time in which counter operation for which the clear timing is longer than cycle A is continued in cycle B, and PWM output A is cleared at the beginning of cycle B.

The output of PWM waveform A for cycle A is the same as the output of a delay pulse with dead time, but, because the clear timing is used at the beginning of cycle B, the clear timing of PWM output B, which is supposed to be output during cycle B, does not occur.

In addition, the set timing of PWM waveform C for cycle C is ignored because a PWM waveform is already set. The result is that there is no PWM waveform change until the clear timing of cycle C, and a waveform that combines PWM waveform B and PWM waveform C is output.

In this way, it is possible to achieve freer PWM output timing when outputting a delay pulse with dead time than when outputting a three-phase PWM signal with dead time.

The peripheral interconnections provide a connection for using the PWM output timing of delay pulse output as input for one-phase PWM output.

Figure 20-33 shows a timing chart for outputting a delay pulse with dead time.

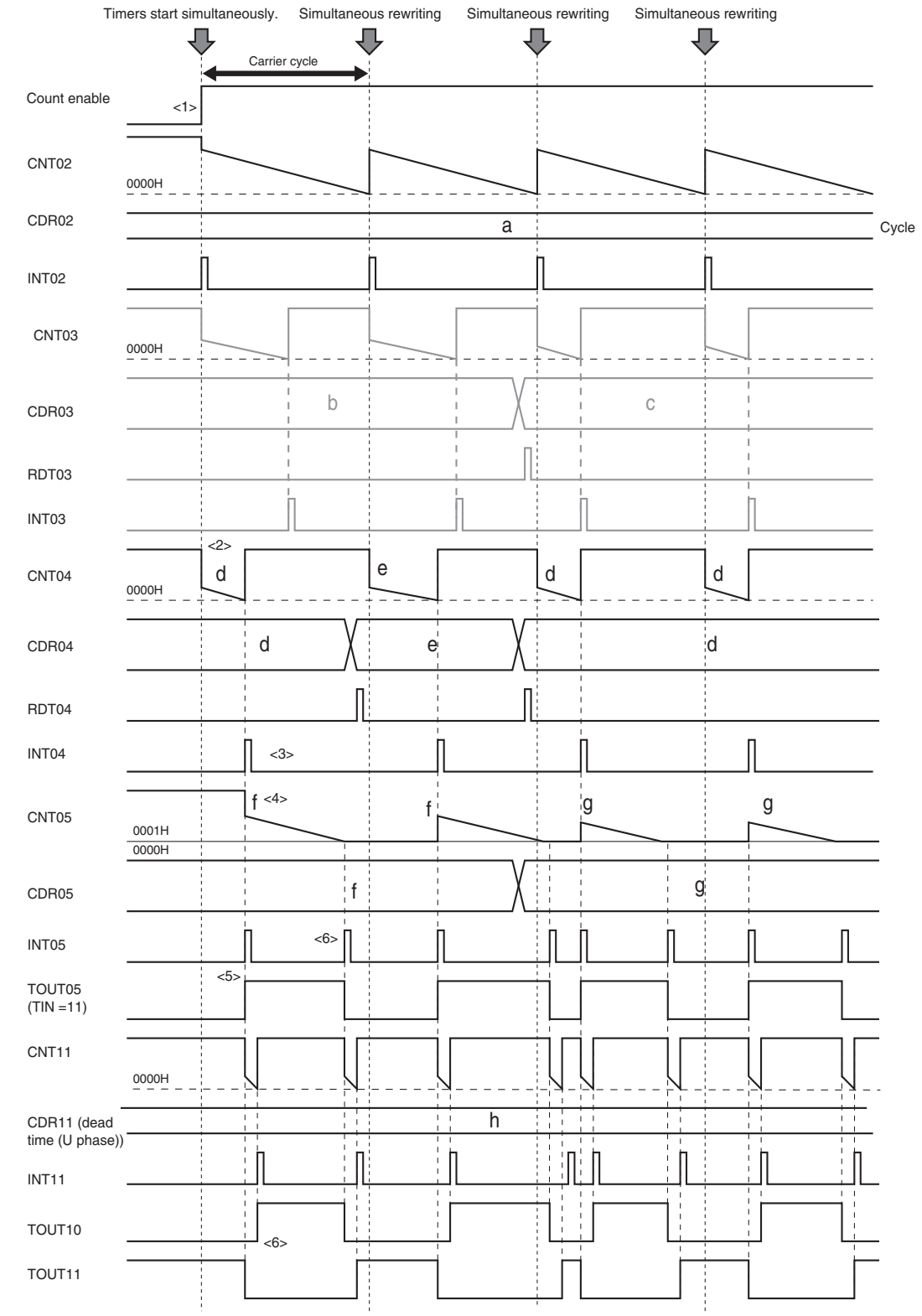


Figure 20-33 Output of a delay pulse with dead time

The output of a delay pulse with dead time shown in *Figure 20-33* is described below.

- CH2 (the carrier cycle timer) and CH4 (the delay timing timer) are started simultaneously by starting timers simultaneously (point <1> in the figure).

  CH5 (the PWM duty timer) and CH11 (the dead time timer) are also enabled, but no counting operations are performed until the edges of INT04, which indicates the count start timing for CH5, and TIN11, which indicates the count start timing for CH11, are detected.

  Because CH3 does not affect PWM output for this function, the channel is not described.

- For CH4, when there is a CH2 underflow, the settings from CDR04 are reloaded to CNT04 (point <2> in the figure).
- The CH4 underflow generates the delay timing signal (INT04) (point <3> in the figure).
- When INT04 is generated, the settings from CDR05 are reloaded to CNT05, and then the CH5 (the PWM duty timer) operation starts (point <4> in the figure).
- At this time, INT05 is generated and the TOUT05 output level changes to the active level (point <5> in the figure).
- Due to the CH5 underflow, INT05 is generated again, and TOUT05 changes to the inactive level (point <6> in the figure). TOUT05, which is changed by the CH4 and CH5 underflow, is supplied to the TIN11 input of one-phase PWM output.

  During one-phase PWM output, a PWM waveform with dead time is output by detecting a TIN11 edge.

20.9.4 Setup flow

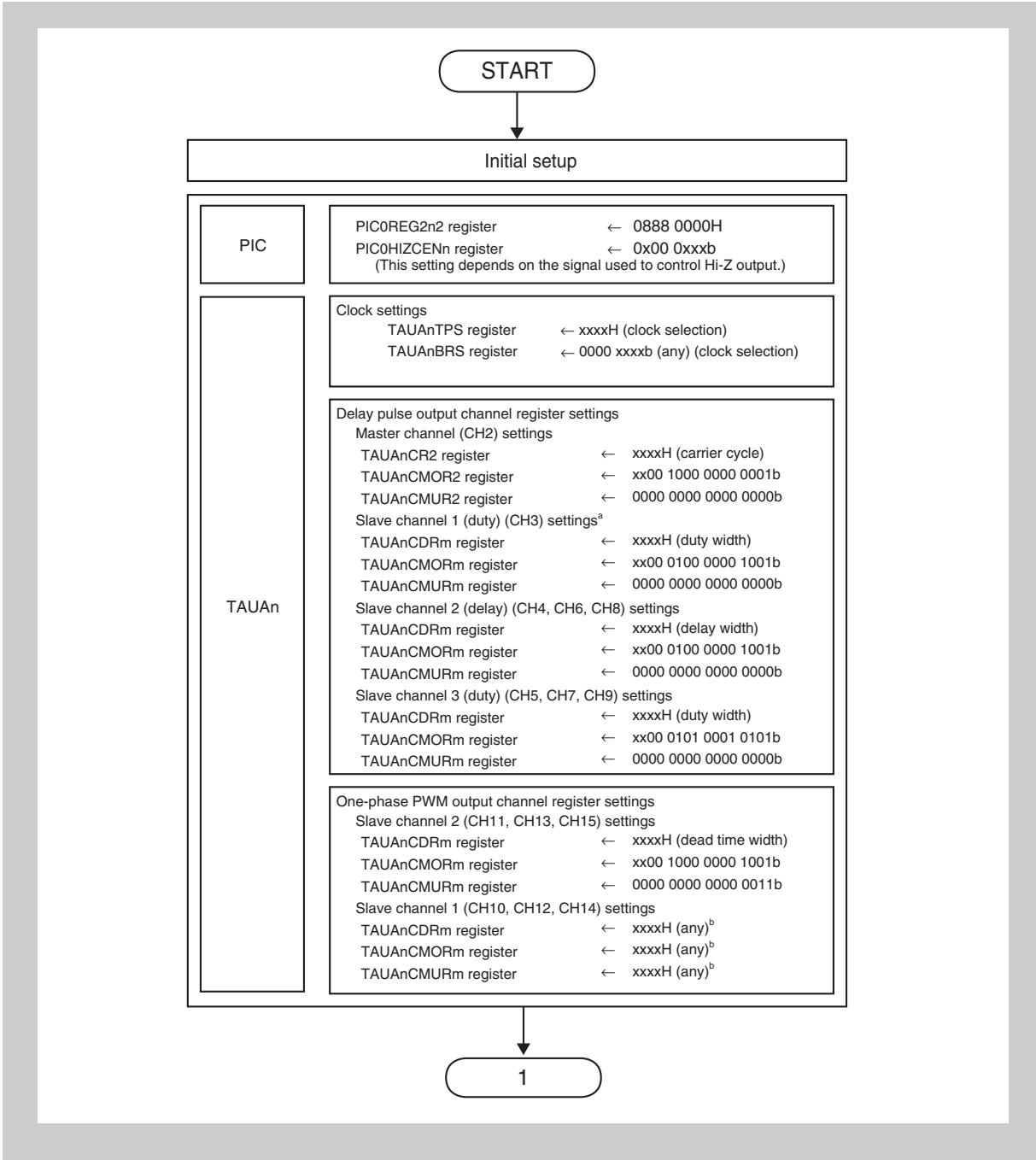


Figure 20-34 Setup flow (active high example)

a) For this feature, the slave channel does not affect operation, but it is set up because it is a configuration channel for delay pulse output.

b) Specify a feature that does not use TOUTm.

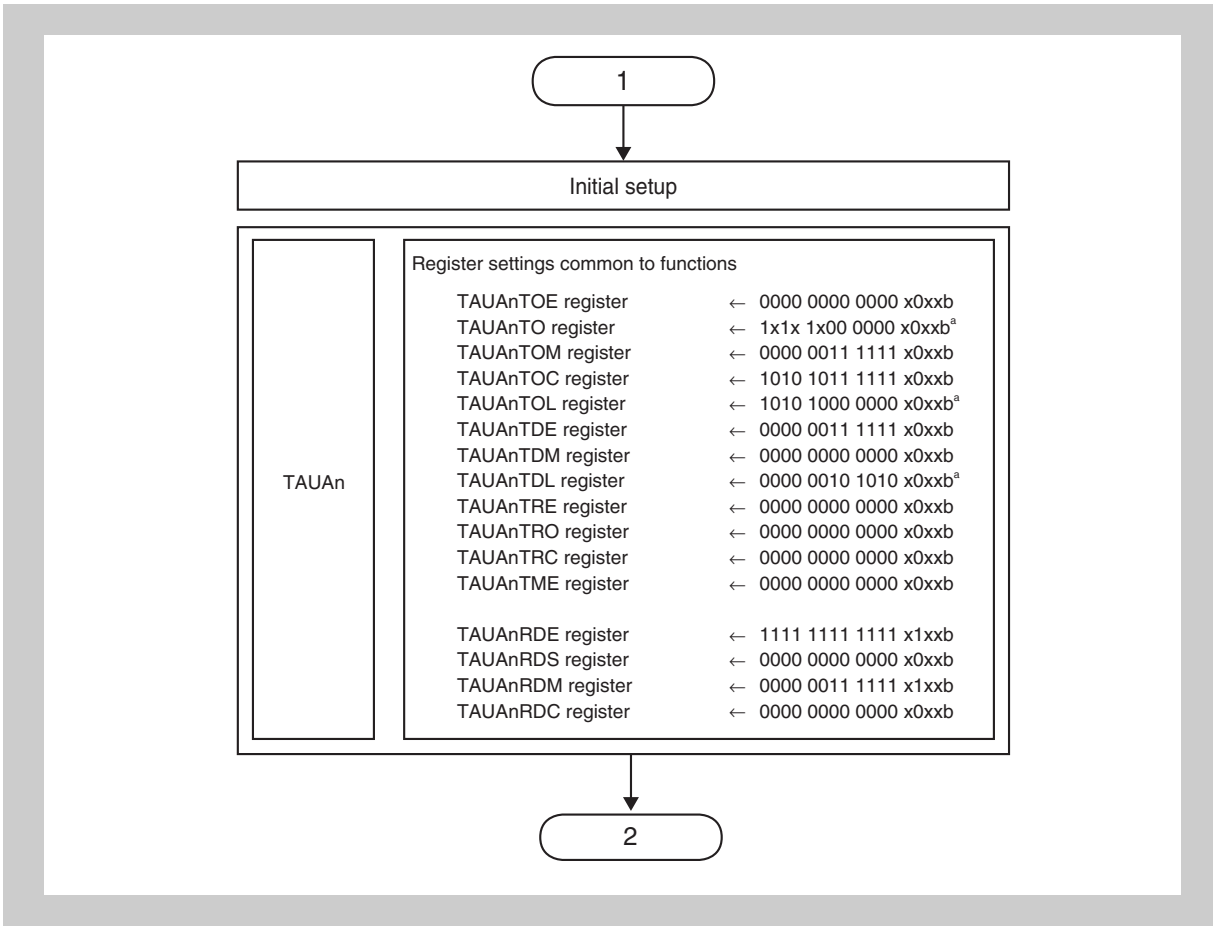


Figure 20-35 Setup flow (active high example) (continued)

a) Change settings according to the active level of the PWM signal to be output.

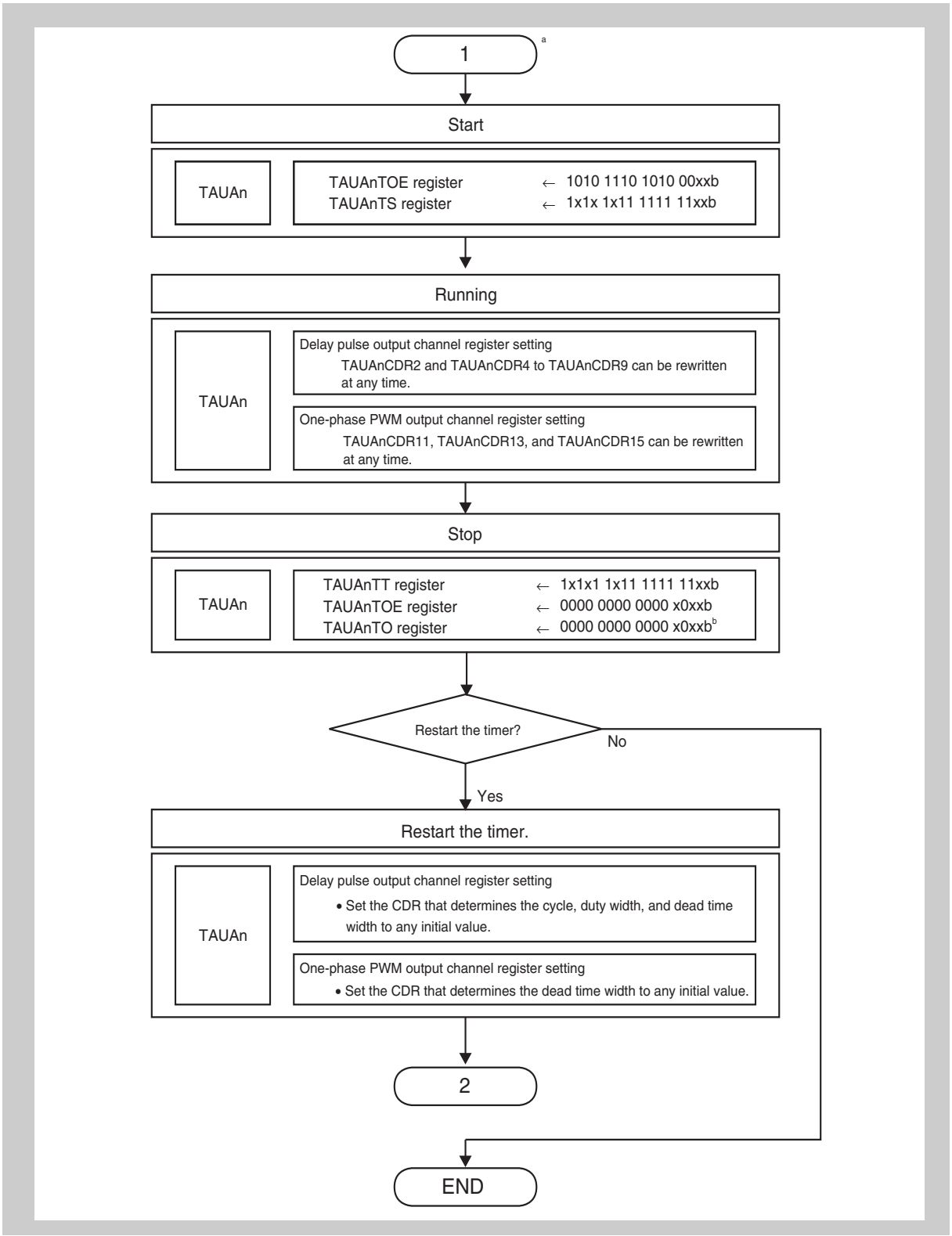


Figure 20-36 Setup flow (active high example) (continued)

a) Specify the selection register (such as TISLTA0BYPS1) and output port to use after specifying the initial settings for the peripheral interconnections and timers.

b) Change settings according to the active level of the PWM signal to be output.

20.9.5 Example of setting up operation functions

This section provides example settings for each register.

(1) TAUAn settings

Table 20-51 TAUAn: CH2-related (master channel used to output a delay pulse[a])

Register	Bit position	Bit name	Setting	Remark
TAUAnCMOR2	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	1	
	10 to 8	TAUAnSTS[2:0]	000	
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0000	
	0	TAUAnMD0	1	Output INTm at the start of operation.
TAUAnCMUR2	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA delay pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.
b) The same operation clock must be specified for the master channel and slave channel.

Table 20-52 TAUAn: CH3-related (slave channel used to output a delay pulse[a], [b])

Register	Bit position	Bit name	Setting	Remark
TAUAnCMOR3	15, 14	TAUAnCKS[1:0]	Any[c]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	100	Start trigger: INTm detection on the master channel
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	Enable start triggers during counting.
TAUAnCMUR3	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA delay pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.
b) The same operation clock must be specified for the master channel and slave channel.
c) For this feature, the channel does not affect operation, but it is set up because it is a configuration channel for delay pulse output.

Note For the TAUAnCMORm register used during delay pulse output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

Table 20-53 TAUAn: CH4, CH6, and CH8-related (slave channel 2 used to output a delay pulse[a]) (m = 4, 6, or 8)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	100	Start trigger: INTm detection on the master channel
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	Enable start triggers during counting.
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA delay pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the slave channel and master channel.

Note For the TAUAnCMORm register used during delay pulse output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

Table 20-54 TAUAn: CH5, CH7, and CH9-related (slave channel 3 used to output a delay pulse[a]) (m = 5, 7, or 9)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[b]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	101	Start trigger: INTm detection on an upper channel
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	1010	
	0	TAUAnMD0	1	Enable start triggers during counting.
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	00	

a) The master channel and slave channel names are defined for TAUA delay pulse output. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

b) The same operation clock must be specified for the slave channel and master channel.

Note For the TAUAnCMORm register used during delay pulse output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

Table 20-55 TAUAn: CH11, CH13, and CH15-related (one-phase PWM output) (m = 11, 13, or 15)

Register	Bit position	Bit name	Setting	Remark
TAUAnCMORm	15, 14	TAUAnCKS[1:0]	Any[a]	Operation clock setting
	13, 12	TAUAnCCS[1:0]	00	
	11	TAUAnMAS	0	
	10 to 8	TAUAnSTS[2:0]	001	Start trigger: Detection of a TINm-input valid edge
	7, 6	TAUAnCOS[1:0]	00	
	5		0	Fixed to 0
	4 to 1	TAUAnMD[4:1]	0100	
	0	TAUAnMD0	1	Enable start triggers during counting.
TAUAnCMURm	1, 0	TAUAnTIS[1:0]	11	Both rising and falling TINm edges are detected as valid. (High width)

a) Specify the same operation clock settings as for the PWM output master channel (CH2).

Note For the TAUAnCMORm register used during one-phase PWM output, TAUAnCKS[1:0] (which selects the operation clock) can be set to any value, but other control bits have fixed values. For details, see the chapter *"Timer Array Unit A (TAUA)*".
CH10, CH12, and CH14 can be used with any feature that does not use TOUTm output (such as A/D trigger output).

Table 20-56 Common TAUAn channel settings (1/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTOE	15 to 10	TAUAnTOE15 to TAUAnTOE10	0 1	Disable the timer. Enable the timer.
	9	TAUAnTOE09	0 1	Disable the timer. Enable the timer.
	8	TAUAnTOE08	0	This is fixed to 0 because TOUT08 is not used.
	7	TAUAnTOE07	0 1	Disable the timer. Enable the timer.
	6	TAUAnTOE06	0	This is fixed to 0 because TOUT06 is not used
	5	TAUAnTOE05	0 1	Disable the timer. Enable the timer.
	4	TAUAnTOE04	0	This is fixed to 0 because TOUT04 is not used
	3	TAUAnTOE03	0	This is fixed to 0 because TOUT03 is not used
	2	TAUAnTOE02	0	This is fixed to 0 because TOUT02 is not used.
	1, 0	TAUAnTOE01 TAUAnTOE00	Any	
TAUAnTO	15 to 10	TAUAnTO15 to TAUAnTO10	0[a]	Output a low-level signal to TOUT15 to TOUT10.
	9 to 2	TAUAnTO09 to TAUAnTO02	0	Output a low-level signal to TOUT09 to TOUT02.
	1, 0	TAUAnTO01 TAUAnTO00	Any	
TAUAnTOM	15 to 10	TAUAnTOM15 to TAUAnTOM10	1	Synchronous channel operation mode
	9 to 4	TAUAnTOM09 to TAUAnTOM04	0	Independent channel operation mode
	3	TAUAnTOM03	1	Synchronous channel operation mode
	2	TAUAnTOM02	0	Independent channel operation mode
	1, 0	TAUAnTOM01 TAUAnTOM00	Any	
TAUAnTOC	15 to 10	TAUAnTOC15 to TAUAnTOC10	1	Synchronous channel operation mode 2
	9 to 4	TAUAnTOC09 to TAUAnTOC04	0	Independent channel operation mode
	3	TAUAnTOC03	0	Synchronous channel operation mode 1
	2	TAUAnTOC02	0	Toggle mode
	1, 0	TAUAnTOC01 TAUAnTOC00	Any	
TAUAnTOL	15 to 10	TAUAnTOL15 to TAUAnTOL10	0[a]	Positive logic output (active high)
	9 to 2	TAUAnTOL09 to TAUAnTOL02	0	Positive logic output (active high)
	1, 0	TAUAnTOL01 TAUAnTOL00	Any	

Table 20-56 Common TAUAn channel settings (2/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnTDE	15 to 10	TAUAnTDE15 to TAUAnTDE10	1	Enable dead time control.[b]
	9 to 2	TAUAnTDE09 to TAUAnTDE02	0	Disable dead time control.
	1, 0	TAUAnTDE01 TAUAnTDE00	Any	
TAUAnTDM	15 to 10	TAUAnTDM15 to TAUAnTDM10	1	Output dead time upon detecting a TINm input edge at a lower odd channel.
	9 to 2	TAUAnTDM09 to TAUAnTDM02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDM01 TAUAnTDM00	Any	
TAUAnTDL	15	TAUAnTDL15	1[a]	Add dead time to the negative W phase period.
	14	TAUAnTDL14	0[a]	Add dead time to the positive W phase period.
	13	TAUAnTDL13	1[a]	Add dead time to the negative V phase period.
	12	TAUAnTDL12	0[a]	Add dead time to the positive V phase period.
	11	TAUAnTDL11	1[a]	Add dead time to the negative U phase period.
	10	TAUAnTDL10	0[a]	Add dead time to the positive U phase period.
	9 to 2	TAUAnTDL09 to TAUAnTDL02	0	Invalid because dead time control is disabled.
	1, 0	TAUAnTDL01 TAUAnTDL00	Any	
TAUAnTRE	15 to 2	TAUAnTRE15 to TAUAnTRE02	0	Disable real-time output.
	1, 0	TAUAnTRE01 TAUAnTRE00	Any	
TAUAnTRO	15 to 2	TAUAnTRO15 to TAUAnTRO02	0	Invalid because real-time output is disabled.
	1, 0	TAUAnTRO01 TAUAnTRO00	Any	
TAUAnTRC	15 to 2	TAUAnTRC15 to TAUAnTRC02	0	Do not use this channel to generate the real-time output trigger.
	1, 0	TAUAnTRC01 TAUAnTRC00	Any	
TAUAnTME	15 to 2	TAUAnTME15 to TAUAnTME02	0	Disable modulation output for timer output and real-time output.
	1, 0	TAUAnTME01 TAUAnTME00	Any	

Table 20-56 Common TAUAn channel settings (3/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnRDE	15	TAUAnRDE15	0	Disable simultaneous rewriting.
	14	TAUAnRDE14	Any	
	13	TAUAnRDE13	0	Disable simultaneous rewriting.
	12	TAUAnRDE12	Any	
	11	TAUAnRDE11	0	Disable simultaneous rewriting.
	10	TAUAnRDE10	Any	
	9 to 2	TAUAnRDE09 to TAUAnRDE02	1	Enable simultaneous rewriting.
	1, 0	TAUAnRDE01 TAUAnRDE00	Any	
TAUAnRDS	15	TAUAnRDS15	0	Do not enable simultaneous rewriting by using another upper channel.
	14	TAUAnRDS14	Any	
	13	TAUAnRDS13	0	Do not enable simultaneous rewriting by using another upper channel.
	12	TAUAnRDS12	Any	
	11	TAUAnRDS11	0	Do not enable simultaneous rewriting by using another upper channel.
	10	TAUAnRDS10	Any	
	9 to 2	TAUAnRDS09 to TAUAnRDS02	0	Enable simultaneous rewriting by using a master channel.
	1, 0	TAUAnRDS01 TAUAnRDS00	Any	
TAUAnRDM	15	TAUAnRDM15	0	Invalid because simultaneous rewriting is not enabled.
	14	TAUAnRDM14	Any	
	13	TAUAnRDM13	0	Invalid because simultaneous rewriting is not enabled.
	12	TAUAnRDM12	Any	
	11	TAUAnRDM11	0	Invalid because simultaneous rewriting is not enabled.
	10	TAUAnRDM10	Any	
	9 to 2	TAUAnRDM09 to TAUAnRDM02	0	Load the signal when the master channel starts counting.
	1, 0	TAUAnRDM01 TAUAnRDM00	Any	

Table 20-56 Common TAUAn channel settings (4/4)

Register	Bit position	Bit name	Setting	Remark
TAUAnRDC	15	TAUAnRDC15	0	Invalid because simultaneous rewriting is not enabled.
	14	TAUAnRDC14	Any	
	13	TAUAnRDC13	0	Invalid because simultaneous rewriting is not enabled.
	12	TAUAnRDC12	Any	
	11	TAUAnRDC11	0	Invalid because simultaneous rewriting is not enabled.
	10	TAUAnRDC10	Any	
	9 to 2	TAUAnRDC09 to TAUAnRDC02	0	Do not use this channel to generate the simultaneous rewrite trigger.
	1, 0	TAUAnRDC01 TAUAnRDC00	Any	

a) Change the setting according to the used system.

b) These are used to control positive/negative phase waveform output for which even channels are paired with odd channels to perform dead time control. For details, see the chapter *"Timer Array Unit A (TAUA)"*.

(2) Peripheral interconnections settings

Table 20-57 Peripheral interconnections settings

Register	Bit position	Bit name	Setting	Remark
PIC0REG2n2	27, 26	PIC0REG2n227 PIC0REG2n226	1 0	Select the input selected by the PIC0REG2n204 bit.
	23, 22	PIC0REG2n223 PIC0REG2n222	1 0	Select the input selected by the PIC0REG2n203 bit.
	19, 18	PIC0REG2n219 PIC0REG2n218	1 0	Select the input selected by the PIC0REG2n202 bit.
	4	PIC0REG2n204	0	Select TAUAnTTOUT09.
	3	PIC0REG2n203	0	Select TAUAnTTOUT07.
	2	PIC0REG2n202	0	Select TAUAnTTOUT05.

20.9.6 Registers

(1) PIC0REG2n2 - Timer I/O control register 2n2

Access This register can be read or written in 32-bit units.

Address PIC0REG202: FF81 C094$_H$

Initial value 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	PIC0REG2n227	PIC0REG2n226	PIC0REG2n225	PIC0REG2n224
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
PIC0REG2n223	PIC0REG2n222	PIC0REG2n221	PIC0REG2n220	PIC0REG2n219	PIC0REG2n218	PIC0REG2n217	PIC0REG2n216
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	PIC0REG2n204	PIC0REG2n203	PIC0REG2n202	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-58 PIC0REG2n2 register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>27
26</td><td>PIC0REG2n227
PIC0REG2n226</td><td>Select the signal input to TAUAnTTIN15.
<table>
<tr><th>PIC0REG2n227</th><th>PIC0REG2n226</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n204 bit</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>25
24</td><td>PIC0REG2n225
PIC0REG2n224</td><td>Select the signal input to TAUAnTTIN14.[a]</td></tr>
<tr><td>23
22</td><td>PIC0REG2n223
PIC0REG2n222</td><td>Select the signal input to TAUAnTTIN13.
<table>
<tr><th>PIC0REG2n223</th><th>PIC0REG2n222</th><th>Input signal</th></tr>
<tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n203 bit</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>21
20</td><td>PIC0REG2n221
PIC0REG2n220</td><td>Select the signal input to TAUAnTTIN12[a].</td></tr>
</table>

Table 20-58 PIC0REG2n2 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>19
18</td><td>PIC0REG2n219
PIC0REG2n218</td><td>Select the signal input to TAUAnTTIN11.
<table><tr><th>PIC0REG2n219</th><th>PIC0REG2n218</th><th>Input signal</th></tr><tr><td>1</td><td>0</td><td>Signal selected by the PIC0REG2n202 bit</td></tr><tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr></table></td></tr>
<tr><td>17
16</td><td>PIC0REG2n217
PIC0REG2n216</td><td>Select the signal input to TAUAnTTIN10[a].</td></tr>
<tr><td>4</td><td>PIC0REG2n204</td><td>Select the signal supplied to TAUAnTTIN15.
0: Select TAUAnTTOUT9.
1: Select the set/clear output according to INTTAUAnI8 and INTTAUAnI9[b].</td></tr>
<tr><td>3</td><td>PIC0REG2n203</td><td>Select the signal supplied to TAUAnTTIN13.
0: Select TAUAnTTOUT7.
1: Select the set/clear output according to INTTAUAnI6 and INTTAUAnI7[b].</td></tr>
<tr><td>2</td><td>PIC0REG2n202</td><td>Select the signal supplied to TAUAnTTIN11.
0: Select TAUAnTTOUT5.
1: Select the set/clear output according to INTTAUAnI4 and INTTAUAnI5[b]</td></tr>
</table>

a) Refer to chapter “Connection with the Timer Motor Control Feature (High-accuracy Triangle PWM Output with Dead Time)” for details.

b) Refer to chapter “Timer Motor Control and Connection (Three-Phase PWM Output with Dead Time)” for details.

Caution Some of the bits defined as 0 in the PIC0REG2n2 register are defined for the other timers. For such bits, use the bit definition of those timers.

(2) TISLTA0BYPS1 - TAUA input selection register

The TISLTA0BYPS1 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 100C$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
TISLTA0BYPS17	TISLTA0BYPS16	TISLTA0BYPS15	TISLTA0BYPS14	TISLTA0BYPS13	TISLTA0BYPS12	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-59 TISLTA0BYPS1 register contents

Bit position	Bit name	Function
7	TISLTA0BYPS17	Select the signal input to TAUA0TTIN15. (This is valid when TISLTA07 = 0 and TSOSLTA05 = 0.) 0: Input signal from TAUA0I15 1: Select the signal selected by PIC0REG20204, PIC0REG20226, and PIC0REG2027.
6	TISLTA0BYPS16	Select the signal input to TAUA0TTIN14. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.
5	TISLTA0BYPS15	Select the signal input to TAUA0TTIN13. (This is valid when TISLTA06 = 0 and TSOSLTA03 = 0.) 0: Input signal from TAUA0I13 1: Select the signal selected by PIC0REG20203, PIC0REG20222, and PIC0REG20223.
4	TISLTA0BYPS14	Select the signal input to TAUA0TTIN12. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.
3	TISLTA0BYPS13	Select the signal input to TAUA0TTIN11. (This is valid when TISLTA05 = 0 and TSOSLTA01 = 0.) 0: Input signal from TAUA0I11 1: Select the signal selected by PIC0REG20202, PIC0REG20218, and PIC0REG20219.
2	TISLTA0BYPS12	Select the signal input to TAUA0TTIN10. See chapter *"High-accuracy Triangle PWM Output with Dead Time"* for details.

(3) TSOSLTA0 - TAUA input selection register

The TSOSLTA0 register selects TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 2014$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	0	TSOSLTA05	TSOSLTA04	TSOSLTA03	TSOSLTA02	TSOSLTA01	TSOSLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-60 TSOSLTA0 register contents

Bit position	Bit name	Function
5	TSOSLTA05	Select the signal input to TAUA0TTIN15. 0: Select the signal selected by TISLTA0BYPS17. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TSOSLTA04	Select the signal input to TAUA0TTIN14. 0: Select the signal selected by TISLTA0BYPS16. 1: Setting prohibited.
3	TSOSLTA03	Select the signal input to TAUA0TTIN13. 0: Select the signal selected by TISLTA0BYPS15. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TSOSLTA02	Select the signal input to TAUA0TTIN12. 0: Select the signal selected by TISLTA0BYPS14. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TSOSLTA01	Select the signal input to TAUA0TTIN11. 0: Select the signal selected by TISLTA0BYPS13. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TSOSLTA00	Select the signal input to TAUA0TTIN10. 0: Select the signal selected by TISLTA0BYPS12. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(4) TISLTA0 - TAUA input selection register

The TISLTA0 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 1000_H

Initial value 00_H

7	6	5	4	3	2	1	0
TISLTA07	TISLTA06	TISLTA05	TISLTA04	TISLTA03	TISLTA02	TISLTA01	TISLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-61 TISLTA0 register contents

Bit position	Bit name	Function
7	TISLTA07	Select the signal input to TAUA0TTIN15. 0: Select the signal selected by TSOSLTA05. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
6	TISLTA06	Select the signal input to TAUA0TTIN13. 0: Select the signal selected by TSOSLTA03. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
5	TISLTA05	Select the signal input to TAUA0TTIN11. 0: Select the signal selected by TSOSLTA01. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TISLTA04	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
3	TISLTA03	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TISLTA02	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TISLTA01	1: See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TISLTA00	1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(5) PIC0HIZCENn - Hi-Z output control register n

The PIC0HIZCENn register selects the Hi-Z output control input signal of TAPAn.

Access This register can be read or written in 8-bit units.

Address PIC0HIZCEN0: FF81 C0B4$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
0	PIC0HIZCENn6	0	0	0	PIC0HIZCENn2	PIC0HIZCENn1	PIC0HIZCENn0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 20-62 PIC0HIZCENn register contents

Bit position	Bit name	Function
6	PIC0HIZCENn6	Select whether to enable or disable Hi-Z output control by the INTADCA0ERR interrupt signal. 0: Disable 1: Enable
2	PIC0HIZCENn2	Select whether to enable or disable Hi-Z output control by the WDTA0TNMI interrupt signal. 0: Disable 1: Enable
1	PIC0HIZCENn1	Select whether to enable or disable Hi-Z output control by the CLMATERR error signal from CLMA0, CLMA2, and CLMA3. 0: Disable 1: Enable
0	PIC0HIZCENn0	Select whether to enable or disable Hi-Z output control by the TAPAnESO pin input. 0: Disable 1: Enable

Chapter 21 Encoder Timer

21.1 Basic structure of Encoder Timer

The Encoder Timer (ENCA) is used for several functions, which involve also other modules, as shown in the basic block diagram below.

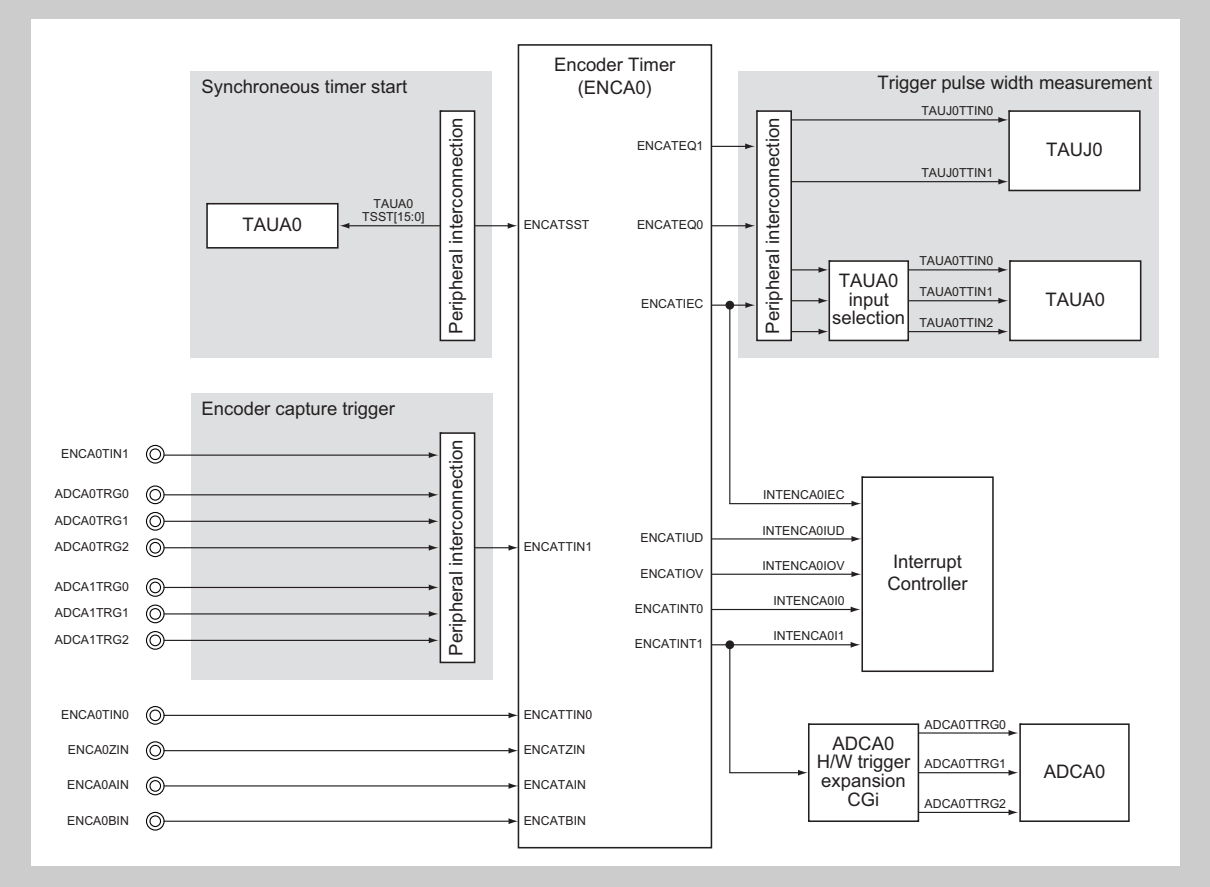


Figure 21-1 Basic structure of Encoder Timer

Peripheral interconnection circuits are used to set up the required connections between the ENCA0 and the other involved modules.

The following sections

- *V850E2/Fx4-H Encoder Timer (ENCA) Features*
- *ENCA Functional Overview*
- *ENCA Control Registers*
- *ENCA Functional Description*
- *ENCA Setting Sequences*

describe the function of the Encoder Timer (ENCA) module.

Afterwards the sections

- *A/D Trigger Encoder Capture*
- *Synchronized Timer Operation*
- *Trigger Pulse Width Measurement*

describe how to set up the involved modules for dedicated applications.

21.2 V850E2/Fx4-H Encoder Timer (ENCA) Features

This chapter contains a generic description of the Encoder Timer (ENCA).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

Instances

This microcontroller has following number of instances of the Encoder Timer.

Table 21-1 Instances of ENCA

Encoder Timer	
Instance	1
Name	ENCA0

Instances index n

Throughout this chapter, the individual instances of a Encoder Timer is identified by the index "n" (n = 0), for example, ENCAnCTL for the ENCAn control register.

Register addresses

All ENCAn register addresses are given as address offsets to the individual base address <ENCAn_base>.
The base address <ENCAn_base> of each ENCAn is listed in the following table:

Table 21-2 Register base addresses <ENCAn_base>

ENCAn instance	<ENCAn_base> address
ENCA0	FF81 9000_H

Clock supply All Encoder Timers provide one clock input:

Table 21-3 ENCAn clock supply

ENCAn instance	ENCAn clock	Connected to
ENCA0	PCLK	Clock Controller CKSCLK_006

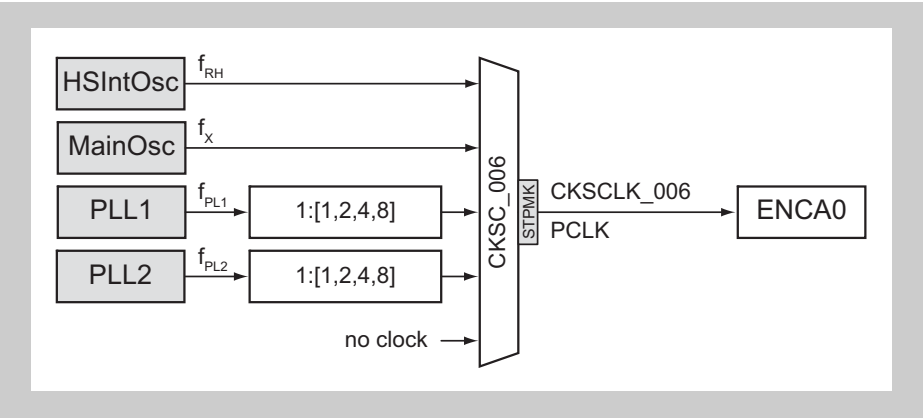


Figure 21-2 ENCA clock supply

Interrupts The Encoder Timer can generate the following interrupt requests:

Table 21-4 ENCAn interrupt requests

ENCAn signals	Function	Connected to
ENCA0:		
ENCATIOV	ENCA0 overflow interrupt	Interrupt Controller INTENCA0IOV
ENCATIUD	ENCA0 underflow interrupt	Interrupt Controller INTENCA0IUD
ENCATINT0	ENCA0 compare match 0 or capture 0 interrupt	Interrupt Controller INTENCA0I0
ENCATINT1	ENCA0 compare match 1or capture 1 interrupt	Interrupt Controller INTENCA0I1[a]
ENCATIEC	ENCA0 clear interrupt by encoder input (phase Z)	Interrupt Controller INTENCA0IEC, trigger pulse width measurement (refer to section *"Trigger Pulse Width Measurement"* below for details)

a) These signals can be used to as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

ENCA H/W reset The Encoder Timers and their registers are initialized by the following reset signal:

Table 21-5 ENCAn reset signal

ENCAn	Reset signal
ENCA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

Internal signals The internal signal connections of the Encoder Timers are listed in the following table.

Table 21-6 ENCAn internal signal connections

ENCAn signal	Function	Connected to
ENCA0:		
ENCATSST	ENCA0 simultaneous start trigger input	Synchroneous timers start (refer to section *"Synchronized Timer Operation"* below for details)
ENCATSTT	ENCA0 simultaneous stop trigger input	not connected
ENCATEQ0	Match detection signal 0	Trigger pulse width measurement (refer to section *"Trigger Pulse Width Measurement"* below for details.
ENCATEQ1	Match detection signal 1	

I/O signals The I/O signals of the Encoder Timers are listed in the following table.

Table 21-7 ENCAn I/O signals

ENCAn signal	Function	Connected to
ENCA0:		
ENCATTIN0	ENCA0 capture trigger input 0	Port ENCA0TIN0[a]
ENCATTIN1	ENCA0 capture trigger input 1	Port ENCA0TIN1[ab]
ENCATAIN	ENCA0 encoder input (phase A)	Port ENCA0AIN[a]
ENCATBIN	ENCA0 encoder input (phase B)	Port ENCA0BIN[a]
ENCATZIN	ENCA0 encoder input (phase Z)	Port ENCA0ZIN[a]

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

b) See section *"A/D Trigger Encoder Capture"* below for details.

Caution The initial state of the input signals noise filters block the input signals. Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

21.3 ENCA Functional Overview

Features summary

- Generation of the counter control signal from the encoder input signal, and performs count operation in synchronization with PCLK.
- Capture function for capturing the counter value with an external trigger signal
- Compare function for compare match judgment with the counter value
- Two capture compare registers that can be set separately for capture operation and for compare operation
- Interrupt mask function for masking the interrupt signal output as a result of compare match judgment during compare operation
- Function for loading the value of the capture compare register to the counter upon underflow occurrence
- Encoder input signal can be applied to the timer counter clear condition
- Edge or level for clearing the encoder input signal of the timer counter clear condition can be selected
- Detection of counter overflow and underflow and output error flags and occurrence interrupts
- Five interrupts: 2 capture compare interrupts, 1 counter clear interrupt, 1 overflow interrupt, and 1 underflow interrupt.

21.3.1 Block Diagram

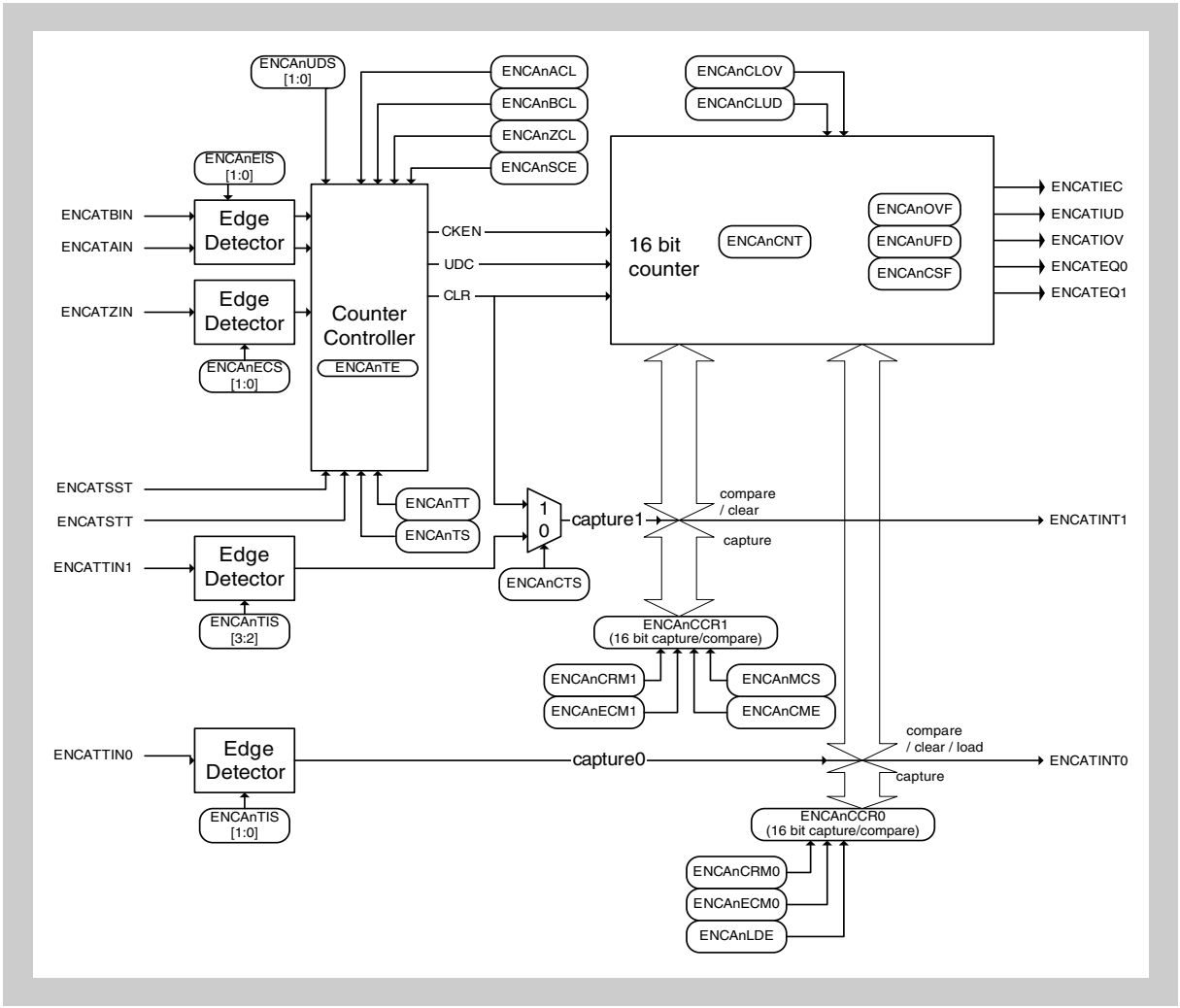


Figure 21-3 Encoder Timer block diagram

21.3.2 Preliminary knowledge for understanding basic specifications

As this macro has an encoder counter function, the rotary encoder is explained below.

Rotary Encoder

The rotary encoder outputs a pulse according to the amount of displacement of the axis of rotation. The output pulse is output as a waveform that has two phase differences, from the fact that the light passing through two slits (phase A, phase B) is transmitted or blocked, when the disc, on which an optical pattern is written, rotates, along with the displacement of the axis of rotation.

The rotation can be detected by counting the output pulses with a counter.

Moreover, there is a pulse called phase Z, which indicates the origin within 1 rotation of the encoder.

Errors can be corrected by clearing the counter to 0000_H by using the phase Z pulse.

Position measurement and speed measurement applications can be achieved by using the rotation measured with the timer counter.

21.4 ENCA Control Registers

The ENCAn is controlled and operated by means of the following registers:

Table 21-8 ENCAn register overview

Register name	Shortcut	Address
ENCA capture compare register 0	ENCAnCCR0	<ENCAn_base>
ENCA capture compare register 1	ENCAnCCR1	<ENCAn_base> + 04_H
ENCA counter register	ENCAnCNT	<ENCAn_base> + 08_H
ENCA status flag register	ENCAnFLG	<ENCAn_base> + $0C_H$
ENCA status flag clear register	ENCAnFGC	<ENCAn_base> + 10_H
ENCA timer enable status register	ENCAnTE	<ENCAn_base> + 14_H
ENCA timer start trigger register	ENCAnTS	<ENCAn_base> + 18_H
ENCA timer stop trigger register	ENCAnTT	<ENCAn_base> + $1C_H$
ENCA I/O control register 0	ENCAnIOC0	<ENCAn_base> + 20_H
ENCA control register	ENCAnCTL	<ENCAn_base> + 40_H
ENCA I/O control register 1	ENCAnIOC1	<ENCAn_base> + 44_H

<ENCAn_base> The base addresses <ENCAn_base> of the ENCAn is defined in the first section of this chapter under the key word “Register addresses”.

(1) ENCAnCTL – ENCA control register

This register is used to configure various operation settings of the Encoder Timer.

Access This register can be read/written in 16-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 40_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8
ENCAn CME	ENCAn MCS	0	0	0	0	ENCAn CRM1	ENCAn CRM0
R/W	R/W	R	R	R	R	R/W	R/W

7	6	5	4	3	2	1	0
ENCAn CTS	0	0	ENCAn LDE	ENCAn ECM1	ENCAn ECM0	ENCAn UDS[1:0]	
R/W	R	R	R/W	R/W	R/W	R/W	R/W

Table 21-9 ENCAnCTL register contents (1/2)

Bit	Name	Function
15	ENCAnCME	Encoder clear mask enable bit This bit is used to enable/disable masking of compare match interrupt detection when the compare function is used. 0: Disables the compare match interrupt (ENCATINT1) mask function for the ENCAnCCR1 register 1: Enables the compare match interrupt (ENCATINT1) mask function for the ENCAnCCR1 register. This bit is valid only when ENCAnCRM1 = 0. When this bit is set to “1”, setting ENCAnECM1 to “1” is prohibited.
14	ENCAnMCS	Encoder mask clear select bit This bit is used to select the trigger for cancelling masking of compare match interrupt detection when the compare function is used. This bit is valid only when ENCAnCRM1 = 0. 0 Masking of compare match interrupt detection is cancelled when the ENCAnCCR1 register is written. 1: Masking of compare match interrupt detection is cancelled when one of the following three operations is performed. -Timer counter clear operation accompanying phase Z -Timer counter clear operation upon compare match between ENCAnCNT and ENCAnCCR0 when ENCAnECM0 = 1 -Loading from ENCAnCCR0 to timer counter upon underflow detection when ENCAnLDE = 1
9	ENCAnCRM1	ENCACCR1 register mode bit 0: ENCAnCCR1 used as compare register. 1: ENCAnCCR1 used as capture register.
8	ENCAnCRM0	ENCACCR0 register mode bit 0: ENCAnCCR0 used as compare register. 1: ENCAnCCR0 used as capture register.

Table 21-9 ENCAnCTL register contents (2/2)

Bit	Name	Function
7	ENCAnCTS	ENCAnCCR1 capture trigger select bit This is a trigger selection bit register for the capture operation to the ENCAnCCR1 register. This bit selects the capture trigger to the ENCAnCCR1 register. This bit is valid only when ENCAnCRM1 = 1. 0: Uses ENCATTIN1 of capture trigger 1 signal as the trigger for capturing to the ENCAnCCR1 register. 1: Uses ENCATZIN of the phase Z encoder input signal as the trigger for capturing to the ENCAnCCR1 register.
4	ENCAnLDE	ENCAn counter load enable bit This register is used to enable/disable setting value loading to the counter upon underflow occurrence. This bit is valid only when ENCAnCRM0 = 0. When ENCAnCRM0 = 1, loading of the ENCAnCCR0 register setting value to the counter upon occurrence of an underflow is not performed, regardless of the value of this bit. 0: Disable loading of ENCAnCCR0 register setting value to counter upon occurrence of a counter underflow. 1: Enable loading of ENCAnCCR0 register setting value to counter upon occurrence of a counter underflow.
3	ENCAnECM1	Encoder clear mode bit 1 This register is used to set the counter clear operation upon match between the counter value and ENCAnCCR1 setting value. This bit is valid only when ENCAnCRM1 = 0. 0: Does not clear the counter to 0000_H upon match of timer counter value and ENCAnCCR1 setting value. 1: Clears the counter to 0000_H upon match of timer counter value and ENCAnCCR1 setting value if the next count is a up-count.
2	ENCAnECM0	Encoder clear mode bit 0 This register is used to set the counter clear operation upon match between the counter value and ENCAnCCR0 setting value. This bit is valid only when ENCAnCRM0 = 0. 0: Does not clear the counter to 0000_H upon match of timer counter value and ENCAnCCR0 setting value. 1: Clears the counter to 0000_H upon match of timer counter value and ENCAnCCR0 setting value if the next count is a down-count.
1, 0	ENCAn UDS[1:0]	UPDOWN count selection bits 1 and 0 This is the counter up/down control register using ENCATAIN and ENCATBIN. 00: Upon detection of valid edge of ENCATAIN, - down-count when ENCATBIN = H, - up-count when ENCATBIN = L 01: Upon detection of valid edge of ENCATAIN, up-count, Upon detection of valid edge of ENCATBIN, down-count 10: At rising edge of ENCATAIN, down-count At falling edge of ENCATAIN, up-count However, count operation performed only when ENCATBIN = L. 11: Detection of both edges of ENCATAIN, ENCATBIN. Judgment of count operation combining both detected edge and level.

(2) ENCAnIOC0 – ENCA I/O control register 0

This register is used to select the input edge of capture triggers 0 and 1 (ENCAnTTIN0, ENCAnTTIN1).

Access This register can be read/written in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 20_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	ENCAnTIS[3:2]		ENCAnTIS[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-10 ENCAnIOC0 register contents

Bit	Name	Function
3, 2	ENCAn TIS[3:2]	Input edge selection bits for capture trigger 1. These bits are valid only when the ENCAnCTL register's ENCAnCRM1 = 1 and ENCAnCTS = 0. All other settings of ENCAnCRM1 and ENCAnCTS are invalid. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection
1, 0	ENCAn TIS[1:0]	Input edge selection bits for capture trigger 0. These bits are valid only when ENCAnCTL.ENCAnCRM0 = 1 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection

(3) ENCAnIOC1 – ENCA I/O control register 1

This register is used to perform the clear condition setting and edge selection upon encoder input.

Access This register can be read/written in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + 44_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
ENCAn SCE	ENCAn ZCL	ENCAn BCL	ENCAn ACL	ENCAn ECS[1:0]		ENCAn EIS[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-11 ENCAnIOC1 register contents (1/2)

Bit	Name	Function
7	ENCAnSCE	Encoder special-clear enable bit This is an encoder special clear enable bit. When cleared to 0, the counter is cleared upon phase Z edge detection. When set to 1, the counter is cleared using the level of ENCATZIN, ENCATAIN, and ENCATBIN (special clear). When setting this bit to 1, set ENCAnUDS1 and ENCAnUDS0 to {1, 0} or {1, 1}. The operation is not guaranteed if this bit is set to 1 with ENCAnUDS1 and ENCAnUDS0 set to {0, 0} or {0, 1}. 0: Clears the counter upon detection of ENCATZIN valid edge (set with ENCAnECS1 and ENCAnECS0). 1: Clears the counter upon detection of input level condition of ENCATZIN, ENCATBIN and ENCATAIN (set with ENCAnZCL bit, ENCAnBCL bit, and ENCAnACL bit).
6	ENCAnZCL	Input-Z clear condition selection bit This bit is used to set the condition for clearing the phase Z encoder input (ENCAnZIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level
5	ENCAnBCL	Input-B clear condition selection bit This bit is used to set the condition for clearing the phase B encoder input (ENCAnBIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level

Table 21-11 ENCAnIOC1 register contents (2/2)

Bit	Name	Function
4	ENCAnACL	Input-A clear condition selection bit This bit is used to set the condition for clearing the phase A encoder input (ENCAnAIN) when using the encoder special clear function. This bit is valid only when ENCAnSCE = 1; it is invalid when ENCAnSCE = 0. 0: Clear condition: Low level 1: Clear condition: High level
3, 2	ENCAn ECS[1:0]	Encoder clear input edge selection bits 1 and 0 These are the encoder clear input edge selection bits (phase Z). These bits are valid only when ENCAnSCE = 0; they are invalid when ENCAnSCE = 1. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection
1, 0	ENCAn EIS[1:0]	Encoder input selection bits 1 and 0 These are the encoder input edge selection bits (phase A, phase B). These bits are valid when ENCAnUDS1 and ENCAnUDS0 = {0, 0} or {0, 1}, and are invalid when ENCAnUDS1 and ENCAnUDS0 = {1, 0} or {1, 1}. 00: No edge detection 01: Rising edge detection 10: Falling edge detection 11: Both edges detection

(4) ENCAnFLG – ENCA status flag register

This register is holds the status flags of the timer counter of the ENCAn.

Access This register can be read in 8-bit units.
Writing to this register during operation is prohibited.

Address <ENCAn_base> + $0C_H$

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
					ENCAnCSF	ENCAnUDF	ENCAnOVF
R	R	R	R	R	R	R	R

Table 21-12 ENCAnFLG register contents

Bit	Name	Function
2	ENCAnCSF	Counter status flag This bit reflects the current timer counter operation. This bit is cleared at the start of count operation. 0: Timer counter in up-count status 1: Timer counter in down-count status
1	ENCAnUDF	Underflow flag This bit reflects the occurrence of an underflow during the timer counter operation. This bit is cleared at the start of count operation. 0: This flag is cleared upon any of the following events: - “1” is written to ENCAnFGC.ENCAnCLUD - ENCAnTS is set while ENCAnTE = 0 - “1” is written to XX.yy of PIC[a] 1: This flag is set to “1” upon occurrence of an underflow during the encoder timer count operation.
0	ENCAnOVF	Overflow flag This bit reflects the occurrence of an overflow during the timer counter operation. This bit is cleared at the start of count operation. 0: This flag is cleared upon any of the following events: - “1” is written to ENCAnFGC.ENCAnCLOV - ENCAnTS is set while ENCAnTE = 0 - “1” is written to XX.yy of PIC[a] 1: This flag is set to “1” upon occurrence of an overflow during the encoder timer count operation.

a) For details, see the chapter that describes PIC.

(5) ENCAnFGC – ENCA status flag clear register

This register is used to clear the timer counter status flags of ENCAnFLG.

Access This register can be written in 8-bit units.
This register always returns 0 when read.

Address <ENCAn_base> + 10_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	ENCAnCLUD	ENCAnCLOV
W	W	W	W	W	W	W	W

Table 21-13 ENCAnFGC register contents

Bit	Name	Function
1	ENCAnCLUD	Underflow flag clear This bit clears the underflow flag. 0: Writing is ignored. 1: Clears ENCAnUDF of the ENCAnFL register (clears underflow detection).
0	ENCAnCLOV	Overflow flag clear This bit clears the overflow flag. 0: Writing is ignored. 1: Clears ENCAnOVF of the ENCAnFL register (clears overflow detection).

(6) ENCAnCCR0 – ENCA capture compare register 0

This register is a 16-bit capture compare register 0.

Access This register can be written in 16-bit units when used as a compare register (ENCAnCTL.ENCAnCRM0 = 0).
When used as a capture register (ENCAnCTL.ENCAnCRM0 = 1), writing to this register during operation is invalid.

Address <ENCAn_base> + 00_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCCR0[15:0]															
R/W															

Table 21-14 ENCAnCCR0 register contents

Bit	Name	Function
15 to 0	ENCAn CCR0[15:0]	Capture compare register 0 Upon occurrence of an underflow, the setting value of this register may be loaded to the counter according to the ENCAnCTL.ENCAnLDE setting. See the description of the ENCAnLDE bit in ENCA control register ENCAnCTL for details. • if ENCAnCTL.ENCAnCRM0 = 0: ENCAnCCR0 is compare register Set the value to be compared with the timer counter value. • if ENCAnCTL.ENCAnCRM0 = 1: ENCAnCCR0 is capture register The captured timer counter value is stored.

(7) ENCAnCCR1 – ENCA capture compare register 1

This register is a 16-bit capture compare register 1.

Access This register can be written in 16-bit units when used as a compare register (ENCAnCTL.ENCAnCRM1 = 0).
When used as a capture register (ENCAnCTL.ENCAnCRM1 = 1), writing to this register during operation is invalid.

Address <ENCAn_base> + 04_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCCR1[15:0]															
R/W															

Table 21-15 ENCAnCCR1 register contents

Bit	Name	Function
15 to 0	ENCAn CCR1[15:0]	Capture compare register 1 During capture operation, the trigger for capturing to this register differs according to the ENCAnCTL.ENCAnCTS setting. See the description of the ENCAnCTS bit in ENCA control register ENCAnCTL for details. • if ENCAnCTL.ENCAnCRM1 = 0: ENCAnCCR1 is compare register Set the value to be compared with the timer counter value. • if ENCAnCTL.ENCAnCRM1 = 1: ENCAnCCR1 is capture register The captured timer counter value is stored.

(8) ENCAnCNT – ENCA counter register

This register is the 16-bit timer counter register.

Access This register can be read/written in 16-bit units.
This register can be written only when the operation is stopped.

Address <ENCAn_base> + 08_H

Initial value Reset input initializes this register to 0000_H.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENCAnCNT[15:0]															
R/W															

Table 21-16 ENCAnCNT register contents

Bit	Name	Function
15 to 0	ENCAn CNT[15:0]	Counter register • ENCAnTE.ENCAnTE status: 0 (initial setting): Count stop An arbitrary value can be set to timer counter. • ENCAnTE.ENCAnTE status: 0 → 1 (operation start)): Count operation start Up/down count operation is started with the set arbitrary value. • ENCAnTE.ENCAnTE status: 1 (operating): Counting Up/down count operation is performed. • ENCAnTE.ENCAnTE status: 1 → 0 (stopped): Count stop The counter value immediately before the operation was stopped is held, and the count operation is stopped.

(9) ENCAnTE – ENCA timer enable status register

This register indicates the operating status of the ENCAn.

Access This register can be read in 8-bit units.

Address <ENCAn_base> + 14_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTE
R	R	R	R	R	R	R	R

Table 21-17 ENCAnTE register contents

Bit	Name	Function
0	ENCAnTE	Timer status enable bit This is a status bit that indicates the operation enabled/stopped status of the ENCAn. This bit is cleared to “0” when “1” is written to ENCAnTT.ENCAnTT, or when xxxx.yy of PIC[a] is set. This bit is set to “1” when “1” is written to ENCAnTS.ENCAnTS, or when xxxx.zz of PIC[a] is set. 0: Operation stopped status 1: Operation enabled status

a) For details, see the chapter that describes PIC.

(10) ENCAnTS – ENCA timer start trigger register

This register provides the trigger bit for setting the ENCAn to the operation enabled state.

Access This register can be read/written in 8-bit units.
This register always returns 0 when read. This register can be written only when ENCAnTE.ENCAnTE is 0.

Address <ENCAn_base> + 18_H

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTS
R	R	R	R	R	R	R	W

Table 21-18 ENCAnTS register contents

Bit	Name	Function
0	ENCAnTS	Timer start trigger bit This is the trigger bit that sets the ENCAn to the operation enabled state. 0: Writing is ignored. 1: ENCAn is set to the operation enabled state by setting ENCAnTE.ENCAnTE = 1.

(11) ENCAnTT – ENCA timer stop trigger register

This register provides the trigger bit for setting the ENCAn to the operation stopped state.

Access This register can be read/written in 8-bit units.
This register always returns 0 when read.

Address <ENCAn_base> + $1C_H$

Initial value Reset input initializes this register to 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ENCAnTT
R	R	R	R	R	R	R	W

Table 21-19 ENCAnTT register contents

Bit	Name	Function
0	ENCAnTT	Timer stop trigger bit This is the trigger bit that sets the ENCAn to the operation stopped state. 0: Writing is ignored. 1: Clears ENCAnTE.ENCAnTE to “0”, to set the ENCAn to the count operation stopped state.

21.5 ENCA Functional Description

The ENCAn operates the timer counter with counter up/down control and clear control by encoder inputs (phase A, phase B, phase Z). The ENCAnCCR0 and ENCAnCCR1 registers can be used as dedicated compare registers or as dedicated capture registers.

21.5.1 Timer counter operation

The timer counter operations of the ENCAn are described below.

The figure below below shows the operation phases. See the corresponding section with the section number for detailed descriptions on each operation.

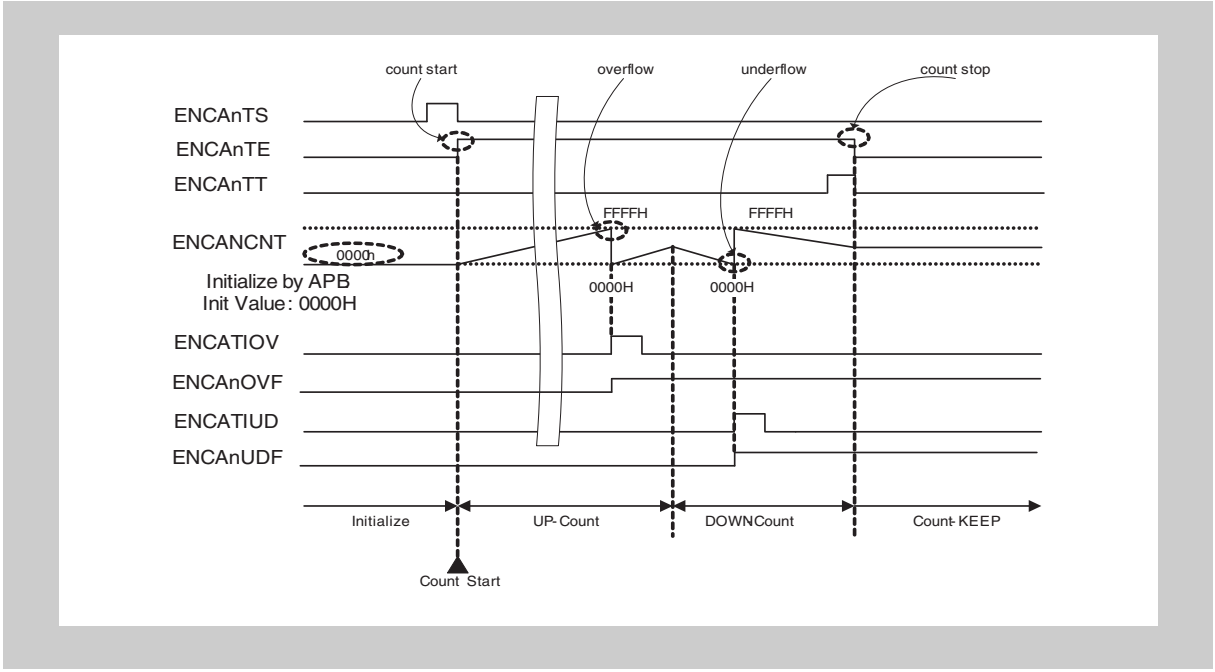


Figure 21-4 Timer counter initial value setting/start/stop

(1) Timer counter initial value setting

The initial value of the ENCAn counter register (ENCAnCNT) can be set in the counter operation stopped status (ENCAnTE = 0).

(2) Timer counter startup

By writing “1” to the timer start trigger bit (ENCAnTS), the timer status enable bit (ENCAnTE) is set to “1”, the count operation is enabled, and counting operation is performed upon detection of the valid edge of the encoder input.

(3) Overflow operation

An overflow occurs when up-counting is performed when the counter value is $FFFF_H$. If the counter value changes from $FFFF_H$ to 0000_H, an overflow interrupt (ENCATIOV) is generated, and the overflow flag (ENCAnOVF) is set to “1”. The overflow flag (ENCAnOVF) is cleared to “0” when “1” is set to the overflow flag clear bit (ENCAnCLOV).

(4) Underflow operation

An underflow occurs when down-counting is performed when the counter value is 0000_H. If the counter value changes from 0000_H to $FFFF_H$, an underflow interrupt (ENCATIUD) is generated, and the underflow flag (ENCAnUDF) is set to “1”. The underflow flag (ENCAnUDF) is cleared to “0” when “1” is set to the underflow flag clear bit (ENCAnCLUD).

(5) Timer counter stop

By writing “1” to the timer stop trigger bit (ENCAnTT), the timer status enable bit (ENCAnTE) is cleared to “0”, and the count operation is stopped. At this time, the timer counter is not reset to 0000_H and holds the value before count operation stop.

21.5.2 Up/down control of timer counter

Up/down control is performed by judging the phase of the encoder inputs (ENCTAIN, ENCTBIN) according to the settings of ENCAnUDS1 and ENCAnUDS0.

(1) When ENCAnUDS1 and ENCAnUDS0 = {0, 0}

ENCAnUDS1	ENCAnUDS0	Operation Description		
		ENCATAIN input	ENCTBIN input	Count operation
0	0	Rising edge	High level	Down
		Falling edge		
		Both edges		
		Rising edge	Low level	Up
		Falling edge		
		Both edges		

The valid edge for ENCATAIN is specified by setting ENCAnEIS1 and ENCAnEIS0.

Count operation is performed when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 0}.

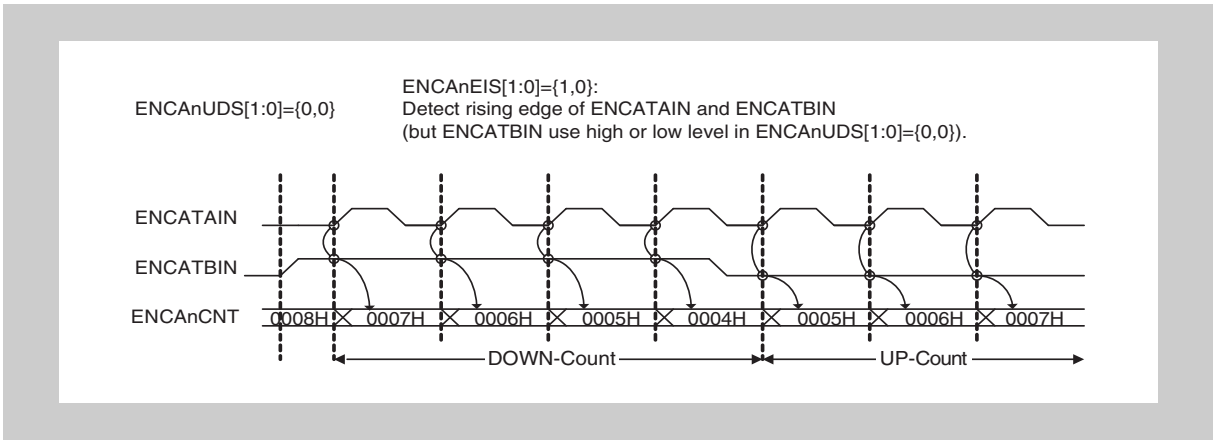


Figure 21-5 Count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 0}

(2) When ENCAnUDS1 and ENCAnUDS0 = {0, 1}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
0	1	Low level	Rising edge	Down
			Falling edge	
			Both edges	
		High level	Rising edge	
			Falling edge	
			Both edges	
		Rising edge	Low level	Up
		Falling edge		
		Both edges		
		Rising edge	High level	
		Falling edges		
		Both edges		
		Simultaneous input		Hold

The valid edges for ENCATAIN and ENCATBIN are specified by setting ENCAnEIS1 and ENCAnEIS0.

Count operation is performed when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {0, 1}.

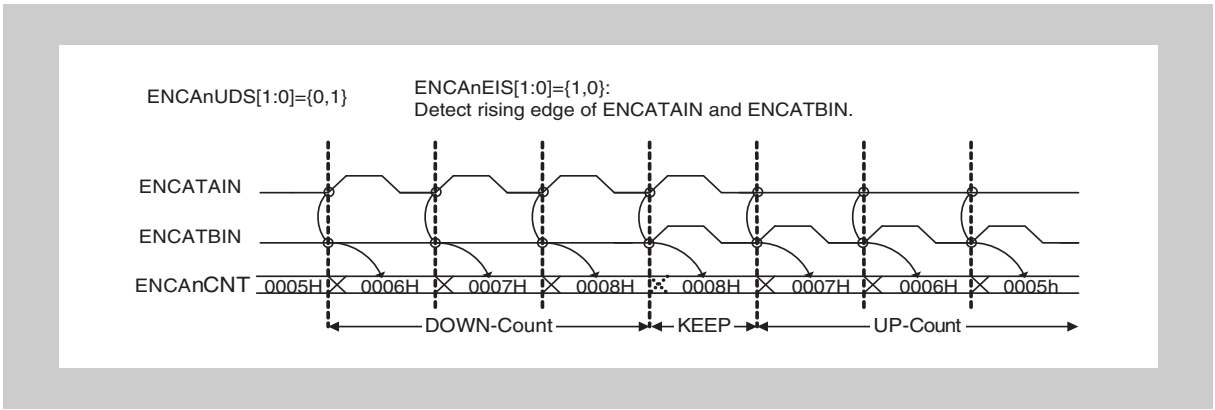


Figure 21-6 Count Operation when ENCAnUDS1 and ENCAnUDS0 = {0, 1}

(3) When ENCAnUDS1 and ENCAnUDS0 = {1, 0}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
1	0	Rising edge	Low level	Down
		Rising edge	Falling edge	
		Falling edge	Low level	Up
		Falling edge	Falling edge	
		Low level	Rising edge	Hold
		Rising edge	Rising edge	
		High level	Rising edge	
		Falling edge	Rising edge	
		Low level	Falling edge	
		Rising edge	High level	
		High level	Falling edge	
		Falling edge	High level	

Valid edge specification for ENCATAIN and ENCATBIN (settings of ENCAnEIS1 and ENCAnEIS0) is invalid.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 0}.

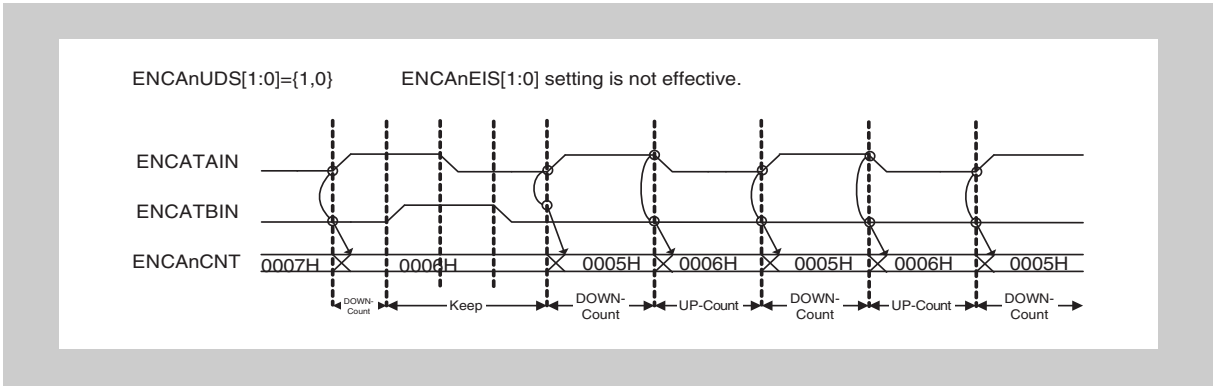


Figure 21-7 Count Operation when ENCAnUDS1 and ENCAnUDS0 = {1, 0}

(4) When ENCAnUDS1 and ENCAnUDS0 = {1, 1}

ENCAnUDS1	ENCAnUDS0	Operation description		
		ENCATAIN input	ENCTBIN input	Count operation
1	1	Low level	Falling edge	Down
		Rising edge	Low level	
		High level	Rising edge	
		Falling edge	High level	
		Rising edge	High level	Up
		High level	Falling edge	
		Falling edge	Low level	
		Low level	Rising edge	
		Simultaneous input		Hold

Valid edge specification for ENCATAIN and ENCATBIN (settings of ENCAnEIS1 and ENCAnEIS0) is invalid.

The count value is held when the valid edges of ENCATAIN and ENCATBIN overlap.

The following timing chart shows the count operation when ENCAnUDS1 and ENCAnUDS0 = {1, 1}.

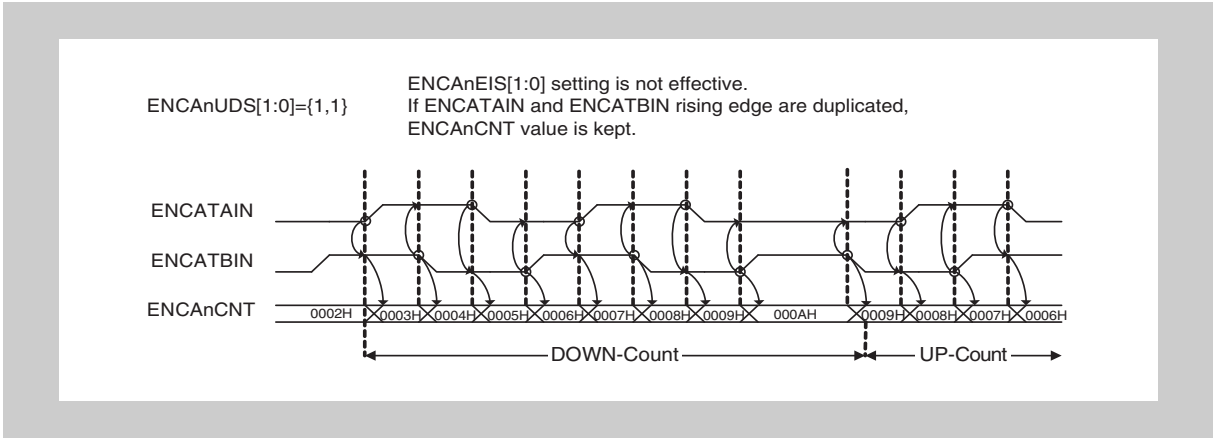


Figure 21-8 Count Operation when ENCAnUDS1 and ENCAnUDS0 = {1, 1}

21.5.3 Timer counter clear control by encoder input

The timer counter is cleared to 0000H by phase Z encoder input (ENCATZIN).

Two types of clearing methods can be selected by controlling the ENCAnSCE, ENCAnZCL, ENCAnBCL, ENCAnACL, ENCAnECS1, and ENCAnECS0 bits of the ENCAnIOC1 register.

Clearing Method	ENCAnSCE	ENCAnZCL	ENCAnBCL	ENCAnACL	ENCAnECS1, ENCAnECS0
(1)	0	Invalid	Invalid	Invalid	Valid
(2)	1	Valid	Valid	Valid	Invalid

(1) Clearing method when ENCAnSCE = 0

- Upon detection of the valid edge of ENCATZIN, the timer counter is cleared to 0000_H in synchronization with the operation clock.
- The valid edge of ENCATZIN is specified by the setting of the ENCAnECS1 and ENCAnECS0 bits.
- The settings of the ENCAnZCL, ENCAnBCL, and ENCAnACL bits are invalid.
- A clear interrupt signal (ENCATIEC) is output simultaneously with timer counter clearing.

(2) Clearing method when ENCAnSCE = 1

- The clear levels of the ENCATZIN, ENCATBIN, ENCATAIN inputs are detected according to the settings of the ENCAnZCL, ENCAnBCL, and ENCAnACL bits, and the timer counter is cleared to 0000_H in synchronization with the operating clock.
- The settings of the ENCAnECS1 and ENCAnECS0 bits are invalid.
- An encoder clear interrupt signal (ENCATIEC) is output simultaneously with timer counter clearing.

The clearing conditions of the timer counter according to the ENCAnZCL, ENCAnBCL, and ENCAnACL settings are listed in the table below.

Counter Clear Condition Setting			Encoder Input Level		
ENCAnZCL	ENCAnBCL	ENCAnACL	ENCATZIN	ENCATBIN	ENCATAIN
0	0	0	Low	Low	Low
0	0	1	Low	Low	High
0	1	0	Low	High	Low
0	1	1	Low	High	High
1	0	0	High	Low	Low
1	0	1	High	Low	High
1	1	0	High	High	Low
1	1	1	High	High	High

21.5.4 Functions of ENCAnCCR0

(1) Compare function

- When ENCAnCRM0 = 0, the ENCAnCCR0 register functions as a dedicated compare register.
- Upon compare match between the value of the timer counter and the ENCAnCCR0 setting value, a compare 0 match interrupt (ENCATINT0) is output.
- When ENCAnECM0 = 1, the timer counter is cleared to 0000_H in synchronization with the operating clock upon compare match if the next count operation is up-count.

ENCAnCCR0 function	Compare match clear control	Next count operation	Timer counter clearing upon compare match with ENCAnCCR0
ENCAnCRM0	ENCAnECM0		
0 (Compare)	0	Up-count	Does not clear (continues count operation).
		Down-count	
	1	Up-count	Clears timer counter to 0000_H.
		Down-count	Does not clear (continues count operation).

When ENCAnLD = 1

- Upon occurrence of an underflow, the setting value of the ENCAnCCR0 register is loaded to the timer counter.
- An underflow interrupt (ENCATIUD) is output.

(2) Capture function

- When ENCAnCRM0 = 1, the ENCAnCCR0 register functions as a dedicated capture register.
- Upon valid edge detection of the capture trigger input 0 (ENCATTIN0), the value of the timer counter is stored into ENCAnCCR0.
- A capture 0 interrupt (ENCATINT0) is output during capture operation.

21.5.5 Functions of ENCAnCCR1

(1) Compare function

- When ENCAnCRM1 = 0, the ENCAnCCR1 register functions as a dedicated compare register.
- Upon compare match between the value of the timer counter and the ENCAnCCR1 setting value, a compare 1 match interrupt (ENCATINT1) is output.
- When ENCAnECM1 = 1, the timer counter is cleared to 0000_H in synchronization with the operating clock upon compare match if the next count operation is down-count.

ENCAnCCR1 function	Compare match clear control	Next count operation	Timer counter clearing upon compare match with ENCAnCCR1
ENCAnCRM1	ENCAnECM1		
0 (Compare)	0	Up-count	Does not clear (continues count operation).
		Down-count	
	1	Up-count	Does not clear (continues count operation).
		Down-count	Clears timer counter to 0000_H.

Compare match interrupt detection mask function

- When ENCAnCME = 1, the compare 1 match interrupt detection mask function is enabled. In this state, the compare 1 match interrupt is output upon the first match of the value of the timer counter and the ENCAnCCR1 setting value, and interrupts are then masked for the second and subsequent compare matches.
- When ENCAnMCS = 0, the compare 1 match interrupt detection mask function is disabled by a write operation to the ENCAnCCR1 register.
- When ENCAnMCS = 1, the compare 1 match interrupt detection mask function is disabled by a timer counter clear operation accompanying phase Z or by a timer counter clear operation upon match between the ENCAnCCR0 register value and the timer counter value.
- When ENCAnMCS = 1 and ENCAnLDE = 1, the compare 1 match interrupt detection mask function is disabled by a loading operation of the ENCAnCCR0 register to the timer counter upon underflow detection.
- Setting ENCAnECM to “1” is prohibited when enabling the compare 1 match interrupt detection mask function.

ENCAnCCR1 function	Compare 1 match interrupt mask	Interrupt mask cancel trigger	Underflow occurrence upon ENCAnLDE = 0	Compare 1 match interrupt output upon compare match with ENCAnCCR1
ENCAnCRM1	ENCAnCME	ENCAnMCS		
0 (Compare)	0 (Mask function disabled)	– (Setting invalid)	-	Outputs compare 1 match interrupt upon each compare match.
	1 (Mask function enabled)	0 (Write operation to ENCAnCCR1)	Occurred (Loading of ENCAnCCR0 to timer counter)	Outputs compare 1 match interrupt once upon the first compare match. (Interrupts are masked for the second and subsequent matches until the cancel trigger occurs.)
		1 (Timer counter clear operation)		

(2) Capture function

- When ENCAnCRM1 = 1, the ENCAnCCR1 register functions as a dedicated capture register.

The operations for each of the ENCAnCTS settings are shown in the table below.

ENCAnCCR1 function	Capture trigger selection	Capture trigger signal	Timer counter clearing	Interrupt occurrence
ENCAnCRM1	ENCAnCTS			
1 (Capture)	0	Capture trigger 1 input (ENCATTIN1)	Does not clear timer counter.	(1) Capture 1 interrupt (ENCATINT1)
	1	Encoder clear input accompanying phase Z (ENCATZIN)	Clears timer counter.	(1) Capture 1 interrupt (ENCATINT1) (2) Encoder clear interrupt (ENCATIEC)

(3) Timer counter clearing upon compare register match

Timer counter clearing upon compare match between the value of the timer counter and the ENCAnCCR0/1 setting value, according to the settings of the ENCAnECM1 and ENCAnECM0 bits, is detailed in the following table.

ENCAnECM1 and ENCAnECM0	Next count operation	Timer counter clearing upon compare match with ENCAnCCR1	Timer counter clearing upon compare match with ENCAnCCR0
00	Up-count	Does not clear (continues count operation).	Does not clear (continues count operation).
	Down-count	Does not clear (continues count operation).	Does not clear (continues count operation).
01	Up-count	Does not clear (continues count operation).	Clears timer counter to 0000_H.
	Down-count	Does not clear (continues count operation).	Does not clear (continues count operation).
10	Up-count	Does not clear (continues count operation).	Does not clear (continues count operation).
	Down-count	Clears timer counter to 0000_H.	Does not clear (continues count operation).
11	Up-count	Does not clear (continues count operation).	Clears timer counter to 0000_H.
	Down-count	Clears timer counter to 0000_H.	Does not clear (continues count operation).

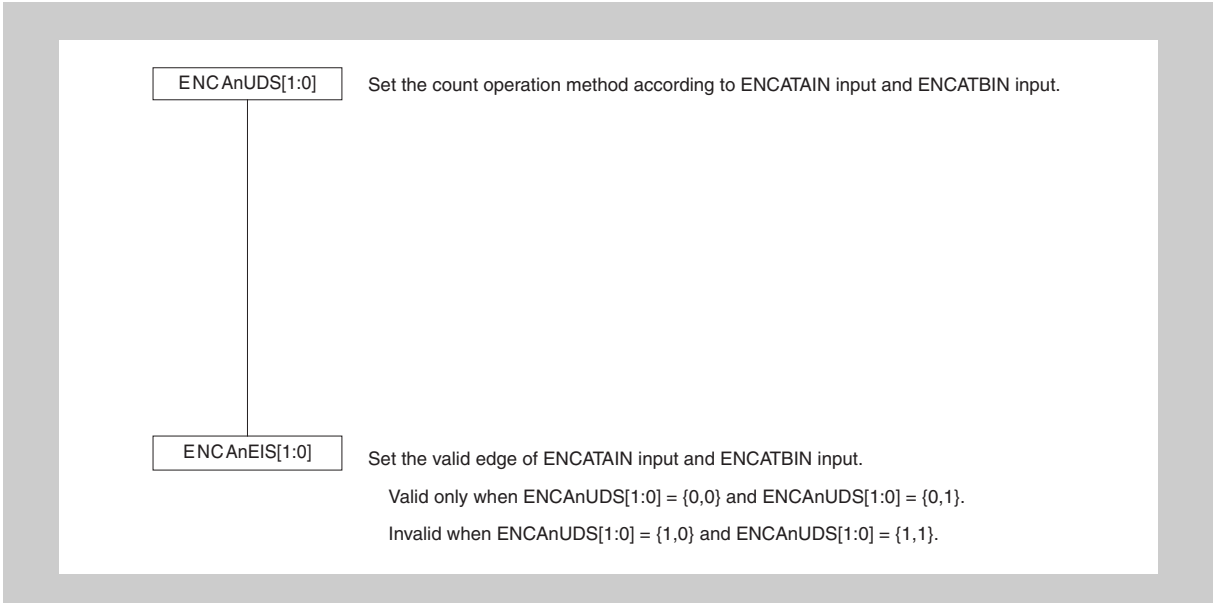
21.6 ENCA Setting Sequences

21.6.1 Encoder timer setting procedure

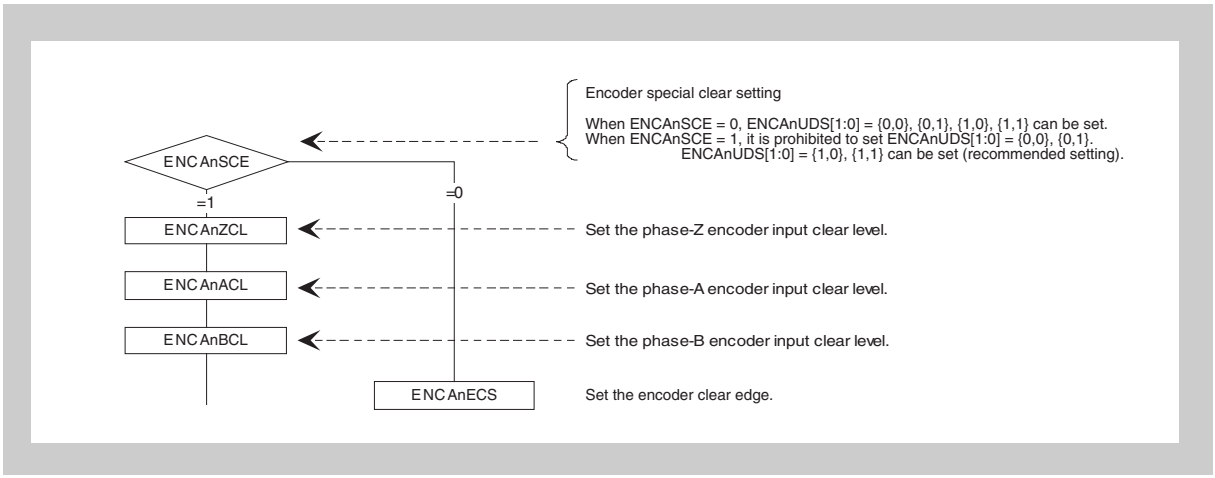
The encoder timer setting procedure is described below.

	Action	Setting status
Initial Setting		Power-off status (Writing to each register is disabled)
	Reset release	Power-on status, ENCAn operation stopped status. (Writing to each register is enabled)
ENCA Initial Settings	Perform the following initial settings. • Setting for counter • Setting for counter clear • Setting for ENCAnCCR0 register • Setting for ENCAnCCR1 register	This is the count operation stopped status. The value of the ENCAnTE bit indicating the operating status is 0.
	Perform the counter initial value settings. • Set any 16-bit value to ENCAnCNT register. (When, after setting this register, the ENCAnTS bit is set to “1”, the counter operation starts from the set count value.)	The set value is set as the initial value of the counter register.
Operation Start	Perform the counter operation start setting. • Set the ENCAnTS bit to “1”.	This is the counter operation starts status. The value of the ENCAnTE bit indicating the operating status is 1, and the count clock is supplied to the internal circuit.
Operating	Only those registers whose setting can be changed during operation can be rewritten. • ENCAnCCR0 register setting can be changed. • ENCAnCCR1 register setting can be changed. • ENCAnIOC0 register setting can be changed.	The count operation set with the initial setting is performed, and up/down counting is performed according to ENCATAIN and ENCATBIN.
Operation Stop	Perform the counter operation stop setting during operation. • Set the ENCAnTT bit to “1”.	This is the counter operation stopped status. The value of the ENCAnTE bit indicating the operating status is 0.
ENCA stop	Reset	This is the power-off status. The entire circuit and all the setting registers are initialized.

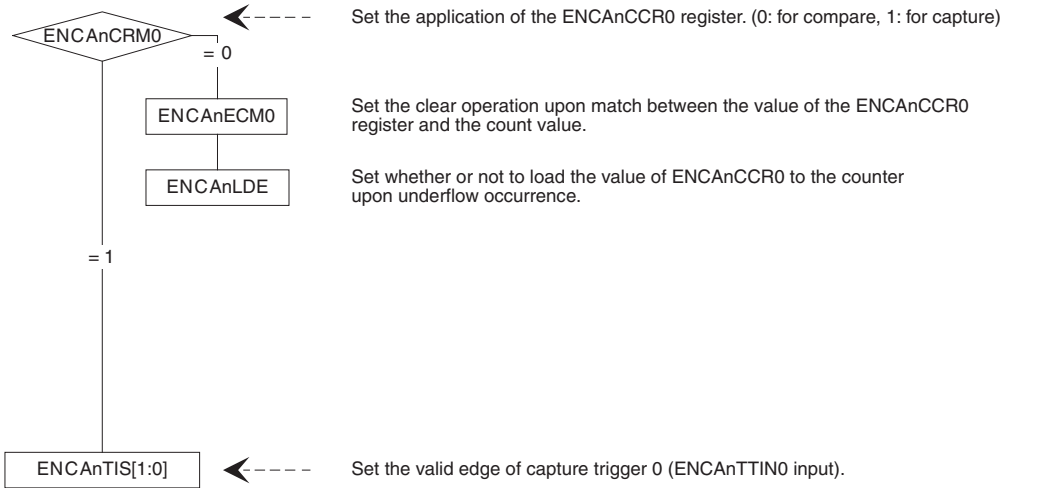
(1) Initial setting procedure for counter



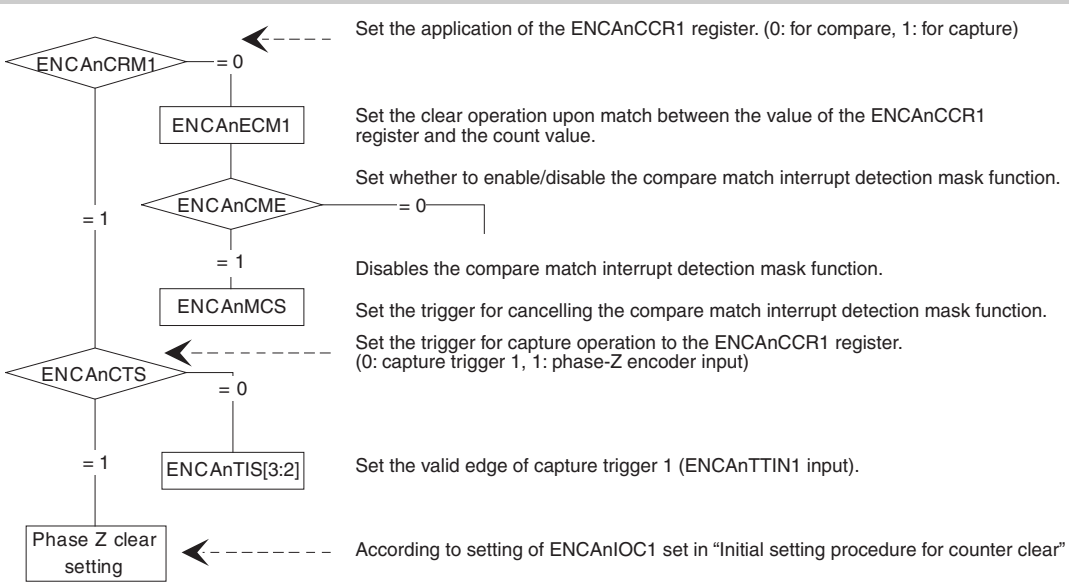
(2) Initial setting procedure for counter clear



(3) Setting procedure for ENCAnCCR0 register



(4) Setting procedure for ENCAnCCR1 register



21.7 A/D Trigger Encoder Capture

21.7.1 Functional overview

This feature makes it possible to obtain an encoder count value synchronized with the A/D Converter by using the A/D Converter trigger signal as the ENCA capture signal.

Peripheral interconnection

A peripheral interconnection circuit provides facilities for connecting the A/D Converter hardware trigger signals to the ENCA0 capture trigger input.

21.7.2 Configuration

The unit configuration for this feature is shown below.

Table 21-20 A/D trigger encoder capture feature configuration

A/D Converter	Encoder timer
ADCA0, ADCA1 (Select the count target from the A/D Converter trigger signals.)	ENCA0

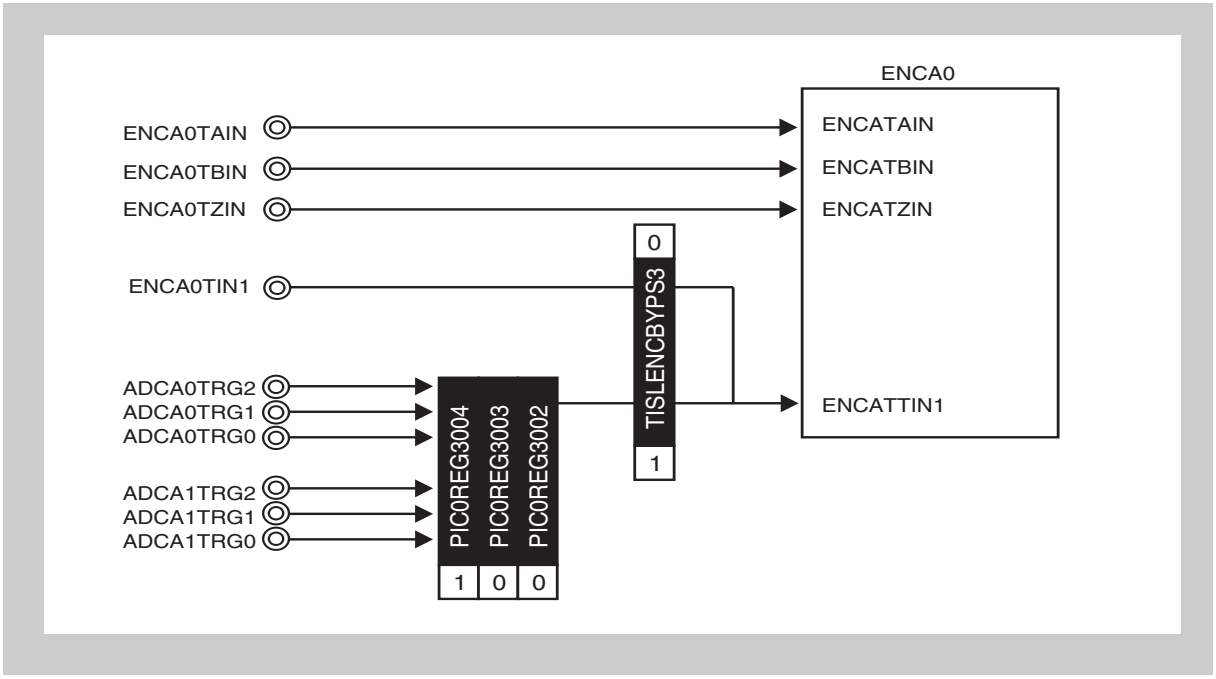


Figure 21-9 Block diagram

21.7.3 Operation example

The above feature is achieved by connecting ENCA0 to the ADCA0 or ADCA1 hardware trigger input ADCAnTRGi (n = 0 or 1, i = 0 to 2).

Caution If using this feature, do not select the ENCA0 interrupt signal INTENCA0I1 as an A/D Converter trigger. If this signal is selected, correct operation becomes impossible because the following loop is established: ADCAnTRG1 generation ENCA0 capture operation INTENCA0I1 generation due to capture execution ADCAnTRG1 generation.

The timing chart for the A/D trigger encoder capture feature with ADCA0TRG1 as the trigger is shown below.

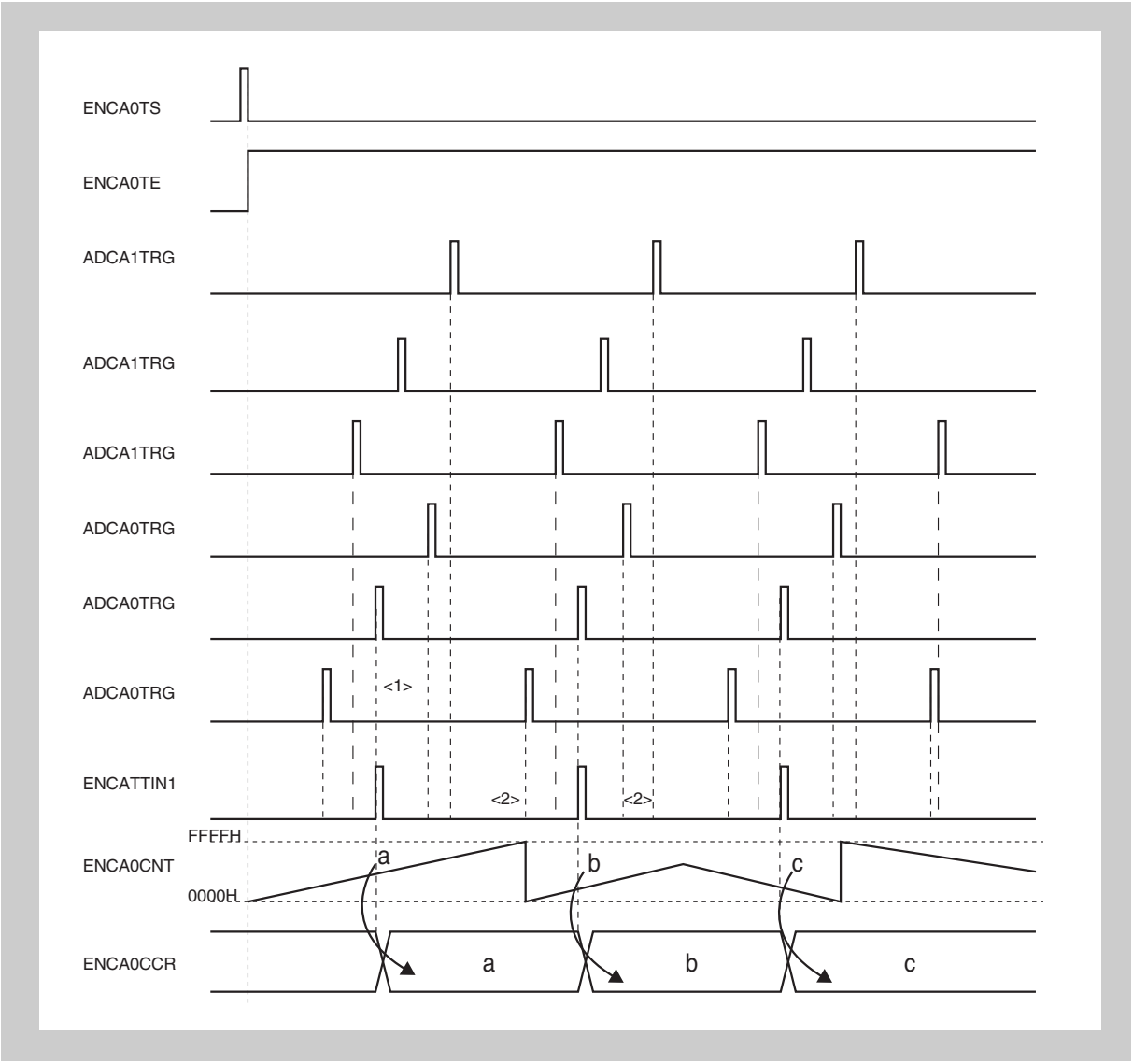


Figure 21-10 Operation of the A/D trigger encoder capture feature

- If ADCA0TRG1 is selected as ENCATTIN1, the capture trigger 1 signal for ENCA0, because the valid ADCA0TRG1 is input to ENCA0 as the ENCATTIN1 signal, ENCA0 capturing occurs (point <1> in the figure).

- For the generation of a hardware trigger signal other than ADCA0TRG1 (ADCA0TRG0, ADCA0TRG2, or ADCA1TRG0 to ADCA1TRG2), no ENCA0 capture operation is performed because the ENCATTIN1 signal is not generated (point <2> in the figure).

21.7.4 Setup flow

The setup flow in this section shows the execution of encoder timer ENCA0 capture operations by using the ADCA0TRG1 signal.

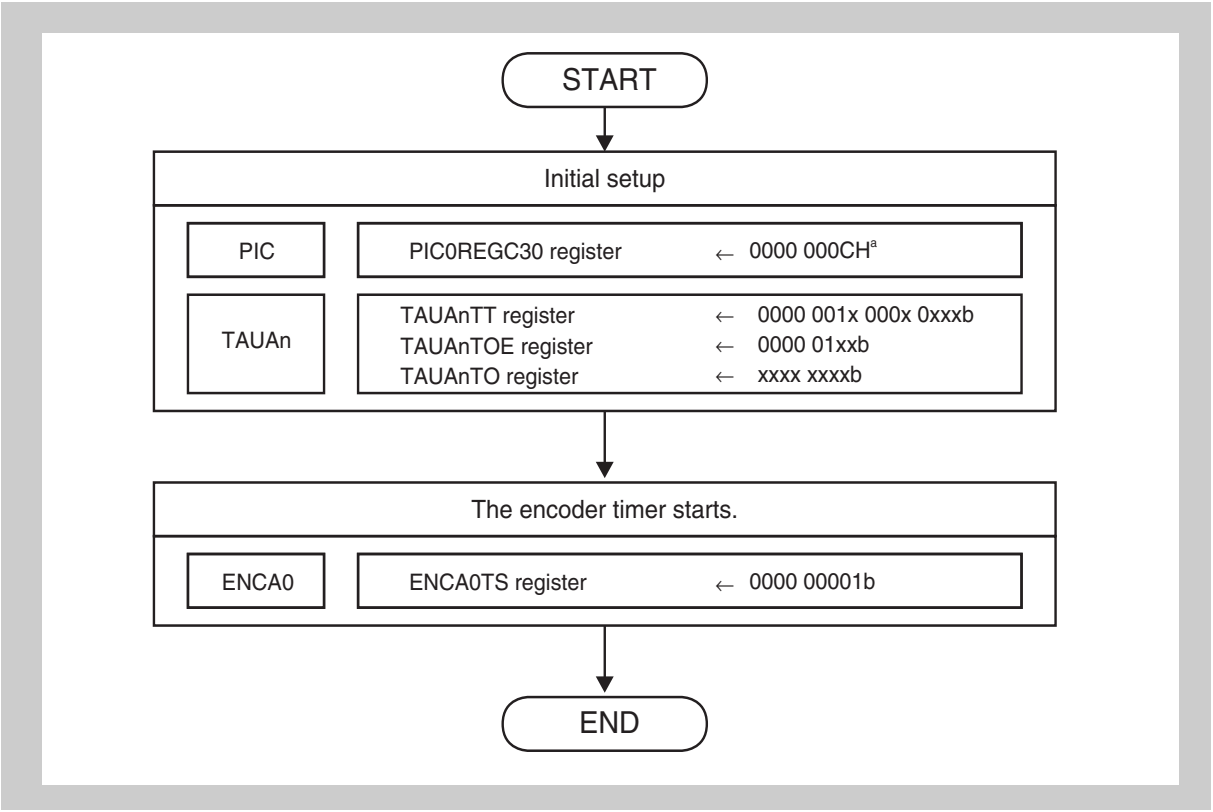


Figure 21-11 Setup flow

a) Change the settings as appropriate for the hardware trigger used to execute the capture operations.

21.7.5 Example of setting up operation functions

This section provides example settings for each register.

This section shows the settings for executing encoder timer ENCA0 capture operations by using the ADCA0TRG1 signal.

Table 21-21 ENCAn settings

Register	Bit position	Bit name	Setting	Remark
ENCAnCTL	15	ENCAnCME	0	(valid only when ENCAnCRM1 = 0)
	14	ENCAnMCS	0	(valid only when ENCAnCRM1 = 0)
	13 to 10		0	Fixed to 0
	9	ENCAnCRM1	1	Set the purpose of the ENCAnCCR1 register to capturing.
	8	ENCAnCRM0	Any	Select the purpose of the ENCAnCCR0 register.
	7	ENCAnCTS	0	Select ENCATTIN1 as the capture trigger.
	6, 5		0	Fixed to 0
	4	ENCAnLDE	Any	Select whether to enable or disable reloading when the ENCAnCCR0 register underflows.
	3	ENCAnECM1	0	(valid only when ENCAnCRM1 = 0)
	2	ENCAnECM0	Any	Select whether to enable or disable ENCAnCCR0 register compare match counter clearing.
	1, 0	ENCAnUDS[1:0]	Any	Select whether to increment or decrement the counter according to ENCATAIN and ENCATBIN.
ENCAnIOC0	7 to 4		0	Fixed to 0
	3, 2	ENCAnTIS[3:2]	0[a] 1[a]	Select the rising edge as the valid edge for capture trigger 1 (ENCATTIN1).
	1, 0	ENCAnTIS[1:0]	Any	Select the valid edge of capture trigger 0 (ENCATTIN0).
ENCAnIOC1	7	ENCAnSCE	Any	Select whether to enable special encoder clearing.
	6	ENCAnZCL	Any	Select the Z-phase clear condition (input level) for the encoder special clearing.
	5	ENCAnBCL	Any	Select the B-phase clear condition (input level) for the encoder special clearing.
	4	ENCAnACL	Any	Select the A-phase clear condition (input level) for the encoder special clearing.
	3, 2	ENCAnECS[1:0]	Any	Select the encoder clearing input (Z phase) edge.
	1, 0	ENCAnEIS[1:0]	Any	Select the encoder input (A and B phases) edge.

a) Change the settings as appropriate for the hardware trigger used to execute capture operations.

Note The only fixed ENCA0CTL values are ENCA0CRM1 = 1 (ENCA0CCR1 register purpose) and ENCA0CTS = 0 (capture trigger source for the ENCA0CCR1 register). Other settings can be freely specified.

Table 21-22 Peripheral interconnection settings

Register	Bit position	Bit name	Setting	Remark
PIC0REG30	4, 3, 2	PIC0REG3004 PIC0REG3003 PIC0REG3002	0[a] 1[a] 1[a]	Select ADCA0TRG1 as the ENCATTIN1 input signal for ENCA0.

a) Change the settings as appropriate for the encoder timer used to execute capture operations.

21.7.6 Registers

(1) PIC0REG30 - Timer I/O control register 30

Access This register can be read or written in 32-bit units.

Address FF81 C0BC$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	PIC0REG 3004	PIC0REG 3003	PIC0REG 3002	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-23 PIC0REG30 register contents

Bit position	Bit name	Function
4 3 2	PIC0REG3004 PIC0REG3003 PIC0REG3002	Select the signal input to ENCATTIN1 for ENCA0.

PIC0REG 3004	PIC0REG 3003	PIC0REG 3002	Input signal
0	0	0	If ENCA0 does not perform capture operations based on the A/D trigger signal
0	1	0	ADCA1TRG2
0	1	1	ADCA1TRG1
1	0	0	ADCA1TRG0
1	0	1	ADCA0TRG2
1	1	0	ADCA0TRG1
1	1	1	ADCA0TRG0
Other than the above			Setting prohibited

Note Some of the bits defined as 0 in the PIC0REG30 register are defined for the other timers. For such bits, use the bit definition of those timers.

(2) TISLENCBYPS - ENCA input selection register

The TISLENCBYPS register selects the ENCA input signal.

Access This register can be read or written in 8-bit units.

Address FF77 1018_H

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	TISLENCBYPS3	0	0	0
R	R	R	R	R/W	R/W	R/W	R/W

Table 21-24 TISLENCBYPS register contents

Bit position	Bit name	Function
3	TISLENCBYPS3	Select the signal input to ENCATTIN1 for ENCA0. 1: Select the signal selected by PIC0REG3004, PIC0REG3003, and PIC0REG3002. 0: Select the signal ENCA0TIN1.

21.8 Synchronized Timer Operation

21.8.1 Functional Overview

Any combination of timers (TAUA and ENCA) can be started at the same time.

Peripheral interconnection

A peripheral interconnection circuit provides registers to select the timer channels to be started simultaneously.

21.8.2 Configuration

The unit and channel configuration for this feature are shown below.

TAUA0 (all channels), ENCA0

(Select the timers and channels to start at the same time from the above.)

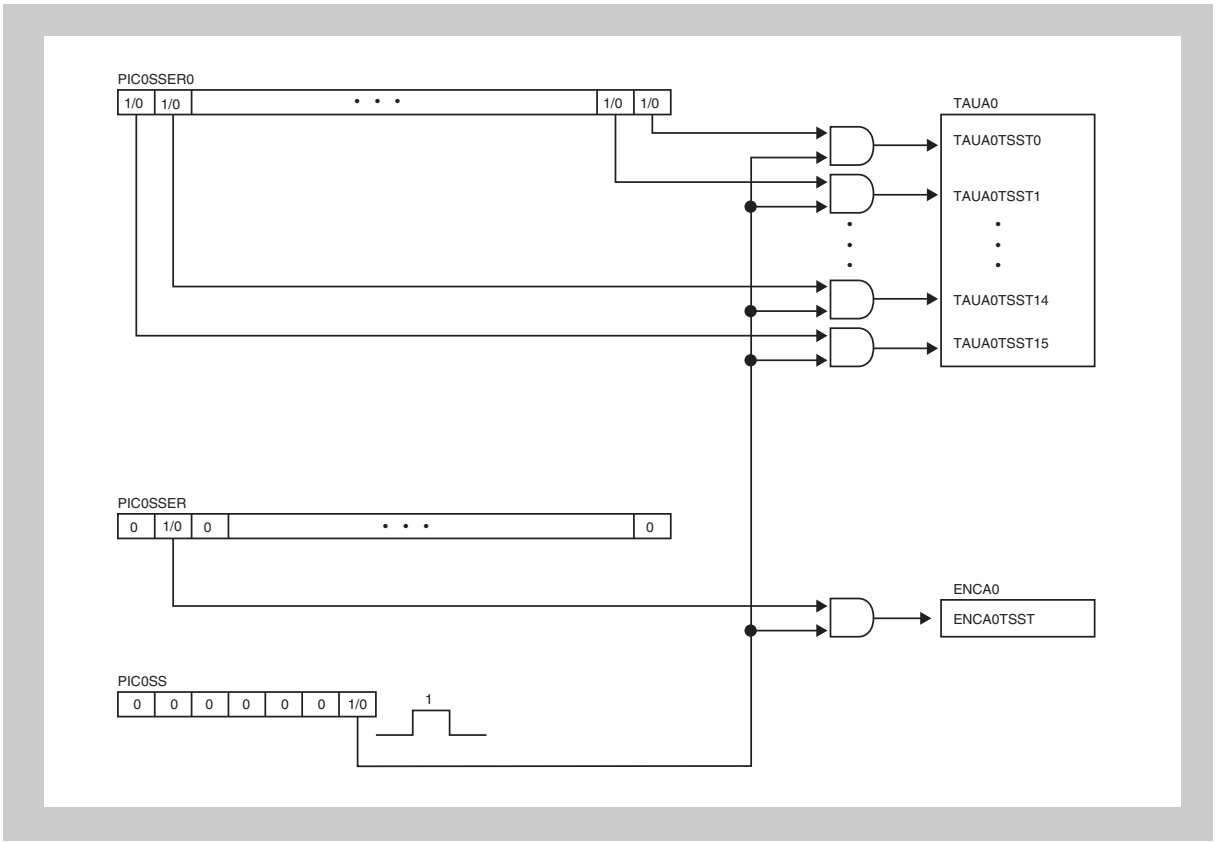


Figure 21-12 Block diagram

21.8.3 Operation example

(1) Timer configuration operation example

The following describes how to start up any combination of timers (TAUA0 and ENCA0) at the same time in any operation mode.

(2) Setup procedure

<1> Initial setup Specify the operation mode for each timer (including register settings). For details about the initial setup of TAUA0 and ENCA0, see the chapter on each timer.

<2> Enabling simultaneous startup Enable the PIC0SSER0 and PIC0SSER2 register bits corresponding to the timers to be started simultaneously.

<3> Outputting the start trigger Write 1 to the SYNCTRG bit of the PIC0SST register to output a start trigger to the timers specified in <2> (a trigger for the target timers among TAUA0TSSTm and ENCATSST) and simultaneously start the timers.

By repeating steps <2> and <3> for channels that have not been started, different groups of target timers can be simultaneously started at different times.

Note For details about the operation when simultaneously starting timers (outputting start triggers) while timers are operating, see the start trigger information in the chapter about each timer.

21.8.4 Setup flow

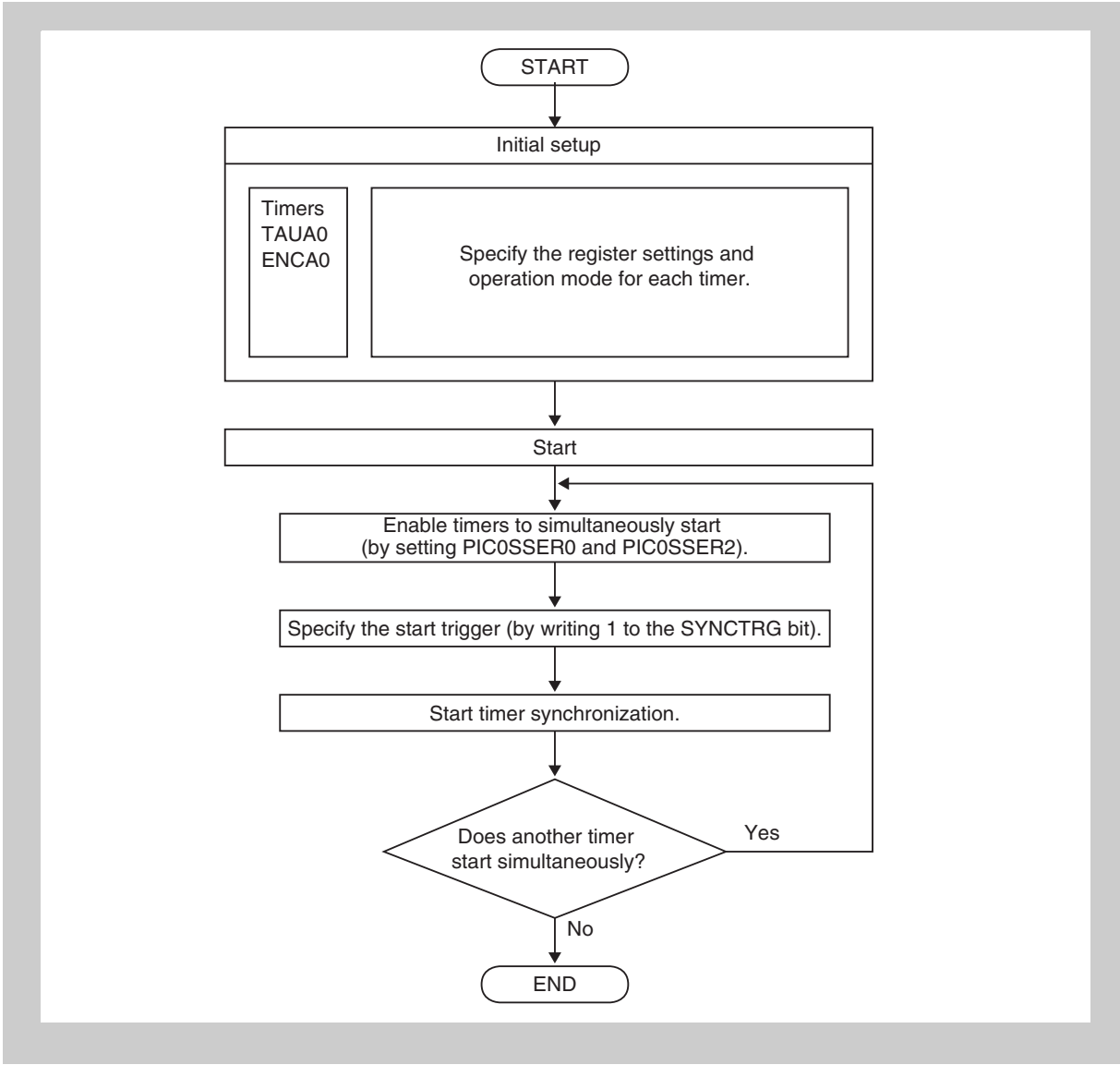


Figure 21-13 Setup flow

Note For details about the operation when simultaneously starting timers (outputting start triggers) while timers are operating, see the start trigger information in the chapter about each timer.

21.8.5 Setting up operation functions

See *21.8.6 "Registers"*.

21.8.6 Registers

(1) PIC0SSER0 - Simultaneous start control register 0

The PIC0SSER0 register enables the start trigger for each channel of TAUA0.

Access This register can be read or written in 16-bit units.

Address FF81 C080$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
PIC0SSER015	PIC0SSER014	PIC0SSER013	PIC0SSER012	PIC0SSER011	PIC0SSER010	PIC0SSER009	PIC0SSER008
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PIC0SSER007	PIC0SSER006	PIC0SSER005	PIC0SSER004	PIC0SSER003	PIC0SSER002	PIC0SSER001	PIC0SSER000
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-25 PIC0SSER0 register contents

Bit position	Bit name	Function
m	PIC0SSER0m	This bit is used to enable or disable simultaneous start triggers for CHm of TAUA0. 0: Disable simultaneous start triggers for CHm of TAUA0. 1: Enable simultaneous start triggers for CHm of TAUA0.

(2) PIC0SSER2 - Simultaneous start control register 2

The PIC0SSER2 register enables the start trigger for each channel of ENCA0.

Access This register can be read or written in 16-bit units.

Address FF81 C088$_H$

Initial value 0000$_H$

15	14	13	12	11	10	9	8
0	PIC0SSER214	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-26 PIC0SSER2 register contents

Bit position	Bit name	Function
14	PIC0SSER214	This bit is used to enable or disable simultaneous start triggers for ENCA0. 0: Disable simultaneous start triggers for ENCA0. 1: Enable simultaneous start triggers for ENCA0.

(3) PIC0SST - Simultaneous start trigger control register

Access This register can be written in 8-bit units.

Address FF81 C004$_H$

Initial value Reading this register returns always 00$_H$.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	SYNCTRG
W	W	W	W	W	W	W	W

Table 21-27 PIC0SST register contents

Bit position	Bit name	Function
0	SYNCTRG	This bit is used to generate start triggers for timers for which simultaneous startup is enabled. 0: Disable simultaneous start triggers. 1: Enable simultaneous start triggers (one PCLK width pulse output).

21.9 Trigger Pulse Width Measurement

21.9.1 Functional overview

This feature uses TAUJn and TAUAn to measure the interval (such as the motor rotation speed) of the trigger output from the encoder.
The pulse width of the following three types of trigger signals can be measured:

- ENCATIEC: ENCA0 clear interrupt signal that uses encoder input (Z phase)
- ENCATEQ0: Match detection signal generated when ENCA0CNT and ENCA0CCR0 match
- ENCATEQ1: Match detection signal generated when ENCA0CNT and ENCA0CCR1 match

Peripheral interconnection

A peripheral interconnection circuit provides facilities for connecting the the Encoder Timer signals to the TAUA0 and TAUJ0 capture trigger input.

21.9.2 Configuration

The unit and channel configuration for this feature are shown below.

Table 21-28 Trigger pulse width measurement configuration

Timer (pulse output side)		Timer (measurement side) (selected according to the measurement-target trigger)
ENCA0	ENCATIEC measurement	TAUJ0 CH0 or CH1
	ENCATEQ0 measurement	TAUA0 CH0 or CH2
	ENCATEQ1 measurement	TAUA0 CH0 or CH1

Note The signal names used in the descriptions below are abbreviations. The actual signal names corresponding to each abbreviation are as follows:

- INTm → INTTAUAnIm or INTTAUJ0Im
  (TAUAn or TAUJn channel m interrupt)
- TINm → TAUAnTTINm or TAUJ0TTINm
  (TAUAn or TAUJn channel m input)
- TOUTm → TAUAnTTOUTm or TAUJ0TTOUTm
  (TAUAn or TAUJn channel m output)
- CDRm → TAUAnCDRm or TAUJ0CDRm
  (TAUAn ot TAUJn channel m data register)
- CNTm → TAUAnCNTm or TAUJ0CNTm
  (TAUAn ot TAUJn channel m counter register)

Table 21-29 TAUA0/TAUJ0: Each channel function setting

TAU	CH	Function name	M/S[a]	Pulse width measurement target trigger of ENCA0
TAUA0	0	TAUAnTTINm input pulse interval measurement	-	ENCATEQ0 or ENCATEQ1
	1	TAUAnTTINm input pulse interval measurement	-	ENCATEQ1
	2	TAUAnTTINm input pulse interval measurement	-	ENCATEQ0
TAUJ0	0	TAUJnTTINm input pulse interval measurement	-	ENCATIEC
	1	TAUJnTTINm input pulse interval measurement	-	ENCATIEC

a) M: Master channel, S: Slave channel, -: Standalone

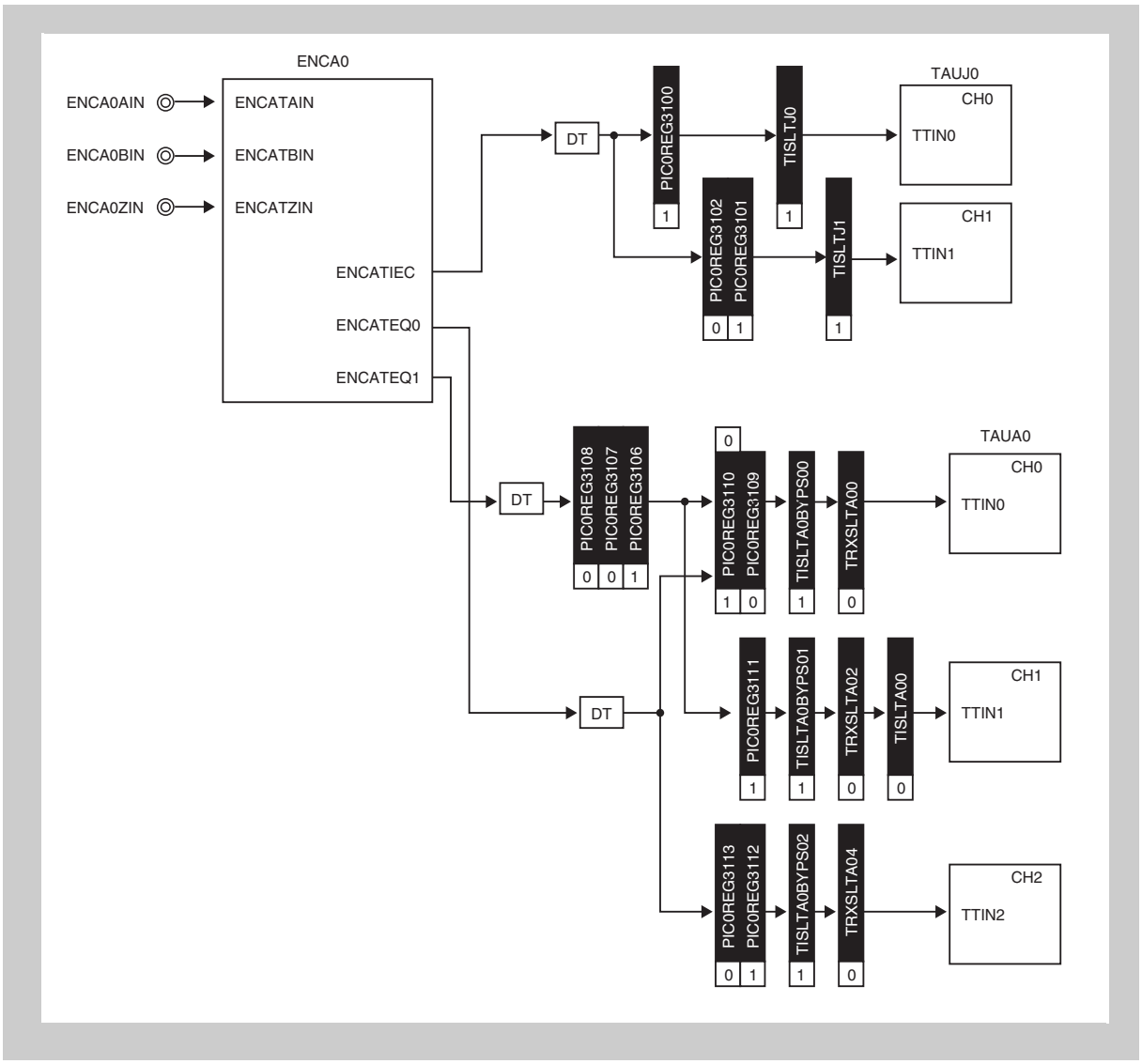


Figure 21-14 Block diagram

21.9.3 Operation example

The feature above is achieved using a combination of the ENCA0 trigger signals (ENCATIEC, ENCATEQ0, and ENCATEQ1) and the following features of TAUA0 and TAUJ0:

- TAUAnTTINm input pulse interval measurement (TAUA0)
- TAUJnTTINm input pulse interval measurement (TAUJ0)

The following peripheral interconnection feature is also used to convert the trigger signal input to TINm to a level-changing toggle signal:

- DT circuit

The trigger pulse width measurement feature makes it possible to measure the interval of the trigger signal output from ENCA0 by using the TAUAnTTINm input pulse interval measurement feature of TAUA0 and the TAUJnTTINm input pulse interval measurement feature of TAUJ0.

(1) TAUAnTTINm and TAUJnTTINm input pulse interval measurement

When the valid edge of the TINm signal of TAUA0 or TAUJ0 is detected, the CNTm value is captured by CDRm, and CNTm is cleared.

Caution For this feature, specify both rising and falling edges (high-level width measurement) (TAUA0CMURm.TAUA0TIS[1:0] = 11_B, TAUJ0CMURm.TAUJ0TIS[1:0] = 11_B) as the valid edges detected for TINm.

For details about the TAUA and TAUJ features, see the corresponding chapters.

(2) DT circuit

The trigger signal output by ENCA0 is converted to a level-changing toggle signal.

As shown in *Figure 21-15 "DT circuit operation"*, the output signal is toggled each time an input trigger signal is generated.

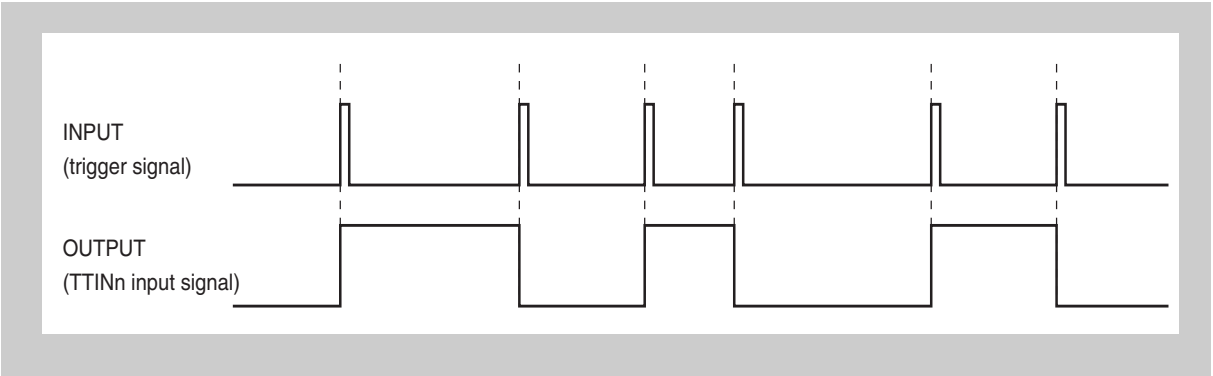


Figure 21-15 DT circuit operation

To measure the generation interval of trigger signals from ENCA0, the peripheral interconnection provides features for converting signals input to TAUA0 and TAUJ0 and connecting signals.

The following figure shows a timing chart for trigger pulse width measurement.

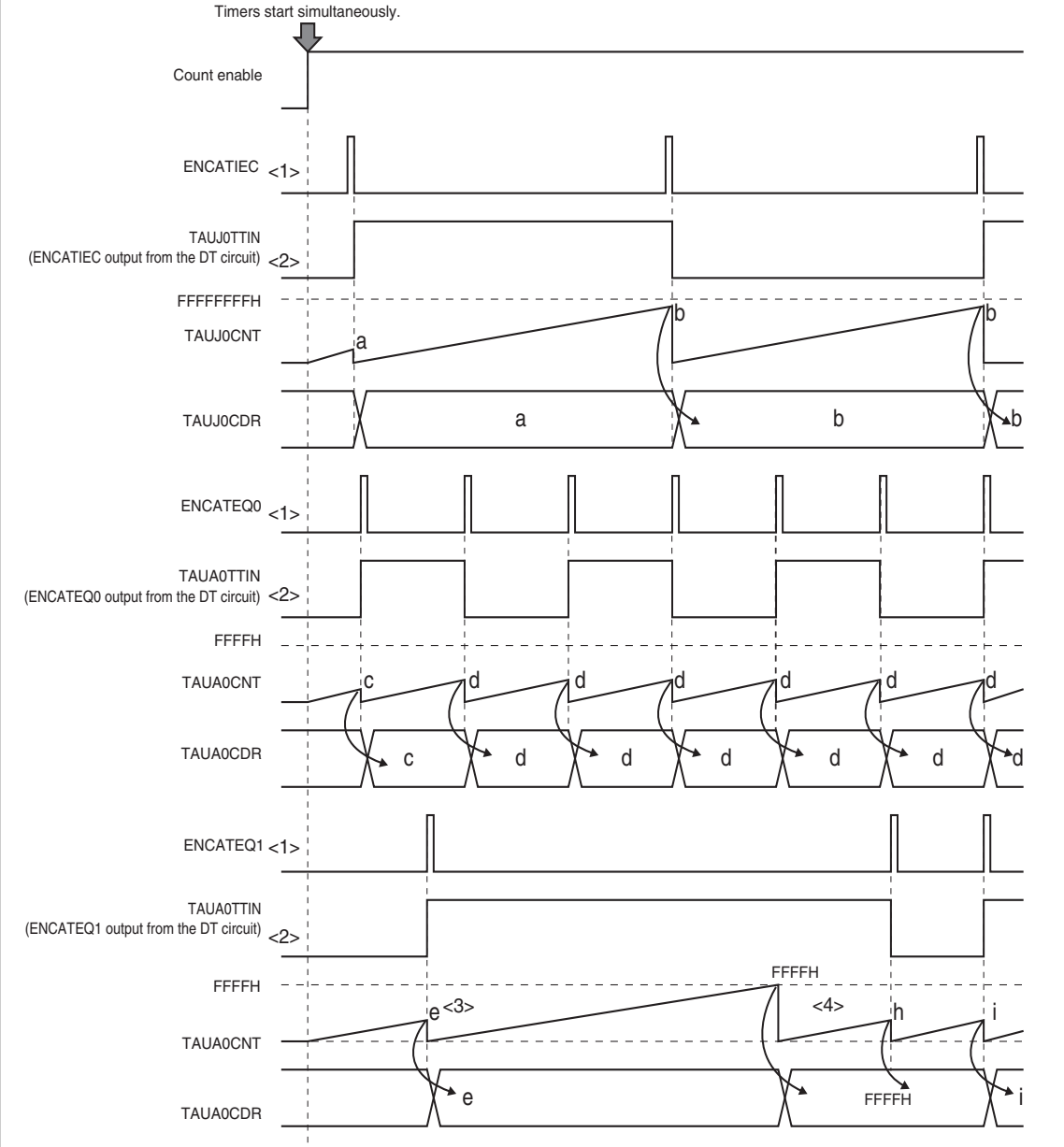


Figure 21-16 Trigger pulse width measurement operation (m = 0 to 2, k = 0 to 3)

- ENCA0 outputs the following trigger signals (point <1> in the figure):
  - ENCATIEC (interrupt trigger signal output when the timer counter value is cleared by ENCATZIN input)
  - ENCATEQ0 (trigger signal output when the timer counter value matches the compare register 0 value)
  - ENCATEQ1 (trigger signal output when the timer counter value matches the compare register 1 value)
- Each trigger signal output from ENCA0 is converted to a level-changing toggle signal by the DT circuit and then output to TINm of TAUA0 and TAUJ0 (point <2> in the figure).

- By specifying both rising and falling edges as the valid TINm edge of TAUA0 and TAUJ0, the CNTm value is captured by CDRm at the TINm toggle timing (point <3> in the figure). The operation that clears CNTm to 0000_H is performed at the same time.
  The first value captured after operation starts (a in the figure) is the period from the start of TAU operation until trigger input.
- When there is an overflow, $FFFF_H$ (or FFFF $FFFF_H$ in the case of TAUJ) is captured, and the count value is not captured for the first following trigger (point <4> in the figure).

The trigger generation interval can be measured by performing the above operation.

The table below shows combinations of trigger signals and measurement timers, as well as the bit settings for the route specification peripheral interconnection register and input selection registers. Specify the settings appropriate for the trigger signal to be measured and measurement timer to be used.

Table 21-30 Combinations of trigger signals and measurement timers

Interrupt trigger signal	Measurement timer	Peripheral interconnection register bit settings		Selector register bit settings
ENCATIEC	TAUJ0 CH0	PIC0REG3100 = 1		TISLTJ0 = 1
	TAUJ0 CH1	PIC0REG3101 = 1 PIC0REG3102 = 0		TISLTJ1 = 1
ENCATEQ0	TAUA0 CH0	PIC0REG3109 = 0 PIC0REG3110 = 1		TISLTA0BYPS00 = 1 TRXSLTA00 = 0
	TAUA0 CH2	PIC0REG3112 = 1 PIC0REG3113 = 0		TISLTA0BYPS02 = 1 TRXSLTA04 = 0
ENCATEQ1	TAUA0 CH0	PIC0REG3106 = 1 PIC0REG3107 = 0 PIC0REG3108 = 0	PIC0REG3109 = 0 PIC0REG3110 = 1	TISLTA0BYPS00 = 1 TRXSLTA00 = 0
	TAUA0 CH1		PIC0REG3111 = 1	TISLTA0BYPS01 = 1 TISLTA00 = 0 TRXSLTA02 = 0

21.9.4 Setup flow

The setup flow shown in this section describes how to set up measurement of the pulse interval for all the combinations below. For registers for which Note is indicated, change the settings as appropriate for the combination of the trigger signal you want to measure and the timer to be used for measurement. For details about combinations of trigger signals and measurement timers, see *Table 21-30 "Combinations of trigger signals and measurement timers"*.

Encoder timer	Trigger signal	Measurement timer
ENCA0	ENCATIEC	TAUJ0 CH0,TAUJ0 CH1
	ENCATEQ0	TAUA0 CH0,TAUA0 CH2
	ENCATEQ1	TAUA0 CH1

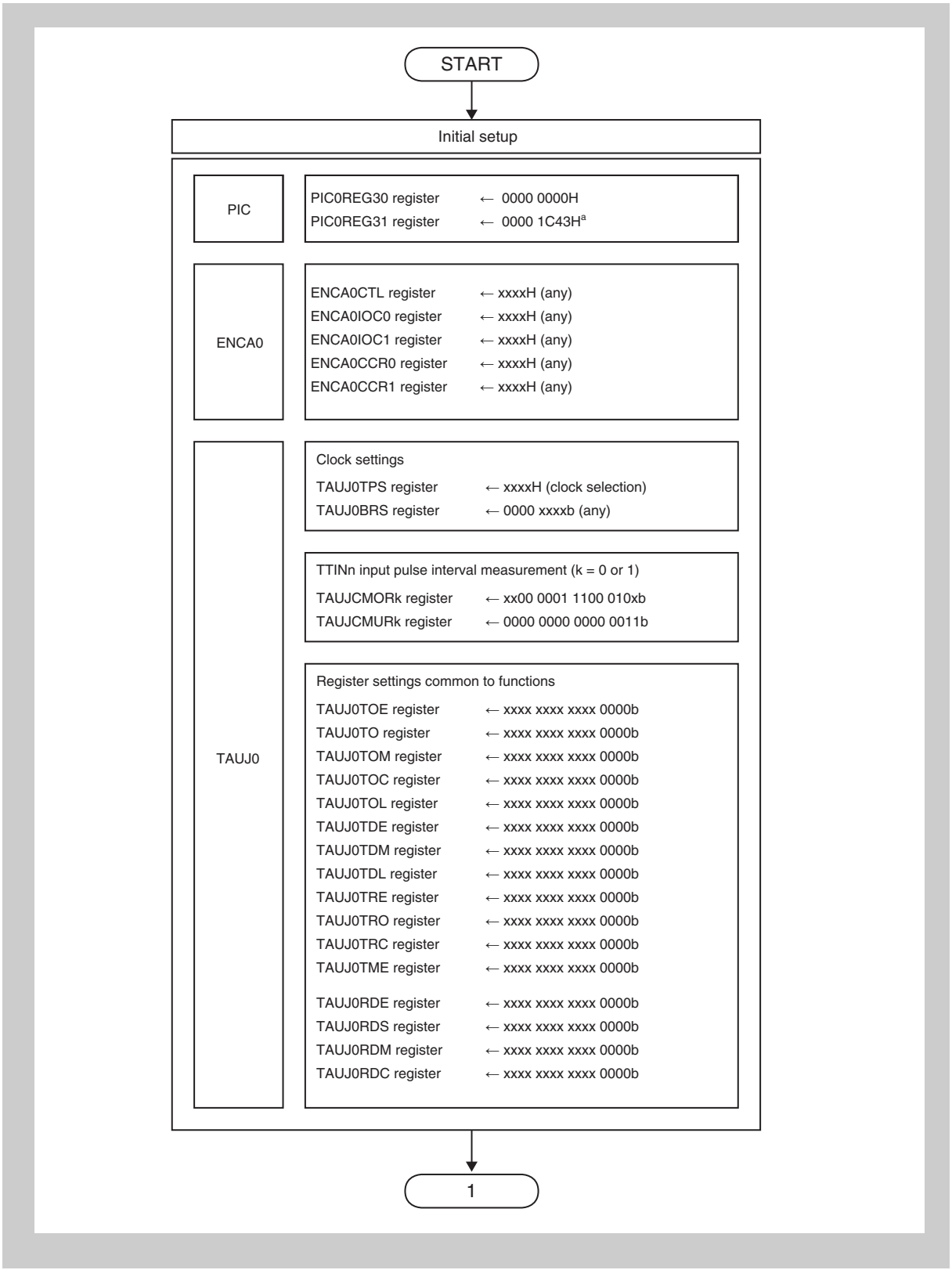


Figure 21-17 Setup flow

a) Change the setting as appropriate for the combination of the trigger signal you want to measure and the timer to be used for measurement.

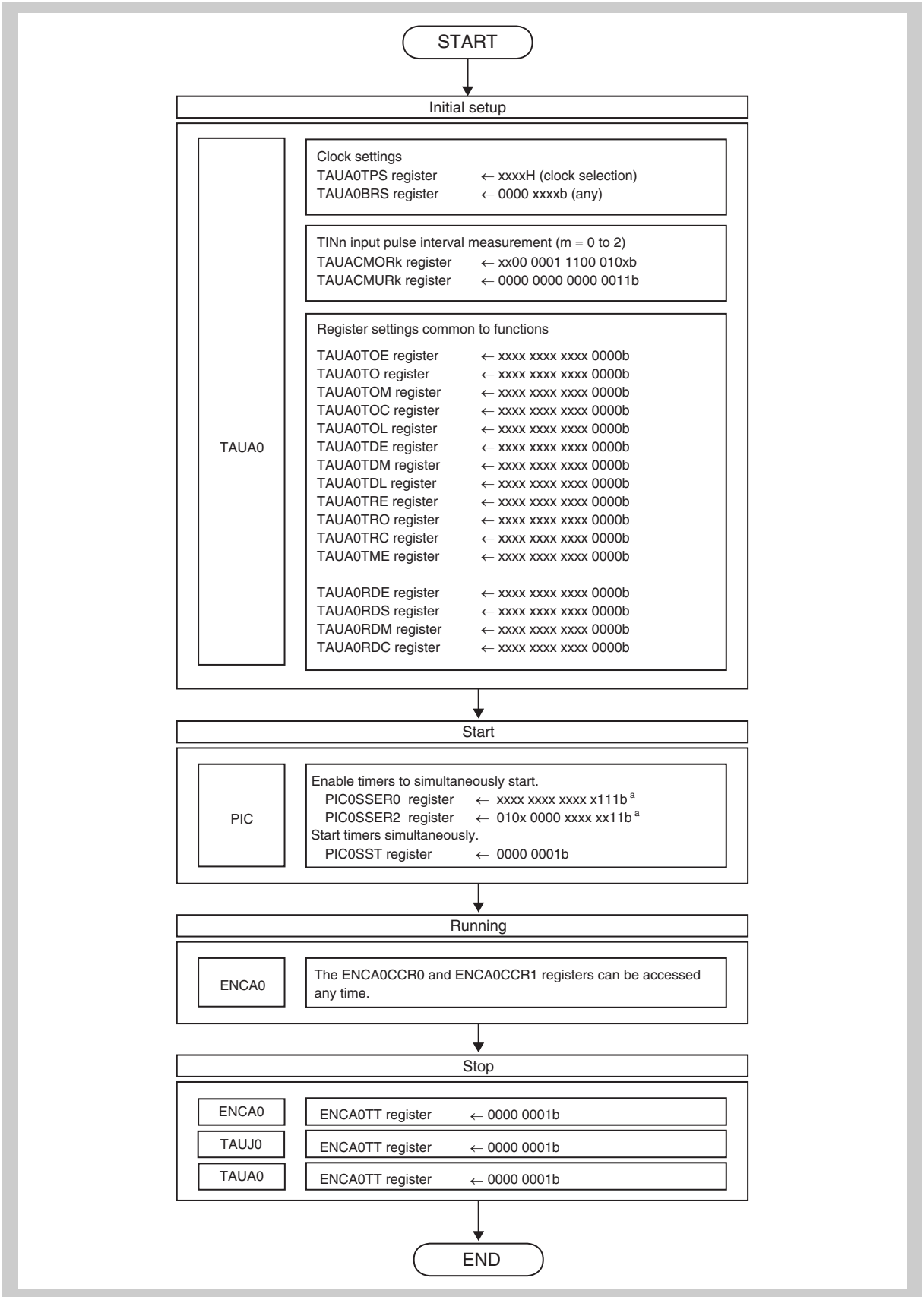


Figure 21-18 Setup flow (continued)

a) Change the setting as appropriate for the combination of the trigger signal you want to measure and the timer to be used for measurement.

21.9.5 Example of setting up operation functions

This section provides example settings for each register.

The setup example shown in this section describes how to set up measurement of the pulse interval for all the combinations below. For details about combinations of trigger signals and measurement timers, see *Table 21-30 "Combinations of trigger signals and measurement timers"*.

Encoder timer	Trigger signal	Measurement timer
ENCA0	ENCATIEC	TAUJ0 CH0, TAUJ0 CH1
	ENCATEQ0	TAUA0 CH0, TAUA0 CH2
	ENCATEQ1	TAUA0 CH1

Table 21-31 ENCA0 settings (1/2)

Register	Bit position	Bit name	Setting	Remark
ENCA0CTL	15	ENCA0CME	Any	Select whether to enable or disable the compare match interrupt detection mask.
	14	ENCA0MCS	Any	Select the cancellation trigger for the compare match interrupt detection mask.
	13 to 10		0	Fixed to 0
	9	ENCA0CRM1	Any	Select the purpose of the ENCA0CCR1 register.
	8	ENCA0CRM0	Any	Select the purpose of the ENCA0CCR0 register.
	7	ENCA0CTS	Any	Select the ENCA0CCR1 capture operation trigger. This does not have to be specified because the trigger is only valid when CRM1 = 1.
	6, 5		0	Fixed to 0
	4	ENCA0LDE	Any	Select whether to enable or disable reloading when an underflow occurs.
	3	ENCA0ECM1	Any	Select whether to enable or disable ENCA0CCR1 compare match counter clearing.
	2	ENCA0ECM0	Any	Select whether to enable or disable ENCA0CCR0 compare match counter clearing.
	1, 0	ENCA0UDS[1:0]	Any	Select whether the counter is incremented or decremented by ENCA0E0 and ENCA0E1.
ENCA0IOC0	7 to 4		0	Fixed to 0
	3, 2	ENCA0TIS[3:2]	Any	Select the valid edge of capture trigger 1 (ENCA0I1).
	1, 0	ENCA0TIS[1:0]	Any	Select the valid edge of capture trigger 0 (ENCA0I0).

Table 21-31 ENCA0 settings (2/2)

Register	Bit position	Bit name	Setting	Remark
ENCA0IOC1	7	ENCA0SCE	Any	Select whether to enable special encoder clearing.
	6	ENCA0ZCL	Any	Select the Z-phase clear level for special encoder clearing.
	5	ENCA0BCL	Any	Select the B-phase clear level for special encoder clearing.
	4	ENCA0ACL	Any	Select the A-phase clear level for special encoder clearing.
	3, 2	ENCA0ECS[1:0]	Any	Select the encoder clearing input (Z phase) edge.
	1, 0	ENCA0EIS[1:0]	Any	Select the encoder input (A and B phases) edge.

Table 21-32 TAUJ0 settings (k = 0 to 3)

Register	Bit position	Bit name	Setting	Remark
TAUJ0CMORk	15, 14	TAUJ0CKS[1:0]	Any	Operation clock setting
	13, 12	TAUJ0CCS[1:0]	00	
	11	TAUJ0MAS	1	
	10, 9, 8	TAUJ0STS[2:0]	000	
	7, 6	TAUJ0COS[1:0]	11	
	5		0	Fixed to 0
	4	TAUJ0MD[4:1]	0010	
	0	TAUJ0MD0	Any	
TAUJ0CMURk	1, 0	TAUJ0TIS[1:0]	00	

Note When using TAUJ0CMORk to perform TAUAnTTINm input pulse interval measurement, each bit of TAUJ0CKS[1:0] (operation clock selection) and TAUJ0MD0 (INTm output control when starting counting) can be set to any value.
The TAUJ0COS[1:0] (overflow operation selection) bits can also be set to any value, but use a fixed value for this feature.
Other control bits use fixed values. For details, see the related sections of the *"Timer Array Unit J (TAUJ)"* chapter.
For the TAUJ common registers (TAUJ0TOE, TAUJ0TO, TAUJ0TOM, TAUJ0TOC, TAUJ0TOL, TAUJ0RDE, and TAUJ0RDM), clear only the bits corresponding to the channels to be used to 0.

Table 21-33 TAUA0 settings (m = 0 to 2)

Register	Bit position	Bit name	Setting	Remark
TAUA0CMORm	15, 14	TAUA0CKS[1:0]	Any	Operation clock setting
	13, 12	TAUA0CCS[1:0]	00	
	11	TAUA0MAS	1	
	10 to 8	TAUA0STS[2:0]	000	
	7, 6	TAUA0COS[1:0]	11	
	5		0	Fixed to 0
	4 to 1	TAUA0MD[4:1]	0010	
	0	TAUA0MD0	Any	
TAUA0CMURm	1, 0	TAUA0TIS[1:0]	00	

Note When using TAUA0CMORm to perform TAUA0TTINm input pulse interval measurement, each bit of TAUA0CKS[1:0] (operation clock selection) and TAUA0MD0 (INTm output control when starting counting) can be set to any value.
The TAUA0COS[1:0] (overflow operation selection) bits can also be set to any value, but use a fixed value for this feature.
Other control bits use fixed values. For details, see the related section of the *"Timer Array Unit A (TAUA)"* chapter. For the TAUA common registers (TAUA0TOE, TAUA0TO, TAUA0TOM, TAUA0TOC, TAUA0TOL, TAUA0TDE, TAUA0TDM, TAUA0TDL, TAUA0TRE, TAUA0TRO, TAUA0TRC, TAUA0TME, TAUA0RDE, TAUA0RDS, TAUA0RDM, and TAUA0RDC), clear only the bits corresponding to the channels to be used to 0.

Table 21-34 Peripheral interconnection settings

Register	Bit position	Bit name	Setting	Remark
PIC0REG31	13, 12	PIC0REG3113 PIC0REG3112	0 1	Select the ENCATEQ0 signal output via DT as the signal input to TAUA0TTIN2.
	11	PIC0REG3111	1	Select the signal selected by PIC0REG3106 to PIC0REG3108 (the ENCATEQ1 signal output via DT) as the signal input to TAUA0TTIN1.
	10, 9	PIC0REG3110 PIC0REG3109	1 0	Select the ENCAT1EQ0 signal output via DT as the signal input to TAUA0TTIN0.
	8 to 6	PIC0REG3108 PIC0REG3107 PIC0REG3106	0 0 1	Select the ENCATEQ1 signal output via DT as the signal input to TAUA0TTIN1 and TAUA0TTIN0.
	2, 1	PIC0REG3102 PIC0REG3101	0 1	Select the ENCATIEC signal output via DT as the signal input to TAUJ0TTIN1.
	0	PIC0REG3100	1	Select the ENCATIEC signal output via DT as the signal input to TAUJ0TTIN0.

21.9.6 Registers

(1) PIC0REG31 - Timer I/O control register 31

Access This register can be read or written in 32-bit units.

Address FF81 C0C0$_H$

Initial value 0000 0000$_H$

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8
0	0	PIC0REG3113	PIC0REG3112	PIC0REG3111	PIC0REG3110	PIC0REG3109	PIC0REG3108
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PIC0REG3107	PIC0RG3106	0	0	0	PIC0REG3102	PIC0REG3101	PIC0RG3100
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-35 PIC0REG31 register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>13
12</td><td>PIC0REG3113
PIC0REG3112</td><td>Selection of TAUA0TTIN2 input.
<table>
<tr><th>PIC0REG3113</th><th>PIC0REG3112</th><th>Input signal</th></tr>
<tr><td>0</td><td>0</td><td>Port TAUA0I2</td></tr>
<tr><td>0</td><td>1</td><td>ENCATEQ0 signal output via DT</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>11</td><td>PIC0REG3111</td><td>Selection of TAUA0TTIN1 input:
0: port TAUA0I1
1: signal selected by PIC0REG3106 to PIC0REG3108</td></tr>
<tr><td>10
9</td><td>PIC0REG3110
PIC0REG3109</td><td>Selection of TAUA0TTIN0 input.
<table>
<tr><th>PIC0REG3110</th><th>PIC0REG3109</th><th>Input signal</th></tr>
<tr><td>0</td><td>0</td><td>The signal selected by PIC0REG3106 to PIC0REG3108</td></tr>
<tr><td>1</td><td>0</td><td>ENCATEQ0 signal output via DT</td></tr>
<tr><td colspan="2">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>8
7
6</td><td>PIC0REG3108
PIC0REG3107
PIC0REG3106</td><td>Selection of TAUA0TTIN0 and TAUA0TTIN1 inputs.
<table>
<tr><th>PIC0REG3108</th><th>PIC0REG3107</th><th>PIC0REG3106</th><th>Input signal</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Port TAUA0I0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>ENCATEQ1 signal output via DT</td></tr>
<tr><td colspan="3">Other than the above</td><td>Setting prohibited</td></tr>
</table></td></tr>
<tr><td>5
4</td><td>PIC0REG3105,
PIC0REG3104</td><td>Selection of TAUJ0TTIN3 input.
See section “TAUJ Input Selection” in chapter “Timer Array Unit J (TAUJ)” for details.</td></tr>
<tr><td>2
1</td><td>PIC0REG3102
PIC0REG3101</td><td>Selection of TAUJ0TTIN1 input.
Selections by PIC0REG3101 and PIC0REG3102 are only effective, if TISLTJ.TISLTJ1 = 1.
<table>
<tr><th>PIC0REG3102</th><th>PIC0REG3101</th><th>Input signal</th></tr>
<tr><td>X</td><td>1</td><td>ENCATIEC signal output via DT</td></tr>
<tr><td>0</td><td>0</td><td rowspan="2">See section “TAUJ Input Selection” in chapter “Timer Array Unit J (TAUJ)” for details.</td></tr>
<tr><td>1</td><td>0</td></tr>
</table></td></tr>
<tr><td>0</td><td>PIC0REG3100</td><td>Selection of TAUJ0TTIN0 input:
0: port TAUJ0I0
1: select the ENCATIEC signal output via DT</td></tr>
</table>

(2) TISLTA0BYPS0 - TAUA0 input selection register

The TISLTA0BYPS0 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 1008_H

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	TISLTA0BYPS02	TISLTA0BYPS01	TISLTA0BYPS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-36 TISLTA0BYPS0 register contents

Bit position	Bit name	Function
2	TISLTA0 BYPS02	Selection of TAUA0TTIN2 input. Selection by TISLTA0BYPS02 is valid when TRXSLTA02 = 0. 0: port TAUA0I2 1: select the signal selected by PIC0REG3112 and PIC0REG3113
1	TISLTA0 BYPS01	Selection of TAUA0TTIN1 input. Selection by TISLTA0BYPS01 is valid when TRXSLTA01 = 0 and TISLTA00 = 0. 0: port TAUA0I1 1: select the signal selected by PIC0REG3111
0	TISLTA0 BYPS00	Selection of TAUA0TTIN0 input. Selection by TISLTA0BYPS00 is valid when TRXSLTA00 = 0. 0: port TAUA0I0 1: select the signal selected by PIC0REG3109 and PIC0REG3110

(3) TRXSLTA0 - TAUA0 reception input selection register

The TRXSLTA0 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 1004_H

Initial value 00_H

7	6	5	4	3	2	1	0
0	0	TRXSLTA0 5	TRXSLTA0 4	TRXSLTA0 [3:2]		TRXSLTA0 [1:0]	
R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-37 TRXSLTA0 register contents

Bit position	Bit name	Function
5	TRXSLTA0 5	Select the signal input to TAUA0TTIN3. Refer to the chapter *"Timer Array Unit A (TAUA)"* for details.
4	TRXSLTA0 4	Select the signal input to TAUA0TTIN2. 0: Select the signal selected by TISLTA0BYPS02. 1: Refer to the chapter *"Timer Array Unit A (TAUA)"* for details.
3, 2	TRXSLTA0 [3:2]	Select the signal input to TAUA0TTIN1. $X0_B$: Select the signal selected by TISLTA0BYPS01. $X1_B$: Refer to the chapter *"Timer Array Unit A (TAUA)"* for details.
1, 0	TRXSLTA0 [1:0]	Select the signal input to TAUA0TTIN0. $X0_B$: Select the signal selected by TISLTA0BYPS00. $X1_B$: Refer to the chapter *"Timer Array Unit A (TAUA)"* for details.

(4) TISLTA0 - TAUA0 input selection register

The TISLTA0 register selects the TAUA0 input signals.

Access This register can be read or written in 8-bit units.

Address FF77 1000_H

Initial value 00_H

7	6	5	4	3	2	1	0
TISLTA07	TISLTA06	TISLTA05	TISLTA04	TISLTA03	TISLTA02	TISLTA01	TISLTA00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-38 TISLTA0 register contents

Bit position	Bit name	Function
7	TISLTA07	Select the signal input to TAUA0TTIN15. See chapter *"Timer Array Unit A (TAUA)"* for details.
6	TISLTA06	Select the signal input to TAUA0TTIN13. See chapter *"Timer Array Unit A (TAUA)"* for details.
5	TISLTA05	Select the signal input to TAUA0TTIN11. See chapter *"Timer Array Unit A (TAUA)"* for details.
4	TISLTA04	Select the signal input to TAUA0TTIN9. See chapter *"Timer Array Unit A (TAUA)"* for details.
3	TISLTA03	Select the signal input to TAUA0TTIN7. See chapter *"Timer Array Unit A (TAUA)"* for details.
2	TISLTA02	Select the signal input to TAUA0TTIN5. See chapter *"Timer Array Unit A (TAUA)"* for details.
1	TISLTA01	Select the signal input to TAUA0TTIN3. See chapter *"Timer Array Unit A (TAUA)"* for details.
0	TISLTA00	Select the signal input to TAUA0TTIN1. 0: Select the signal selected by TRXSLTA0[3:2]. 1: See chapter *"Timer Array Unit A (TAUA)"* for details.

(5) TISLTJ - TAUJ input selection register

The TISLTJ register selects the TAUJ input signals.

Access This register can be read or written in 8-bit units.

Address FF77 3000$_H$

Initial value 00$_H$

7	6	5	4	3	2	1	0
TISLTJ7	TISLTJ6	TISLTJ5	TISLTJ4	TISLTJ3	0	TISLTJ1	TISLTJ0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 21-39 TISLTJ register contents

Bit position	Bit name	Function
7	TISLTJ7	Selection of TAUJ1TTIN3. See section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"* for details.
6	TISLTJ6	Selection of TAUJ1TTIN2. See section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"* for details.
5	TISLTJ5	Selection of TAUJ0TTIN3. See section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"* for details.
4	TISLTJ4	Selection of TAUJ0TTIN2. See section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"* for details.
3	TISLTJ3	Selection of TAUJ0TTIN3: See section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"* for details.
1	TISLTJ1	Selection of TAUJ0TTIN1: 0: port TAUJ0I1 1: signal selection by PICREG31.PICREG3101 See also section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"*.
0	TISLTJ0	Selection of TAUJ0TTIN0: 0: port TAUJ0I0 1: signal selection by PICREG31.PICREG3100 See also section *"TAUJ Input Selection"* in chapter *"Timer Array Unit J (TAUJ)"*.

Chapter 22 PWM Diagnostic

This chapter describes the PWM diagnostic functionality implemented on the V850E2/Fx4-H.

The first section *22.1 "PWM Diagnostic functional overview"* provides an overview of the underlaying concept and its implementation.

The next chapter *22.2 "PWM Delay (DLYA)"* provides the details for the applicable channel-to-channel delay of the PWM signal sources.

The third chapter *22.3 "PWM Diagnostic timing and trigger generation (PMCA)"* provides the details for function blocks related to timing and trigger generation.

22.1 PWM Diagnostic functional overview

This chapter describes the general concept of the PWM diagnosis functionality and its implementation and usage on the V850E2/Fx4-H.

22.1.1 Basic concept and definitions

In various applications the current through a PWM controlled load needs to be measured in order to monitor the correct function of the load and take the appropriate measures in case of a malfunction.

Purpose of the PWM diagnostic modules and related modules is to provide the means to

- generate up to 64 PWM signals. In this chapter each PWM signal is referenced by PWMa with a = 0 to 63
- generate a channel delay between the PWM signals in order to minimize the overall EME radiation.
- trigger the A/D Converter during the high time of the PWM signal relative to the active edge of the PWM signals involved.
- control the address signals of an external multiplexer used to expand the number of PWM channels being monitored and/or minimize the number of required A/D Converter channels.

PWM group

A PWM group contains the PWM signals with the same PWM period frequency. Up to eight PWM groups are available. Within this chapter the PWM groups are named PWMGrp(b) with b = 0 to 7.

PWM group period frequencies

For each PWM group a PWM period frequency $f_{PWM}(b)$ with b = 0 to 7 can be defined. The PWM periods are divided into two 'flavours':

- Synchronous PWM period frequencies.
- Asynchronous PWM period frequencies.

Synchronous PWM frequencies

The synchronous PWM period frequencies must be multiples of each other, e.g.

$f_{PWM}(0)$ = 100 Hz,

$f_{PWM}(1) = 2 \cdot f_{PWM}(0) = 200$ Hz

$f_{PWM}(2) = 4 \cdot f_{PWM}(0) = 2 \cdot f_{PWM}(1) = 400$ Hz

Multiplexer address signals

The external multiplexer addresses are controlled relative to the period of the slowest synchronous PWM period frequency (in the example above $f_{PWM}(0)$).

Multiplexer frame

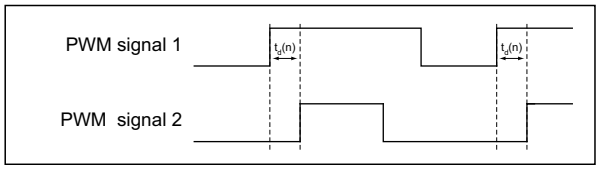
A complete pass of eight multiplexer address changes from 000_B to 111_B is called ‘multiplexer frame’.

Asynchronous PWM frequencies

The period frequencies of the asynchronous PWM frequencies are independent to any other PWM frequency and can be freely chosen within the possible range.
The external multiplexer addresses cannot be switched relative to the periods of the asynchronous PWM frequencies.

Channel-to-channel delay

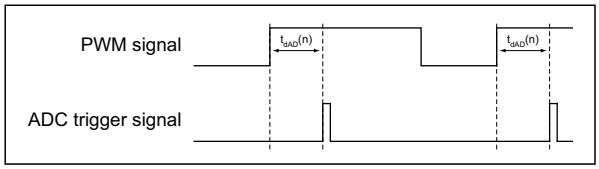
For each PWMa signal a delay $t_d(a)$ can be specified. In doing so, the active edge of the PWM signals of a PWM period frequency can be spread within the period.



For details on the channel-to-channel delay refer to chapter *22.1.3 “Channel-to-channel delay” on page 1758* .

A/D Converter trigger delay

For each PWM group period frequency $f_{PWM}(b)$ an A/D Converter trigger $t_{dADC}(b)$ can be set. It specifies the delay between the active edge of the PWM signal and the related A/D Converter trigger signal.



For details on the A/D Converter trigger delay refer to section *22.1.4 “A/D Converter trigger delay” on page 1759* .

22.1.2 PWM generation

The PWM signals are generated by the Timer Array Units A (TAUA), B (TAUB) and C (TAUC).

For details on general PWM signal generation by TAUA, TAUB, TAUC refer to the related chapter of the TAUA, TAUB, TAUC.

When using the PWM signals in conjunction with the PWM diagnostic modules the following configuration must be used:

Master channels for PWM period

For the definition of the PWM period the following TAUA, TAUB, TAUC channels must be used as master channels:

CH0, CH4, CH8, CH12

For the synchronous PWM signals the period must be set to the frequencies

$f_{PWM}(0)$ to $f_{PWM}(2)$.

For asynchronous PWM signals the period must be set to the frequencies of

$f_{PWM}(3)$ to $f_{PWM}(7)$.

Slave channels for PWM duty cycle

The following TAUA, TAUB, TAUC channels must be configured as slave channels to the related master channel:

TAUA, TAUB, TAUC master channel	TAUA, TAUB, TAUC slave channels
CH0	CH1, CH2
CH4	CH5, CH6
CH8	CH9, CH10
CH12	CH13, CH14

22.1.3 Channel-to-channel delay

A channel delay can be configured individually for each TAUA, TAUB, TAUC output by means of the PWM Delay module. Refer to the chapter *"PWM Delay (DLYA)"* for details.

Still, the configuration needs to meet the setup described below:

4. For each PWM group a related channel-to-channel delay $t_{dg}(c)$ can be defined.
5. A maximum of eight different delay times are supported for each PWM group ($t_{dg(n)}(c)$ with c = 0 to 7.
6. The delay time for each PWM signal of a PWM group $t_{dg(n)}(c)$ must be a multiple of $t_{dg}(n)$.
7. The longest delay time of a PWM signal of a PWM group must be smaller than the period of the PWM group.

This leads to the following configurations:

$t_{dg}(n) < 1/f_{PWM}(n) / (8\text{-}1)$

$t_{dg(n)}(c) = c \cdot t_{dg}(n)$

Example

The period frequency for the PWM group 0 is 100 Hz:

$f_{PWM}(0) = 100$ Hz

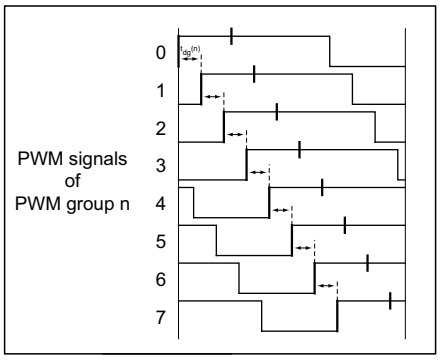
The resulting maximum delay time for a signal of that group is

$t_{dg}(0) = 1/100$ Hz / (8-1) = 1.42 ms

Thus the resulting delay times for the signals of that group are

$t_{dg(0)}(0) = 0 \cdot t_{dg}(0) = 0.00$ ms
$t_{dg(0)}(1) = 1 \cdot t_{dg}(0) = 1.42$ ms
$t_{dg(0)}(2) = 2 \cdot t_{dg}(0) = 2.84$ ms
$t_{dg(0)}(3) = 3 \cdot t_{dg}(0) = 4.26$ ms

$t_{dg(0)}(4) = 4 \cdot t_{dg}(0) = 5.68$ ms
$t_{dg(0)}(5) = 5 \cdot t_{dg}(0) = 7.10$ ms
$t_{dg(0)}(6) = 6 \cdot t_{dg}(0) = 8.52$ ms
$t_{dg(0)}(7) = 7 \cdot t_{dg}(0) = 9.94$ ms

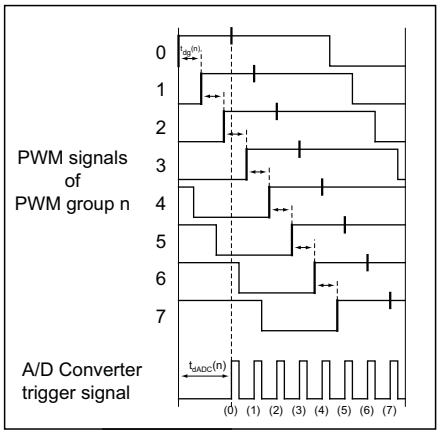


22.1.4 A/D Converter trigger delay

In order to measure a PWM signal during its active phase the A/D Converter needs to be triggered with a precise timing relative to the PWM signal.

For each PWM group an A/D Converter trigger delay $t_{dADC}(b)$ can be defined. The A/D Converter trigger delay is the time between the start of the group period to the first A/D Converter trigger. Only a single A/D Converter trigger must be defined per group, as the A/D Converter triggers for PWM signals with a delay of that group are generated automatically. In total 8 A/D Converter triggers are generated.

The diagram below shows 8 PWM signals of a PWM group (for simplicity reasons with the same duty cycle) and the related A/D Converter trigger signals.



The above described A/D Converter trigger signals are valid for PWM groups with asynchronous PWM frequencies. The A/D Converter trigger signals for PWM groups with synchronous PWM frequencies are derived the same way

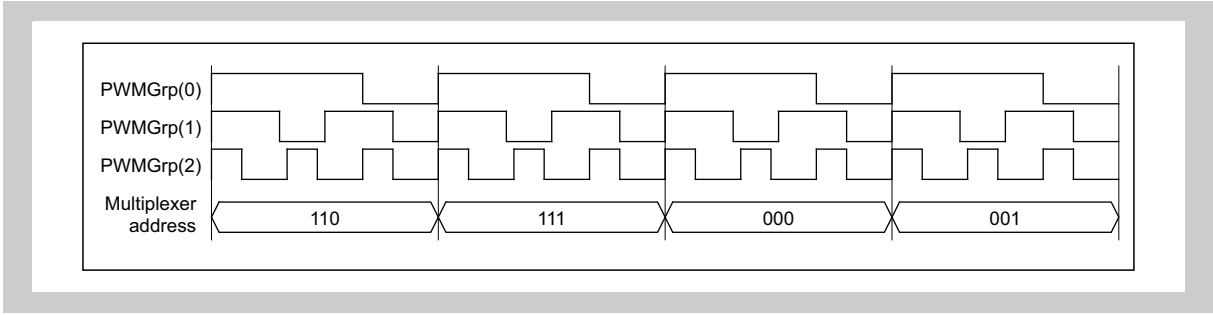
but are further processed before being passed to the A/D Converter trigger inputs. See the next section for details.

22.1.5 Synchronous PWM groups multiplexer and A/D Converter control

For the PWM groups PWMGrp(0) to PWMGrp(2) the address signals of an external multiplexer can be generated synchronously to the periods of the groups.

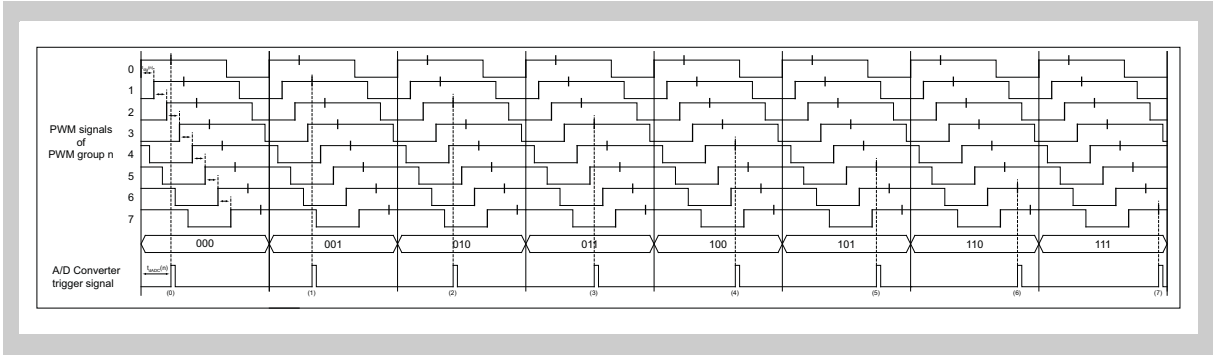
The three multiplexer address signals start at [000] and are incremented by one until the value of 111_B is reached, where they restart at 000_B.

The multiplexer address signals change at the beginning of the slowest synchronous PWM group period. The diagram below shows an example with a single non-delayed signal for each of the PWM groups and the related multiplexer address signals.



For the synchronous PWM groups the eight A/D Converter trigger signals are distributed over the periods of the multiplexer frame.

As an example, the diagram below shows eight PWM signals with different delays, the A/D Converter trigger signals and the multiplexer addresses for a complete frame of a PWM group:

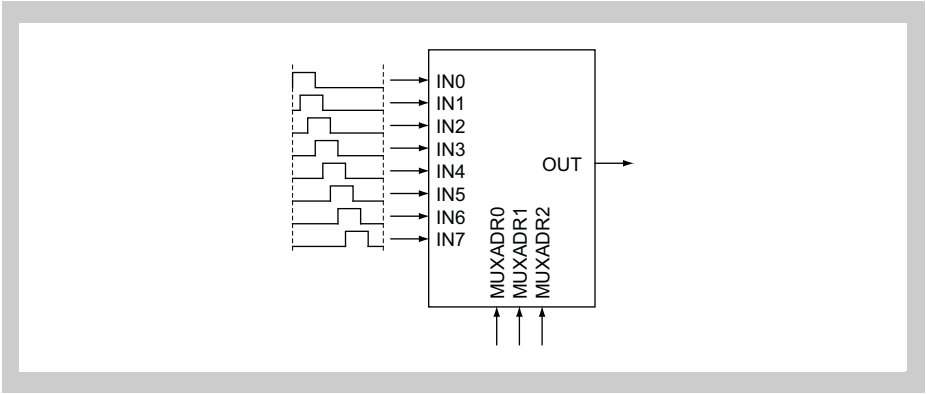


As a consequence of this relationship between a A/D Converter trigger signals and a multiplexer address the PWM signals connected to the input to the multiplexer must meet the following delay requirements:

PWM signal delay $t_d(a)$ of PWM signal PWMa at multiplexer input m (m = 0 to 7):

$$t_d(a) = t_{dg(b)}(m)$$

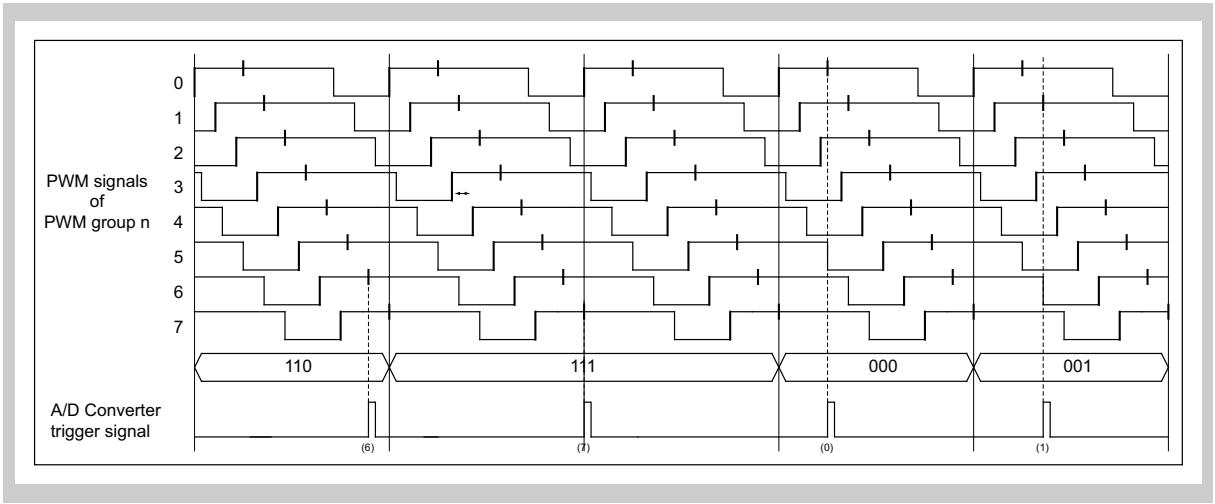
The diagram below shows an example:



Depending on the timings of the PWM group periods, channel-to-channel delays and A/D Converter trigger delays not all A/D Converter triggers might reside within the same period. E.g. a PWM group period of 10 ms, a channel-to-channel delay of 1 ms and a A/D Converter trigger delay of 3 ms the A/D Converter trigger would result in the following A/D Converter trigger times (relative to the start of the period):

$t_{dADC}(0)$ = 3 ms
$t_{dADC}(1)$ = 4 ms
$t_{dADC}(2)$ = 5 ms
$t_{dADC}(3)$ = 6 ms
$t_{dADC}(4)$ = 7 ms
$t_{dADC}(5)$ = 8 ms
$t_{dADC}(6)$ = 9 ms
$t_{dADC}(7)$ = 10 ms

The last A/D Converter trigger resides at the beginning the next period.
In such a case the switching of the multiplexer is delayed for a period. Actually, the multiplexer addresses will only change if all A/D Converter triggers of all active PWM groups have occurred within a given multiplexer address. See the diagram below for an example:



The multiplexer address of 111_B is kept for a second period, as the A/D Converter trigger for the PWM signal 7 does not take place with the same period as the A/D Converter triggers for the PWM signals 0 to 6.

22.1.6 A/D Converter conversion result memory transfer

In order to minimize the CPU load involved in the PWM diagnostic, the DMA should be utilized to transfer the A/D Converter conversion results from the A/D Converter conversion result register to a location in RAM.

The number of necessary DMA transfers is shown here:

(1) DMA transfers for synchronous PWM frequencies:

For the synchronous PWM frequencies it is suggested to let the DMA transfer all A/D Converter conversion results of a multiplexer frame and then use the DMA completion interrupt to trigger the CPU to evaluate the conversion results.

The number of necessary DMA transfers for all conversion results within a multiplexer frame depends on the following variables:

- Number of active PWM groups (max. 3): a
- Number of A/D Converter triggers per PWM group (one trigger per period): fixed to 8
- Number of A/D Converter inputs used for synchronous PWM signal diagnostics: b

DMA transfer count = a · b · 8

Example:

Active PWM groups = 3
A/D Converter inputs used = 8

DMA trigger count = 3 · 8 · 8 = 192

(2) DMA transfers for asynchronous PWM frequencies:

For the asynchronous PWM groups it is recommended to transfer the A/D Converter conversion results of one PWM group period from A/D Converter registers to RAM and then evaluate the results by CPU.

The number of necessary DMA transfers for all conversion results within a PWM group period depends on the following variable:

- Number of A/D Converter triggers per PWM group (one trigger per period): fixed to 8
- Number of A/D Converter inputs used for synchronous PWM signal diagnostics: b

DMA transfer count = b · 8

Example:

A/D Converter inputs used = 4

DMA trigger count = 4 · 8 = 32

22.1.7 Diagnostic value evaluation

Whenever the PWM group trigger signal triggers the conversion of an A/D Converter channel group *all* inputs associated with that channel group are converted. But as at a specific time only the input related to exactly that timing

is valid, the conversion results of the other inputs must be neglected.
E.g. in the example of the DMA transfer for asynchronous PWM frequencies, a total of 32 A/D Converter conversion results are transferred by the DMA to RAM. But only four of them actually hold valid values.

For the synchronous PWM group frequencies another item must be considered:
Within each of the eight multiplexer address periods each synchronous PWM group is triggered once. The order in which they are triggered depends on their occurrences within that period. Due to the shifts implied by the individual channel-to-channel delay of each PWM group the order of PWM group triggers may be different for the multiplexer addresses periods.

The PWM group trigger order is not different from one multiplexer address frame to another though.

22.2 PWM Delay (DLYA)

This chapter contains the generic description of the PWM Delay unit (DLYA)

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc. The subsequent sections describe the features that apply to all implementations.

22.2.1 DLYA features

Instances

This microcontroller has following number of instances of the DLYA unit.

Table 22-1 Instances of DLYA

DLYA	
Instance	1
Name	DLYA0

Instances index n

Throughout this chapter, the individual instances of a DLYA unit is identified by the index "n" (n = 0), for example, DLYAnCTL00 for the DLYAn control register 00.

Channel group index p

The PWM Delay (DLYA) has 8 channel groups. Throughout this chapter, the individual channel groups are identified by the index “p” (p = 0 to 7).

Channel index q

Each of the 8 channel groups p = 0 to 7 has 8 channels. Throughout this chapter, the individual channels are identified by the index “q” (q = 0 to 7).

Register addresses

All DLYAn register addresses are given as address offsets to the individual base address <DLYAn_base>.
The base address <DLYAn_base> of each DLYAn is listed in the following table:

Table 22-2 Register base addresses <DLYAn_base>

DLYAn instance	<DLYAn_base> address
DLYA0	FF81 0800_H

Clock supply All DLYA units provide two clock inputs:

Table 22-3 DLYA clock supply

DLYAn instance	DLYAn clock	Connected to
DLYA0	PCLK	Clock Generator CKSCLK_006

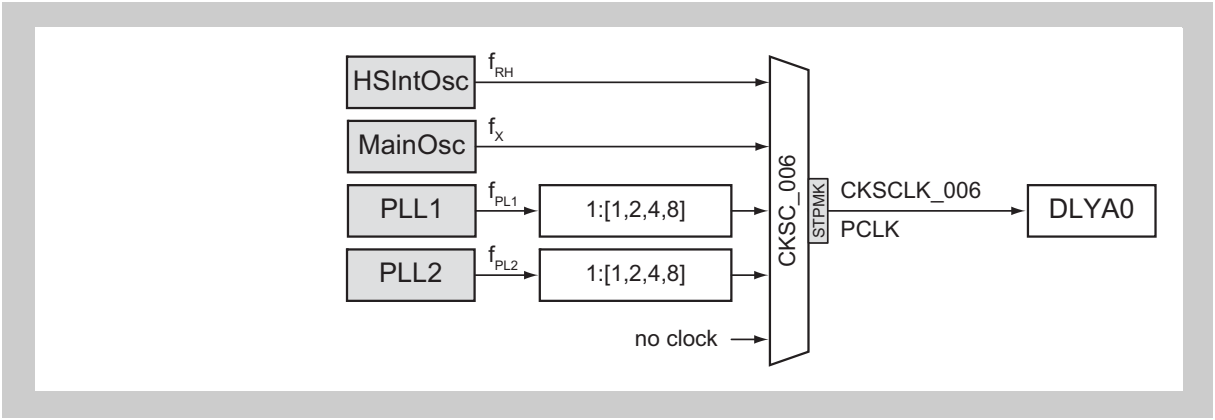


Figure 22-1 DLYA clock supply

DLYA count clock CNTCLK The internal count clock CNTCLK is derived from the PCLK, as shown in the diagram below:

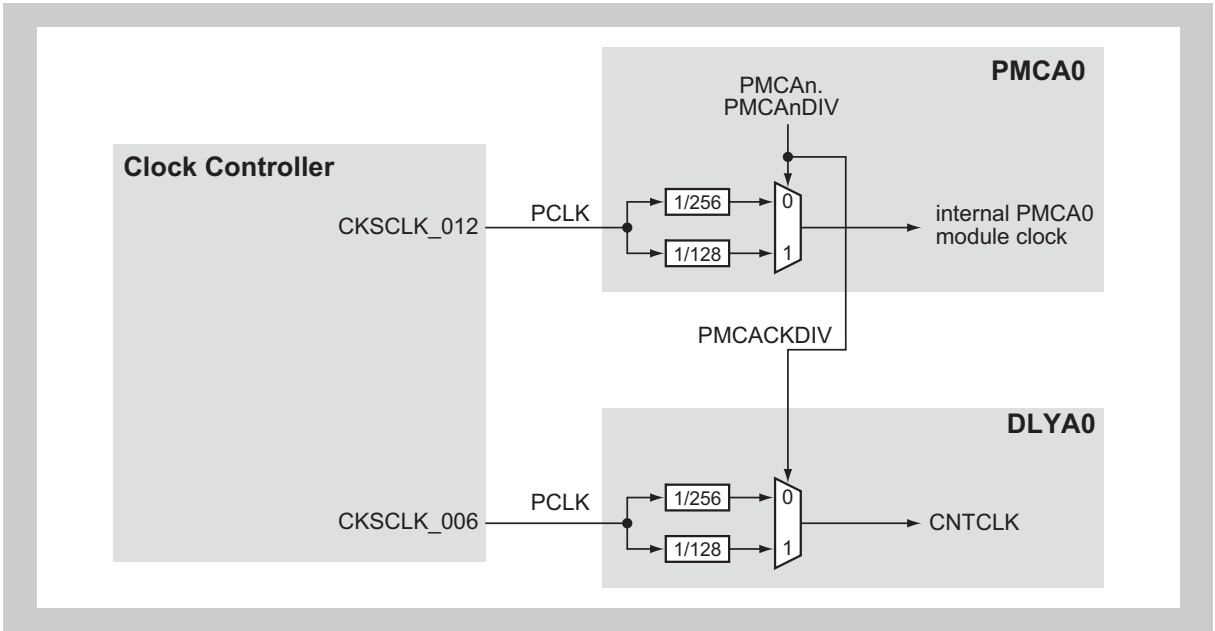


Figure 22-2 DLYA count clock CNTCLK

The count clock selection (PCLK/256 or PCLK/128) is controlled by the PWM Diagnostic timing and trigger generation module PMCA0. control register bit PMCA0CTL1.PMCA0DIV.

DLYA H/W reset The DLYA units and their registers are initialized by the following reset signal:

Table 22-4 DLYA reset signal

DLYAn instance	Reset signal
DLYA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

PWM signals connection Each delay circuit input of DLYA is connected to an individual output of a Timer Array Unit.
The internal signal connections of the DLYAn units to the Timer Array Unit outputs are listed in the following table:

Table 22-5 DLYAn PWM signals connections (1/2)

Channel group	DLYA0 input signal	Connected to	DLYA0 ouput signal	Connected to
p = 0:	DLYA0TDIN00	TAUA0TTOUT1	DLYA0TDOUT00	TAUA0 output ports via PWM Delay (DLYA) bypass control[a]
	DLYA0TDIN01	TAUA0TTOUT2	DLYA0TDOUT01	
	DLYA0TDIN02	TAUA0TTOUT5	DLYA0TDOUT02	
	DLYA0TDIN03	TAUA0TTOUT6	DLYA0TDOUT03	
	DLYA0TDIN04	TAUA0TTOUT9	DLYA0TDOUT04	
	DLYA0TDIN05	TAUA0TTOUT10	DLYA0TDOUT05	
	DLYA0TDIN06	TAUA0TTOUT13	DLYA0TDOUT06	
	DLYA0TDIN07	TAUA0TTOUT14	DLYA0TDOUT07	
p = 1:	DLYA0TDIN10	TAUB1TTOUT1	DLYA0TDOUT10	TAUB1 output ports via PWM Delay (DLYA) bypass control[a]
	DLYA0TDIN11	TAUB1TTOUT2	DLYA0TDOUT11	
	DLYA0TDIN12	TAUB1TTOUT5	DLYA0TDOUT12	
	DLYA0TDIN13	TAUB1TTOUT6	DLYA0TDOUT13	
	DLYA0TDIN14	TAUB1TTOUT9	DLYA0TDOUT14	
	DLYA0TDIN15	TAUB1TTOUT10	DLYA0TDOUT15	
	DLYA0TDIN16	TAUB1TTOUT13	DLYA0TDOUT16	
	DLYA0TDIN17	TAUB1TTOUT14	DLYA0TDOUT17	
p = 2:	DLYA0TDIN20	TAUB2TTOUT1	DLYA0TDOUT20	TAUB2 output ports via PWM Delay (DLYA) bypass control[a]
	DLYA0TDIN21	TAUB2TTOUT2	DLYA0TDOUT21	
	DLYA0TDIN22	TAUB2TTOUT5	DLYA0TDOUT22	
	DLYA0TDIN23	TAUB2TTOUT6	DLYA0TDOUT23	
	DLYA0TDIN24	TAUB2TTOUT9	DLYA0TDOUT24	
	DLYA0TDIN25	TAUB2TTOUT10	DLYA0TDOUT25	
	DLYA0TDIN26	TAUB2TTOUT13	DLYA0TDOUT26	
	DLYA0TDIN27	TAUB2TTOUT14	DLYA0TDOUT27	

Table 22-5 DLYAn PWM signals connections (2/2)

Channel group	DLYA0 input signal	Connected to	DLYA0 ouput signal	Connected to
p = 3:	DLYA0TDIN30	TAUC3TTOUT1	DLYA0TDOUT30	TAUC3 output ports via PWM Delay (DLYA) bypass control[a]
	DLYA0TDIN31	TAUC3TTOUT2	DLYA0TDOUT31	
	DLYA0TDIN32	TAUC3TTOUT5	DLYA0TDOUT32	
	DLYA0TDIN33	TAUC3TTOUT6	DLYA0TDOUT33	
	DLYA0TDIN34	TAUC3TTOUT9	DLYA0TDOUT34	
	DLYA0TDIN35	TAUC3TTOUT10	DLYA0TDOUT35	
	DLYA0TDIN36	TAUC3TTOUT13	DLYA0TDOUT36	
	DLYA0TDIN37	TAUC3TTOUT14	DLYA0TDOUT37	
p = 4:	DLYA0TDIN40	TAUC4TTOUT1	DLYA0TDOUT40	TAUC4 output ports via PWM Delay (DLYA) bypass control[a]
	DLYA0TDIN41	TAUC4TTOUT2	DLYA0TDOUT41	
	DLYA0TDIN42	TAUC4TTOUT5	DLYA0TDOUT42	
	DLYA0TDIN43	TAUC4TTOUT6	DLYA0TDOUT43	
	DLYA0TDIN44	TAUC4TTOUT9	DLYA0TDOUT44	
	DLYA0TDIN45	TAUC4TTOUT10	DLYA0TDOUT45	
	DLYA0TDIN46	TAUC4TTOUT13	DLYA0TDOUT46	
	DLYA0TDIN47	TAUC4TTOUT14	DLYA0TDOUT47	
p = 5:	DLYA0TDIN50	TAUC5TTOUT1	DLYA0TDOUT50	Port TAUC5O1
	DLYA0TDIN51	TAUC5TTOUT2	DLYA0TDOUT51	Port TAUC5O2
	DLYA0TDIN52	TAUC5TTOUT5	DLYA0TDOUT52	Port TAUC5O5
	DLYA0TDIN53	TAUC5TTOUT6	DLYA0TDOUT53	Port TAUC5O6
	DLYA0TDIN54	TAUC5TTOUT9	DLYA0TDOUT54	Port TAUC5O9
	DLYA0TDIN55	TAUC5TTOUT10	DLYA0TDOUT55	Port TAUC5O10
	DLYA0TDIN56	TAUC5TTOUT13	DLYA0TDOUT56	Port TAUC5O13
	DLYA0TDIN57	TAUC5TTOUT14	DLYA0TDOUT57	Port TAUC5O14
p = 6:	DLYA0TDIN60	TAUC6TTOUT1	DLYA0TDOUT60	Port TAUC6O1
	DLYA0TDIN61	TAUC6TTOUT2	DLYA0TDOUT61	Port TAUC6O2
	DLYA0TDIN62	TAUC6TTOUT5	DLYA0TDOUT62	Port TAUC6O5
	DLYA0TDIN63	TAUC6TTOUT6	DLYA0TDOUT63	Port TAUC6O6
	DLYA0TDIN64	TAUC6TTOUT9	DLYA0TDOUT64	Port TAUC6O9
	DLYA0TDIN65	TAUC6TTOUT10	DLYA0TDOUT65	Port TAUC6O10
	DLYA0TDIN66	TAUC6TTOUT13	DLYA0TDOUT66	Port TAUC6O13
	DLYA0TDIN67	TAUC6TTOUT14	DLYA0TDOUT67	Port TAUC6O14
p = 7:	DLYA0TDIN70	TAUC7TTOUT1	DLYA0TDOUT70	Port TAUC7O1
	DLYA0TDIN71	TAUC7TTOUT2	DLYA0TDOUT71	Port TAUC7O2
	DLYA0TDIN72	TAUC7TTOUT5	DLYA0TDOUT72	Port TAUC7O5
	DLYA0TDIN73	TAUC7TTOUT6	DLYA0TDOUT73	Port TAUC7O6
	DLYA0TDIN74	TAUC7TTOUT9	DLYA0TDOUT74	Port TAUC7O9
	DLYA0TDIN75	TAUC7TTOUT10	DLYA0TDOUT75	Port TAUC7O10
	DLYA0TDIN76	TAUC7TTOUT13	DLYA0TDOUT76	Port TAUC7O13
	DLYA0TDIN77	TAUC7TTOUT14	DLYA0TDOUT77	Port TAUC7O14

a) Refer to section *22.2.2 “PWM Delay (DLYA) bypass control” on page 1768* for details.

22.2.2 PWM Delay (DLYA) bypass control

All Timer Array Unit's output channels, assigned to the DLYA channel groups p = 0 to 4, can bypass the PWM Delay module, as shown in the diagram below.

Bypassing of a channel group is controlled by the DLYA bypass enable register DLYAEN.

Following channel groups can be bypassed:

Table 22-6 Channels and groups with bypass option

TUAxn	Channel group	Bypass control DLYAEN		Port TAUxnO
TAUA0	p = 0	DLYAEN0 =	0:	TAUA0O[1,2,5,6,9,10,13,14]
			1:	DLYATDOUT0[7:0]
TAUB1	p = 1	DLYAEN1 =	0:	TAUB1O[1,2,5,6,9,10,13,14]
			1:	DLYATDOUT1[7:0]
TAUB2	p = 2	DLYAEN2 =	0:	TAUB2O[1,2,5,6,9,10,13,14]
			1:	DLYATDOUT2[7:0]
TAUC3	p = 3	DLYAEN3 =	0:	TAUC3O[1,2,5,6,9,10,13,14]
			1:	DLYATDOUT3[7:0]
TAUC4	p = 4	DLYAEN4 =	0:	TAUC4O[1,2,5,6,9,10,13,14]
			1:	DLYATDOUT4[7:0]

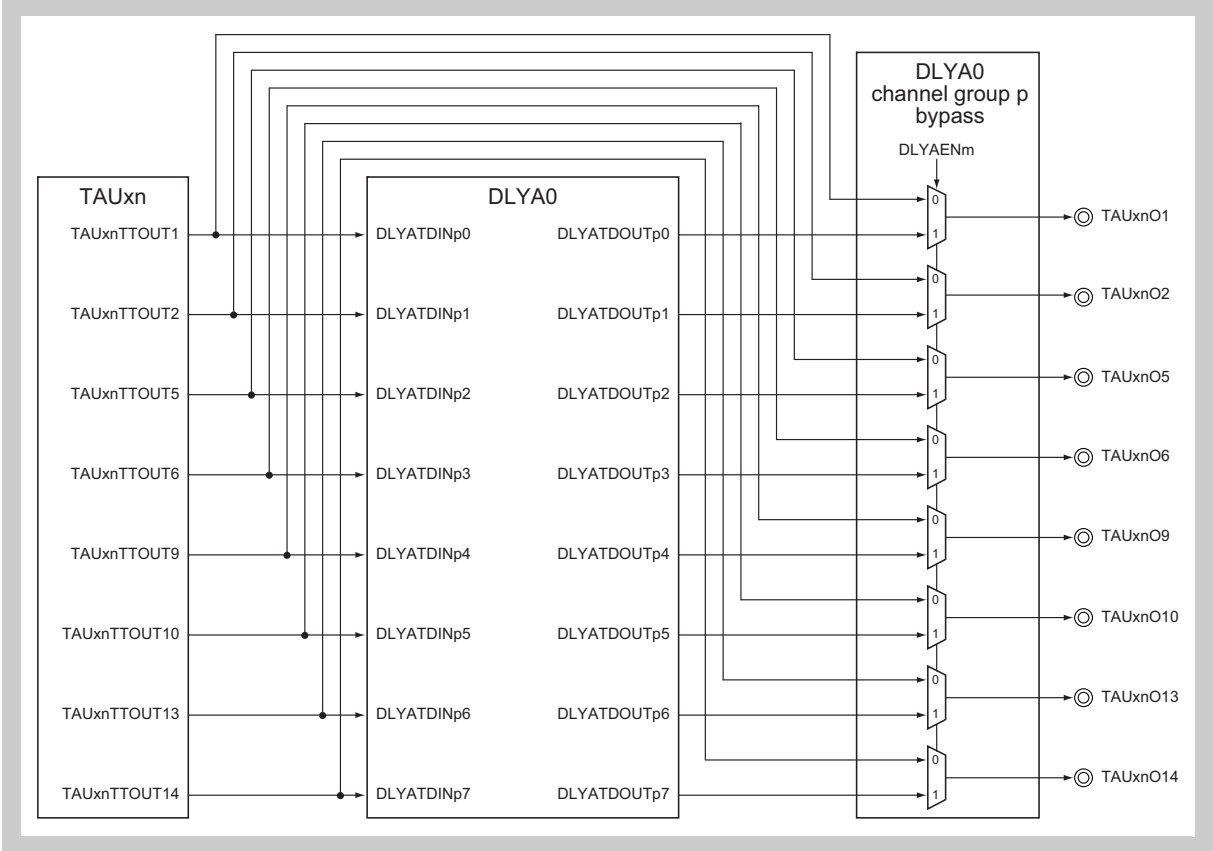


Table 22-7 PWM Delay bypass

(1) DLYAEN – DLYA bypass enable register

This register controls the DLYA bypass for the selected Timer Array Unit output signals.

Access This register can be read/written in 8-bit units.

Address $FF77103C_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	DLYAEN4	DLYAEN3	DLYAEN2	DLYAEN1	DLYAEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22-8 DLYAEN register contents

Bit position	Bit name	Function
4	DLYAEN4	DLYA bypass for TAUC4 (channel group p = 4) output signals: 0: bypass enabled (direct TAUC4 outputs at ports) 1: bypass disabled (delayed DLYATDOUT4[7:0] outputs at ports)
3	DLYAEN3	DLYA bypass for TAUC3 (channel group p = 3) output signals: 0: bypass enabled (direct TAUC3 outputs at ports) 1: bypass disabled (delayed DLYATDOUT3[7:0] outputs at ports)
2	DLYAEN2	DLYA bypass for TAUB2 (channel group p = 2) output signals: 0: bypass enabled (direct TAUB2 outputs at ports) 1: bypass disabled (delayed DLYATDOUT2[7:0] outputs at ports)
1	DLYAEN1	DLYA bypass for TAUB1 (channel group p = 1) output signals: 0: bypass enabled (direct TAUB1 outputs at ports) 1: bypass disabled (delayed DLYATDOUT1[7:0] outputs at ports)
0	DLYAEN0	DLYA bypass for TAUA0 (channel group p = 0) output signals: 0: bypass enabled (direct TAUA0 outputs at ports) 1: bypass disabled (delayed DLYATDOUT0[7:0] outputs at ports)

22.2.3 Functional Overview

Features summary

The DLYA has the following features:

- 64 input channels
- separate delay enable/disable for each channel
- separate delay times for each channel

The PWM delay block diagram is shown below.

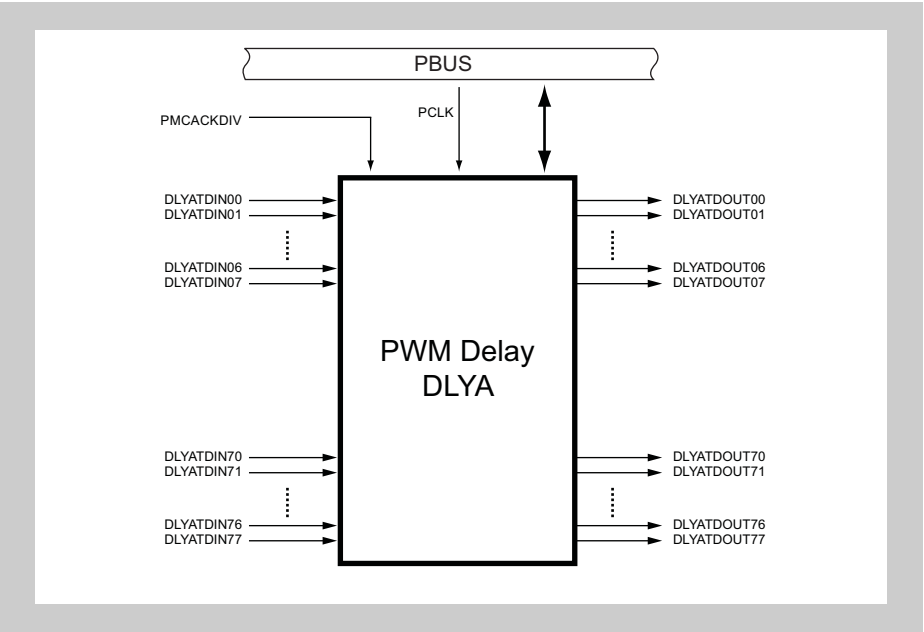


Figure 22-3 Block diagram of the DLYA

(1) Count clock CNTCLK

The internal count clock CNTCLK, which is the base clock for specifying the delays, is a fraction of the PCLK and selected by the PMCACKDIV signal. Refer to the section *"DLYA features"* above and the key word "DLYA count clock CNTCLK".

(2) Delay enable

The delay function can be enabled respectively disabled separately for each input signal DLYATDINpq by use of the control registers

- DLYAnCTL00 for the input signals DLYATDIN0[7:0] to DLYATDIN3[7:0]
- DLYAnCTL01 for the input signals DLYATDIN4[7:0] to DLYATDIN7[7:0]

(3) Delay time

The delay time for each DLYATDINpq input signal is specified by separate registers DLYACMPpq for each channel.

The delay time is specified in number of CNTCLK clock periods by the 12-bit value DLYACMPpq.DLYACMPpq[11:0] in the range of 002_H to FFF_H.

If DLYACMPpq.DLYACMPpq[11:0] = 0, no delay is applied to DLYATDINpq.

22.2.4 Registers

This section contains a description of all registers of the DLYA.

(1) DLYA registers overview

The DLYA functions are controlled and operated by the following registers:

Table 22-9 DLYA register overview (1/2)

Register Name	Shortcut	Address
Control register 00	DLYAnCTL00	<DLYAn_base> + 00_H
Control register 01	DLYAnCTL01	<DLYAn_base> + 04_H
DLYATDIN00 delay selection register	DLYAnCMP00	<DLYAn_base> + 08_H
DLYATDIN01 delay selection register	DLYAnCMP01	<DLYAn_base> + $0C_H$
DLYATDIN02 delay selection register	DLYAnCMP02	<DLYAn_base> + 10_H
DLYATDIN03 delay selection register	DLYAnCMP03	<DLYAn_base> + 14_H
DLYATDIN04 delay selection register	DLYAnCMP04	<DLYAn_base> + 18_H
DLYATDIN05 delay selection register	DLYAnCMP05	<DLYAn_base> + $1C_H$
DLYATDIN06 delay selection register	DLYAnCMP06	<DLYAn_base> + 20_H
DLYATDIN07 delay selection register	DLYAnCMP07	<DLYAn_base> + 24_H
DLYATDIN10 delay selection register	DLYAnCMP10	<DLYAn_base> + 28_H
DLYATDIN11 delay selection register	DLYAnCMP11	<DLYAn_base> + $2C_H$
DLYATDIN12 delay selection register	DLYAnCMP12	<DLYAn_base> + 30_H
DLYATDIN13 delay selection register	DLYAnCMP13	<DLYAn_base> + 34_H
DLYATDIN14 delay selection register	DLYAnCMP14	<DLYAn_base> + 38_H
DLYATDIN15 delay selection register	DLYAnCMP15	<DLYAn_base> + $3C_H$
DLYATDIN16 delay selection register	DLYAnCMP16	<DLYAn_base> + 40_H
DLYATDIN17 delay selection register	DLYAnCMP17	<DLYAn_base> + 44_H
DLYATDIN20 delay selection register	DLYAnCMP20	<DLYAn_base> + 48_H
DLYATDIN21 delay selection register	DLYAnCMP21	<DLYAn_base> + $4C_H$
DLYATDIN22 delay selection register	DLYAnCMP22	<DLYAn_base> + 50_H
DLYATDIN23 delay selection register	DLYAnCMP23	<DLYAn_base> + 54_H
DLYATDIN24 delay selection register	DLYAnCMP24	<DLYAn_base> + 58_H
DLYATDIN25 delay selection register	DLYAnCMP25	<DLYAn_base> + $5C_H$
DLYATDIN26 delay selection register	DLYAnCMP26	<DLYAn_base> + 60_H
DLYATDIN27 delay selection register	DLYAnCMP27	<DLYAn_base> + 64_H
DLYATDIN30 delay selection register	DLYAnCMP30	<DLYAn_base> + 68_H
DLYATDIN31 delay selection register	DLYAnCMP31	<DLYAn_base> + $6C_H$
DLYATDIN32 delay selection register	DLYAnCMP32	<DLYAn_base> + 70_H
DLYATDIN33 delay selection register	DLYAnCMP33	<DLYAn_base> + 74_H
DLYATDIN34 delay selection register	DLYAnCMP34	<DLYAn_base> + 78_H
DLYATDIN35 delay selection register	DLYAnCMP35	<DLYAn_base> + $7C_H$
DLYATDIN36 delay selection register	DLYAnCMP36	<DLYAn_base> + 80_H
DLYATDIN37 delay selection register	DLYAnCMP37	<DLYAn_base> + 84_H
DLYATDIN40 delay selection register	DLYAnCMP40	<DLYAn_base> + 88_H
DLYATDIN41 delay selection register	DLYAnCMP41	<DLYAn_base> + $8C_H$

Table 22-9 DLYA register overview (2/2)

Register Name	Shortcut	Address
DLYATDIN42 delay selection register	DLYAnCMP42	<DLYAn_base> + 90_H
DLYATDIN43 delay selection register	DLYAnCMP43	<DLYAn_base> + 94_H
DLYATDIN44 delay selection register	DLYAnCMP44	<DLYAn_base> + 98_H
DLYATDIN45 delay selection register	DLYAnCMP45	<DLYAn_base> + $9C_H$
DLYATDIN46 delay selection register	DLYAnCMP46	<DLYAn_base> + $A0_H$
DLYATDIN47 delay selection register	DLYAnCMP47	<DLYAn_base> + $A4_H$
DLYATDIN50 delay selection register	DLYAnCMP50	<DLYAn_base> + $A8_H$
DLYATDIN51 delay selection register	DLYAnCMP51	<DLYAn_base> + AC_H
DLYATDIN52 delay selection register	DLYAnCMP52	<DLYAn_base> + $B0_H$
DLYATDIN53 delay selection register	DLYAnCMP53	<DLYAn_base> + $B4_H$
DLYATDIN54 delay selection register	DLYAnCMP54	<DLYAn_base> + $B8_H$
DLYATDIN55 delay selection register	DLYAnCMP55	<DLYAn_base> + BC_H
DLYATDIN56 delay selection register	DLYAnCMP56	<DLYAn_base> + $C0_H$
DLYATDIN57 delay selection register	DLYAnCMP57	<DLYAn_base> + $C4_H$
DLYATDIN60 delay selection register	DLYAnCMP60	<DLYAn_base> + $C8_H$
DLYATDIN61 delay selection register	DLYAnCMP61	<DLYAn_base> + CC_H
DLYATDIN62 delay selection register	DLYAnCMP62	<DLYAn_base> + $D0_H$
DLYATDIN63 delay selection register	DLYAnCMP63	<DLYAn_base> + $D4_H$
DLYATDIN64 delay selection register	DLYAnCMP64	<DLYAn_base> + $D8_H$
DLYATDIN65 delay selection register	DLYAnCMP65	<DLYAn_base> + DC_H
DLYATDIN66 delay selection register	DLYAnCMP66	<DLYAn_base> + $E0_H$
DLYATDIN67 delay selection register	DLYAnCMP67	<DLYAn_base> + $E4_H$
DLYATDIN70 delay selection register	DLYAnCMP70	<DLYAn_base> + $E8_H$
DLYATDIN71 delay selection register	DLYAnCMP71	<DLYAn_base> + EC_H
DLYATDIN72 delay selection register	DLYAnCMP72	<DLYAn_base> + $F0_H$
DLYATDIN73 delay selection register	DLYAnCMP73	<DLYAn_base> + $F4_H$
DLYATDIN74 delay selection register	DLYAnCMP74	<DLYAn_base> + $F8_H$
DLYATDIN75 delay selection register	DLYAnCMP75	<DLYAn_base> + FC_H
DLYATDIN76 delay selection register	DLYAnCMP76	<DLYAn_base> + 100_H
DLYATDIN77 delay selection register	DLYAnCMP77	<DLYAn_base> + 104_H

<DLYAn_base> The base addresses <DLYAn_base> of the DLYAn is defined in the first section of this chapter under the key word “Register addresses”.

(2) DLYAnCTL00 – Control register 00

This register specifies if the delay of the signals if the delay for the signals DLYATDIN[3:0][7:0] is enabled or disabled.

Access This register can be read/written in 32-bit units.

Address <DLYAn_base> + 00_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DLYAn EN37	DLYAn EN36	DLYAn EN35	DLYAn EN34	DLYAn EN33	DLYAn EN32	DLYAn EN31	DLYAn EN30	DLYAn EN27	DLYAn EN26	DLYAn EN25	DLYAn EN24	DLYAn EN23	DLYAn EN22	DLYAn EN21	DLYAn EN20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLYAn EN17	DLYAn EN16	DLYAn EN15	DLYAn EN14	DLYAn EN13	DLYAn EN12	DLYAn EN11	DLYAn EN10	DLYAn EN07	DLYAn EN06	DLYAn EN05	DLYAn EN04	DLYAn EN03	DLYAn EN02	DLYAn EN01	DLYAn EN00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22-10 DLYAnCTL00 register contents

Bit position	Bit name	Function
31 to 0	DLYAnENpq	Selects if the delay for signal DLYATDINpq (p = 3 to 0, q = 7 to 0) is enabled or disabled. 0: The signal DLYATDINpq will not be delayed. 1: The signal DLYATDINpq will be delayed.

Caution The setting of this register may only be changed when no signals are input.

(3) DLYAnCTL01 – Control register 01

This register specifies if the delay of the signals if the delay for the signals DLYATDIN[7:4][7:0] is enabled or disabled.

Access This register can be read/written in 32-bit units.

Address <DLYAn_base> + 04_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DLYAnEN77	DLYAnEN76	DLYAnEN75	DLYAnEN74	DLYAnEN73	DLYAnEN72	DLYAnEN71	DLYAnEN70	DLYAnEN67	DLYAnEN66	DLYAnEN65	DLYAnEN64	DLYAnEN63	DLYAnEN62	DLYAnEN61	DLYAnEN60
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLYAnEN57	DLYAnEN56	DLYAnEN55	DLYAnEN54	DLYAnEN53	DLYAnEN52	DLYAnEN51	DLYAnEN50	DLYAnEN47	DLYAnEN46	DLYAnEN45	DLYAnEN44	DLYAnEN43	DLYAnEN42	DLYAnEN41	DLYAnEN40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22-11 DLYAnCTL01 register contents

Bit position	Bit name	Function
31 to 0	DLYAnENpq	Selects if the delay for signal DLYATDINpq (p = 7 to 4, q = 7 to 0) is enabled or disabled. 0: The signal DLYATDINpq will not be delayed. 1: The signal DLYATDINpq will be delayed.

Caution The setting of this register may only be changed when no signals DLYATDIN[7:4][7:0] are input.

(4) DLYAnCMPpq – Delay selection registers (p, q = 0 to 7)

These registers specify the delays for each DLYATDINpq signal in number of CNTCLK periods

Access This register can be read/written in 32-bit units.

Address DLYAnCMP0q: <DLYAn_base> + 08_H + p x 4
DLYAnCMP1q: <DLYAn_base> + 28_H + p x 4
DLYAnCMP2q: <DLYAn_base> + 48_H + p x 4
DLYAnCMP3q: <DLYAn_base> + 68_H + p x 4
DLYAnCMP4q: <DLYAn_base> + 88_H + p x 4
DLYAnCMP5q: <DLYAn_base> + $A8_H$ + p x 4
DLYAnCMP6q: <DLYAn_base> + $C8_H$ + p x 4
DLYAnCMP7q: <DLYAn_base> + $E8_H$ + p x 4

Initial Value $0000\ 0000_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	DLYAnCMP[11..0]											
R	R	R	R	R/W											

Table 22-12 DLYAnCMPpq register contents

Bit position	Bit name	Function
11 to 0	DLYAnCMP [11..0]	The number of CNTCLKs the input signals DLYATDINpq will be delayed. The value must be in the range of 002_H to FFF_H The delay time must not be bigger than one period of the input signal.

Caution The setting of this register may only be changed when no signals DLYATDINpq are input.

22.2.5 Procedure for Setup

Follow the below setup procedure to put the PWM Delay into operation:

1. Set DLYA0CMPpq register (p, q = 0 to 7)
2. Set DLYA0CTL00 and DLYA0CTL01 registers
3. Input signals to DLYATDINpq (p, q = 0 to 7)

22.2.6 Timing diagrams

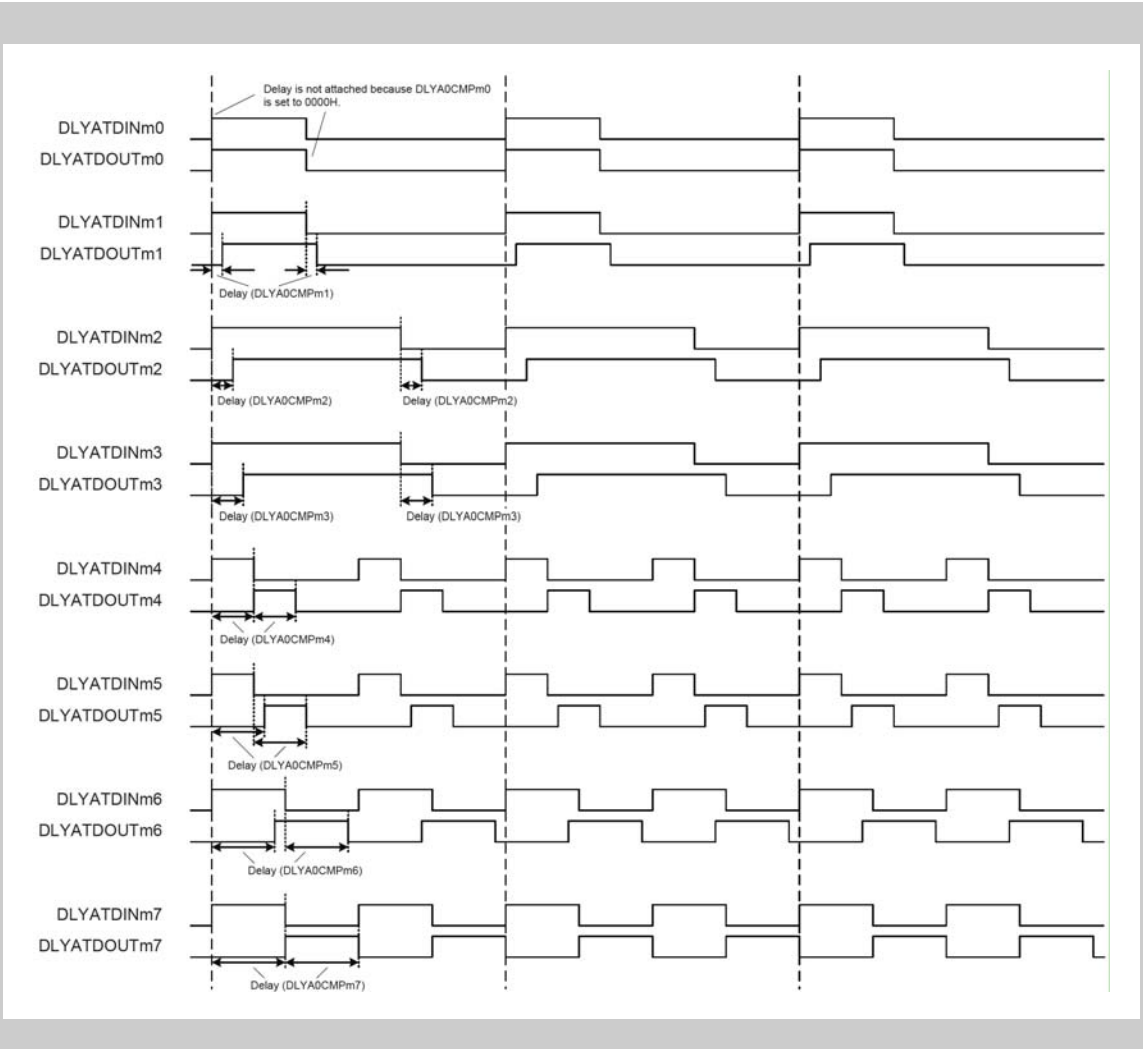


Figure 22-4 Timing diagram of the PWM Delay DLYA

22.3 PWM Diagnostic timing and trigger generation (PMCA)

This chapter contains a description of the PWM Diagnostic timing and trigger module (PMCA).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

22.3.1 Features

Instances

This microcontroller has following number of instances of the PMCA unit.

Table 22-13 Instances of PMCA

Timer Array Unit A	
Instance	1
Name	PMCA0

Instances index n

Throughout this chapter, the individual instances of a PMCA is identified by the index “n” (n = 0), for example, PMCAnCTL0 for the PMCAn control register.

Register addresses

All PMCAn register addresses are given as address offsets to the individual base address <PMCAn_base>.
The base address <PMCAn_base> of each PMCAn is listed in the following table:

Table 22-14 Register base addresses <PMCAn_base>

PMCAn instance	<PMCAn_base> address
PMCA0	FF81 $0C00_H$

Clock supply All PMCAn provide one clock input:

Table 22-15 PMCAn clock supply

PMCAn instance	PMCAn clock	Connected to
PMCA0	PCLK	Clock Generator CKSCLK_012

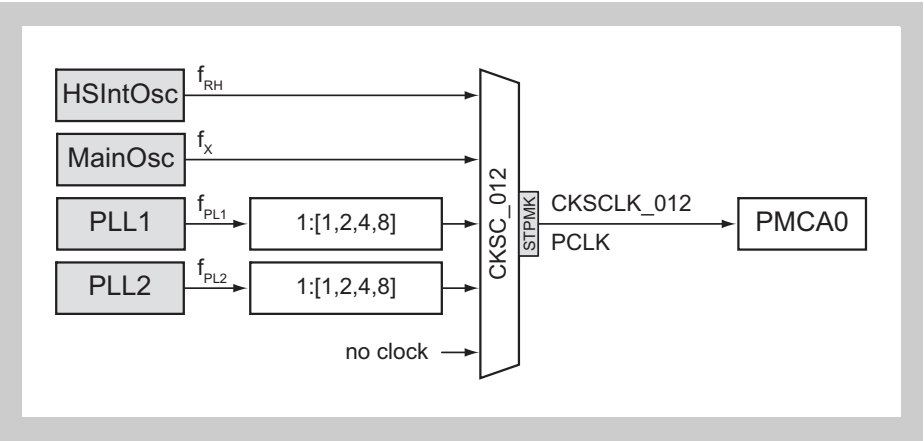


Figure 22-5 PMCA clock supply

PMCA H/W reset The PMCAn and their registers are initialized by the following reset signal:

Table 22-16 PMCAn reset signal

PMCAn	Reset signal
PMCA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the PMCAn are listed in the following table.

Table 22-17 PMCAn I/O signals

PMCA signal	Function	Connected to
PMCA0:		
PMCAMSEL[2:0]	External multiplexer address select signals	Port PMCA0MSEL[2:0]

The PMCAn is connected to various TAUAn, TAUBn, and TAUCn signals and also interface to the ADCAn.

Internal signals The internal signal connections of PMCAn are listed in the following table.

Table 22-18 PMCAn internal signals

PMCA signal	Function	Connected to
PMCA0:		
PMCACUP	Count-up signal input	Count-up signal input selector, refer to section *"Count-up signal input selector"* below
PMCATRGIN0 to PMCATRGIN7	Trigger generators inputs	Trigger generators input signal selector, refer to *"Trigger generators input signal selectors"* below
PMCAADTR0	A/D Converter group conversion trigger signal 0	ADCA0 H/W trigger expansion PMCAADTR0[a]
PMCAADTR1	A/D Converter group conversion trigger signal 1	ADCA0 H/W trigger expansion PMCAADTR1[a]
PMCAADTR2	A/D Converter group conversion trigger signal 2	ADCA0 H/W trigger expansion PMCAADTR2[a]
PMCAADTR3	A/D Converter group conversion trigger signal 3	ADCA1 H/W trigger expansion PMCAADTR3[a]
PMCAADTR4	A/D Converter group conversion trigger signal 4	ADCA1 H/W trigger expansion PMCAADTR4[a]
PMCAADTR5	A/D Converter group conversion trigger signal 5	ADCA1 H/W trigger expansion PMCAADTR5[a]
PMCAIAD	A/D Converter conversion complete feedback signal 6	A/D Converter conversion completion signal selector, refer to section *"A/D Converter conversion completion signal selector"* below
PMCACKDIV	Clock divider control	PWM Delay DLYA PCMACKDIV signal

a) These signals can be used as a trigger source to start the A/D Converter. Refer to the section *"H/W Trigger Expansion"* in the chapter *"A/D Converter (ADCA)"*.

22.3.2 Connections

The following diagram summarizes the PMCA0 connections.

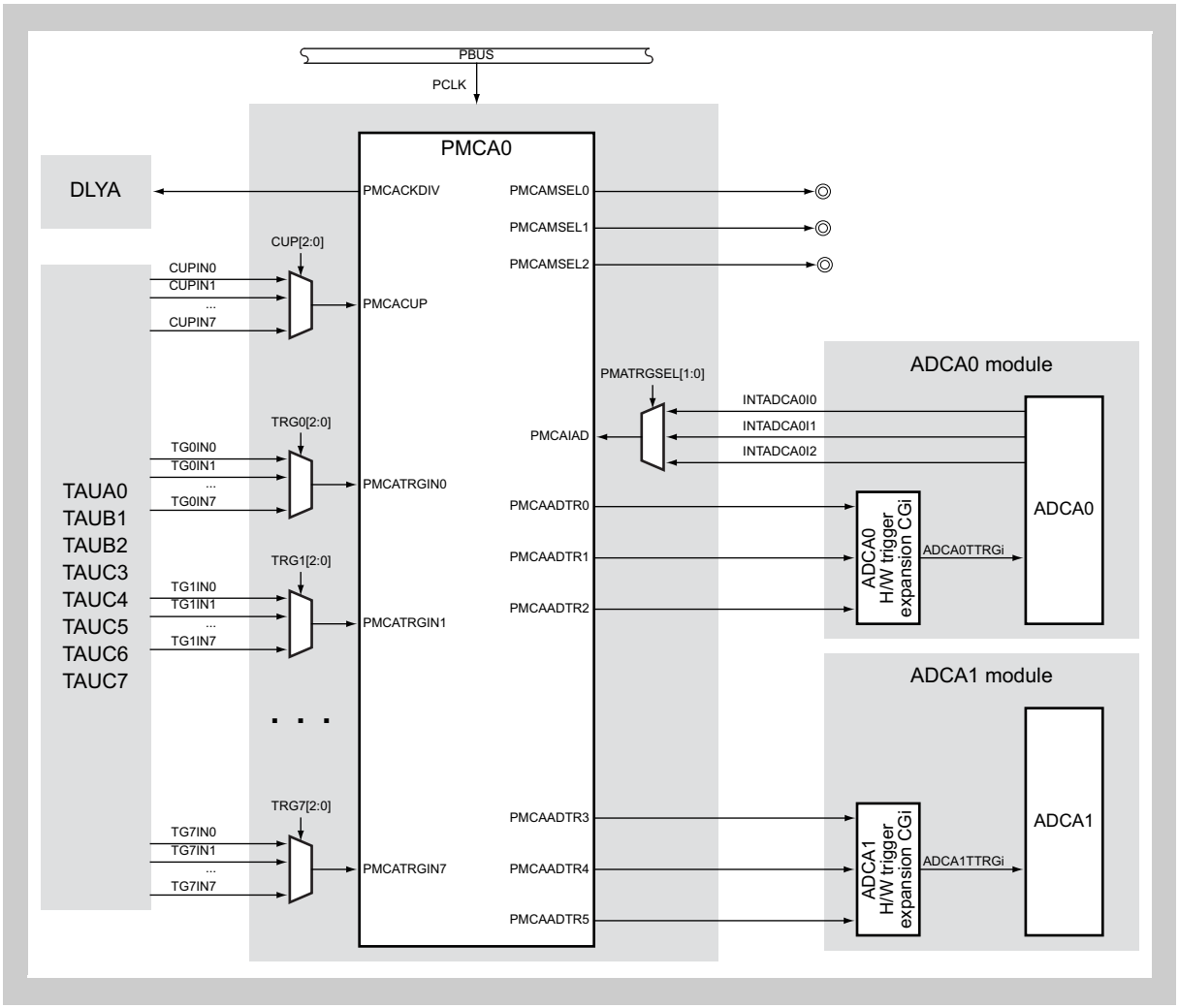


Figure 22-6 PMCA0 connection scheme

Various selectors are provided to select signals for the

- trigger generators inputs PMCATRGIN0 to PMCATRGIN7
- count-up signal PMCACUP
- A/D Converter conversion completion signal PMCAIAD

The following sections describe the selector's control registers.

(1) Trigger generators input signal selectors

The input signals PMCATRGIN0 to PMCATRGIN07 of the trigger generators can be selected from various TAUA, TAUB and TAUC sources by the trigger input selector registers TRGSEL0 to TRGSEL3.

(a) TRGSEL0 - Trigger input selection register 0

This register controls the setting of the PMCATRGIN0 and PMCATRGIN1 input selector to choose the desired timer signal for generation of the A/D Converter trigger signals.

Access This register can be written in 8-bit units.

Address FF77 1020_H

Initial Value Reading this register returns always 00_H.

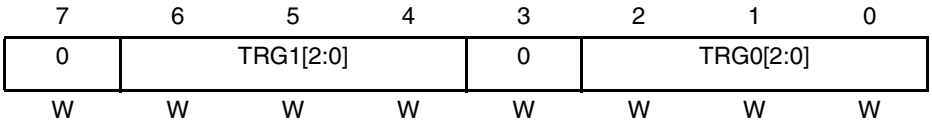


Table 22-19 TRGSEL0 register contents

Bit position	Bit name	Function
6 to 4	TRG1[2:0]	PCMATRGIN1 input selection: 000_B: TAUA0TTOUT7 001_B: TAUB1TTOUT7 010_B: TAUB2TTOUT7 011_B: TAUC3TTOUT7 100_B: TAUC4TTOUT7 101_B: TAUC5TTOUT7 110_B: TAUC6TTOUT7 111_B: TAUC7TTOUT7
2 to 0	TRG0[2:0]	PCMATRGIN0 input selection: 000_B: TAUA0TTOUT3 001_B: TAUB1TTOUT3 010_B: TAUB2TTOUT3 011_B: TAUC3TTOUT3 100_B: TAUC4TTOUT3 101_B: TAUC5TTOUT3 110_B: TAUC6TTOUT3 111_B: TAUC7TTOUT3

(b) TRGSEL1 - Trigger input selection register 1

This register controls the setting of the PMCATRGIN2 and PMCATRGIN3 input selector to choose the desired timer signal for generation of the A/D Converter trigger signals.

Access This register can be written in 8-bit units.

Address FF77 1024_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	TRG3[2:0]			0	TRG2[2:0]		
W	W	W	W	W	W	W	W

Table 22-20 TRGSEL1 register contents

Bit position	Bit name	Function
6 to 4	TRG3[2:0]	PCMATRGIN3 input selection: 000_B: TAUA0TTOUT11 001_B: TAUB1TTOUT11 010_B: TAUB2TTOUT11 011_B: TAUC3TTOUT11 100_B: TAUA0TTOUT15 101_B: TAUB1TTOUT15 110_B: TAUC2TTOUT15 111_B: TAUC3TTOUT15
2 to 0	TRG2[2:0]	PCMATRGIN2 input selection: 000_B: TAUA0TTOUT3 001_B: TAUB1TTOUT3 010_B: TAUB2TTOUT3 011_B: TAUC3TTOUT3 100_B: TAUA0TTOUT7 101_B: TAUB1TTOUT7 110_B: TAUC2TTOUT7 111_B: TAUC3TTOUT7

(c) TRGSEL2 - Trigger input selection register 2

This register controls the setting of the PMCATRGIN4 and PMCATRGIN5 input selector to choose the desired timer signal for generation of the A/D Converter trigger signals.

Access This register can be written in 8-bit units.

Address FF77 1028_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	TRG5[2:0]			0	TRG4[2:0]		
W	W	W	W	W	W	W	W

Table 22-21 TRGSEL2 register contents

Bit position	Bit name	Function
6 to 4	TRG5[2:0]	PCMATRGIN5 input selection: 000_B: TAUA0TTOUT7 001_B: TAUB1TTOUT7 010_B: TAUB2TTOUT7 011_B: TAUC3TTOUT7 100_B: TAUC4TTOUT7 101_B: TAUC5TTOUT7 110_B: TAUC6TTOUT7 111_B: TAUC7TTOUT7
2 to 0	TRG4[2:0]	PCMATRGIN4 input selection: 000_B: TAUA0TTOUT11 001_B: TAUB1TTOUT11 010_B: TAUB2TTOUT11 011_B: TAUC3TTOUT11 100_B: TAUA0TTOUT15 101_B: TAUB1TTOUT15 110_B: TAUC2TTOUT15 111_B: TAUC3TTOUT15

(d) TRGSEL3 - Trigger input selection register 3

This register controls the setting of the PMCATRGIN6 and PMCATRGIN7 input selector to choose the desired timer signal for generation of the A/D Converter trigger signals.

Access This register can be written in 8-bit units.

Address FF77 102C$_H$

Initial Value Reading this register returns always 00$_H$.

7	6	5	4	3	2	1	0
0	TRG7[2:0]			0	TRG6[2:0]		
W	W	W	W	W	W	W	W

Table 22-22 TRGSEL3 register contents

Bit position	Bit name	Function
6 to 4	TRG7[2:0]	PCMATRGIN7 input selection: 000$_B$: TAUA0TTOUT3 001$_B$: TAUB1TTOUT3 010$_B$: TAUB2TTOUT3 011$_B$: TAUC3TTOUT3 100$_B$: TAUA0TTOUT7 101$_B$: TAUB1TTOUT7 110$_B$: TAUC2TTOUT7 111$_B$: TAUC3TTOUT7
2 to 0	TRG6[2:0]	PCMATRGIN6 input selection: 000$_B$: TAUA0TTOUT3 001$_B$: TAUB1TTOUT3 010$_B$: TAUB2TTOUT3 011$_B$: TAUC3TTOUT3 100$_B$: TAUC4TTOUT3 101$_B$: TAUC5TTOUT3 110$_B$: TAUC6TTOUT3 111$_B$: TAUC7TTOUT3

(2) Count-up signal input selector

The count-up signal PMCACUP can be selected by the TRGSEL4 register.

(a) TRGSEL4 - PMCACUP input selection register

This register controls the setting of the PMCACUP input selector.

Access This register can be written in 8-bit units.

Address FF77 1030_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0					CUP[2:0]		
W	W	W	W	W	W	W	W

Table 22-23 TRGSEL4 register contents

Bit position	Bit name	Function
2 to 0	CUP[2:0]	PMCACUP input selection: 000_B: TAUA0TTOUT3 001_B: TAUB1TTOUT3 010_B: TAUB2TTOUT3 011_B: TAUC3TTOUT3 100_B: TAUA0TTOUT7 101_B: TAUB1TTOUT7 110_B: TAUC2TTOUT7 111_B: TAUC3TTOUT7

(3) A/D Converter conversion completion signal selector

The feedback signal PMCAIAD for the end of the A/D Converter conversion can be selected by the PMATRGSL register.

(a) PMATRGSL - PMCAIAD input selection register

This register controls ADCA0 end of conversion interrupt selection.

Access This register can be read/written in 8-bit units.

Address $FF771038_H$

Initial Value 00_H

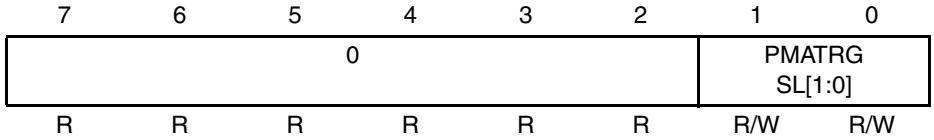


Table 22-24 PMATRGSL register contents

Bit position	Bit name	Function
1 to 0	PMATRG SL[1:0]	PMCAIAD input selection: 00_B: ADCA0 ADCATINT0 01_B: ADCA0 ADCATINT1 $1x_B$: ADCA0 ADCATINT2

22.3.3 Functional Overview

Features summary

The PWM Diagnostic module (PMCA) has the following features:

- ADC trigger control of up to 6 groups of PWM signals
- Address signal control for an external multiplexer for up to three synchronous PWM groups.

(1) Terms

In this chapter, the following terms are used:

PWM signal

A PWM signal defined by the period and the duty cycle.
The period describes the time between two rising edges of the PWM signal.
The value is equivalent to 1/f of the signal (f = signal frequency).
The duty cycle describes the portion of time with high signal level relative to the total time of the period.
The PWM signals used for automatic diagnostic must start with a high level.

PWM group

PWM signals with the same period.

Channel -to-channel delay

The duration between the rising edges of two channels.

(2) Block diagram

The block diagram shows the main components of the PWM Diagnostic module.

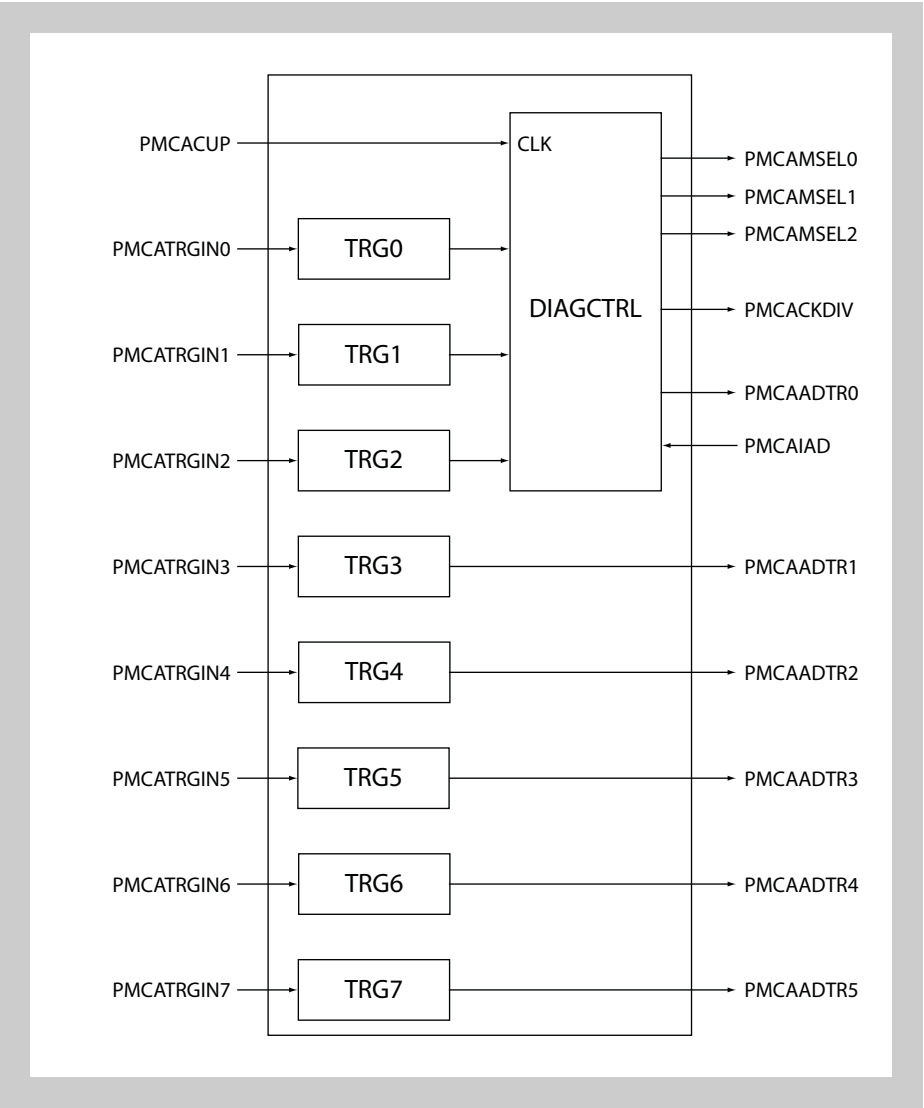


Figure 22-7 Block diagram of the PWM Diagnostic module

Trigger generators The trigger generators PMCATRGIN7-0 generate the trigger events for the ADC conversion group based on the timing provided by their selected input signal.

Diagnostic control The DIAGCTRL block provides the timing information and generates the related signals to switch the external multiplexers and to trigger the connected ADC trigger groups.

PMCACUP The PMCACUP signal provides the period of the slowest PWM frequency to be used in the automatic diagnostic function.

PMCAMSEL2-0 The 3 output signals PMCAMSEL2-0 provide the address signals for external 8:1 signal multiplexers. The external multiplexers can be used to reduce the number of necessary ADC channels for the automatic diagnostic function.

PMCAADTR5-0 The PMCAADTR5-0 signals provide the trigger information for the ADC unit synchronous to the timing of the different PWM groups.

PMCAIAD The PMCAIAD signal provides feedback information from the related ADC unit of when the conversion triggered by PMCAADTR0 is finished.

22.3.4 Registers

This section contains a description of all registers of the PMCAn Diagnostic module.

(1) PWM Diagnostic registers overview

The PWM Diagnostic modules are controlled and operated by the following registers:

Table 22-25 PMCA register overview

Register Name	Shortcut	Address
Control registers:		
Control register 0	PMCAnCTL0	<PMCAn_base> + 00_H
Control register 1	PMCAnCTL1	<PMCAn_base> + 04_H
Control register 2	PMCAnCTL2	<PMCAn_base> + 28_H
Status registers:		
Status register	PMCAnSTR	<PMCAn_base> + $2C_H$
Compare registers:		
Compare register 0	PMCAnCMP0	<PMCAn_base> + 08_H
Compare register 1	PMCAnCMP1	<PMCAn_base> + $0C_H$
Compare register 2	PMCAnCMP2	<PMCAn_base> + 10_H
Compare register 3	PMCAnCMP3	<PMCAn_base> + 14_H
Compare register 4	PMCAnCMP4	<PMCAn_base> + 18_H
Compare register 5	PMCAnCMP5	<PMCAn_base> + $1C_H$
Compare register 6	PMCAnCMP6	<PMCAn_base> + 20_H
Compare register 7	PMCAnCMP7	<PMCAn_base> + 24_H

<PMCAn_base> The base addresses <PMCAn_base> of the PMCAn is defined in the first section of this chapter under the key word “Register addresses”.

(2) PMCAnCTL0 – PMCA control register 0

This register controls the operation of the Delay modules and the Multiplexer controller.

Access This register can be read/written in 8-bit units.

Address <PMCAn_base> + 00_H

Initial Value 00_H

7	6	5	4	3	2	1	0
PMCAnEN7	PMCAnEN6	PMCAnEN5	PMCAnEN4	PMCAnEN3	PMCAnEN2	PMCAnEN1	PMCAnEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22-26 PMCAnCTL0 register contents

Bit position	Bit name	Function
7 to 0	PMCAnENm	Enable/disable of the Delay_m module 0: Diagnostics for DiagBlockm is disabled. 1: Diagnostics for DiagBlockm is enabled.
2 to 0		Enable/disable of the external multiplexer address signals PMCAMSEL2-0 000: PMCAMSEL2-0 = 000_B, PMCAADTR0 = 0 Other: PMCAMSEL2-0 and PMCAADTR0 are in operation

Caution Setting any of the bits to '1' must be done only after initializing all other related registers of the PMCAn.

Note When changing the values of PMCAnCTL0.PMCAnEN2 to 0, the value becomes effective only upon a reload of the 3-bit counter.
When changing the values of PMCAnCTL0.PMCAnEN7 to 3, the value becomes effective immediately.

(3) PMCAnCTL1 – PMCA control register 1

This register controls the global divider of the operation clock of the PMCAn module.

Access This register can be read/written in 8-bit units.

Address <PMCAn_base> + 04_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	PMCAn DIV
R	R	R	R	R	R	R	R/W

Table 22-27 PMCAnCTL1 register contents

Bit position	Bit name	Function
0	PMCAnDIV	Division ratio of module operation clock: 0: PCLK/256 is selected as module operation clock. 1: PCLK/128 is selected as module operation clock.

Caution Setting this register is allowed only in case of PMCAnCTL0 = 0.

(4) PMCAnCMPm – PMCA comparison registers

These registers provide the timing value for ADC trigger signal generation.

They must be set to the same timing value as the channel-to-channel delay of the PWM group that the ADC trigger shall be generated for by the trigger block m (m = 0 to 7).

Access This register can be read/written in 16-bit units.

Address PMCAnCMP0: <PMCAn_base> + 08_H
PMCAnCMP1: <PMCAn_base> + $0C_H$
PMCAnCMP2: <PMCAn_base> + 10_H
PMCAnCMP3: <PMCAn_base> + 14_H
PMCAnCMP4: <PMCAn_base> + 18_H
PMCAnCMP5: <PMCAn_base> + $1C_H$
PMCAnCMP6: <PMCAn_base> + 20_H
PMCAnCMP7: <PMCAn_base> + 24_H

Initial Value 0000_H

15	14	13	12	11	10	9	8
0	0	0	0	PMCAnTP[11:8]			
R	R	R	R	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PMCAnTP[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 22-28 PMCAnCMPm register contents

Bit position	Bit name	Function
11 to 0	PMCAnTP [11:0]	Timing period. A value of 1 or larger must be set before setting PMCAnCTL0.PMCAnEN = 1.

Caution This register must not be changed while this macro is in operation (PMCAnCTL0.PMCAnEN = 1).

22.3.5 Procedures for Setup, Writing and Reading

The following subsections provide the procedures for PMCA setup and for using the PMCA module.

(1) Initial setting of the PMCAn module

When initializing the PMCAn module after a system reset the registers must be written in the following order:

4. Set the PMCAnCTL1 register
5. Set the PMCAnCMPm register
6. Set the PMCAnCTL0 register.
7. Input signals to PMCAnCUP and PMCAnTRGIN[7:0].

(2) Changing the PMCAnCTL0 register while diagnostics are ongoing (PMCAnCTL0 is ≠ 0)

When changing the value of the PMCAnCTL0 register while diagnostics are ongoing the registers must be written in the following order:

1. Set the new value to the PMCAnCTL0 register.
2. In case the bits PMCANCTL0.PMCAnEN2 to 0 are changed, the new value only becomes active after a reload of the 3-bit counter occurred. Read the PMCASTR register to check of the new value of the PMCAnCTL0 value became active.

(3) Stopping the diagnostic operation

To stop the diagnostic operation immediately, set PMCAnCTL0.PMCAnSTP = 1.

(4) Changing the PMCAnCTL0 register after the diagnostic operation has been stopped

In case the diagnostic operation has been stopped by setting the PMCAnCTL2.PMCAnSTP = 1, use the following procedure the re-enable the diagnostic operation again.

1. Stop all input signals to PMCAnCUP and PMCAnTRGIN[7:0]
2. Set the PMCAnCMPm register.
3. Set the PMCAnCTL0 register.
4. Input signals to PMCAnCUP and PMCAnTRGIN[7:0].

Chapter 23 Asynchronous Serial Interface E (URTE)

This chapter contains a generic description of the Asynchronous Serial Interface E.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

23.1 V850E2/Fx4-H URTEn Features

Instances

This microcontroller has following number of instances of the Asynchroneous Serial Interface E URTEn.

Table 23-1 Instances of URTE

Asynchroneous Serial Interface E	
Instance	12
Name	URTE0 to URTE11

Instances index n

Throughout this chapter, the instance of an Asynchroneous Serial Interface E is identified by the index "n" (n = 0 to 11), for example, URTEnCTL0 for the URTEn control register 0.

Register addresses

All URTEn register addresses are given as address offsets to the individual base address <URTEn_base>.
The <URTEn_base> address of each URTEn are listed in the following table:

Table 23-2 Register base addresses <URTEn_base>

URTEn instance	<URTEn_base> address
URTE0	FF5C 0000$_H$
URTE1	FF5D 0000$_H$
URTE2	FF5E 0000$_H$
URTE3	FF5F 0000$_H$
URTE4	FF60 0000$_H$
URTE5	FF61 0000$_H$
URTE6	FF62 0000$_H$
URTE7	FF63 0000$_H$
URTE8	FF64 0000$_H$
URTE9	FF65 0000$_H$
URTE10	FF66 0000$_H$
URTE11	FF67 0000$_H$

Clock supply All Asynchroneous Serial Interface E provide one clock input.

Table 23-3 URTEn clock supply

URTEn instance	URTEn clock	Connected to
URTE0	PCLK	Clock Controller CKSCLK_112
URTE1	PCLK	Clock Controller CKSCLK_112
URTE2	PCLK	Clock Controller CKSCLK_114
URTE3	PCLK	Clock Controller CKSCLK_114
URTE4	PCLK	Clock Controller CKSCLK_114
URTE5	PCLK	Clock Controller CKSCLK_114
URTE6	PCLK	Clock Controller CKSCLK_114
URTE7	PCLK	Clock Controller CKSCLK_114
URTE8	PCLK	Clock Controller CKSCLK_114
URTE9	PCLK	Clock Controller CKSCLK_114
URTE10	PCLK	Clock Controller CKSCLK_011
URTE11	PCLK	Clock Controller CKSCLK_011

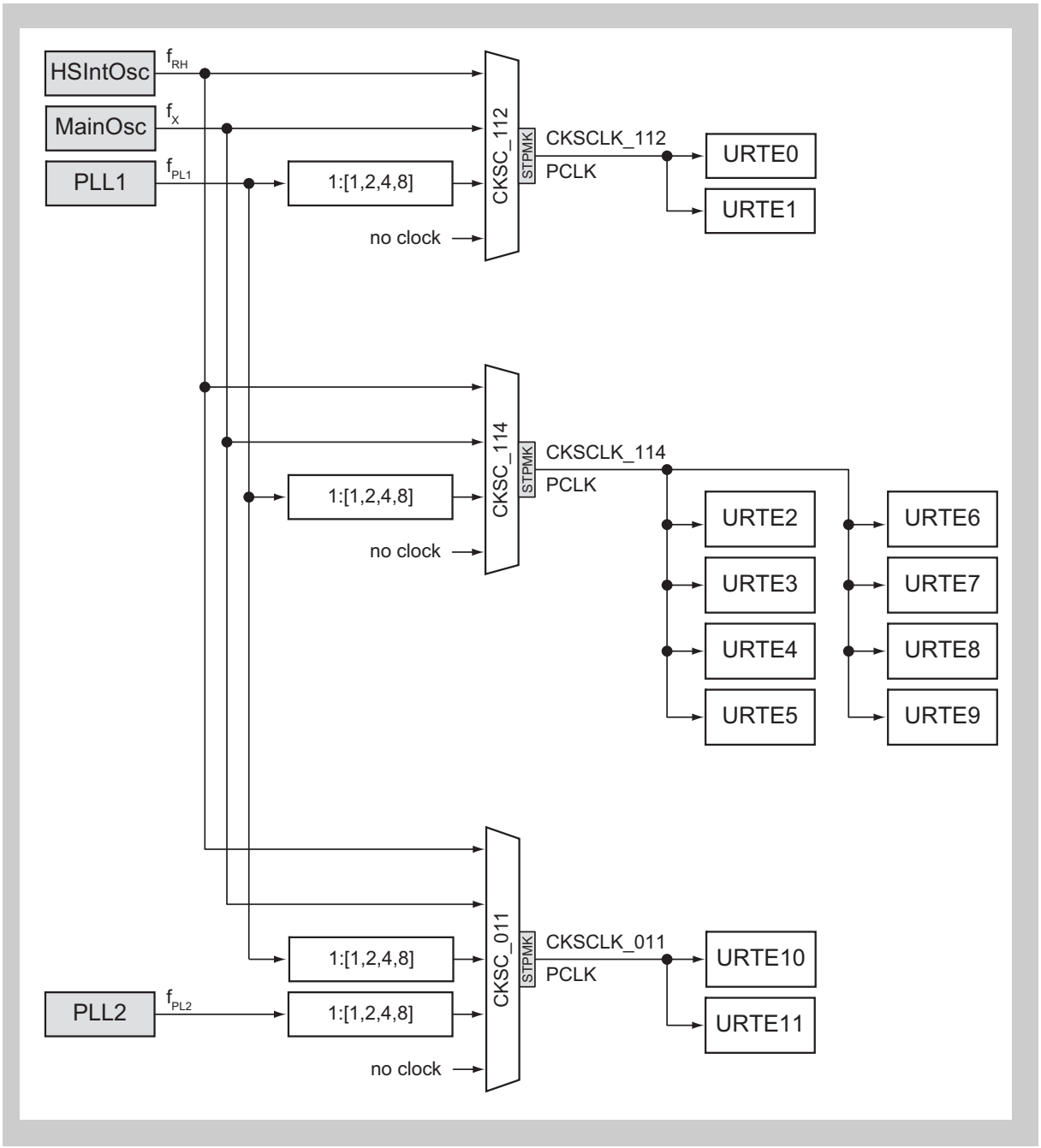


Figure 23-1 URTE clock supply

Interrupts

The Asynchroneous Serial Interface E can generate following interrupts requests, which are input to their respective LIN Master Controllers LMAn:

Table 23-4 URTEn interrupts requests

URTEn signals	Function	Connected to
INTUAEnTIT	Transmission interrupt	LMAn INTUAEnIT
INTUAEnTIR	Reception interrupt	LMAn INTUAEnIR
INTUAEnTIS	Status interrupt	LMAn INTUAEnIS

URTE H/W reset

The Asynchroneous Serial Interfaces E and their registers are initialized by the following reset signal:

Table 23-5 URTEn reset signal

URTEn	Reset signal
URTE2 to URTE9	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)
URTE10, URTE11	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

I/O signals

The I/O signals of the Asynchroneous Serial Interface E are listed in the table below.

Table 23-6 URTEn I/O signals

URTEn signals	Function	Connected to
URTEnTTXD	Transmit data output	Port URTEnTX
URTEnTRXD	Receive data input	Port URTEnRX[a]

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

Caution

The initial state of the receive data input signals noise filters block the input signals.
Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

Note that the Asynchroneous Serial Interfaces E feature additional noise filters. Refer to the section *"Digital receive data noise filter"* below for further details.

Baudrate measurement

Following URTEn receive data signals can be internally connected to a capture input of the Timer Array Units A.

Table 23-7 URTEn timer connections

URTEn signals	Function	Connected to
URTE10:		
Port URTE10RX	Receive data input	TAUA0 TAUA0TTIN0[a]
URTE11:		
Port URTE11RX	Receive data input	TAUA0 TAUA0TTIN1[a]

a) Refer to section Chapter 15 “TAUA Input Selection” on page 714 for details.

23.2 Functional Overview

- Full-duplex communication:
  - Internal URTEn receive data register n (URTEnRX)
  - Internal URTEn transmit data register n (URTEnTX)
- 2-pin configuration:
  - URTEnTTXD: Transmit data output pin
  - URTEnTRXD: Receive data input pin
- Reception error and status output function
  - Parity error
  - Framing error
  - Overrun error
  - Data consistency error
  - BF receive error
- Interrupt sources: 3
  - Transmission interrupt INTUAEnTIT
  - Reception interrupt INTUAEnTIR
  - Status interrupt INTUAEnTIS
- Character length: 7, 8 bits
- Parity function: odd, even, 0, none
- Transmission stop bit: 1, 2 bits
- MSB-/LSB-first transfer selectable
- Transmit/receive data inverted input/output possible
- 13 to 20 bits selectable for the BF (Break Field) in the LIN (Local Interconnect Network) communication format
  - Recognition of 11 bits or more possible for BF reception in LIN communication format
  - BF reception flag provided
- BF reception can be detected during data communication
- Bus monitor function to keep data consistency of the transmit data

23.3 Configuration

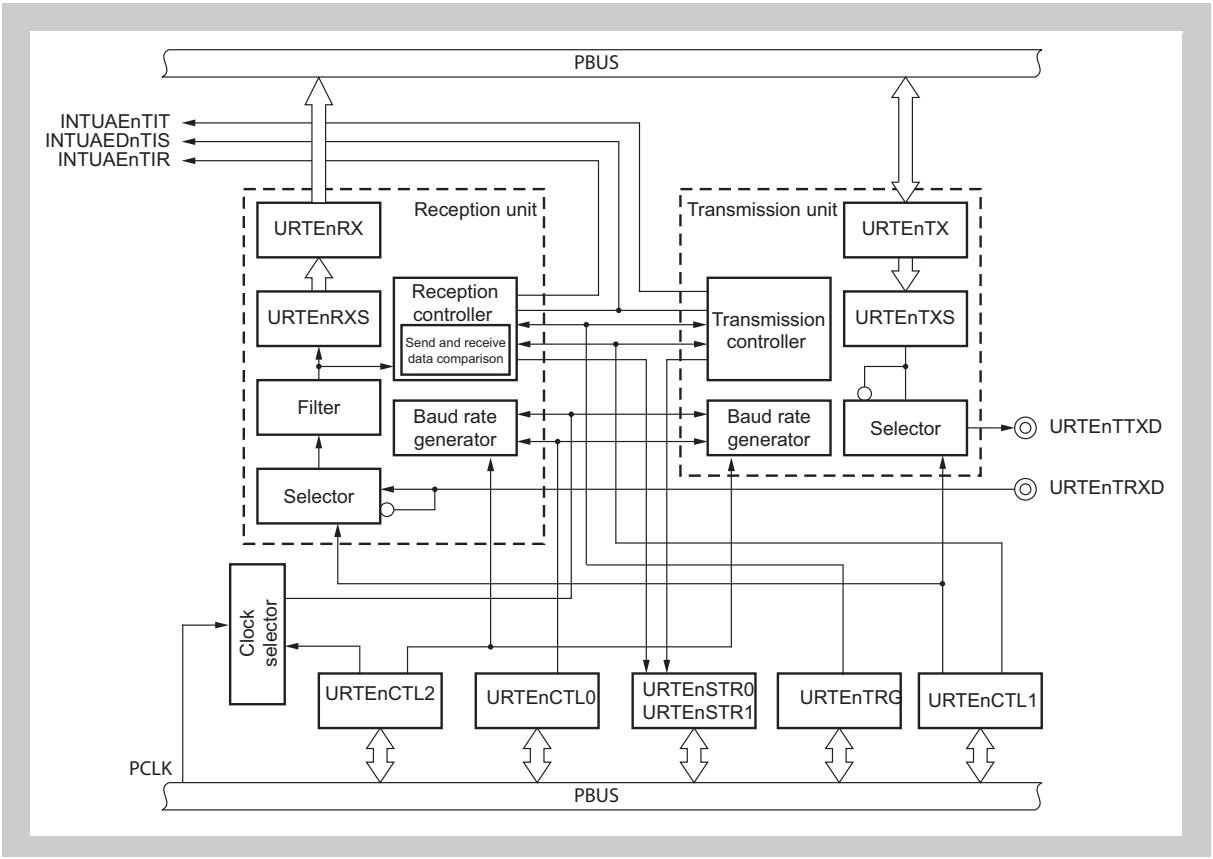


Figure 23-2 Block diagram of Asynchronous Serial Interface URTEn

23.4 URTE Registers

The URTE is controlled and operated by means of the following registers:

Table 23-8 URTEn registers

Register function	Name	Address
Control register 0	URTEnCTL0	<URTEn_base> + 00_H
Control register 1	URTEnCTL1	<URTEn_base> + 20_H
Control register 2	URTEnCTL2	<URTEn_base> + 24_H
Trigger register	URTEnTRG	<URTEn_base> + 04_H
Status register 0	URTEnSTR0	<URTEn_base> + 08_H
Status register 1	URTEnSTR1	<URTEn_base> + $0C_H$
Status clear register	URTEnSTC	<URTEn_base> + 10_H
Receive data register	URTEnRX	<URTEn_base> + 14_H
Transmit data register	URTEnTX	<URTEn_base> + 18_H
Emulation register	URTEnEMU	<URTEn_base> + 34_H

<URTEn_base> The base addresses <URTEn_base> of the URTEn are defined in the first section of this chapter under the key word “Register addresses”.

Register access All registers are accessible in 32-bit width.
Writing to non-existing register bits is ignored, reading of these bits return always 0.

(1) URTEnCTL0 - URTEn control register 0

This register controls the URTEn basic serial transfer operation.

Access This register can be read/written in 8-bit units.

Address <URTEn_base> + 00_H

Initial Value 00_H

7	6	5	4	3	2	1	0
URTEn PW	URTEn TXE	URTEn RXE	0	0	0	0	URTEn SLDC
R/W	R/W	R/W	R	R	R	R	R/W

Table 23-9 URTEnCTL0 register contents

Bit position	Bit name	Function
7	URTEn PW	URTEn enable 0: Stop URTEn operation 1: Enable URTEn operation Changing this bit initializes all transmission and reception units.
6	URTEn TXE	Transmission operation enable 0: Disable transmission operation 1: Enable transmission operation • To start transmission, set URTEnPW to 1 and then set URTEnTXE to 1. To stop transmission clear URTEnTXE to 0 and then URTEnPW to 0. • To initialize the transmission unit, clear URTEnTXE to 0, wait for two cycles of the base clock, and then set URTEnTXE to 1 again.
5	URTEn RXE	Reception operation enable 0: Disable reception operation 1: Enable reception operation • To enable reception, set URTEnPW to 1 and then set URTEnRXE to 1. To disable reception clear URTEnRXE to 0 and then URTEnPW to 0. • To initialize the reception unit, clear URTEnRXE to 0, wait for two periods of the base clock, and then set URTEnRXE to 1 again. The reception is enabled after URTEnRXE is set to 1 and two cycles of base clock have passed. The rising edge detection of the RXDD pin is enabled after URTEnRXE is set to 1 and four cycles of the base clock have passed.
0	URTEn SLDC	Data consistency check enable 0: Disable data consistency check 1: Enable data consistency check This bit selects the handling of data consistency error checks when transmitting data. When this bit is set to 1, the transmit data and receive data are compared, and if a mismatch is detected, URTEnSTR1.URTEnDCE is set to 1 and a status interrupt request INTUAEnTIS is issued. This bit is accessed only when starting a transmission. Consequently, if this bit value is changed later on during transmission processing, the transmission processing continues to use the value set at the start of processing.

Cautions 1. In case URTEn is in the status of

- enabled reception and transmission (URTEnCTL0 bits URTEnPW = 1 & URTEnRXE & URTEnTXE = 1) and
- data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1) and
- transmission is ongoing or is completed

transmission shall be disabled, while keeping reception enabled, proceed as follows:

- check that no transmission is pending (URTEnSTR0 bits URTEnSSBT = URTEnSST = 0) and
- check that no reception is ongoing (URTEnSTR0 bits URTEnSSBR = URTEnSSR = 0) then
- disable transmission by URTEnCTL0.URTEnTXE = 0.

The reason for this procedure is that the data consistency error flag URTEnSTR1.URTEnDCE is cleared to 0 by URTEnCTL0.URTEnTXE. Thus a potential data consistency error would not be reported, if transmission is disabled during a data transfer or after its completion.

2. In case URTEn is in the status of
   - enabled reception and transmission (URTEnCTL0 bits URTEnPW = 1 & URTEnRXE & URTEnTXE = 1) and
   - data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1) and
   - transmission is ongoing or is completed

   reception shall be disabled, while keeping transmission enabled, proceed as follows:
   - check that no transmission is pending (URTEnSTR0 bits URTEnSSBT = URTEnSST = 0) and
   - check that no reception is ongoing (URTEnSTR0 bits URTEnSSBR = URTEnSSR = 0) then
   - disable reception by URTEnCTL0.URTEnRXE = 0.

   The reason for this procedure is that the data consistency error flag URTEnSTR1.URTEnDCE is invalid URTEnCTL0.URTEnTXE is cleared. Thus a potential data consistency error of already transmitted data would not be reported.
3. Don’t start any data transmission if
   - data consistency check is enabled (URTEnCTL0.URTEnSLDC = 1) and
   - BF reception is enabled (URTEnSTR0.URTEnSSBR = 1) and
   - detection of a BF during reception is not active (URTEnCTL1.URTEnSLBM = 0)

   A data consistency error will occur under above conditions when the BF reception is completed. The status interrupt INTUAEnTIS will be asserted and the BF reception completion will not be indicated (URTEnSTR1.URTEnBSF remains 0). Consequently the BF reception completion will not be recognized.

(2) URTEnCTL1 - URTEn control register 1

This register defines the data frame properties of the URTEn serial data transfers.

Access This register can be read/written in 16-bit units.

Address <URTEn_base> + 20_H

Initial Value 5002_H

15	14	13	12	11	10	9	8
URTEn SLBM	URTEn BLG[2:0]			0	0	0	URTEn CLG
R/W	R/W	R/W	R/W	R	R	R	R/W

7	6	5	4	3	2	1	0
URTEn SLP[1:0]		URTEn TDL	URTEn RDL	0	URTEn SLG	URTEn SLD	URTEn SLIT
R/W	R/W	R/W	R/W	R	R/W	R/W	R/W

Table 23-10 URTEnCTL1 register contents (1/3)

Bit position	Bit name	Function
15	URTEn SLBM	BF receive mode selection 0: BF reception during data reception disabled. 1: BF reception during data reception enabled. • When this bit is set to 1, data reception processing uses a character bit length of 8 bits and no parity bit, regardless of the values set to the URTEnCLG and URTEnSLP[1:0]. • Changing this bit is only allowed, if reception is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0).
14 to 12	URTEn BLG[2:0]	BF bit length during transmission URTEnBLG2 / URTEnBLG1 / URTEnBLG0 / BF length 1 / 0 / 1 / 13 bits 1 / 1 / 0 / 14 bits 1 / 1 / 1 / 15 bits 0 / 0 / 0 / 16 bits 0 / 0 / 1 / 17 bits 0 / 1 / 0 / 18 bits 0 / 1 / 1 / 19 bits 1 / 0 / 0 / 20 bits Changing these bits is only allowed, if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).
8	URTEn CLG	Receive/transmit data bit length 0: 7 bits 1: 8 bits • When the transmission/reception is performed in the LIN format, set URTEnCLG to 1. • Changing this bit is only allowed, if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).

Table 23-10 URTEnCTL1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>7 to 6</td><td>URTEn
SLP[1:0]</td><td>Parity bit selection
<table>
<tr><th rowspan="2">URTEn SLP1</th><th rowspan="2">URTEn SLP0</th><th colspan="2">Operation</th></tr>
<tr><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>Output without parity bit</td><td>Received with no parity</td></tr>
<tr><td>0</td><td>1</td><td>Output with space parity (0-fixed)</td><td>No parity judgment</td></tr>
<tr><td>1</td><td>0</td><td>Output with odd parity</td><td>Judged as odd parity</td></tr>
<tr><td>1</td><td>1</td><td>Output with even parity</td><td>Judged as even parity</td></tr>
</table>
• If “Reception with no parity judgment” is selected during reception, a parity check is not performed. Therefore, since the URTEnSTR1.UTREnPE bit is not set, no error interrupt is output.
• When the transmission/reception is performed in the LIN format, set URTEnSLP[1:0] to 00_B.
• Changing these bits is only allowed, if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>5</td><td>URTEn
TDL</td><td>Transmission data level control
0 No inverted output of transmit data
1 Inverted output of transmit data
• The output level of the URTEnTTXD pin can be inverted using this bit. It inverts the URTEnTTXD output level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnTXE. Therefore, if URTEnTDL is set to 1 while the operation is disabled, the URTEnTTXD outputs low level.
• Changing this bit is only allowed, if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).</td></tr>
<tr><td>4</td><td>URTEn
RDL</td><td>Reception data level control
0 No inverted output of receive data
1 Inverted output of receive data
• The output level of the URTEnTRXD pin can be inverted using this bit. It inverts the URTEnTRXD input level immediately, regardless of the values of URTEnCTL0.URTEnPW and URTEnCTL0.URTEnRXE. Therefore, if URTEnRDL is set to 1 while the operation is disabled, the URTEnTRXD inputs low level.
• Changing this bit is only allowed, if reception is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = 0).</td></tr>
</table>

Table 23-10 URTEnCTL1 register contents (3/3)

Bit position	Bit name	Function
2	URTEn SLG	Stop bit number selection for transmission data 0 1 bit 1 2 bits • The stop bit length during data or BF reception is always handled as "1". • Changing this bit is only allowed, if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).
1	URTEn SLD	Transfer direction selection 0 MSB-first transfer 1 LSB-first transfer • The stop bit length during data or BF reception is always handled as "1". • When the transmission/reception is performed in the LIN format, set URTEnSLD to 1. • Changing this bit is only allowed, if reception and transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnRXE = URTEnCTL0.URTEnTXE = 0).
0	URTEn SLIT	Transmission interrupt request (INTUAEnTIT) timing selection 0 INTUAEnTIT generated at the start of transmission, i.e. when the transmit data is stored to the transmission shift register 1 INTUAEnTIT generated at transmission completion • Changing this bit is only allowed, if transmission is disabled (URTEnCTL0.URTEnPW = 0 or URTEnCTL0.URTEnTXE = 0).

(3) URTEnCTL2 - URTEn control register 2

This register defines the baud rates of the URTEn serial data transfers.

Access This register can be read/written in 16-bit units.

Address <URTEn_base> + 24_H

Initial Value $EFFF_H$

15	14	13	12	11	10	9	8
URTEnPRS[2:0]			0	URTEnBRS[11:8]			
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
URTEnBRS[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-11 URTEnCTL2 register contents

Bit position	Bit name	Function
15 to 13	URTEn PRS[2:0]	Prescaler clock (PRSCLK) division value 0: PRSCLK = PCLK / 2^0 1: PRSCLK = PCLK / 2^1 2: PRSCLK = PCLK / 2^2 3: PRSCLK = PCLK / 2^3 4: PRSCLK = PCLK / 2^4 5: PRSCLK = PCLK / 2^5 6: PRSCLK = PCLK / 2^6 7: PRSCLK = PCLK / 2^7
11 to 0	URTEn BRS[11:0]	Baudrate clock (BRCLK) division value (PRSCLK = PCLK / $2^{URTEnPRS[2:0]}$):

URTEn BRS[11:0]	Transmit/receive BRCLK	BF receive clock
000_H	PRSCLK / (2 x 4)	PRSCLK / 4
001_H		
002_H		
003_H		
004_H		
005_H	PRSCLK / (2 x 5)	PRSCLK / 5
...	PRSCLK / (2 x URTEnBRS[11:0])	PRSCLK / URTEnBRS[11:0]
FFE_H	PRSCLK / (2 x 4094)	PRSCLK / 4094
FFF_H	PRSCLK / (2 x 4095)	PRSCLK / 4095

Caution Writing to this register is only allowed if the operation of URTEn is disabled (URTEnCTL0.URTEnPW = 0).

PCLK The value of the URTEn input clock is defined in the first section of this chapter under the key word “Clock supply”.

(4) URTEnTRG - URTEn trigger register

This register controls the URTEn transmission/reception trigger of BF.

Access This register can be read/written in 8-bit units.

Address <URTEn_base> + 04_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
0	URTEn BRT	URTEn BTT	0	0	0	0	0
R	R/W	R/W	R	R	R	R	R

Table 23-12 URTEnTRG register contents (1/2)

Bit position	Bit name	Function
6	URTEn BRT	BF reception trigger 0: read value is always 0, writing 0 is ignored 1: Enable BF reception • When reception is enabled, writing 1 to this bit enables BF reception (URTEnSTR0.URTEnSSBR = 1) and BF reception processing begins when the falling edge of the receive serial signal is detected. • If 1 is written to this bit during reception processing, the current reception processing is terminated. Consequently, the received data is not stored, the framing, parity and overflow error bits are not updated based on the data that was being received and no interrupts are generated. Meanwhile, the BF counter value is continuously being used. • After BF reception, the reception status is set according to the URTEnCTL1.URTEnSLBM setting. • Setting this bit to 1 is only allowed, if reception is enabled (URTEnCTL0.URTEnPW = URTEnCTL0.URTEnRXE = 1). After URTEnBRT is set to 1, completion of BF reception is reported by either of the following two methods, based on the URTEnCTL1.URTEnSLBM setting: • if URTEnCTL1.URTEnSLBM = 0 When BF reception is complete, a reception interrupt request INTUAEnTIR is output. • URTEnCTL1.URTEnSLBM When BF reception is complete, URTEnSTR1.URTEnBSF is set to 1 and a status interrupt request INTUAEnTIS is output.

Table 23-12 URTEnTRG register contents (2/2)

Bit position	Bit name	Function
5	URTEn BTT	BF transmission trigger 0: read value is always 0, writing 0 is ignored 1: Triggers BF transmission • When 1 is written to this bit while URTEnSTR0.URTEnSSBT = 0 and (URTETDCM = "H" or URTE0DCE = 0) transmission is enabled, a BF transmit request is set, and URTEnSSBT is set to 1. • When 1 is written to this bit during a data transmission, a BF is transmitted after the current transmission processing is completed. Even if 1 is written to this bit before the BF transmission is completed, a BF is transmitted only once. • When transmission is enabled (URTE0PW = URTE0TXE = 1), writing 1 to this bit clears all previously set data transmit requests (which have not been transmitted), leaving only BF transmit requests. If a write operation occurs in the URTE0TX7 to URTE0TX0 bits after writing 1 to this bit, data is transmitted after the BF is transmitted. • If both a BF transmit request and a data transmit request have been set when a transmission starts, the BF transmission takes priority. • When URTE0DCE = 1 and URTETDCM = "L", writing 1 to this bit is ignored. • When reception is enabled (URTEnCTL0.URTEnPW = URTEnCTL0.URTEnRXE = 1), writing 1 to this bit enables BF reception (URTEnSTR0.URTEnSSBR = 1) and BF reception processing begins when the falling edge of the receive serial signal is detected. • If 1 is written to this bit during reception processing, the current reception processing is terminated. Consequently, the received data is not stored, the framing, parity and overlow errors bits are not updated based on the data that was being received and no interrupts are generated. Meanwhile, the BF counter value is continuously being used. • After BF reception, the reception status is set according to the URTEnCTL1.URTEnSLBM setting. • When transmitting a BF while reception is enabled (URTEnCTL0 bits URTEnPW = URTEnRXE = 1) and BF shall not be detected during reception (URTEnCTL1.URTEnSLBM = 0), write 1 to URTEnBTT after 1 is written to URTEnBRT, then transmit the BF. If URTEnBTT is set to 1 first, a framing error may occur. • Setting this bit to 1 is only allowed, if transmission is enabled (URTEnCTL0.URTEnPW = URTEnCTL0.URTEnTXE = 1).

(5) URTEnSTR0 - URTEn status register 0

This register indicates the current status of serial data transmissions.

Access This register can be read in 8-bit units.
In case the URTEn operation is disabled (URTEnCTL0.URTEnPW = 0), this register can also be written. If the URTEn is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the register takes on its initial value.

Address <URTEn_base> + 08_H

Initial Value 0000_H. This register is initialized by any reset and when URTEn operation is disabled by URTEnCTL0.URTEnPW = 0..

7	6	5	4	3	2	1	0
0	URTEn SSBR[a]	URTEn SSBT[b]	0	0	0	URTEn SSR[b]	URTEn SST[b]
R	R	R	R	R	R	R	R

a) This bit is also initialized if reception is disabled by URTEnCTL0.URTEnRXE = 0.

b) These bits are also initialized if transmission is disabled by URTEnCTL0.URTEnTXE = 0.

Table 23-13 URTEnSTR0 register contents

Bit position	Bit name	Function
6	URTEn SSBR	BF reception enable status 0: BF reception disabled 1: BF reception has been enabled by setting URTEnTRG.URTEnBRT to 1 (BF reception standby mode or BF reception busy).
5	URTEn SSBT	BF transmission enable status 0: BF transmission disabled 1: BF transmission has been enabled by setting URTEnTRG.URTEnBTT to 1 (BF transmission standby mode or BF transmission busy).
1	URTEn SSR	Data reception status 0: no data reception ongoing 1: data reception ongoing (data reception busy)
0	URTEn SST	Data transmission status 0: no transmission pending or ongoing 1: data in URTEnTX[7:0] pending to be transmitted or trasmission ongoing

(6) URTEnSTR1 - URTEn status register 1

This register indicates results of serial data transmisssions.

Access This register can be read in 8-bit units.
In case the URTEn operation is disabled (URTEnCTL0.URTEnPW = 0), this register can also be written. If the URTEn is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the register takes on its initial value.

Address <URTEn_base> + $0C_H$

Initial Value 0000_H. This register is initialized by any reset and when URTEn operation is disabled by URTEnCTL0.URTEnPW = 0..

7	6	5	4	3	2	1	0
0	0	0	URTEn BSF[a]	URTEn DCE[b]	URTEn PE[a]	URTEn FE[a]	URTEn OVE[a]
R	R	R	R	R	R	R	R

a) These bits are also initialized if reception is disabled by URTEnCTL0.URTEnRXE = 0.

b) This bit is also initialized if transmission is disabled by URTEnCTL0.URTEnTXE = 0.

Table 23-14 URTEnSTR1 register contents (1/2)

Bit position	Bit name	Function
4	URTEn BSF	BF reception successful flag 0: no valid BF frame received 1: BF reception completed successfully When the BF receive mode selection bit is set in LIN communication mode, it is necessary to read this bit by the status interrupt processing and to confirm the beginning of a new frame slot. URTEnBSF is cleared by • URTEnCTL0.URTEPW = 1 • URTEnCTL0.URTERXE = 0 • URTEnCTL1.URTESLBM = 0 • URTEnSTC.URTEnCLBS = 1
3	URTEn DCE	Data consistency error flag 0: no data consistency check error detected 1: data consistency check error detected URTEnDCE is cleared by • URTEnCTL0.URTEPW = 0 • URTEnCTL0.URTETXE = 0 • URTEnCTL1.URTESLDC = 0 • URTEnSTC.URTEnCLDC = 1
2	URTEn PE	Parity error flag 0: no parity error in received data detected 1: parity error in received data detected The operation of URTEnPE is controlled by the settings of the URTEn.URTEnSLP[1:0]. URTEnPE is cleared by • URTEnCTL0.URTEPW = 0 • URTEnCTL0.URTERXE = 0 • URTEnSTC.URTEnCLP = 1

Table 23-14 URTEnSTR1 register contents (2/2)

Bit position	Bit name	Function
1	URTEn FE	Framing error flag 0: cleared by - URTEnCTL0.URTEPW = 0 - URTEnCTL0.URTERXE = 0 - URTEnSTC.URTEnCLF = 1 1: framing error in received data detected
0	URTEn OVE	Overrun error flag 0: cleared by - URTEnCTL0.URTEPW = 0 - URTEnCTL0.URTERXE = 0 - URTEnSTC.URTEnCLOV = 1 1: overrun error in received data detected When an overrun error occurs, the data is discarded without the next receive data being written to the receive data register URTEnRX.

Note If the bits of these registers are set (1) and cleared (0) at the same time, setting takes priority over clearing.

For further information concerning error detections, refer to the sections *"Transmission data consistency check"* and *"Reception errors"*.

Caution In case reception and transmission is enabled and a data consistency check error occurs (URTEnSTR1.URTEnDCE = 1), follow the procedure below prior next data transmission:

- disable transmission by URTEnCTL0.URTEnTXE = 0
- initiate a transmission by URTEnTRG.URTEnBTT = 1 (BT transmission trigger) or writing any data to URTEnTX
- enable transmission by URTEnCTL0.URTEnTXE = 1

Afterwards new transmissions can be started.

(7) URTEnSTC - URTEn status clear register

This register is used to clear the status bits of the status register 1 URTEnSTR1.

Access This register can be read/written in 8-bit units.
Reading this register returns always 00_H.

Address <URTEn_base> $+10_H$

Initial Value 0000_H.

7	6	5	4	3	2	1	0
0	0	0	URTEn CLBS	URTEn CLDC	URTEn CLP	URTEn CLF	URTEn CLOV
R	R	R	R/W	R/W	R/W	R/W	R/W

Table 23-15 URTEnSTC register contents

Bit position	Bit name	Function
4	URTEn CLBS	Clear BF reception successful flag 0: writing 0 is ignored 1: writing 1 clears URTEnSTR1.URTEnBSF
3	URTEn CLDC	Clear data consistency error flag 0: writing 0 is ignored 1: writing 1 clears URTEnSTR1.URTEnDCE In case URTEnDCE will be cleared by setting URTEnCLDC = 1, any pending data or BF transmit requests will be ignored.
2	URTEn CLP	Clear parity error flag 0: writing 0 is ignored 1: writing 1 clears URTEnSTR1.URTEnPE
1	URTEn CLF	Clear framing error flag 0: writing 0 is ignored 1: writing 1 clears URTEnSTR1.URTEnFE
0	URTEn CLOV	Clear overrun error flag 0: writing 0 is ignored 1: writing 1 clears URTEnSTR1.URTEnOVE

(8) URTEnRX - URTEn receive data register

This register stores received data.

The data stored in the receive shift register is transferred to URTEnRX upon completion of reception of 1 byte of data.

7-bit transfers If the data length has been specified as 7 bits (URTEnCTL1.URTEnCLG = 0) and

- LSB-first reception (URTEnCTL1.URTEnSLD = 1),
  the receive data is transferred to URTEnRX[6:0] and the MSB URTEnRX[7] always becomes 0.
- MSB-first reception (URTEnCTL1.URTEnSLD = 0),
  the receive data is transferred to URTEnRX[7:1] and the LSB URTEnRX[0] always becomes 0.

For further information on data formats, refer to the section *"Data formats"*.

Overrun error When an overrun error (URTEnSTR1.URTEnOVE = 1) occurs, the receive data at this time is not transferred to URTEnRX and is discarded.

When reception processing ends and data reception is confirmed without any overrun errors, the received data is stored to URTEnRX according to the specified storage format.
Access

This register can be read in 8-bit units.
In case the URTEn operation is disabled (URTEnCTL0.URTEnPW = 0), this register can also be written. If the URTEn is enabled (URTEnCTL0.URTEnPW = 1), any written values are disregarded and the register takes on its initial value.

Access This register can be read in 8-bit units.

Address <URTEn_base> + 14_H

Initial Value FF_H. This register is initialized by any reset and when URTEn operation is enabled by URTEnCTL0.URTEnPW = 1.

7	6	5	4	3	2	1	0
URTEnRX[7:0]							
R	R	R	R	R	R	R	R

Table 23-16 URTEnRx register contents

Bit position	Bit name	Function
7 to 0	URTEn RX[7:0]	URTEn receive data

(9) URTEnTX - URTEn transmit data register

This register is used to stores data to be transmitted.

Transmit data in URTEnTX is stored to the transmission shift register URTEnTXS according to the specified transmit data format.

7-bit transfers If the data length has been specified as 7 bits (URTEnCTL1.URTEnCLG = 0) and

- LSB-first transmission (URTEnCTL1.URTEnSLD = 1), URTEnTX[6:0] with MSB URTEnTX[7] always “0” is transferred to the shift register URTEnTXS.
- MSB-first reception (URTEnCTL1.URTEnSLD = 0), URTEnTX[7:1] with LSB URTEnTX[0] always “0” is transferred to the shift register URTEnTXS.

For further information on data formats, refer to the section *“Data formats”*.

Access This register can be read/written in 8-bit units.

Address <URTEn_base> + 18_H

Initial Value FF_H

7	6	5	4	3	2	1	0
URTEnTX[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-17 URTEnTx register contents

Bit position	Bit name	Function
7 to 0	URTEn TX[7:0]	URTEn transmit data

When transmission is enabled (URTEnCTL0 bits URTEnPW = URTETXE = 1), a write to URTEnTX triggers the start of the transmission.

Note If the the next data to URTEnTX is written before the ongoing transmission is completed, the continuous transmission is enabled.

(10) URTEnEMU - URTEn emulation register

This register controls whether the URTEn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <URTEn_base> + 34_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
URTEn SVSDIS							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 23-18 URTEnEMU register contents

Bit position	Bit name	Function
7	URTEn SVSDIS	Emulation control 0: URTEn can be stopped during emulation 1: URTEn continuous operating during emulation

23.5 Interrupt Request Signals

The following four interrupt request signals are generated by URTEn.

- Transmission interrupt request INTUAEnTIT
- Reception interrupt request INTUAEnTIR
- Status interrupt request INTUAEnTIS

23.5.1 Transmission interrupt request INTUAEnTIT

If transmit data is transferred from the URTEnTX register to transmit shift register with transmission enabled, the transmission interrupt request INTUAEnTIT is generated.

The condition for generation of a transmit interrupt request depends on the setting of the URTEnCTL1.URTEnSLIT:

- at start of transmission process: URTEnCTL1.URTEnSLIT = 0

A transmission interrupt request is issued when starting transmission of the first bit (this is the start bit for data transmission or the first BF bit for BF transmission). During data transmission, a transmission interrupt request is issued when transmit data in URTEnTX is transferred to the transmission shift register.

- at end of transmission process: URTEnCTL1.URTEnSLIT = 1

A transmission interrupt request is issued after completing transmission of the last bit (the first bit of the stop bit when the stop bit length is 1, or the second bit of the stop bit when the stop bit length is 2).

Note If a data consistency error is detecting, this interrupt is not generated.

The following diagrams show the timing of the transmission interrupt request for both cases.

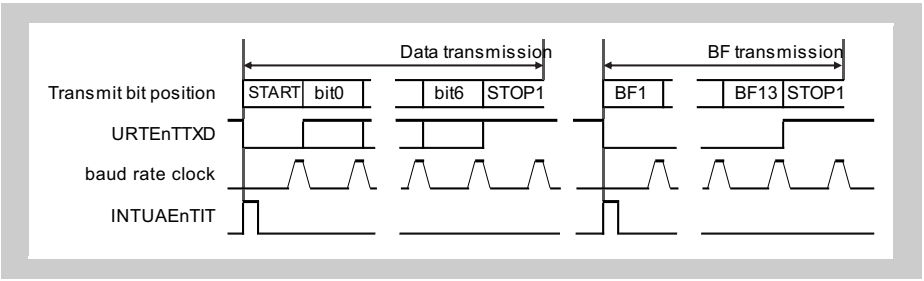


Figure 23-3 Transmission interrupt request timing for URTEnCTL1.URTEnSLIT = 0

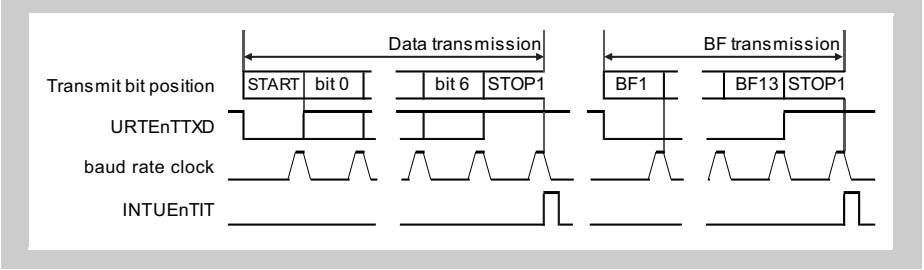


Figure 23-4 Transmission interrupt request timing for URTEnCTL1.URTEnSLIT = 1

23.5.2 Reception interrupt request INTUAEnTIR

In case of erroneous reception, the status interrupt INTUAEnTIS is generated instead of INTUAEnTIR.

No reception interrupt request INTUAEnTIR is generated in the reception disabled status.

A reception interrupt request is issued when the first bit of the stop bit is sampled (at the end of reception processing).

The following diagrams show the timing of the reception interrupt request during data/BF reception.

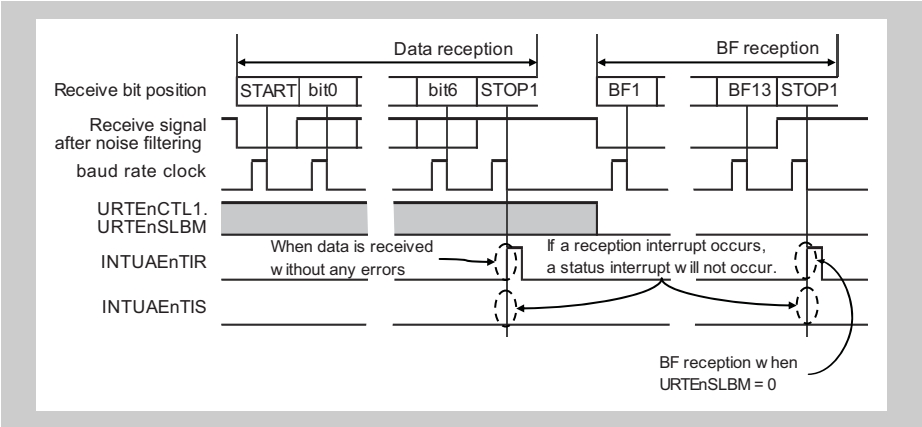


Figure 23-5 Reception interrupt request timing

23.5.3 Status interrupt request INTUAEnTIS

A status interrupt request is generated if an error condition occurred during reception or transmission, as reflected in the status register 1 URTEnSTR1.

When the BF reception mode selection bit is set in LIN communication mode (URTEnCTL1.URTEnSLBM = 1), the status interrupt request signal is generated when a consecutive low level (BF) of 11 bits or more is received.

23.6 Operation

23.6.1 Data formats

Full-duplex serial data reception and transmission is performed.

As shown in the figures below, one data frame of transmit/receive data consists of a start bit, character bits, parity bit, and stop bit(s).

Several properties of a transmit/receive data frame can be specified by control bits of the URTEnCTL1 register:

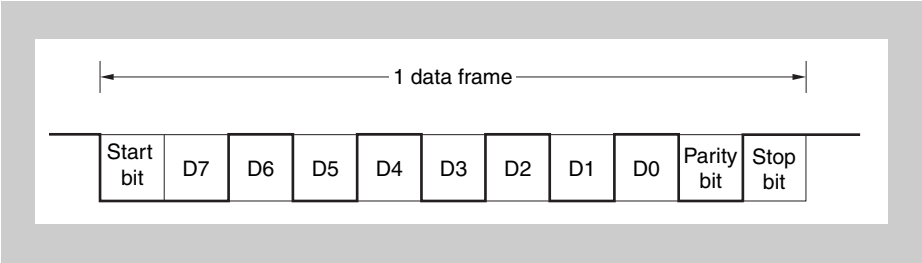
Table 23-19 Data format specification

Item	Options	Control bits
Start bit	1 bit	fixed
Character bits	7 bits / 8 bits	URTEnCTL1.URTEnCLG
Parity	Even parity/odd parity/0	URTEnCTL1.URTEnSLP[1:0]
Stop bit	1 bit / 2 bits	URTEnCTL1.URTEnSLG
Data order	MSB first / LSB first	URTEnCTL1.URTEnSLD
Tx data level	inverted / not inverted	URTEnCTL1.URTEnSLG
Rx data level	inverted / not inverted	URTEnCTL1.URTEnRLG

(1) URTEn transmit/receive data format

(a) 8-bit data length, LSB first, even parity, 1 stop bit, transfer data: 55_H

(b) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55_H



(c) 8-bit data length, MSB first, even parity, 1 stop bit, transfer data: 55_H, URTEnTTXD inversion

(d) 7-bit data length, LSB first, odd parity, 2 stop bits, transfer data: 36_H

(e) 8-bit data length, LSB first, no parity, 1 stop bit, transfer data: 87_H

23.6.2 BF transmission/reception format

The URTEn has a BF (Break Field) transmission/reception control function to enable use of the LIN function.

About LIN

LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol intended to aid the cost reduction of an automotive network.

LIN communication is single-master communication, and up to 15 slaves can be connected to one master.

The LIN slaves are used to control the switches, actuators, and sensors, and these are connected to the LIN master via the LIN network.

Normally, the LIN master is connected to a network such as CAN (Controller Area Network).

In addition, the LIN bus uses a single-wire method and is connected to the nodes via a transceiver that complies with ISO9141.

In the LIN protocol, the master transmits a frame with baud rate information and the slave receives it and corrects the baud rate error. Therefore, communication is possible when the baud rate error in the slave is ±14% or less.

The *Figure 23-6 "LIN transmission outline"* and *Figure 23-7 "LIN reception outline"* below outline the transmission and reception manipulations of LIN.

(1) LIN transmission outline

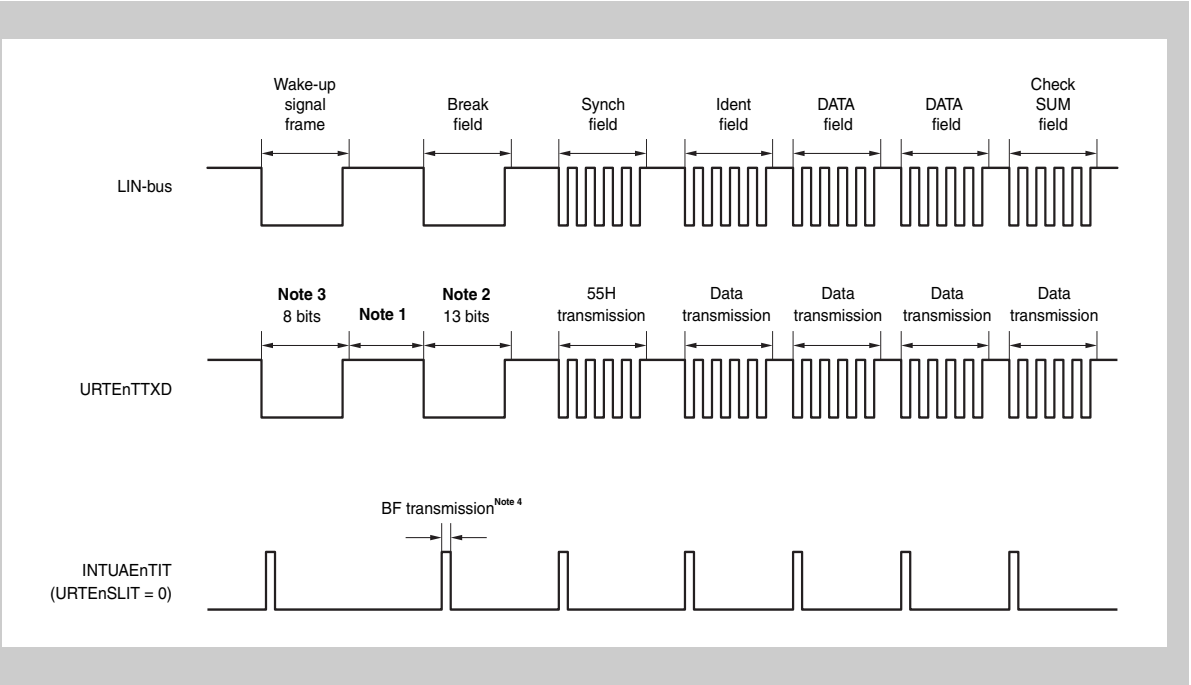


Figure 23-6 LIN transmission outline

Notes

1. The interval between each field is controlled by software.
2. BF output is performed by hardware. The output width is the bit length set by URTEnCTL1.URTEnBLG[2:0].
   If even finer output width adjustments are required, such adjustments can be performed using URTEnCTLn.URTEnBRS[11:0].

3. 80_H transfer in the 8-bit mode is substituted for the wakeup signal frame.
4. A transmission enable interrupt INTUAEnTIT is generated at the start of each transmission. INTUAEnTIT is also generated at the start of each BF transmission.

(2) LIN reception outline

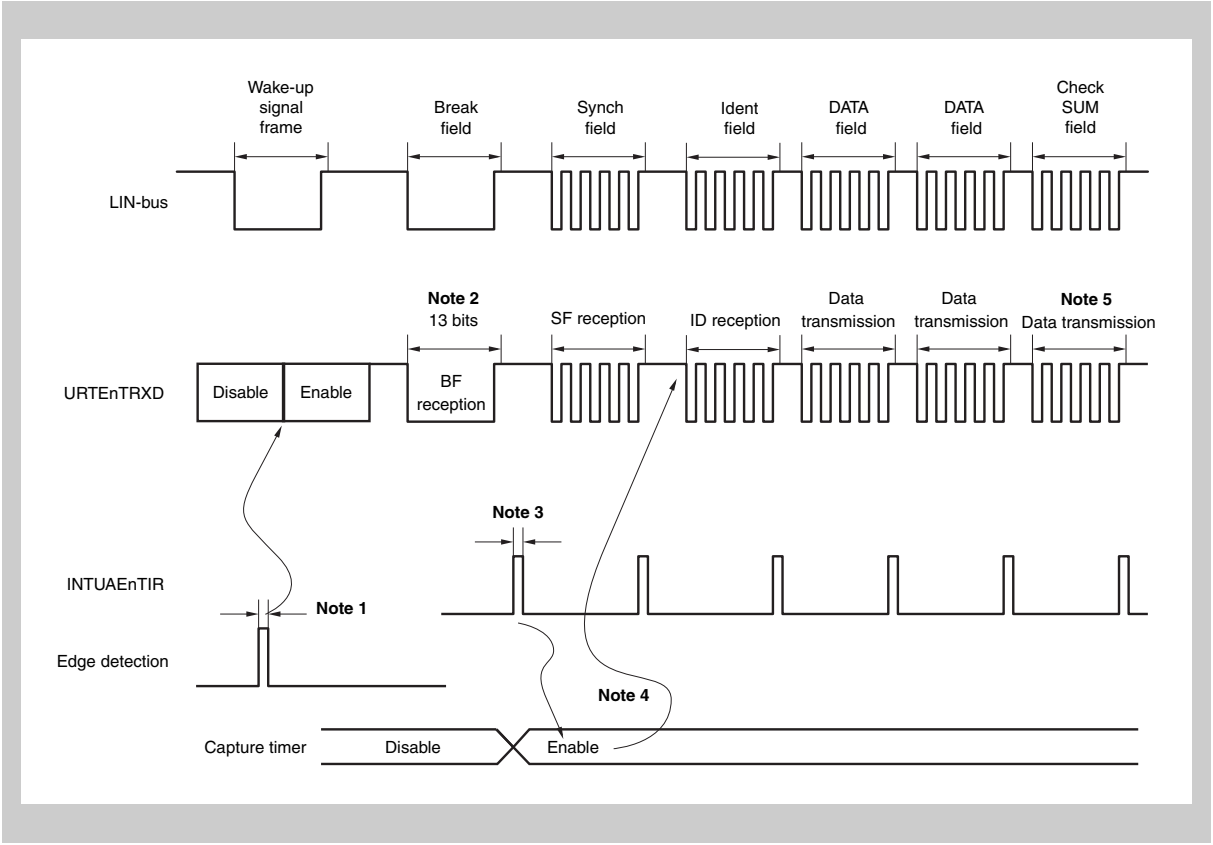


Figure 23-7 LIN reception outline

Notes

1. The wakeup signal is sent by the port edge detector, URTEn is enabled, and the BF reception mode is set.
2. Upon detection of the BF reception of 11 or more bits, normal BF reception end is judged.
   Upon detection of BF reception of less than 11 bits, a BF reception error is judged, no interrupt is generated, and the mode returns to the BF reception mode.
3. When BF reception ends normally and reception mode selection bit URTEnCTL1.URTEnSLBM
   - is set to "0"
     the reception interrupt INTUAEnTIR is generated
   - is set to "1"
     the status interrupt INTUAEnTIS is generated and the BF reception success flag URTEnSTR1.URTEnBSF is set.

   If the BF reception trigger bit URTEnTRG.URTEnBRT = 1, the error detection for the overrun, parity, and framing is not performed during the BF reception. Moreover, the data transfer from the receive shift register URTEnRXS to the receive data register URTEnRX is not performed, either. At this time, the URTEnRX holds the prior value.

4. In order to adjust the baud rate clock properly, the URTEnTRXD signal needs to be connected to the capture input of a timer. By measuring the time between two URTEnTRXD edges the transfer rate and the baud rate error can be calculated and the baud rate can be adjusted accordingly via the baud rate setting URTEnCTL2.URTEnBRS[11:0].
5. Check-sum field (CSF) distinctions are made by software. URTEn is initialized following CSF reception, and the processing for setting the BF reception mode again is performed by software. When URTEnCTL1.URTEnSLBM = 1, BF reception can be performed automatically without setting to BF reception mode again.

23.6.3 BF transmission

When the URTEnCTL0 bits URTEnPW = URTEnTXE = 1, the transmission enabled status is entered, and BF transmission is started by setting the BF transmission trigger URTEnTRG.URTEnBTT = 1.

Thereafter, URTEnSTR0.URTEnSSBT is set to “1” and a low level width of 13 to 20 bits, as specified by URTEnCTL1.URTEnBLG[2:0], is output.
A transmission interrupt INTUAEnTIT is generated upon BF

- transmission start, if URTEnCTL1.URTEnSLIT = 0
- transmission end, if URTEnCTL1.URTEnSLIT = 1.

Following the end of BF transmission, URTEnSTR0.URTEnSSBT is automatically cleared. Thereafter, the URTEn transmission mode is restored.

Transmission is suspended until the data to be transmitted next is written to the URTEnTX register, or until the BF transmission trigger URTEnTRG.URTEnBTT is set to 1 and URTEnSTR0.URTEnSSBT changes to 1.

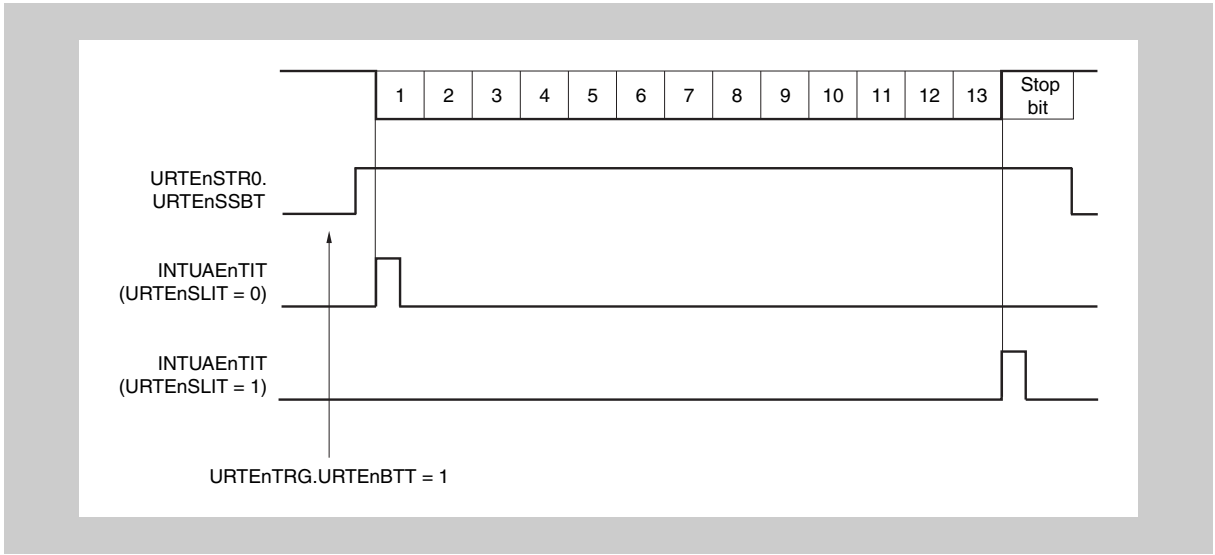


Figure 23-8 BF transmission

23.6.4 BF reception

The reception enabled status is achieved by setting the URTEnCTL0.URTEnCTL0.URTEnPW bit to 1 and then setting the URTEnCTL0.URTEnRXE bit to 1.

The BF reception wait status is set by setting the BF reception trigger URTEnTRG.URTEnBRT = 1. URTEnSTR0.URTEnSSBR is set to 1.

In the BF reception wait status, similarly to the URTEn reception wait status, the URTEnTRXD pin is monitored and start bit detection is performed.

Following detection of the low level, reception is started and the internal counter counts up according to the set baud rate.

When a low level is received for minimum 11 bits (valid BF reception), while the BF receiving mode selection bit

- URTEnCTL1.URTEnSLBM = 0 (BF Rx during data Rx disabled), the reception interrupt INTUAEnTIR is generated.
- URTEnCTL1.URTEnSLBM = 1 (BF Rx during data Rx enabled), the status interrupt INTUAEnTIS is generated and BF reception success flag URTEnSTR1.URTEnBSF is set at the same time

The URTEnSTR0.URTEnSSBR bit is automatically cleared and BF reception ends.

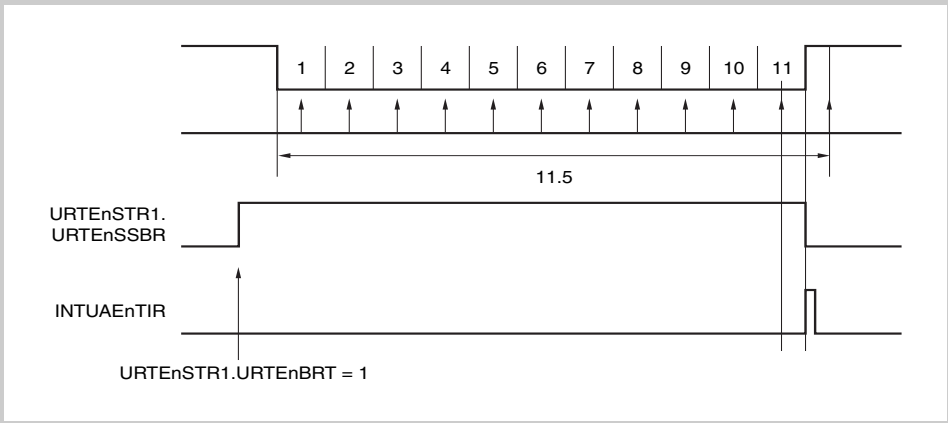


Figure 23-9 Normal BF reception (stop bit after more than 10.5 “L” bits)

Error detection for the URTEnSTR1 error flags URTEnOVE, URTEnPE, and URTEnFE is suppressed and URTEn communication error detection processing is not performed.
Moreover, the erroneous data is not stored in URTEnRX, but the initial value FF_H is held.

If the BF width is 10 or fewer bits, reception is terminated as error processing without generating an interrupt, and the BF reception mode is returned to. URTEnSTR0.URTEnSSBR is not cleared at this time.

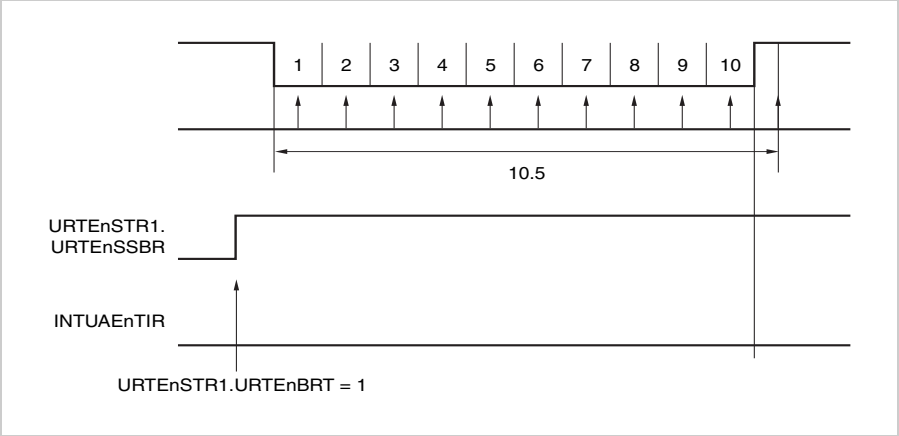


Figure 23-10 BF reception error (stop bit within 10.5 “L” bits)

The BF mode can be selected between a single BF receive mode and an any time BF receive mode in by URTEnCTL1.URTEnSLBM. The status of a successful reception of the BF is shown URTEnSTR1.URTEnBSF.

Note URTEnSTR0.URTEnSSBR is set to “1”

- by setting URTEnTRG.URTEnBRT to “1"
- cleared by normal BF reception.

23.6.5 Transmission data consistency check

The URTEn incorporates a data consistency check function to detect a mismatch between the transmit data written to transmit register URTEnTX and the data on the bus when the device operates in master mode.

Data consistency check is enabled by URTEnCTL0.URTEnSLDC = 1.

The data consistency is checked by comparing the transmit data in the transmit register URTEnTX and the receive data in the receive register URTEnRX. In case of a mismatch the data consistency error flag URTEnSTR1.URTEnDCE is set and a status interrupt request INTUAEnTIS occurs.

The data consistency check of data is not done in reception mode.

The data consistency check of the send data and the input data terminal level is done even if the reception is disabled during sending. In that case also the reception completion interrupt request signal INTUAEnTIR, the URTEnSTR1 status bits URTEnBSF, URTEnFE, URTEnOVE and the status interrupt request signal INTUAEnTIS will not be generated as well. Receive data does not need to be read.

Refer to the description of the *"URTEnSTR1 - URTEn status register 1"* for details.

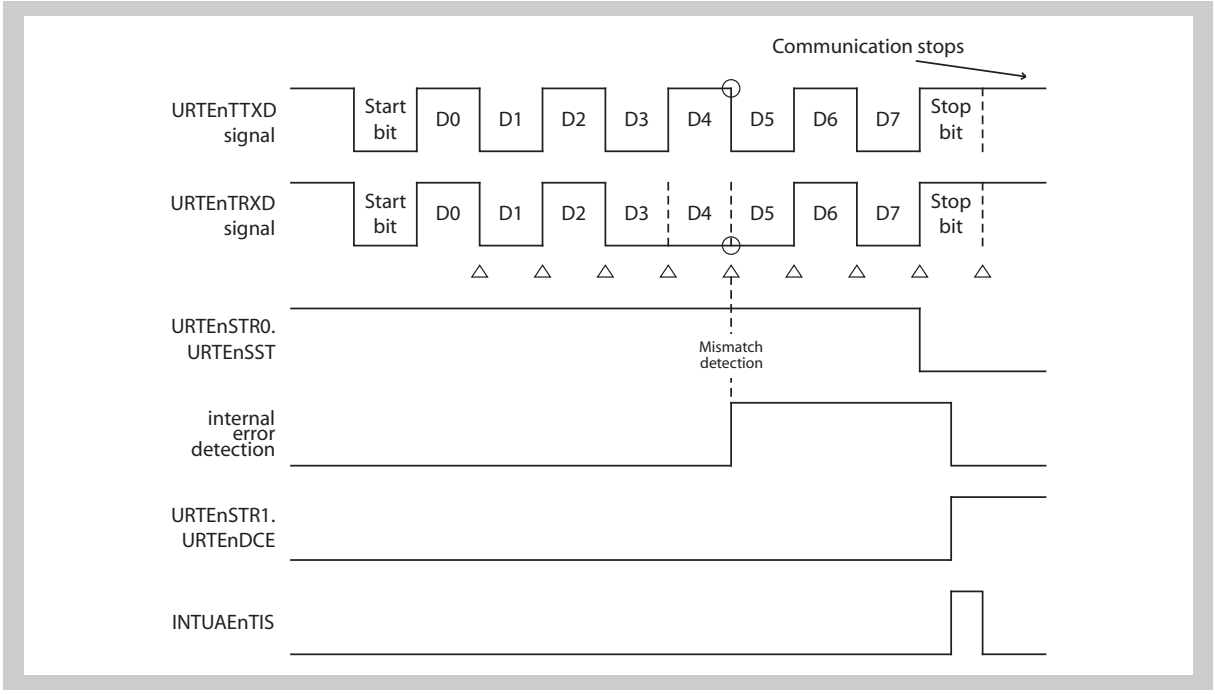


Figure 23-11 Timing example of data consistency error (no BF reception active, i.e. URTEnSTR0.URTEnSSBR = 0)

23.6.6 URTEn transmission

Transmission start

Set the transmission enabled status by performing the following procedures.

- Specify the baud rate by the URTEn control register 2 URTEnCTL2.
- Specify the transmit parity, data character length, stop bit length, transmit data order, transmission interrupt request timing and output logic level by the URTEn control register 1 URTEnCTL1.
- Enable URTEn operation and transmission by URTEnCTL0.URTEnPW = URTEnCTL0.URTEnTXE = 1)

Write of the transmit data to the transmission buffer register URTEnTX starts transmission. The data which is saved in the URTEnTX register is transferred to the transmit shift register URTEnTXS. Then, the start, parity and stop bits are added and the data frame is output serially via URTEnTTXD.

Transmission stop

When URTEnCTL0.URTEnPW or URTEnCTL0.URTEnTXE is set to 0, transmission operations are stopped immediately, even during transmission processing.

Concurrent BF and data transmission

When a BF transmit request and a data transmit request have both been set, BF transmission takes priority.

Data consistency check

When URTETDCM = "L", if a data consistency error is detected, the subsequent data is not transmitted until URTE0CLDC = 1, URTE0PW = 0, or URTE0TXE = 0 is written.

INTUAEnTIT timing

The time to generate the transmission interrupt INTUAEnTIT depends on the setting of the URTEnCTL1.URTEnSLIT bit:

- URTEnCTL1.URTEnSLIT = 0
  INTUAEnTIT is generated at the start of transmission, i.e. when the data from the data register URTEnTX is transferred to the transmit shift register URTEnTXS and transmission is started.
- URTEnCTL1.URTEnSLIT = 1
  INTUAEnTIT is generated when the entire data transmission process is completed, i.e. when the last bit of the data frame has been transitted.

Once INTUAEnTIT is generated, the next data can be written to URTEnTX.

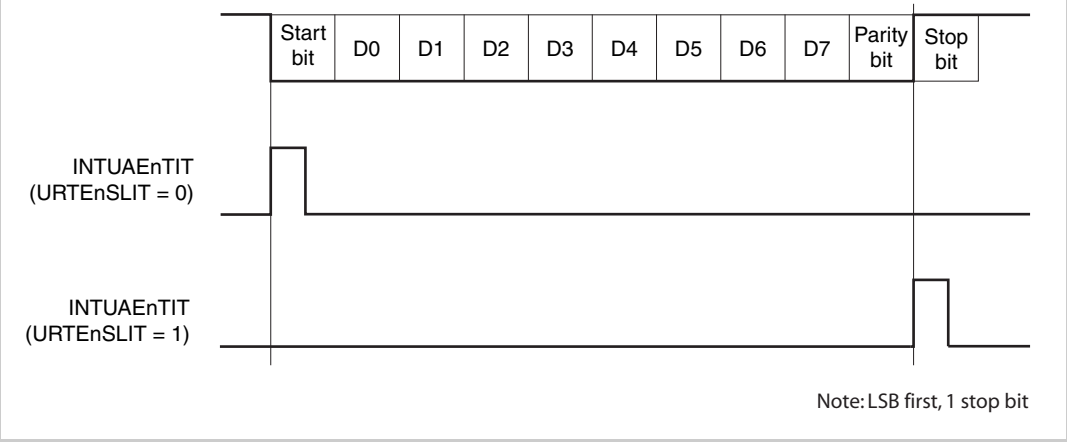


Figure 23-12 Transmission interrupt timing

23.6.7 Continuous transmission procedure

Continuous transmission is achieved by writing the next transmit data to the transmit data register URTEnTX, while shifting out of the previous data from the transmission shift register URTEnTXS is ongoing.

Note In order to maintain correct write timing, the transmission interrupt INTUAEnTIT must be generated at the start of each transmission (URTEnCTL1.URTEnSLIT = 0).

Caution If the value is written to the URTEnTX register before the INTUAEnTIT is generated, the transmit data set before is overwritten by the new transmit data.

To initialize the transmission unit, confirm that no transmission is ongoing (URTEnSTR0 bits URTEnSSBT = URTEnSST = 0). If the initialization is performed during an ongoing transmission, the transmission is aborted.

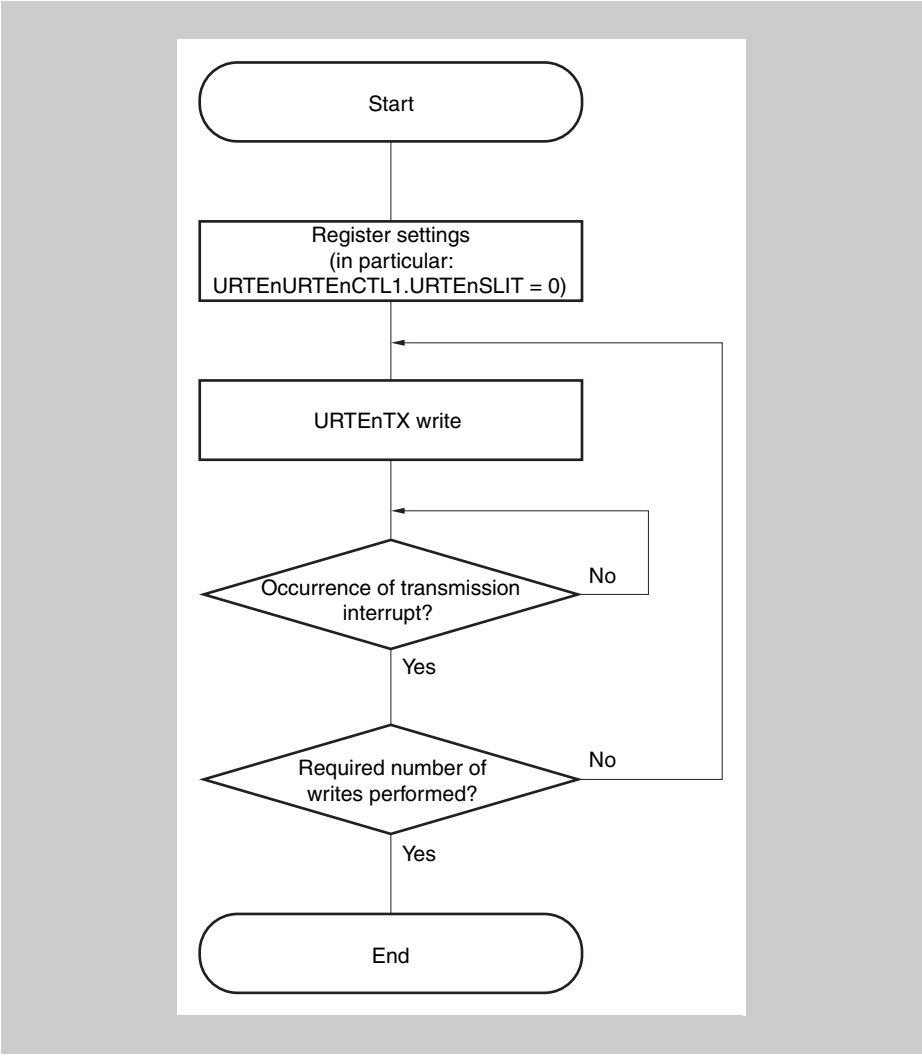


Figure 23-13 Continuous transmission processing flow

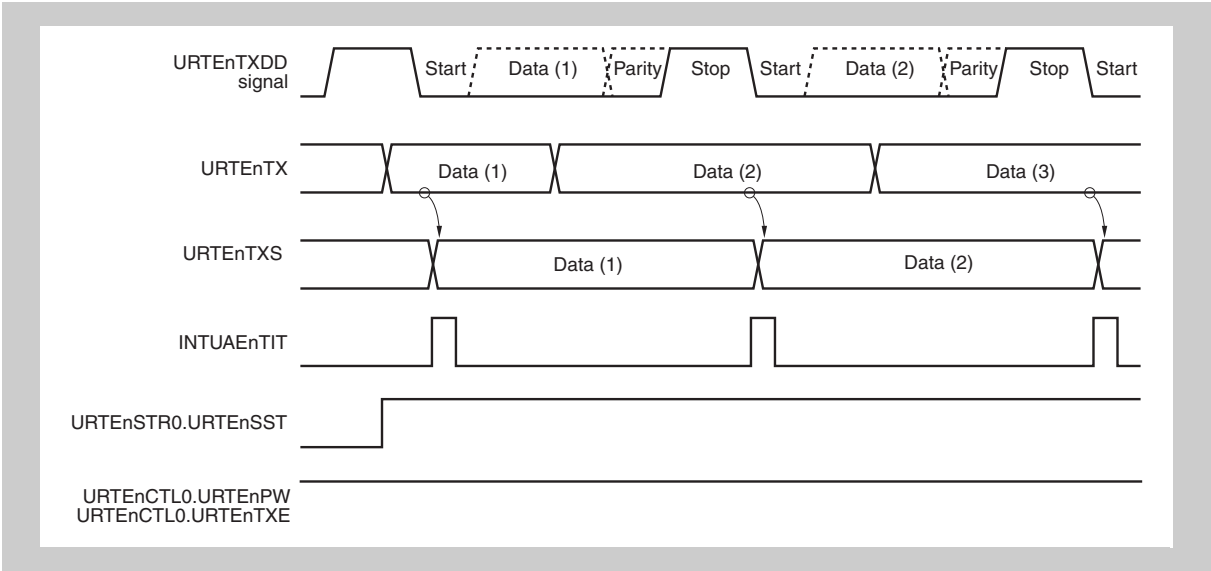


Figure 23-14 Continuous transmission operation timing - transmission start

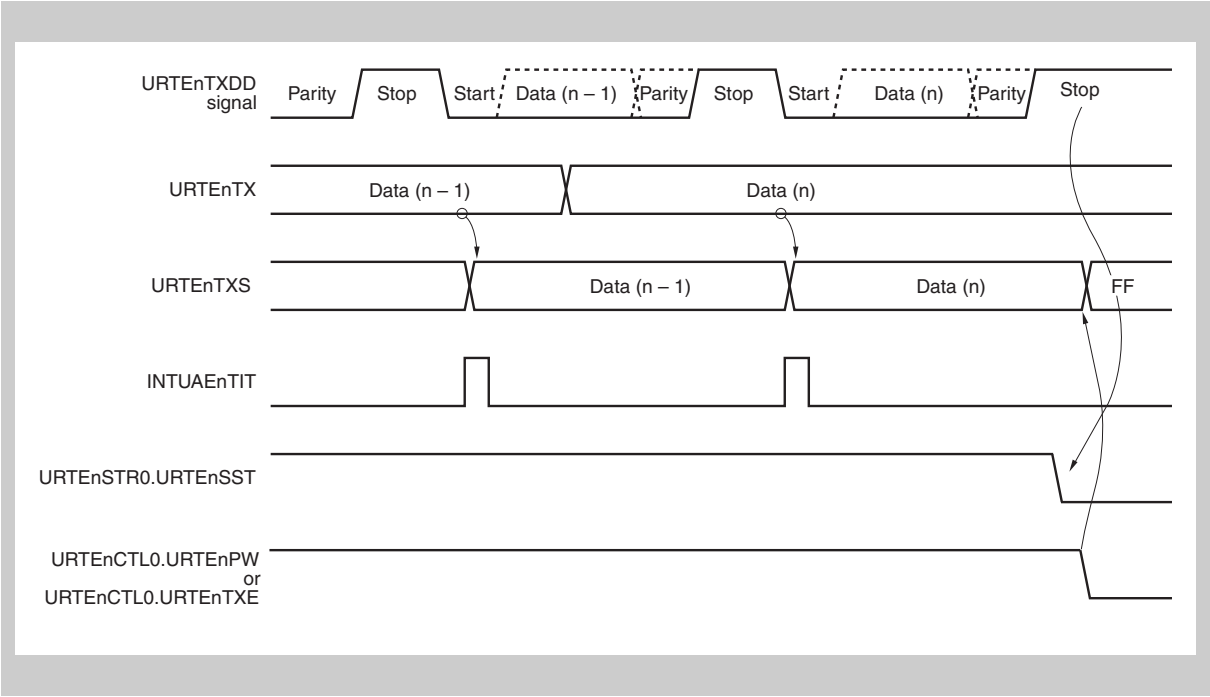


Figure 23-15 Continuous transmission operation timing - transmission end

23.6.8 URTEn reception

Reception start

Set the reception enabled status by the following procedure:

- Specify the baud rate by the URTEn control register 2 URTEnCTL2.
- Specify the receive parity, data character length, stop bit length, receive data order and output logic level by the URTEn control register 1 URTEnCTL1.
- Enable URTEn operation and reception by URTEnCTL0.URTEnPW = URTEnCTL0.URTEnRXE = 1).

When the sampling of the input level of the URTEnTRXD pin is performed and the falling edge is detected, the data sampling of the URTEnTRXD input is started. The start bit is recognized if the URTEnTRXD pin is low level after the time of a half bit is passed after the detection of the falling edge (shown in the figure below). After a start bit has been recognized, the receive operation starts, and serial data is stored in the receive shift register according to the set baud rate. When the reception interrupt INTUAEnTIR is asserted upon reception of the stop bit, the data stored in the receive shift register URTEnRXS is written to the receive data register URTEnRX.

Reception stop

When URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE is set to 0, reception operations are stopped immediately, even during reception processing.

Reception errors

If an overrun error occurs (URTEnSTR1.URTEnOVE = 1), the receive data at this time is not transferred to the URTEnRX register and is discarded.
Even if a parity error (URTEnSTR1.URTEnPE = 1) or a framing error (URTEnSTR1.URTEnFE = 1) occurs during reception, reception continues until the reception position of the first stop bit, and the reception data is transferred to the URTEnRX.
In any case of the reception errors, the status interrupt INTUAEnTIS is

generated after the following reception completion, but not the reception interrupt INTUAEnTIR.

When the receive data order, parity, data character length, and the stop bit length are changed, clear the power bit (URTEnCTL0.URTEnPW = 0) or clear both the transmission enabled bit and the reception enabled bit (URTEnTXE = 0, URTEnRXE = 0), and then change the setting.

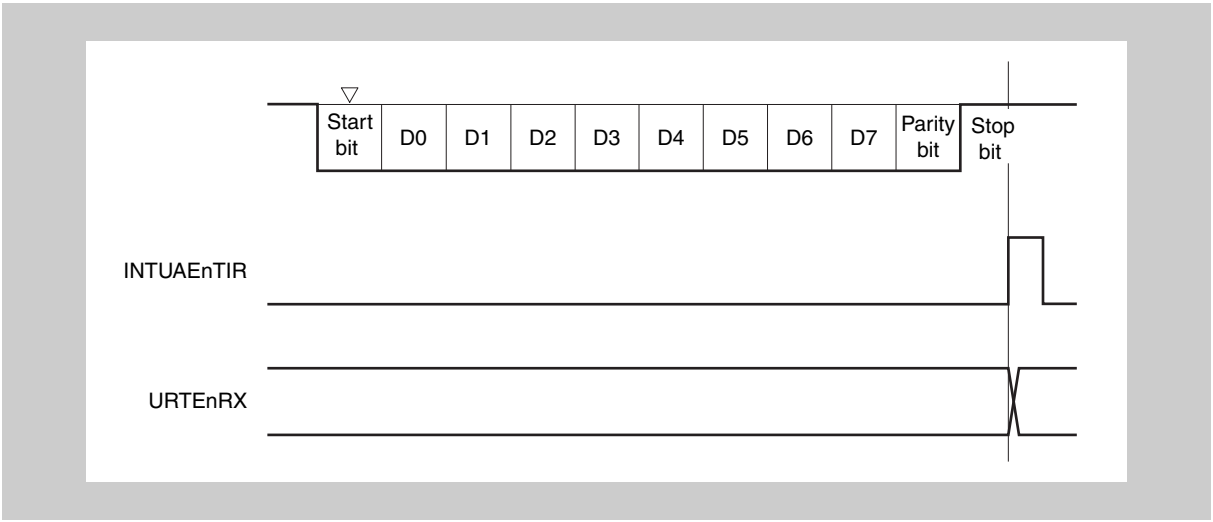


Figure 23-16 URTEn reception

Cautions

1. Be sure to read the URTEnRX register even when a reception error occurs. If the URTEnRX register is not read, an overrun error occurs during reception of the next data.
2. The operation during reception is performed assuming that there is only one stop bit. A second stop bit is ignored.
3. When reception is completed, read the URTEnRX register after the reception interrupt INTUAEnTIR has been generated, and clear the URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE bit to 0.
   If the URTEnCTL0.URTEnPW or URTEnCTL0.URTEnRXE bit is cleared to 0 before the INTUAEnTIR is generated, the read value of the URTEnRX register cannot be guaranteed.
4. If receive completion processing (INTUAEnTIR interrupt generation) and the URTEnCTL0.URTEnPW bit = 0 or URTEnCTL0.URTEnRXE bit = 0 conflict, INTUAEnTIR may be generated in spite of these being no data stored in the URTEnRX register.

Notes

1. If low level is always input to the URTEnTRXD pin, it is not judged as the start bit.
2. In continuous reception, immediately after the stop bit is detected at the first reception bit (when the reception interrupt is generated), the next start bit can be detected.

23.6.9 Reception errors

Errors during a receive operation are of three types: parity errors, framing errors, and overrun errors. Data reception result error flags are set in the URTEnSTR1 register and a status interrupt request signal INTUAEnTIS is generated when an error occurs.

It is possible to ascertain which error occurred during reception by reading the contents of the URTEnSTR1 register.

Clear a reception error flag by writing 1 to its associated bit in the status clear register URTEnSTC.

Table 23-20 Reception error causes

Error flag in URTEnSTR1	Reception error	Cause
URTEnPE	Parity error	Received parity bit does not match the setting
URTEnFE	Framing error	Stop bit not detected
URTEnOVE	Overrun error	Reception of next data completed before data was read from receive buffer

Note Even in case of a parity or framing error, data is transferred from the receive shift register URTEnRXS to the receive data register URTEnRX. Consequently the data from URTEnRX must be read. Otherwise an overrun error URTEnSTR1.URTEnOVE will occur at reception of the next data.

In case of an overrun error, the receive shift register data is not transferred to URTEnRX, thus the previous data is not overwritten.

23.6.10 Parity types and operations

Caution When using the LIN function, fix the URTEnCTL1.URTEnSLP[1:0] to 00_B.

The parity bit is used to detect bit errors in the communication data. Normally the same parity is used on the transmission side and the reception side.

In the case of even parity and odd parity, it is possible to detect odd-count bit errors. In the case of 0 parity and no parity, errors cannot be detected.

(1) Even parity

- During transmission
  The number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so as to be an even number. The parity bit values are as follows:
  - Odd number of bits whose value is “1” among transmit data:1
  - Even number of bits whose value is “1” among transmit data:0
- During reception
  The number of bits whose value is “1” among the reception data, including the parity bit, is counted, and if it is an odd number, a parity error is output.

(2) Odd parity

- During transmission
  Opposite to even parity, the number of bits whose value is “1” among the transmit data, including the parity bit, is controlled so that it is an odd number. The parity bit values are as follows.
  - Odd number of bits whose value is “1” among transmit data: 0
  - Even number of bits whose value is “1” among transmit data: 1
- During reception
  The number of bits whose value is “1” among the receive data, including the parity bit, is counted, and if it is an even number, a parity error is output.

(3) 0 parity

During transmission, the parity bit is always made 0, regardless of the transmit data.

During reception, parity bit check is not performed. Therefore, no parity error occurs, regardless of whether the parity bit is 0 or 1.

(4) No parity

No parity bit is added to the transmit data.

Reception is performed assuming that there is no parity bit. No parity error occurs since there is no parity bit.

23.6.11 Digital receive data noise filter

The receive data signal input URTEnTRXD is equipped with a digital noise filter to eliminate noise and spikes.

This filter samples the URTEnTRXD pin using the prescaler output clock PRSCLK.

When the same sampling value is read twice, the URTEnTRXD signal is validated as the input data.
Therefore, data not exceeding the width of 2 prescaler output clocks is judged to be noise and thus eliminated.

The noise filter causes a delay of 4 prescaler output clock PRSCLK cycles when capturing the serial data URTEnTRXD, until it is forwarded as valid.

23.7 Baud Rate Generator

The transmission and reception baudrate BRCLK are derived from the PBUS bus clock PCLK by use of a prescaler and a baudrate generator, as shown in the figure below.

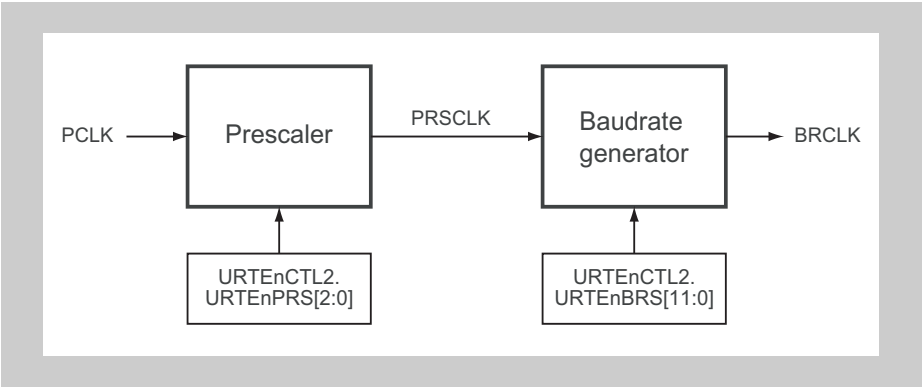


Figure 23-17 Configuration of baud rate generator

The prescaler output clock PRSCLK is a fraction of PCLK, the divisor is set up by the value URTEnCTL2.URTEnPRS[2:0]:

$$PRSCLK = PCLK / 2^{URTEnPRS[2:0]}$$

PRSCLK is further divided by the baudrate generator by a value, determined by URTEnCTL2.URTEnBRS[11:0].
The baudrate generator distinguishes between the baudrate for data frames and BF receptions, as listed in the table below. The BF reception clock is the double of the baudrate clock BRCLK.

Table 23-21 Baudrate generator clocks output

URTEnCTL2. URTEnBRS[11:0]	Transmit/receive BRCLK	BF receive clock
000_H	PCLK / [$2^{URTEnPRS[2:0]}$ x (2 x 4)]	PCLK / [$2^{URTEnPRS[2:0]}$ x 4]
001_H		
002_H		
003_H		
004_H		
005_H	PCLK / [$2^{URTEnPRS[2:0]}$ x (2 x 5)]	PCLK / [$2^{URTEnPRS[2:0]}$ x 5]
...	PCLK / [$2^{URTEnPRS[2:0]}$ x (2 x URTEnBRS[11:0])]	PCLK / [$2^{URTEnPRS[2:0]}$ x URTEnBRS[11:0]
FFE_H	PCLK / [$2^{URTEnPRS[2:0]}$ x (2 x 4094)]	PCLK / [$2^{URTEnPRS[2:0]}$ x 4094]
FFF_H	PCLK / [$2^{URTEnPRS[2:0]}$ x (2 x 4095)]	PCLK / [$2^{URTEnPRS[2:0]}$ x 4095]

Chapter 24 LIN Master Controller (LMA)

This chapter contains a generic description of the LIN Master Controller.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

24.1 V850E2/Fx4-H LMAn Features

LMAn instances

This microcontroller has following number of instances of the LIN Master Controller LMAn.

Table 24-1 Instances of LMA

LIN Master Controller	
Instance	12
Name	LMA0 to LMA11

LMAn instances index n

Throughout this chapter, the instance of a LIN Master Controller is identified by the index "n" (n = 0 to 11), for example, LMAnCTLL for the LMAn control register L.

CNTAm instances

This microcontroller has following number of instances of the LIN Master Scheduler Counters CNTAm.

Table 24-2 Instances of LIN Master Scheduler Counters CNTAm

LIN Master Schduler Counter	
Instance	3
Name	CNTA0 to CNTA2

CNTAm instances index m

Throughout this chapter, the instance of a LIN Master Scheduler Counter is identified by the index "m" (m = 0 to 2), for example, CNTAmCTL for the CNTAm control register.

LMAn register addresses

All LMAn register addresses are given as address offsets to the individual base address <LMAn_base>.
The <LMAn_base> address of each LMAn are listed in the following table:

Table 24-3 LMAn register base addresses <LMAn_base> (1/2)

LMAn instance	<LMAn_base> address
LMA0	FF5C 0080$_H$
LMA1	FF5D 0080$_H$

Table 24-3 LMAn register base addresses <LMAn_base> (2/2)

LMAn instance	<LMAn_base> address
LMA2	FF5E 0080$_H$
LMA3	FF5F 0080$_H$
LMA4	FF60 0080$_H$
LMA5	FF61 0080$_H$
LMA6	FF62 0080$_H$
LMA7	FF63 0080$_H$
LMA8	FF64 0080$_H$
LMA9	FF65 0080$_H$
LMA10	FF66 0080$_H$
LMA11	FF67 0080$_H$

CNTAm register addresses

All CNTAm register addresses are given as address offsets to the individual base address <CNTAm_base>.
The <CNTAm_base> address of each CNTAm are listed in the following table:

Table 24-4 CNTAm register base addresses <CNTAm_base>

CNTAm instance	<CNTAm_base> address
CNTA0	FF5C 4000$_H$
CNTA1	FF5D 4000$_H$
CNTA2	FF5E 4000$_H$

Clock supply

All LIN Master Controller and LIN Master Scheduler Counters provide one clock input.

Table 24-5 LMAn/CNTAm clock supply (1/2)

LMAn/CNTAm instance	LMAn clock	Connected to
LMA0	PCLK	Clock Controller CKSCLK_112
LMA1	PCLK	Clock Controller CKSCLK_112
CNTA0	PCLK	Clock Controller CKSCLK_112
LMA2	PCLK	Clock Controller CKSCLK_114
LMA3	PCLK	Clock Controller CKSCLK_114
LMA4	PCLK	Clock Controller CKSCLK_114
LMA5	PCLK	Clock Controller CKSCLK_114
LMA6	PCLK	Clock Controller CKSCLK_114
LMA7	PCLK	Clock Controller CKSCLK_114
LMA8	PCLK	Clock Controller CKSCLK_114
LMA9	PCLK	Clock Controller CKSCLK_114
CNTA1	PCLK	Clock Controller CKSCLK_114
LMA10	PCLK	Clock Controller CKSCLK_011

Table 24-5 LMAn/CNTAm clock supply (2/2)

LMAn/CNTAm instance	LMAn clock	Connected to
LMA11	PCLK	Clock Controller CKSCLK_011
CNTA2	PCLK	Clock Controller CKSCLK_011

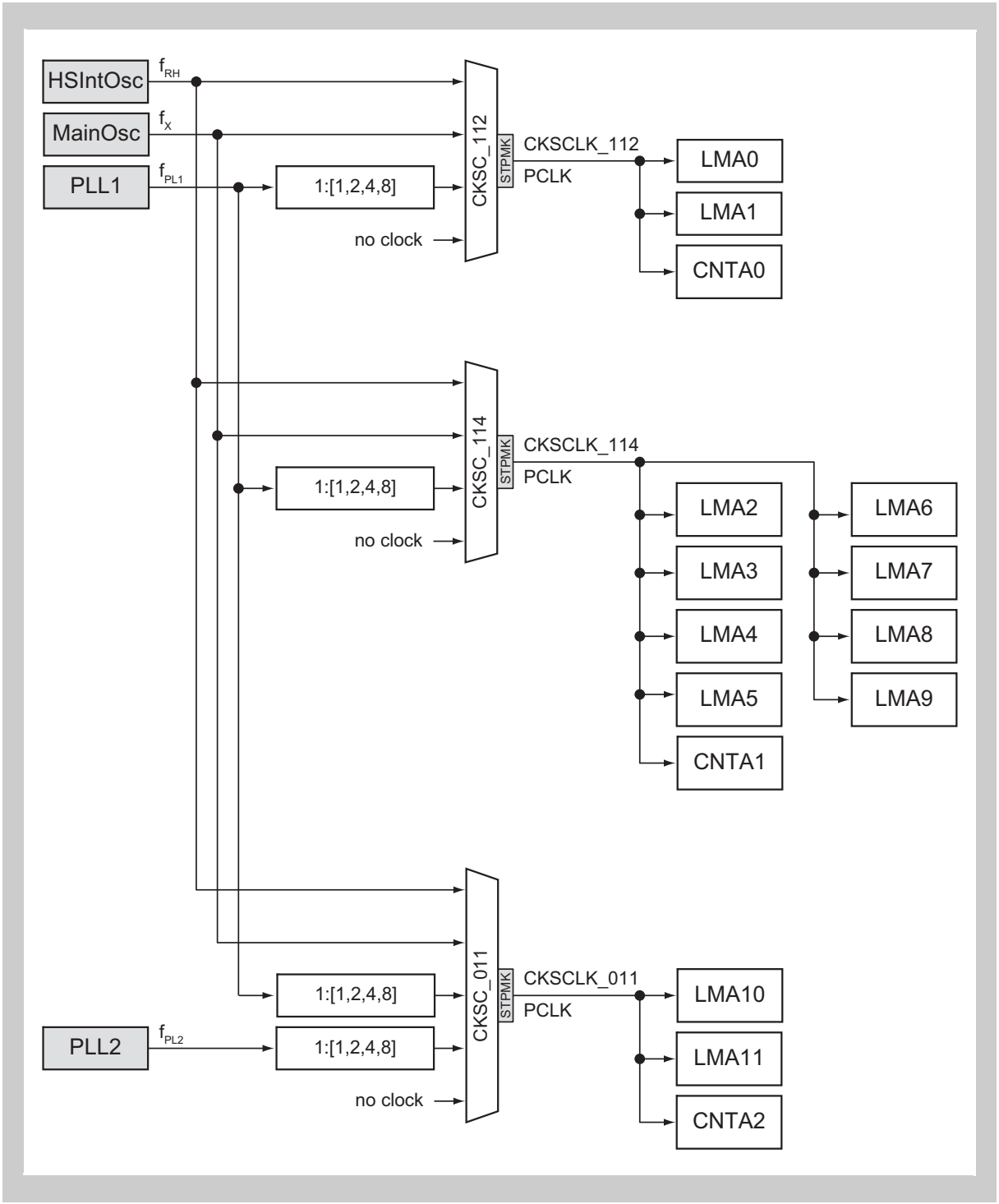


Figure 24-1 LMAnclock supply

Interrupts and DMA/DTS

The LIN Master Controllers can generate following interrupt and DMA/DTS requests:

Table 24-6 LMAn interrupt and DMA/DTS requests (1/2)

LMAn signals	Function	Connected to
LMA0:		
INTLMA0TIT	Transmission interrupt	Interrupt Controller INTLMA0IT DMA Controller trigger 45 DTS Controller trigger 59
INTLMA0TIR	Reception interrupt	Interrupt Controller INTLMA0IR DMA Controller trigger 44 DTS Controller trigger 58
INTLMA0TIS	Status interrupt	Interrupt Controller INTLMA0IS
LMA1:		
INTLMA1TIT	Transmission interrupt	Interrupt Controller INTLMA1IT DMA Controller trigger 47 DTS Controller trigger 61
INTLMA1TIR	Reception interrupt	Interrupt Controller INTLMA1IR DMA Controller trigger 46 DTS Controller trigger 60
INTLMA1TIS	Status interrupt	Interrupt Controller INTLMA1IS
LMA2:		
INTLMA2TIT	Transmission interrupt	Interrupt Controller INTLMA2IT DMA Controller trigger 92
INTLMA2TIR	Reception interrupt	Interrupt Controller INTLMA2IR
INTLMA2TIS	Status interrupt	Interrupt Controller INTLMA2IS
LMA3:		
INTLMA3TIT	Transmission interrupt	Interrupt Controller INTLMA3IT DMA Controller trigger 93
INTLMA3TIR	Reception interrupt	Interrupt Controller INTLMA3IR
INTLMA3TIS	Status interrupt	Interrupt Controller INTLMA3IS
LMA4:		
INTLMA4TIT	Transmission interrupt	Interrupt Controller INTLMA4IT DMA Controller trigger 95
INTLMA4TIR	Reception interrupt	Interrupt Controller INTLMA4IR DMA Controller trigger 94
INTLMA4TIS	Status interrupt	Interrupt Controller INTLMA4IS
LMA5:		
INTLMA5TIT	Transmission interrupt	Interrupt Controller INTLMA5IT DMA Controller trigger 108
INTLMA5TIR	Reception interrupt	Interrupt Controller INTLMA5IR DMA Controller trigger 107
INTLMA5TIS	Status interrupt	Interrupt Controller INTLMA5IS
LMA6:		
INTLMA6TIT	Transmission interrupt	Interrupt Controller INTLMA6IT DMA Controller trigger 110
INTLMA6TIR	Reception interrupt	Interrupt Controller INTLMA6IR DMA Controller trigger 109
INTLMA6TIS	Status interrupt	Interrupt Controller INTLMA6IS

Table 24-6 LMAn interrupt and DMA/DTS requests (2/2)

LMAn signals	Function	Connected to
LMA7:		
INTLMA7TIT	Transmission interrupt	Interrupt Controller INTLMA7IT DMA Controller trigger 112
INTLMA7TIR	Reception interrupt	Interrupt Controller INTLMA7IR DMA Controller trigger 111
INTLMA7TIS	Status interrupt	Interrupt Controller INTLMA7IS
LMA8:		
INTLMA8TIT	Transmission interrupt	Interrupt Controller INTLMA8IT DMA Controller trigger 114
INTLMA8TIR	Reception interrupt	Interrupt Controller INTLMA8IR DMA Controller trigger 113
INTLMA8TIS	Status interrupt	Interrupt Controller INTLMA8IS
LMA9:		
INTLMA9TIT	Transmission interrupt	Interrupt Controller INTLMA9IT DMA Controller trigger 116
INTLMA9TIR	Reception interrupt	Interrupt Controller INTLMA9IR DMA Controller trigger 115
INTLMA9TIS	Status interrupt	Interrupt Controller INTLMA9IS
LMA10:		
INTLMA10TIT	Transmission interrupt	Interrupt Controller INTLMA10IT[a] DMA Controller trigger 118
INTLMA10TIR	Reception interrupt	Interrupt Controller INTLMA10IR[a] DMA Controller trigger 117
INTLMA10TIS	Status interrupt	Interrupt Controller INTLMA10IS[a]
LMA11:		
INTLMA11TIT	Transmission interrupt	Interrupt Controller INTLMA11IT[a] DMA Controller trigger 120
INTLMA11TIR	Reception interrupt	Interrupt Controller INTLMA11IR[a] DMA Controller trigger 119
INTLMA11TIS	Status interrupt	Interrupt Controller INTLMA11IS[a]

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

LMA H/W reset The LIN Master Controllers and their registers are initialized by the following reset signal:

Table 24-7 LMAn reset signal

LMAn	Reset signal
LMA2 to LMA9	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)
LMA10, LMA11	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

Internal signals

Each LIN Master Controller LMAn is connected to an Asynchronous Serial Interfaces E URTEn signals and to the LIN Master Scheduler Counter CNTAm (m = 0 to 2).
The internal signal connections of the LIN Master Controllers are listed in the following table.

Table 24-8 LMAn internal signal connections

LMAn signal	Function	Connected to
LMA10:		
INTUAE10TIT	Transmission interrupt	URTEn INTUAE10TIT
INTUAE10TIR	Reception interrupt	URTEn INTUAE10TIR
INTUAE10TIS	Status interrupt	URTEn INTUAE10TIS
CNTAmCNT[15:0]	Free-running counter value	CNTA2 CNT2CNT[15:0]
LMA11:		
INTUAE11TIT	Transmission interrupt	URTEn INTUAE11TIT
INTUAE11TIR	Reception interrupt	URTEn INTUAE11TIR
INTUAE11TIS	Status interrupt	URTEn INTUAE11TIS
CNTAmCNT[15:0]	Free-running counter value	CNTA2 CNT2CNT[15:0]

LIN Master Scheduler Counters

Following table shows the assignment of the LIN Master Controllers to the LIN Master scheduler counters:

Table 24-9 CNTAm to LMAn assignment

CNTAm instance	LMAn instance
CNTA0	LMA0, LMA1
CNTA1	LMA2 to LMA9
CNTA2	LMA10 to LMA11

For a detailed description of the LIN Master Scheduler Counters CNTAm refer to section *19.1 “LIN Master Scheduler Counters (CNTA)” on page 1* .

24.2 LIN Master Scheduler Counters (CNTA)

The LIN master scheduler counter consists of a free-running 16-bit counter. The count clock is derived from the CNTA input clock PCLK, that is divided by a prescaler.

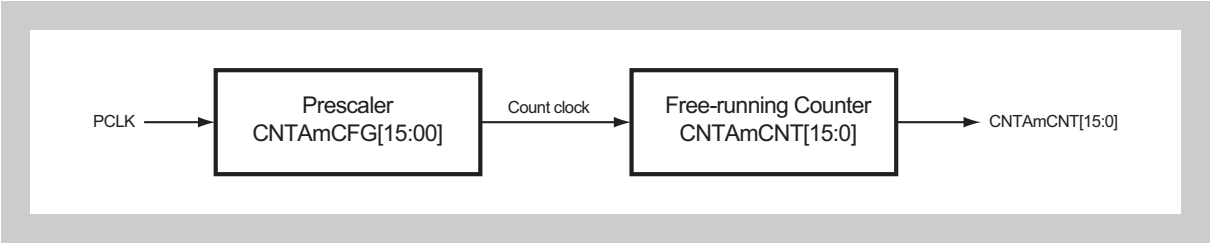


Figure 24-2 LIN master scheduler counter

CNTAm enable Before the LIN Master Controller is enabling the scheduler, the counter CNTAm must be enabled by CNTAmCTL.CNTAmPW = 1.

Prescaler division The division factor of the prescaler is determined by the value CNTAmCFG.CNTAmPRS[15:00]:

- CNTAmPRS[15:00] = $FFFF_H$: count clock = PCLK / 1
- else: count clock = PCLK / (CNTAmPRS[15:00]+2)

24.2.1 CNTAm registers

The CNTAm is controlled and operated by means of the following registers:

Table 24-10 CNTAm registers

Register function	Name	Address
Control register	CNTAmCTL	<CNTAm_base> + 00_H
Configuration register	CNTAmCFG	<CNTAm_base> + 04_H

<CNTAm_base> The base addresses <CNTAm_base> of the CNTAm are defined in the first section of this chapter under the key word “CNTAm register addresses”.

(1) CNTAmCTL - CNTAm control register

This register enables/disables the CNTAm operation.

Access This register can be read/written in 16-bit units.

Address <CNTAm_base> + 00_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTAm PW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-11 CNTAmCTL register contents

Bit position	Bit name	Function
15	CNTAm PW	CNTAm operation enable 0: CNTAm disabled 1: CNTAm enabled

(2) CNTAmCFG - CNTAm configuration register

This register sets the division factor for the clock prescaler.

Access This register can be read/written in 16-bit units.

Address <CNTAm_base> + 04_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTAmPRS[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-12 CNTAmCTL register contents

Bit position	Bit name	Function
15	CNTAm PRS[15:00]	CNTAm prescaler division ratio $FFFF_H$: PCLK/1 0000_H: PCLK/2 0001_H: PCLK/3 0002_H: PCLK/4 ... $FFFE_H$: PCLK/65536

24.3 Functional Overview

The LMAn module is connected to a UARTEn module. This combination provides a LIN master interface, but can be used also as a buffered UARTEn.

- UART through mode
- UART buffer mode, full-duplex operation
  - 12 byte Tx buffer
  - 12 byte Rx buffer
- LIN master mode
  - automatic checksum generation and check
  - automatic transmission of Break Field (BF), Sync Field (SF), and checksum
  - Scheduler and automatic frame start function

The block diagram shows the environment of the LIN Master Controller.

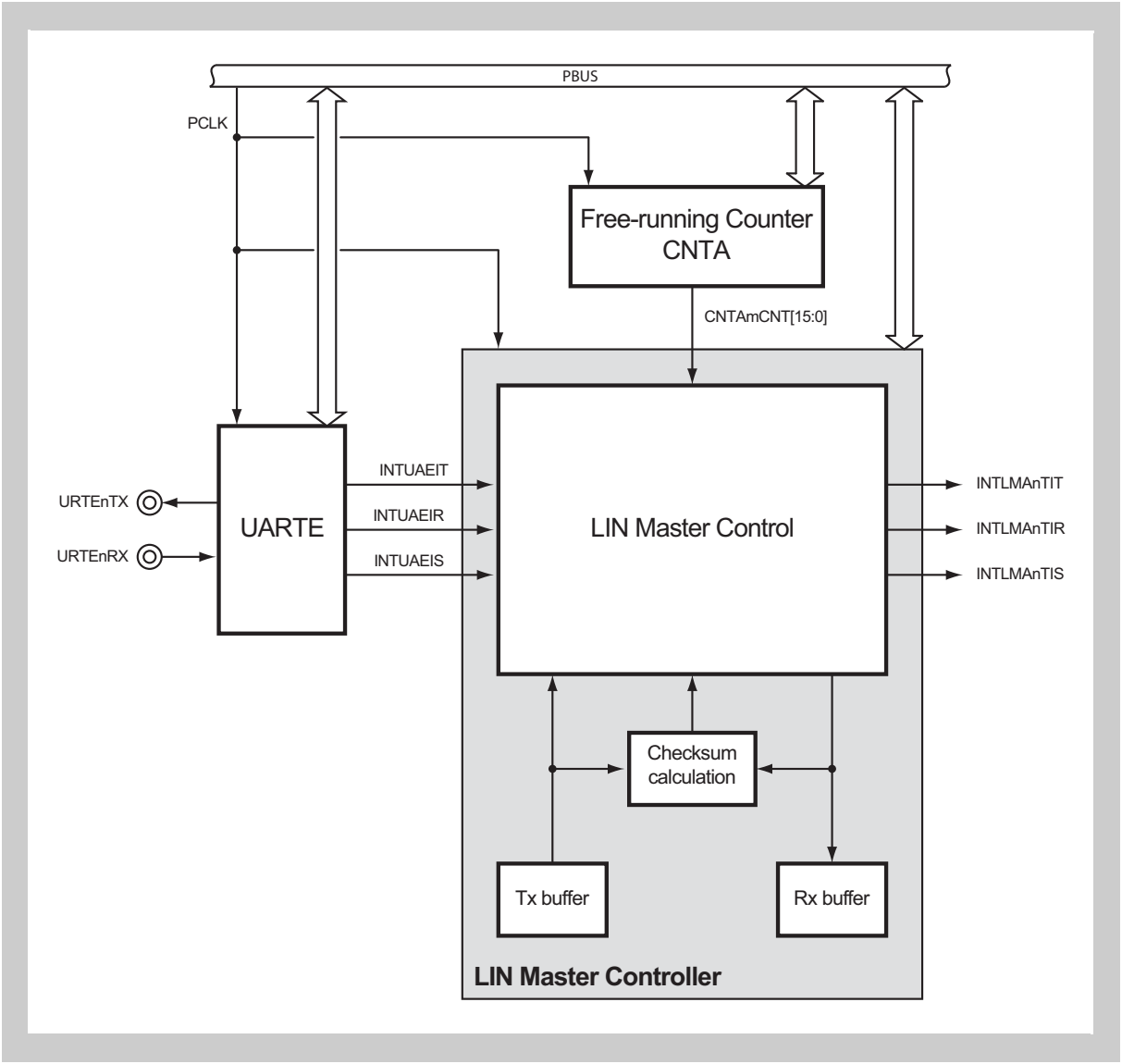


Figure 24-3 LIN Master Controller environment

The LIN Master Controller is tightly coupled to the UARTE and utilizes the UARTE as a asynchronous serial interface function with LIN capabilities.

In LIN master mode the UARTE is completely controlled by the LIN Master Controller, all data transfers between the Tx and Rx buffers are managed by the LIN Master Controller.

The UARTE interrupt signals INTUAEIT, INTUAEIR and INTUAEIS are handled by the LIN Master Controller, which generates the interrupt signals INTLMATIT, INTLMATIR and INTLMATIS towards the microcontroller's Interrupt Controller.

For a detailed description of the UARTE refer to the chapter "Asynchronous Serial Interface E (URTE)".

For using the scheduler and the automatic frame start function, a free-running counter is connected to the LIN Master Controller. Refer to the section "LIN Master Scheduler Counter" earlier in this chapter.

24.4 Functional Description

The LIN Master can be set up in three different basic modes by setting the LMAnCTLL.LMAnMD[1:0] bits:

- LMAnCTLL.LMAnMD[1:0] = 00_B: UARTEn through mode
  In this mode the LMAn is bypassed and the connected UARTEn is operated as without LMAn connected.
- LMAnCTLL.LMAnMD[1:0] = 01_B: UARTEn buffer mode
  In this mode the LMAn operates as a UARTEn Rx and Tx buffer, each providing a storage capacity of 12 byte.
- LMAnCTLL.LMAnMD[1:0] = $1x_B$: LIN master mode
  In this mode the LMAn operates in combination with the UARTEn as a LIN master bus controller, providing Rx and Tx buffers of 12 bytes each in order to handle entire LIN frame transactions without any interaction by the CPU.

24.4.1 UART through mode

In UART through mode (LMAnCTLL.LMAnMD[1:0] = 00_B) the LMAn is bypassed and the UART is used without any LMAn functions.

As this is the default LMAn mode, the UARTEn can be controlled and operated without any LMAn intervention.

Note In order to keep power consumption at a minimum, it is recommended to keep LMAnCTLH.LMAnPW = 0.

Interrupts All LMAn interrupts request are identical to the UART interrupt requests:

- transmission interrupt request: INTLMAnTIT = INTUAEnTIT
- reception interrupt request: INTLMAnTIR = INTUAEnTIR
- status interrupt request: INTLMAnTIS = INTUAEnTIS

Data transmission Data to be transmitted is written to the UARTEn transmit data register URTEnTX.

Data reception Received data is read from the UARTEn receive data register URTEnRX.

The UARTE status registers URTEnSTR0 and URTEnSTR1 provide information about data transaction status and error detections.

24.4.2 UART buffer mode

In this mode the UARTEn - LMAn combination acts a UARTEn with Rx and Tx buffers of 12 byte size each. This mode is a full-duplex mode, thus receive and transmit transactions are separately controllable and are handled simultaneously.

(1) Initialization

UARTE settings

The UARTEn must be set up as follows:

- URTEnCTL2
  - URTEnPRS[2:0], URTEnBRS[11:0]: baudrate setting
- URTEnCTL1
  - URTEnSLBM = 1: BF reception during data reception
- URTEnCTL0
  - URTEnPW = 1: UARTEn enabled
  - URTEnCTL0.URTEnTXE = x: transmission enabled/disabled
  - URTEnCTL0.URTEnRXE = x: reception enabled/disabled
  - URTEnCTL0.URTEnSLDC = 0: no data consistency check
  - URTEnSLIT = 0: transmission interrupt request at start of transmission

All other UARTEn settings can be set as required.

LMAn settings

The LMAn must be set up as follows:

- LMAnCTLL
  - LMAnMD[1:0] = 01_B: UART buffer mode
  - LMAnACSE = 0: automatic checksum disabled
  - LMAnSCHE = 0: scheduler disabled
  - LMAnAFE = 0: automatic frame start function disabled
  - LMAnITMK = 0: INTLMAnTIT not masked
  - LMAnIRMK = 0: INTLMAnRIT not masked
- LMAnCTLH
  - LMAnPW = 1: LMAn enabled

(2) Interrupts

INTLMAnTIT

The transmission interrupt request is generated if the number of data, set up in the Tx buffer and specified by LMAnTCTLL.LMAnTLG[3:0], have been transmitted.

INTLMAnTIR

The reception interrupt request is generated if the number of data, specified by LMAnRCTLL.LMAnRLG[3:0], have been stored in the Rx buffer.
In case of continuous reception (LMAnRCTLL.LMAnRLG[3:0] = 0), INTLMAnTIR is generated after storage of the 12th data, i.e. when the Rx buffer is full.

INTLMAnTIS The status interrupt request is generated under following conditions:

- UART has detected an error during data reception:
  - parity error: URTEnSTR1.URTEnPE = 1
  - framing error: URTEnSTR1.URTEnFE = 1
  - overrun error: URTEnSTR1.URTEnOVE = 1

(3) Data transmission

For data transmission the data to be transmitted have to be written to the Tx buffer (via the LMAnTX01 to LMAnTXAB registers) and the number of bytes to be transmitted has to be specified in LMAnTCTLL.LMAnTLG[3:0] prior starting the transmission by setting the transmit request LMAnTCTLL.LMAnTRQ = 1.

The transmission interrupt request INTLMAnTIT indicates the transmission of the last data byte.

Following values are allowed for the transmit length:

- LMAnTCTLL.LMAnTLG[3:0] = 0: 12 data bytes are transmitted
- LMAnTCTLL.LMAnTLG[3:0] = 1 to 12: 1 to 12 data bytes are transmitted

Setting LMAnTCTLL.LMAnTLG[3:0] > 12 is prohibited.

The following diagram shows the principle transmission process.

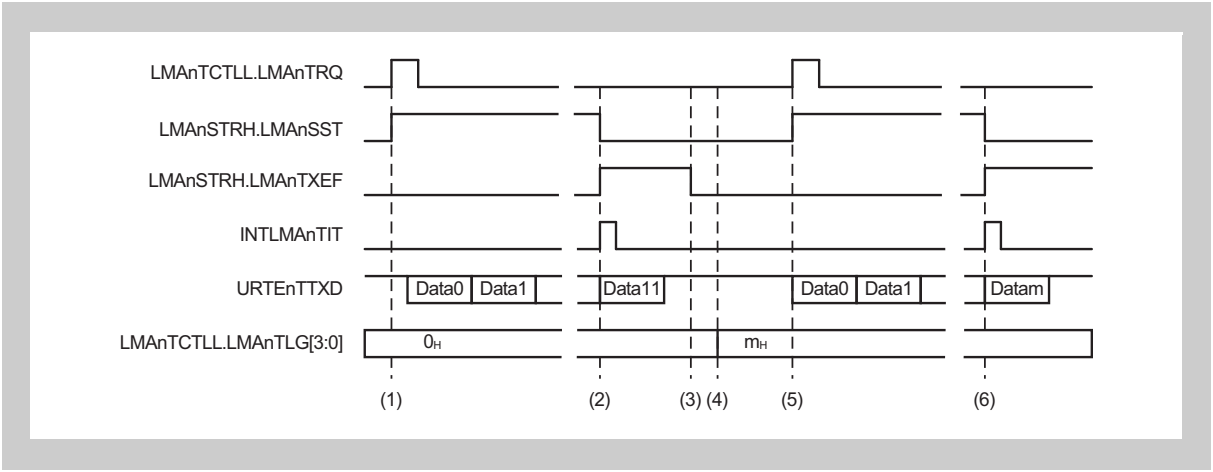


Figure 24-4 Data transmission in UART buffer mode

Precondition LMAn has been

- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- enabled (LMAnCTLH.LMAnPW = 1)
- Tx buffer empty flag LMAnSTRH.LMAnTXEF is cleared

Procedure

1. Write 12 bytes of data to the Tx buffer LMAnTX01 to LMAnTXAB and set the transmit request LMAnTCTLL.LMAnTRQ = 1. Data Tx starts afterwards.
   The start of transmission is indicated by LMAnSTRH.LMAnSST = 1.
   Since LMAnTCTLL.LMAnTLG[3:0] = 0, 12 byte will be transmitted.
2. Upon Tx start of the last data byte Data11 the transmission interrupt request INTLMAnTIT is asserted and LMAnSTRH.LMAnTXEF = 1 to indicate Tx buffer empty. LMAnSTRH.LMAnSST is cleared to 0.

3. Clear the Tx buffer empty flag by LMAnSTCH.LMAnCLTXEF = 1.
4. The next Tx buffer is prepared to transmit m byte of data by writing m data byte to the Tx buffer and set LMAnTCTLL.LMAnTLG[3:0] = $0C_H$.
5. Afterwards the next transmission is started by LMAnTCTLL.LMAnTRQ = 1.
6. Upon Tx start of the m^{th} data byte Datam the transmission interrupt request INTLMAnTIT is asserted and LMAnSTRH.LMAnTXEF = 1 to indicate Tx buffer empty. LMAnSTRH.LMAnSST is cleared to 0.

Note No errors are detected and indicated in case a transmit request is issued, though Tx buffer empty is indicated (LMAnSTRH.LMAnTXEF = 1) or the Tx length is set to incorrect values (LMAnTCTLL.LMAnTLG[3:0] > $0C_H$).

Tx abort For stopping an ongoing data transmission, a Tx abort request must be issued by LMAnTCTLH.LMAnTAB = 1. No new data from the Tx buffer is sent to the UARTE and LMAnSTRH.LMAnSST is cleared.
The UARTE completes any ongoing data transmissions. The UARTE Tx completion can be confirmed by URTEnSTR0.URTEnSST = 0.

Caution After setting an Tx abort request LMAnTCTLH.LMAnTAB = 1, a transmit interrupt request may occur. Thus mask INTLMAnTIT in the Interrupt Controller before the Tx abort request.

(4) Data reception

For data reception the number of bytes to be received has to be specified in LMAnRCTLL.LMAnRLG[3:0] prior starting the reception by setting the receive request LMAnRCTLL.LMAnRRQ = 1.
The reception interrupt request INTLMAnTIR indicates the reception of the last data byte.

Following values are allowed for the transmit length:

- LMAnRCTLL.LMAnRLG[3:0] = 0: continuous Rx mode
  In this mode received data are continuously stored to the Rx buffer without the need to set further receive requests (LMAnRCTLL.LMAnRRQ = 1).Each time 12 data bytes have been stored in the Rx buffer, a reception interrupt request INTLMAnTIR is asserted.
- LMAnRCTLL.LMAnRLG[3:0] = 1 to 12: 1 to 12 data bytes are stored in the Rx buffer

Setting LMAnRCTLL.LMAnTLG[3:0] > 12 is prohibited.

The following diagram shows the principle reception process.

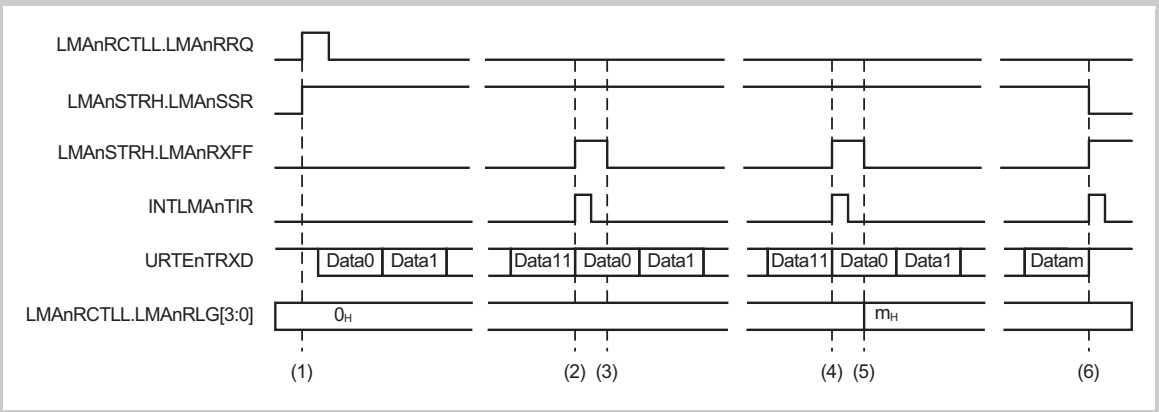


Figure 24-5 Data reception in UART buffer mode

Precondition LMAn has been

- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- enabled (LMAnCTLH.LMAnPW = 1)
- Rx buffer full flag LMAnSTRH.LMAnRXFF is cleared

Procedure

1. Set the receive request LMAnRCTLL.LMAnRRQ = 1. Data Rx starts afterwards.
   The start of reception is indicated by LMAnSTRH.LMAnSSR = 1.
   Since LMAnRCTLL.LMAnRLG[3:0] = 0, 12 byte will be received in continuous reception mode.
2. Upon Rx of the last data byte 11 the reception interrupt request INTLMAnTIR is asserted and LMAnSTRH.LMAnRXFF = 1 to indicate Rx buffer full.
   Because of continuous reception mode, Data1 is received without a new receive request.
3. The Rx buffer needs to be read via the LMAnRX00 to LMAn RXAB registers and the Rx buffer full flag is cleared by LMAnSTCH.LMAnCLRXFF = 1.
4. After reception of the next 12 data byte, only m data bytes shall be received and reception shall be stopped. Therefore Rx buffer length is set to LMAnRCTLL.LMAnRLG[3:0] = m and the Rx buffer full flag is cleared by LMAnSTCH.LMAnCLRXFF = 1.
5. If the m data bytes have been stored in the Rx buffer, reception is stopped (LMAnSTRH.LMAnSSR = 0).

Rx abort For stopping an ongoing data reception, a Rx abort request must be issued by LMAnRCTLH.LMAnRAB = 1. No new data is stored to the TR buffer, and LMAnSTRH.LMAnSSt is cleared.
The UARTE completes any ongoing data reception, but the finally received data is not stored in the Rx buffer. The UARTE Rx completion can be confirmed by URTEnSTR0.URTEnSSR = 0.

Caution After setting an Rx abort request LMAnRCTLH.LMAnRAB = 1, a receive interrupt request may occur. Thus mask INTLMAnTIR in the Interrupt Controller before the Rx abort request.

(5) UARTE Rx errors

Error detections during data reception can be initiated by the UARTE as well as by the LMAn.

UARTEn errors

If the UARTE detects a parity, framing or overrun error during data reception, the received data is stored to the Rx buffer and the assigned Rx data error flag LMAnSTRL.LMAnRXBE[11:00] is set. The number of RXBE[11:00] is associated to the data byte in the Rx buffer:

- LMAnSTRL.LMAnRXBE[00] = 1 for LMAnRX01.LMAnRX00B[7:0] error
- ...
- LMAnSTRL.LMAnRXBE[11] = 1 for LMAnRXAB.LMAnRX11B[7:0] error

If the specified number of data bytes (LMAnRCTLL.LMAnRLG[3:0]) have been stored in the Rx buffer, the status interrupt request INTLMAnTIS is asserted and reception stops (LMAnSTRH.LMAnSSR = 0), even if operating in continues reception mode (LMAnRCTLL.LMAnRLG[3:0] = 0).

Note that in this case no receive interrupt request INTLMAnTIR is generated.

The Rx error flags LMAnSTRL.LMAnRXBE[11:00] are cumulative, i.e. each reception error, that is detected until the specified number of bytes are received, set its error flag. Upon the end of reception, indicated by INTLMAnTIS, LMAnSTRL.LMAnRXBE[11:00] can be read in order to identify all Rx buffer data, that have causes error flag settings.

The following diagram shows an example in continuous reception mode (LMAnRCTLL.LMAnRLG[3:0] = 0).

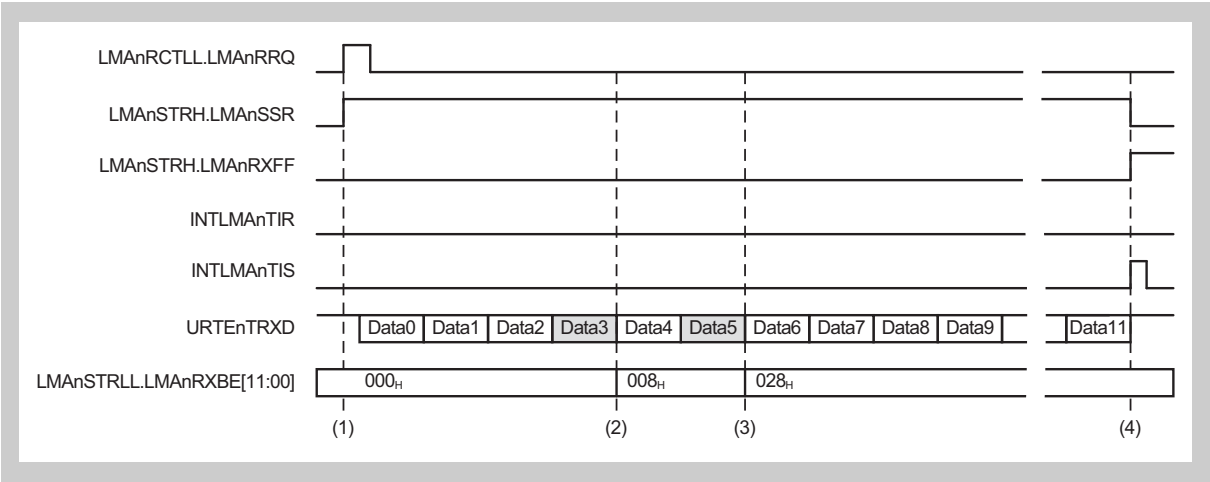


Figure 24-6 Data reception in UART buffer mode with UART reception errors

Precondition LMAn has been

- set into UART buffer mode (LMAnCTLL.LMAnMD[1:0] = 01_B)
- enabled (LMAnCTLH.LMAnPW = 1)
- Rx buffer full flag LMAnSTRH.LMAnRXFF is cleared
- Rx error flags LMAnSTRL.LMAnRXBE[11:00] has been cleared

Procedure

1. Rx is started by LMAnRCTLL.LMAnRRQ = 1.
2. During reception of Data3 the UARTE has detected an error. Data3 is stored to the Rx buffer LMAnRX23.LMAnRX03B[7:0] and the associated error flag LMAnSTRL.LMAnRXBE[03] is set, thus LMAnSTRL.LMAnRXBE[11:00] = 008_H.
3. During reception of Data5 the UARTE has detected another error. Data5 is stored to the Rx buffer LMAnRX45.LMAnRX05B[7:0] and the associated error flag LMAnSTRL.LMAnRXBE[05] is set, thus LMAnSTRL.LMAnRXBE[11:00] = 028_H.
4. After reception of the last data Data11, the status interrupt INTLMAnTIS is generated (instead of INTLMAnTIR) and the Rx process is stopped (LMAnSTRH.LMAnSSR = 0).

LMAn Rx buffer overflow error If the Rx buffer full flag LMAnSTRH.LMAnRXFF is set while new data has been received,

- the Rx buffer overflow flag LMAnSTRH.LMAnROVE is set
- the status interrupt request INTLMAnTIS is asserted
- the received data is not stored in the Rx buffer
- Rx process is stopped (LMAnSTRH.LMAnSSR = 0).

Since the received data, that is not stored in the Rx buffer, remains in the URATE's receive register URTEnRX, it is possible to read from there. However if a next data is received, this data is lost since the UARTE discards this data, i.e. it does not overwrite the URTEnRX register, but sets its overrun error flag URTEmSTR1.URTEnOVE = 1.

24.4.3 LIN master modes

In this mode the LMAn in combination with the UARTE provides a LIN master interface with automatic LIN master Tx and Rx frame transactions, including sending of Break and Sync Fields (BF and SF) to initiate a LIN master frame transaction, automatic checksum functions, scheduler and automatic frame start facilities. The separate 12-byte Tx and Rx buffers allow complete LIN frame transactions without any intervention from the CPU side.

Principle of operation

A LIN master frame transactions is initiated by sending the frame header (BF and SF). This is done automatically when the transaction is started.

In either LIN master Tx or Rx mode the PID is be written to the Tx buffer.

In *Tx mode* the data to be transmitted (maximum 8 data bytes) has to written to the Tx buffer and the length of the Tx frame has to be specified in LMAnTCTLL.LMAnTLG[3:0].
If *automatic checksum* is enabled, the checksum is automatically calculated and appended to the Tx data bytes in the Tx buffer. In case automatic checksum is disabled, the CPU needs to calculate the checksum and to write it to the Tx buffer.
After start of the frame transmission by setting the transmit request LMAnTCTLL.LMAnTRQ = 1 transmission starts sending “BF - SF - PID - TxData bytes - Tx Checksum”.
Simultaneously the sent data is received and stored in the Rx buffer for performing the *data consistency check*, which is executed during transmission of the entire LIN frame.
If automatic checksum calculation is enabled, the checksum of the received data is automatically calculated and checked against the received checksum in the Rx buffer. In case of mismatch a checksum error is reported. If automatic checksum calculation is disabled, the checksum over the received data needs to be calculated and checked against the received checksum by the CPU.

In *Rx mode* the frame length to be received (maximum 8 data bytes) has to be specified in LMAnTCTLL.LMAnTLG[3:0].
Frame reception is also started by setting the transmit request LMAnTCTLL.LMAnTRQ = 1. The LIN master starts sending “BF - SF - PID” and then waits to receive the number of specified data bytes and finally the checksum from the LIN slave.
If *automatic checksum* calculation is enabled, the checksum of the received data is automatically calculated and checked against the received checksum in the Rx buffer. In case of mismatch a checksum error is reported. If automatic checksum calculation is disabled, the checksum over the received data needs to be calculated and checked against the received checksum by the CPU.

Note In either LIN master Tx or Rx mode the frame transaction control is done via the Tx control register LMAnTCT.

If the *scheduler* is enabled, status interrupts are generated in defined periods to ensure a minimum LIN interframe space (which is the time between two LIN frames). This minimum interframe space may be required by some slaves. For that purpose the length of the LIN frame slot length FRSL, which spans the LIN frame plus the interframe space, is to be specified in the Tx buffer prior starting the LIN frame transaction.

If the scheduler and the *automatic frame start function* is enabled, a new LIN frame transaction is automatically started after the LIN frame slot length FRSL.

LIN frame header errors

Two modes are provided to react on errors during the LIN frame header (BF: Break Field, SF: Sync Field) transmission:

- LMAnCTLL.LMAnMD[1:0] = 10_B: LIN master mode without break in header
  Data transfers are carried on after error detections during header transmission.
- LMAnCTLL.LMAnMD[1:0] = 11_B: LIN master mode with break in header
  Data transfers are stopped after error detections during header transmission.

Enhanced checksum

Automatic checksum calculation can be performed in two ways:

- LMAnTCTLL.LMAnSLEC = 0: only the data bytes are used to calculate the checksum (classic checksum)
- LMAnTCTLL.LMAnSLEC = 1: the PID and the data bytes are used to calculate the checksum (enhanced checksum)

LIN frame length

The number of bytes to be transferred, i.e. the length of the LIN frame, needs to be specified in LMAnTCTLL.LMAnTLG[3:0].
The LMAnTCTLL.LMAnTLG[3:0] value includes the number of data bytes (maximum 8), the PID and the checksum. Thus

LMAnTCTLL.LMAnTLG[3:0] = 2 to 10

All other values are prohibited.

Tx/Rx abort

For stopping an ongoing data transaction, a Tx abort request must be issued by LMAnTCTLH.LMAnTAB = 1.
No new data from the Tx buffer is sent to the UARTE respectively received from the UARTE and stored in the Rx buffer.LMAnSTRH.LMAnSST is cleared. The UARTE completes any ongoing data transmissions. The UARTE data transmission/reception completion can be confirmed by URTEnSTR0.URTEnSST =URTEnSTR0.URTEnSST = 0.

Caution

After setting an Tx abort request LMAnTCTLH.LMAnTAB = 1, a transmit, reception or status interrupt request may occur. Thus mask INTLMAnTIT, INTLMAnTIR, and INTLMAnTIS in the Interrupt Controller before the Tx abort request.

(1) Initialization

CNTAn settings If the scheduler is to be used, the scheduler counter must be set up as follows:

- CNTAmCTL
  - CNTAmPW = 1: counter enabled
- CNTAmCFG
  - CNTAmPRS[15:00]: division ratio

UARTE settings The UARTEn must be set up as follows:

- URTEnCTL2
  - URTEnPRS[2:0], URTEnBRS[11:0]: baudrate setting
- URTEnCTL1
  - URTEnSLBM = 0: no BF reception during data reception
  - URTEnBLG[2:0] = 0: BF bit length
  - URTEnCLG = 1: 8-bit data
  - URTEnSLP[1:0] = 00_B: no parity
  - URTEnTDL = 0: no Tx data inversion
  - URTEnRDL = 0: no Rx data inversion
  - URTEnSLG = 0: 1 stop bit
  - URTEnSLD = 1: LSB first
  - URTEnSLIT = 0: transmission interrupt request at start of transmission
- URTEnCTL0
  - URTEnPW = 1: UARTEn enabled
  - URTEnCTL0.URTEnTXE = 1: transmission enabled
  - URTEnCTL0.URTEnRXE = 1: reception enabled
  - URTEnCTL0.URTEnSLDC = 1: data consistency check enabled

LMAn settings The LMAn must be set up as follows:

- LMAnCTLL
  - LMAnMD[1:0] = $1x_B$: LIN master mode
  - LMAnACSE = x: automatic checksum enabled/disabled
  - LMAnSCHE = x: scheduler enabled/disabled
  - LMAnAFE = x: automatic frame start function enabled/disabled
  - LMAnITMK = x: INTLMAnTIT masked/not masked
  - LMAnIRMK = x: INTLMAnRIT masked/not masked
- LMAnCTLH
  - LMAnPW = 1: LMAn enabled

(2) Interrupts

Since both transmission and reception are involved during a LIN master frame transaction, transmit (INTLMAnTIT) and receive (INTLMAnRIT) interrupts are generated in Tx as well as in Rx mode.

Note Both interrupt requests can be separately suppressed:

- LMAnCTLL.ITMK = 1: INTLMAnTIT is masked and will not be generated
- LMAnCTLL.IRMK = 1: INTLMAnRIT is masked and will not be generated

INTLMAnTIT

- in Tx mode (LMAnTCTLL.LMAnSLRT = 0)
  INTLMAnTIT is generated if the number of data, specified by LMAnTCTLL.LMAnTLG[3:0], have been transmitted.
- in Rx mode (LMAnTCTLL.LMAnSLRT = 1)
  if SF has been transmitted

INTLMAnTIR

- in Tx mode (LMAnTCTLL.LMAnSLRT = 0)
  Generation of INTLMAnTIR depends also on the auto-checksum function. If the checksum has been received for checksum control and
  - auto-checksum function is disabled (LMAnCTL.LMAnACSE = 0), INTLMAnTIR is always asserted.
  - auto-checksum function is enabled (LMAnCTL.LMAnACSE = 1), INTLMAnTIR is asserted, if the received checksum matches the automatically calculated checksum. In case of a mismatch a checksum error is indicated (LMAnSTRH.LMAnFCSE = 1) and the status interrupt request INTLMAnTIS is generated instead.
- in Rx mode (LMAnTCTLL.LMAnSLRT = 1)
  INTLMAnTIR is generated if the number of data, specified by LMAnTCTLL.LMAnTLG[3:0], have been stored in the Rx buffer.

INTLMAnTIS The status interrupt request is generated under various conditions:

- UARTE has detected a framing error.
- UARTE has detected an overrun error.
- UARTE has detected a data consistency error.
- UARTE has detected a BF transmission error.
- UARTE has detected a SF transmission error.
- LMAn has detected an auto-checksum error.
- LMAn has detected a buffer preparation error.
- LMAn scheduler ready event occurred.

(3) Data transmission

The LIN master Tx mode is selected by LMAnCTLL.LMAnMD[1:0] = $1x_B$ and LMAnTCTLL.LMAnSLRT = 0.

For transmission of a LIN master frame the Tx buffer has to be prepared in the following format prior starting the frame transmission:

Table 24-13 Tx buffer preparation in LIN master Tx mode

Tx buffer register		Tx buffer for 8 data byte	Tx buffer for 5 data byte
LMAnTXAB.	LMAnTX11B[7:0]	FRSLH[a]	FRSLH[a]
	LMAnTX10B[7:0]	FRSLL[a]	FRSLL[a]
LMAnTX89.	LMAnTX9B[7:0]	Tx checksum[b]	–
	LMAnTX8B[7:0]	TxData7	–
LMAnTX67.	LMAnTX7B[7:0]	TxData6	–
	LMAnTX6B[7:0]	TxData5	Tx checksum[b]
LMAnTX45.	LMAnTX5B[7:0]	TxData4	TxData4
	LMAnTX4B[7:0]	TxData3	TxData3
LMAnTX23.	LMAnTX3B[7:0]	TxData2	TxData2
	LMAnTX2B[7:0]	TxData1	TxData1
LMAnTX01.	LMAnTX1B[7:0]	TxData0	TxData0
	LMAnTX0B[7:0]	PID	PID

a) The frame slot length FRSLL/FRSLH is only effective if the scheduler is enabled (LMAnCTLL.LMAnSCHE = 1).

b) The Tx checksum is automatically stored, if automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1). Otherwise the checksum needs to be calculated and stored by software.

After starting the LIN frame transmission the sent data is stored in the Rx buffer for checksum confirmation. When the entire frame has been transmitted, the Rx buffer looks as follows:

Table 24-14 Rx buffer after LIN frame transmission

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRxAB.	LMAnRx11B[7:0]	–	–
	LMAnRx10B[7:0]	–	–
LMAnRx89.	LMAnRx9B[7:0]	Rx checksum	–
	LMAnRx8B[7:0]	RxData7	–
LMAnRx67.	LMAnRx7B[7:0]	RxData6	–
	LMAnRx6B[7:0]	RxData5	Rx checksum
LMAnRx45.	LMAnRx5B[7:0]	RxData4	RxData4
	LMAnRx4B[7:0]	RxData3	RxData3
LMAnRx23.	LMAnRx3B[7:0]	RxData2	RxData2
	LMAnRx2B[7:0]	RxData1	RxData1
LMAnRx01.	LMAnRx1B[7:0]	RxData0	RxData0
	LMAnRx0B[7:0]	PID	PID

If automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1), the checksum (as selected by LMAnTCTLL.LMAnSLEC) is calculated and compared with the received checksum. Otherwise the checksum needs to be calculated and compared by software.

The following diagram shows the principle LIN frame transmission process with the maximum of 8 data byte.

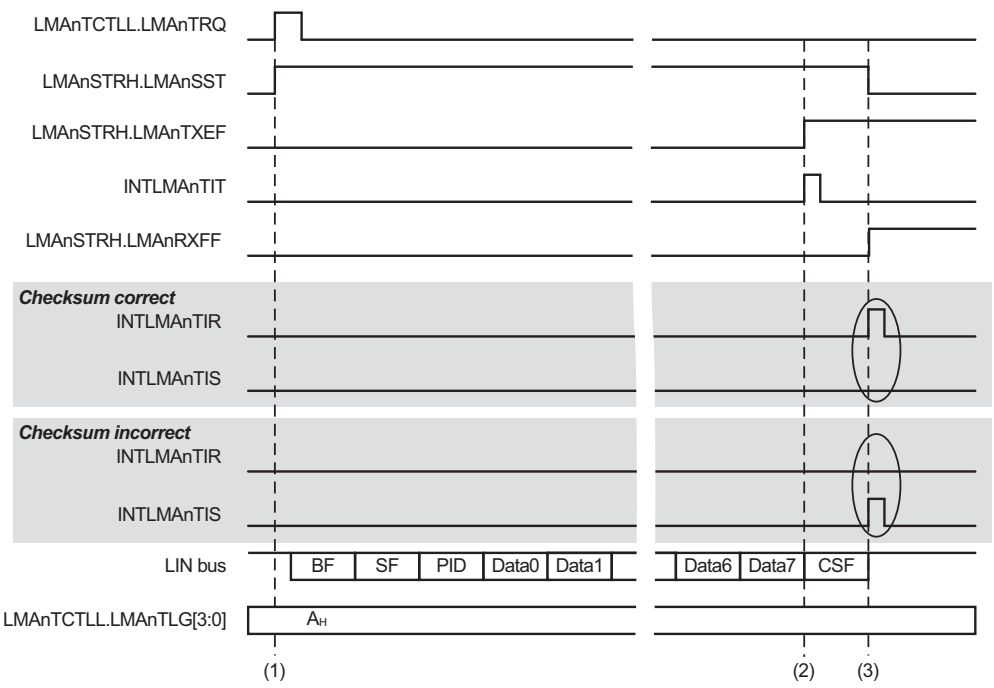


Figure 24-7 LIN frame transmission in LIN master mode

Precondition LMAn has been

- set into LIN master Tx mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$, LMAnTCTLL.LMAnSLRT = 0)
- neither scheduler and, thus, nor automatic frame start function used (LMAnSCHE and LMAnAFE of LMAnCTLL register both cleared)
- enabled (LMAnCTLH.LMAnPW = 1)
- Tx buffer empty flag LMAnSTRH.LMAnTXEF is cleared
- Rx buffer full flag LMAnSTRH.LMAnRXFF is cleared

Procedure

1. Write the PID to Tx buffer LMAnTX01.LMAnTX00B[7:0] and 8 bytes of data to the Tx buffer LMAnTX01.LMAn01B[7:0] to LMAnTX89.LMAn08B[7:0]. If automatic checksum is disabled (LMAnCTLL.LMAnACSE = 0), calculate the checksum CSF (over the bytes including respectively excluding PID, depending on LMAnTCTLL.LMAnSLEC) and store it to LMAnTX89.LMAn09B[7:0]. Otherwise CSF will be calculated and added automatically.
   Set the frame length LMAnTCTLL.LMAnTLG[3:0] = A_H (10 = PID + 8 data bytes + CSF).
   Start frame transmission by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
2. Upon start of the Checksum Field (CSF) transmission, Tx buffer empty is indicated by LMAnSTRH.LMAnTXEF = 1 and the transmit interrupt request INTLMAnTIT is asserted.
3. After the checksum field CSF has been transmitted, the Rx buffer full flag LMAnSTRH.LMAnRXFF is set and checksum control is performed,

provided automatic checksum calculation is enabled (LMAnCTLL.LMAnACSE = 1.
If the checksum is correct, the receive interrupt request INTLMAnTIR is asserted.
If the checksum is incorrect, the status INTLMAnTIS instead of the receive interrupt request is asserted.

(4) Data reception

The LIN master Rx mode is selected by LMAnCTLL.LMAnMD[1:0] = $1x_B$ and LMAnTCTLL.LMAnSLRT = 1.

For data reception in LIN master mode the Tx buffer has to be prepared in the following format prior starting the frame reception:

Table 24-15 Tx buffer preparation in LIN master Rx mode

Tx buffer register		Tx buffer
LMAnTXAB.	LMAnTX11B[7:0]	FRSLH[a]
	LMAnTX10B[7:0]	FRSLL[a]
LMAnTX89.	LMAnTX9B[7:0]	–
	LMAnTX8B[7:0]	–
LMAnTX67.	LMAnTX7B[7:0]	–
	LMAnTX6B[7:0]	–
LMAnTX45.	LMAnTX5B[7:0]	–
	LMAnTX4B[7:0]	–
LMAnTX23.	LMAnTX3B[7:0]	–
	LMAnTX2B[7:0]	–
LMAnTX01.	LMAnTX1B[7:0]	–
	LMAnTX0B[7:0]	PID

a) The frame slot length FRSLL/FRSLH is only effective if the scheduler is enabled (LMAnCTLL.LMAnSCHE = 1).

After starting the LIN frame reception the LIN master frame header (BF, SF, PID) is transmitted to the slaves and the data, received from the slave afterwards, is stored in the Rx buffer. When the entire frame has been transmitted, the Rx buffer looks as follows:

Table 24-16 LIN master Rx buffer after LIN frame reception (1/2)

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRxAB.	LMAnRx11B[7:0]	–	–
	LMAnRx10B[7:0]	–	–
LMAnRx89.	LMAnRx9B[7:0]	Rx checksum	–
	LMAnRx8B[7:0]	RxData7	–
LMAnRx67.	LMAnRx7B[7:0]	RxData6	–
	LMAnRx6B[7:0]	RxData5	Rx checksum
LMAnRx45.	LMAnRx5B[7:0]	RxData4	RxData4
	LMAnRx4B[7:0]	RxData3	RxData3

Table 24-16 LIN master Rx buffer after LIN frame reception (2/2)

Rx buffer register		Rx buffer for 8 data byte	Rx buffer for 5 data byte
LMAnRx23.	LMAnRx3B[7:0]	RxData2	RxData2
	LMAnRx2B[7:0]	RxData1	RxData1
LMAnRx01.	LMAnRx1B[7:0]	RxData0	RxData0
	LMAnRx0B[7:0]	PID	PID

If automatic checksum is enabled (LMAnCTLL.LMAnACSE = 1), the checksum (as selected by LMAnTCTLL.LMAnSLEC) is calculated and compared with the received checksum. Otherwise the checksum needs to be calculated and compared by software.

The following diagram shows the principle LIN frame reception process with the maximum of 8 data byte.

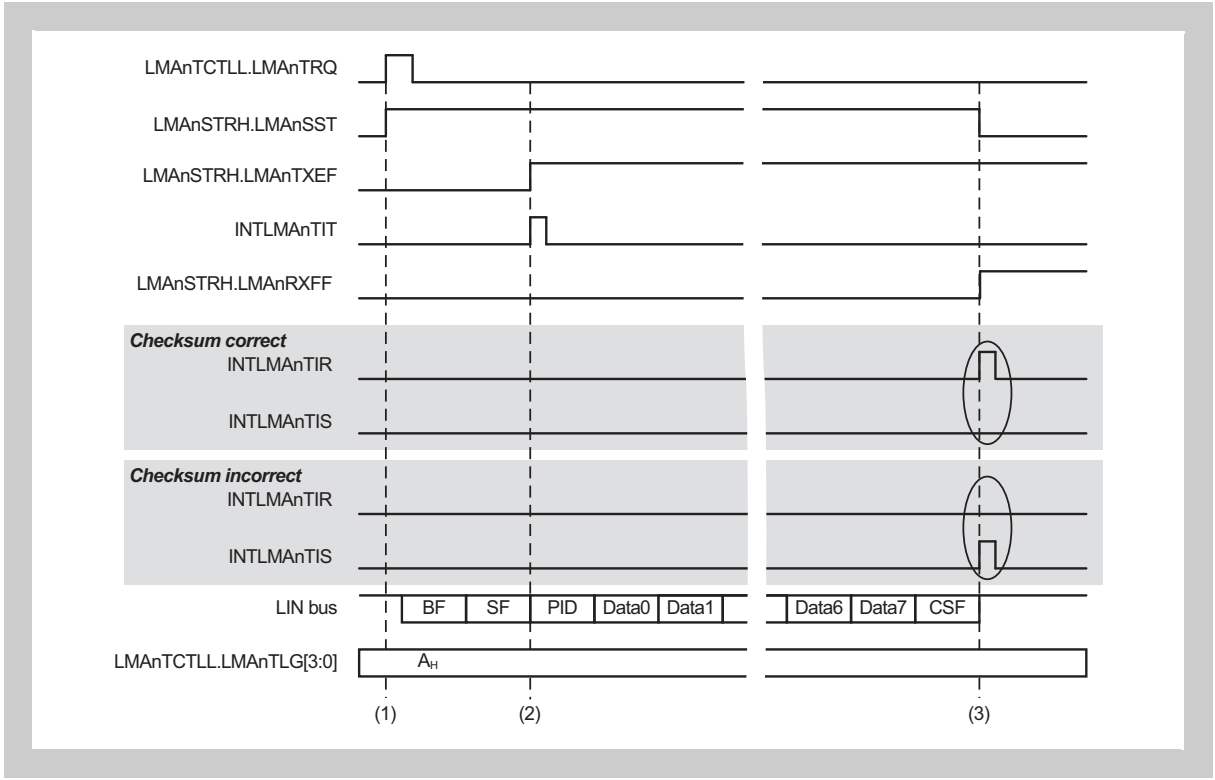


Figure 24-8 LIN frame reception in LIN master mode

Precondition LMAn has been

- set into LIN master Rx mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$, LMAnTCTLL.LMAnSLRT = 1)
- neither scheduler and, thus, nor automatic frame start function used (LMAnSCHE and LMAnAFE of LMAnCTLL register both cleared)
- enabled (LMAnCTLH.LMAnPW = 1)
- Tx buffer empty flag LMAnSTRH.LMAnTXEF is cleared
- Rx buffer full flag LMAnSTRH.LMAnRXFF is cleared

Procedure

1. Write the PID to Tx buffer LMAnTX01.LMAnTX00B[7:0].
   The number of data
   Start frame reception by LMAnTCTLL.LMAnTRQ = 1, and transmission start of the frame header is indicated by LMAnSTRH.LMAnSST = 1.
2. Upon start of the PID transmission, Tx buffer empty is indicated by LMAnSTRH.LMAnTXEF = 1 and the transmit interrupt request INTLMAnTIT is asserted.
3. After the checksum field CSF has been transmitted, the Rx buffer full flag LMAnSTRH.LMAnRXFF is set and checksum control is performed, provided automatic checksum calculation is enabled (LMAnCTLL.LMAnACSE = 1.
   If the checksum is correct, the receive interrupt request INTLMAnTIR is asserted.
   If the checksum is incorrect, the status INTLMAnTIS instead of the receive interrupt request is asserted.

(5) LIN master mode transaction errors

During LIN master mode transactions the UARTEn as well as the LMAn can detected and indicate various transaction errors.

Table 24-17 LIN master mode transaction errors

Detection module	Error	Indicator	Error cause
UARTE	Framing error	URTEnSTR1.URTEnFE = 1	No stop bit was detected after reception of the 8th bit of the SF, PID, data, or CSF (checksum) byte. A framing error generates also a data consistency error.
	Overrun error	URTEnSTR1.URTEnOVE = 1	New data received, while the UARTE receive register URTEnRX holds data, that has not been stored in the Rx buffer.
	Data consistency error	URTEnSTR1.URTEnDCE = 1	Data sent during transmission is erroneous. A data consistency error during BF transmission is indicated as a BF transmission error. A data consistency error during SF transmission is indicated as a SF transmission error. Note that any framing error generates also a data consistency error.
	BF transmission error	URTEnSTR1.URTEnBSF = 0 LMAnSTRH.LMANBFE = 1	Data consistency occurs during BF Tx
	SF transmission error	URTEnSTR1.URTEnDCE = 1 LMAnSTRH.LMANSFE = 1	Data consistency error occurred during SF transmission.
LMAn	Auto-checksum error	LMAnSTRH.LMAnFCSE = 1	Calculated checksum does not match the received checksum after completion of a LIN frame transaction. provided the auto-checksum function is enabled by LMAnCTLL.LMAnACSE = 1.
	Rx/Tx buffer preparation error	LMAnSTRH.LMAnPIE = 1	LIN frame transaction is started (by LMAnTCTLL.TRQ = 1 or in auto frame start mode), while the Tx and Rx buffer are not set up correctly, i.e. • Tx buffer empty (LMAnSTRH.LMAnTXEF = 1) • Rx buffer full (LMAnSTRH.LMAnRXFF = 1) • transmit length incorrect (LMAnTCTLL.LMAnTLG[3:0] = 0, 1, 11 to 15)

The procedure taking place upon an UARTE error detection depends on the mode the LMAn is operating in and the type of erroneous data.

The main difference between LIN master mode with and without break in header refers to the different behaviour, when a data consistency check error is detected in the LIN frame header (BF/SF) in Rx mode: “with break in header” stops any further transactions, while “without break in header” continuous transactions.

- LIN master mode with break in header

  If a *data consistency check error* was detected in any type of data, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

  In case of *framing/overrun error* detections during transmission of the BF/SF transactions are continued.
  If PID/data/checksum are received with framing/overrun errors in Rx mode, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

- LIN master mode without break in header

  If a *data consistency check error* was detected in PID/data/checksum, or during transmission of BF/SF, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated. If data inconsistency during BF/SF in Rx mode, transactions continue, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

  In case of *framing/overrun error* detections during transmission of the BF or SF transactions are continued.
  If PID/data/Checksum are received with framing/overrun errors in Rx mode, further transactions are stopped, the associated Rx buffer error flag LMAnSTRL.LMAnRXBE is set and a status interrupt request INTLMAnTIS is generated.

The following table summarizes these procedures.

Figure 24-9 UARTE errors in LIN master mode

Rx/Tx mode	Framing error[a]/overrun error in		Data consistency check error in	
	BF/SF[b]	PID/data/CSF	BF / SF	PID / Data / CSF
LIN master mode with break in header (LMAnCTLL.LMAnMD[1:0] = 11_B)				
Tx mode (LMAnTCTLL. LMAnSLRT = 0)	Transaction continued		• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted	
Rx mode (LMAnTCTLL. LMAnSLRT = 1)	Transaction continued	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted		
LIN master mode without break in header (LMAnCTLL.LMAnMD[1:0] = 10_B)				
Tx mode (LMAnTCTLL. LMAnSLRT = 0)	Transaction continued		• Transaction stopped after PID transmitted • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted after PID transmitted	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted
Rx mode (LMAnTCTLL. LMAnSLRT = 1)	Transaction continued	• Transaction stopped at STOP bit of erroneous field • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted	• Transaction continued • LMAnSTRL.LMAnRXBE[i] = 1 • INTLMAnTIS asserted after CSF reception completed	

a) A framing error during transmission of any type of data a data consistency error is detected simultaneously (in Tx mode: BF/SF/PID/data/CSF are transmitted, in Rx mode: BF/SF/PID/ are transmitted).

b) Any error detection in BF or SF sets the respective error flag LMAnSTRH.LMAnBFE respectively LMAnSTRH.LMAnSFE.

Note During data reception in LIN Master mode, the LMAn cannot distinguish between missing responses from LIN slaves and incomplete responses from LIN slaves.
In both cases, the data reception will not finish and needs to be aborted by software.
There is no indication, how many data bytes have been received up to the abortion.

24.4.4 Automatic checksum function

The automatic checksum function allows to automatically generate respectively control checksums.

The automatic checksum function is control by:

- LMAnCTLL.LMAnACSE = 0: automatic checksum function disabled
- LMAnCTLL.LMAnACSE = 1: automatic checksum function enabled

In LMAnCTLL.LMAnACSE = 0: automatic checksum function is enabled, the automatic checksum function performs the following:

Tx mode In Tx mode (LMAnTCTLL.LMAnSLRT = 0), the checksum is automatically calculated and appended to the transmit data in the Tx buffer upon LIN frame transaction start.
After transmission completion the checksum the same procedure takes place as in Rx mode (see below).

Rx mode After reception completion the checksum is calculated from the received data and automatically compared to the received checksum, stored in the Rx buffer. In case both match, the receive interrupt request INTLMAnTIR is generated. If they don't match, the status interrupt request INTLMAnTIS is generated and the checksum error flag LMAnSTRH.LMAnFCSE is set.

Checksum format The data incorporated in the calculation of the checksum in automatic checksum mode can be chosen:

- LMAnTCTLL.LMAnSLEC = 0: classic checksum
  Only the data bytes, stored in the Tx or Rx buffer, are used to calculate the checksum.
- LMAnTCTLL.LMAnSLEC = 1: enhanced checksum
  The data bytes and the PID, stored in the Tx or Rx buffer, are used to calculate the checksum.

24.4.5 Scheduler

The scheduler allows to generate status interrupts INTLMAnTIS in defined time distances. INTLMAnTIS can be used to initiate the next LIN master frame transaction.
By this a minimum LIN interframe space (which is the time between two LIN frames) can be ensured. This minimum interframe space may be required by some slaves.

For that purpose the LIN frame slot length FRSL, which spans the LIN frame plus the interframe space, is to be specified in the Tx buffer prior starting the LIN frame transaction.

Scheduler counter

The scheduler makes use of a scheduler counter CNTAm. CNTAm is a free-running counter. Its count clock can be selected by the prescaler CNTAmCFG.CNTAmPS[15:0].
A detailed description of the scheduler counter CNTAm is given in the first section of this chapter in “Scheduler Counter A”.

(1) Scheduler operation

Note

Before any LIN master frame transaction is started with scheduler, the employed scheduler counter CNTAm has to be enabled (CNTAmCTL.CNTAmPWR = 1) and its prescaler has to be set up (CNTAmCFG.CNTAmPS[15:0]). CNTAmCFG.CNTAmPS[15:0] determines the scheduler count clock SCHECLK. The scheduler clocks are counting up the scheduler counters count value CNTAmCNT.

The scheduler needs to be enabled by LMAnCTLL.LMAnSCHE = 1.

Before starting a LIN master frame transaction (LMAnTCTLL.LMAnTRQ = 1) with scheduler function, the length of the 16-bit LIN frame slot length FRSL[15:0] has to be written to the Tx buffer:

- LMAnTXAB.LMAnTX10B[7:0] = FRSLL = FRSL[7:0]
- LMAnTXAB.LMAnTX11B[7:0] = FRSLH = FRSL[15:8]

FRSL[15:0] defines the number of scheduler clocks SCHECLK for the frame slot length.

If the LIN master frame transaction is started, the current value of the scheduler counter CNTAmCNT[15:0] is added to the defined frame slot length FRSL[15:0] and stored to the compare register LMAnCMPL.LMAnCMP[15:0].

The scheduler counter value CNTAmCNT[15:0] is continuously compared with the compare register LMAnCMP[15:0]. If both match, the status interrupt request INTLMAnTIS is generated and the scheduler ready flag LMAnSTRH.LMAnSRF is set.

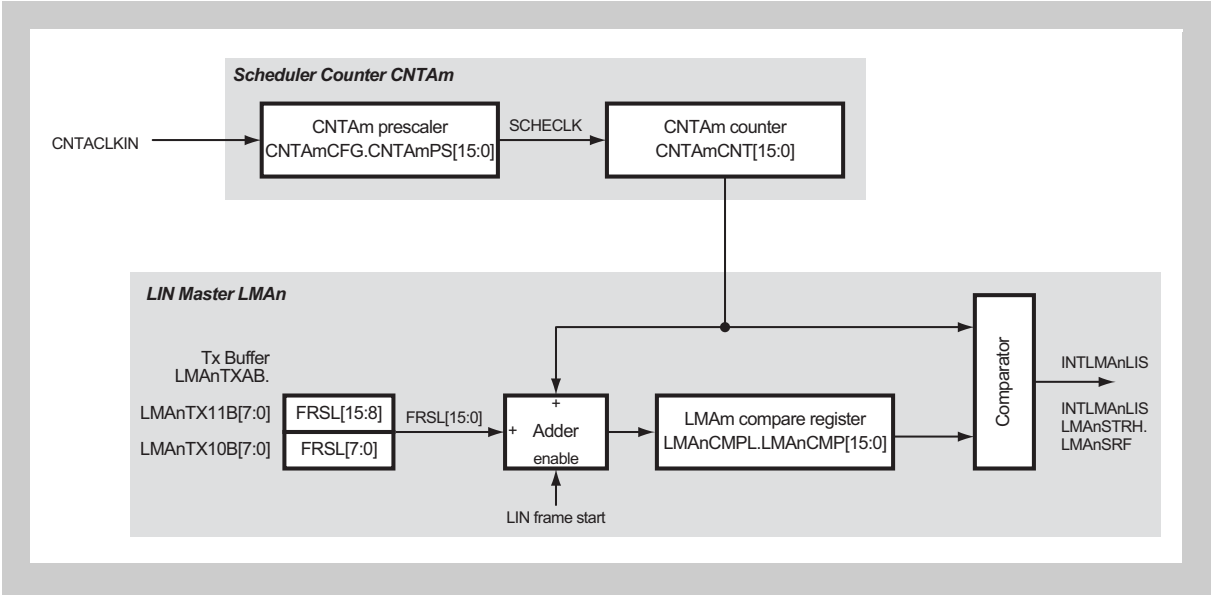


Figure 24-10 Scheduler functional overview

Interrupt handling

Upon occurrence of the status interrupt INTLMAnTIS transaction of a next LIN master frame can be started immediately, provided the frame slot length FRSL includes also a minimum interframe space. Thus the receive and transmit interrupts may not be necessary to be processed and could be suppressed by masking LMAnCTLL.LMAnITMK = LMAnCTLL.LMAnIRMK = 1.

The diagram below shows the principle timing of a LIN frame transmission with use of the scheduler.

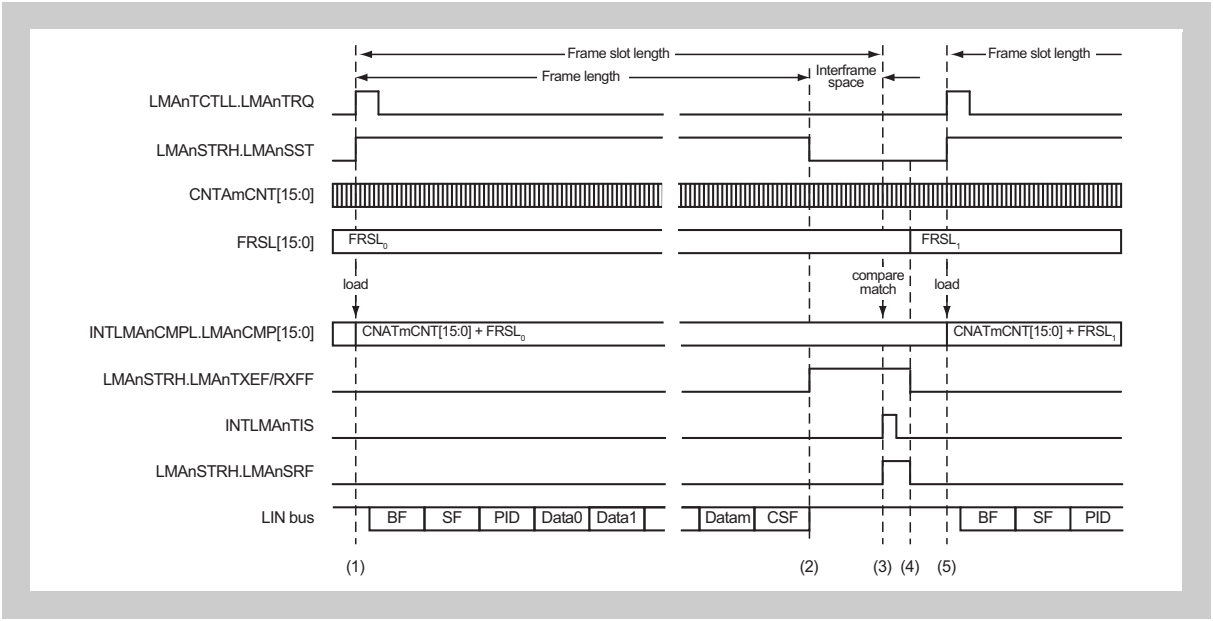


Figure 24-11 LIN frame transaction with scheduler

Precondition

The scheduler counter is operating and scheduler clock frequency has been set up.
LMAn has been

- set into LIN master mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$)
- scheduler is enabled (LMAnCTLL.LMAnSCHE = 1)
- automatic frame start function is disabled (LMAnCTLL.LMAnAFE = 0)
- enabled (LMAnCTLH.LMAnPW = 1)
- Tx buffer empty flag LMAnSTRH.LMAnTXEF and Rx buffer full flag LMAnSTRH.LMAnRXFF are cleared
- Tx buffer has been set up correctly with frame slot length $FRSL_0$

Procedure

1. Start frame transmission by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length $FRSL_0$ is stored to LMAnCMP.CMP[15:0].
2. After transmission of the checksum field CSF, i.e. after the frame length, the Tx buffer is indicated as empty (LMAnSTRH.LMAnTXEF = 1) and the Rx buffer as full (LMAnSTRH.LMAnRXFF = 1).
3. Upon match of the scheduler counter CNTAmCNT[15:0] and LMAnCMP.LMAnCMP[15:0] the status interrupt request INTLMAnTIS indicates the end of the frame slot length and the scheduler ready flag LMAnSTRH.LMAnSRF is set.
   If a sufficient interframe space was regarded, when defining the frame slot length $FRSL_0$, the next frame transaction could already be started at this point in time.
4. The buffer status flags and the scheduler ready flag are cleared (LMAnCLTXEF = LMAnCLRXFF = LMAnCLSRF of LMAnSTCH register set to 1).
   The Tx buffer and its control register LMAnTCTLL are prepared for the next frame transaction, with the next frame slot length $FRSL_1$.
5. Next frame transaction is started by LMAnTCTLL.LMAnTRQ = 1, and transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length $FRSL_1$ is stored to LMAnCMP.CMP[15:0].

(2) Scheduler operation with automatic frame start function

The automatic start function, in combination with the scheduler, allows to eliminate unnecessary gaps on the LIN bus by enabling the CPU to make all preparations within the interframe space. The next LIN frame transaction is started automatically after the interframe space, thus maintaining a maximum data transmission performance on the bus.

The automatic frame start function needs to be enabled by LMAnCTLL.LMAnAFE = 1.
Note that the scheduler has to be enabled as well by LMAnCTLL.LMAnSCHE = 1.

In order to start LIN frame transactions with automatic frame start function the first Tx request bit LMAnTCTLL.LMAnFRQ has to be set to 1 in addition to the first Tx request LMAnTCTLL.LMAnTRQ. After LMAnTRQ has returned to 0, it can be set again to 1, although the frame transfer is ongoing (LMAnSTRH.LMAnSST = 1). Thus the next frame transaction is started immediately after the next interframe space has passed.

During the interframe space all preparations (Rx/Tx buffer setup, etc.) have to be finished for the next frame transaction. These preparations can be triggered by the reception interrupt request INTLMAnTIR.

Caution The reception interrupt request INTLMAnTIR must be used to initiate frame preparations in Tx as well as in Rx mode.

The diagram below shows the principle timing of a LIN frame transmission with use of the scheduler and automatic frame start function.

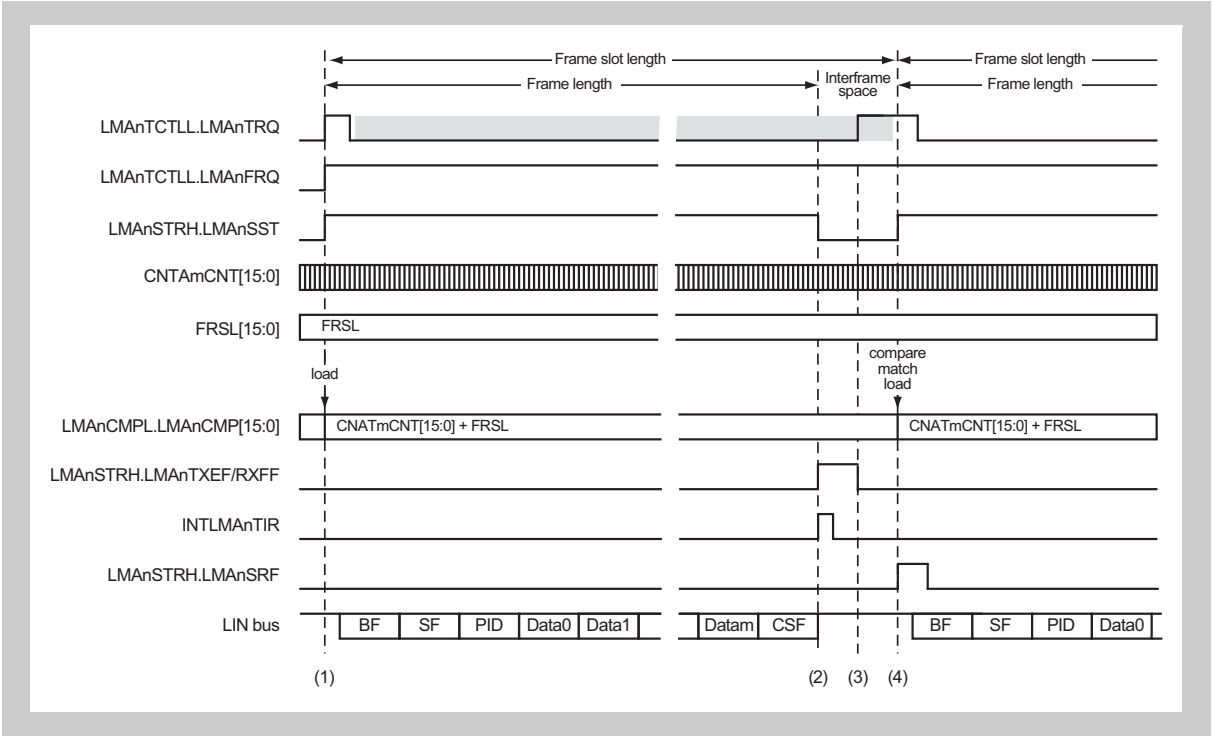


Figure 24-12 LIN frame transmission with scheduler and automatic frame start function

Precondition The scheduler counter is operating and scheduler clock frequency has been set up.
LMAn has been

- set into LIN master mode (LMAnCTLL.LMAnMD[1:0] = $1x_B$)
- scheduler is enabled (LMAnCTLL.LMAnSCHE = 1)
- automatic frame start function is enabled (LMAnCTLL.LMAnAFE = 1)
- enabled (LMAnCTLH.LMAnPW = 1)
- Tx buffer empty flag LMAnSTRH.LMAnTXEF and Rx buffer full flag LMAnSTRH.LMAnRXFF are cleared
- Tx buffer has been set up correctly with frame slot length FRSL

Procedure

1. Start first transmission with automatic frame start function by LMAnTCTLL.LMAnTRQ = 1 and LMAnTCTLL.LMAnFRQ = 1.Transmission start is indicated by LMAnSTRH.LMAnSST = 1.
   Sum of the current schedule counter value CNTAmCNT[15:0] and the frame slot length FRSL is stored to LMAnCMP.CMP[15:0].

After LMAnTCTLL.LMAnTRQ has returned to 0, the next transmission request can be set. At the latest it needs to be set before the interframe space has ended, i.e. in the gray area.
However in the diagram LMAnTRQ is set in the interframe space.

2. After transmission of the checksum field CSF, i.e. after the frame length, the Tx buffer is indicated as empty (LMAnSTRH.LMAnTXEF = 1) and the Rx buffer as full (LMAnSTRH.LMAnRXFF = 1). The reception interrupt request INTLMAnTIR is asserted.
3. All preparations are performed for the next frame transmission.
The buffer status flags and the scheduler ready flag are cleared (LMAnCLTXEF = LMAnCLRXFF = LMAnCLSRF of LMAnSTCH register set to 1).
The next Tx request is set (LMAnTCTLL.LMAnTRQ = 1).
4. Upon match of the scheduler counter CNTAmCNT[15:0] and LMAnCMP.LMAnCMP[15:0], next frame transmission is started.

Preparation error

If the next frame is not completely and correctly prepared,

- LMAnTCTLL.LMAnTRQ and LMAnSTRH.LMAnSST are cleared and frame transaction is not started
- LMAnCMP.LMAnCMP[15:0] is loaded with CNATmCNT[15:0] + FRSL
- the preparation incomplete error flag LMAnSTRH.LMAnPIE is set
- the status interrupt request LMAnTIS is asserted

If frame preparation is completed with start of the next frame, i.e. if CNATmCNT[15:0] = LMAnCMP.LMAnCMP[15:0], frame transmission starts. Otherwise the next LMAnTIS interrupt request is asserted to indicate another preparation incomplete error.

24.5 LMAn Registers

This section contains a description of all registers of the LIN Master A.

24.5.1 LMAn registers overview

The LMAn is controlled and operated by means of the registers, listed in the table below.

One set of 16-bit and one set of 32-bit registers are provided, at which the bits of two 16-bit registers can be accessed via one 32-bit register. Note that the offset addresses for both sets are different.

Throughout this document all descriptions refer to the 16-bit register set. However the described functions of 16-bit registers bits apply also to bits with the same name in their associated 32-bit register.

Table 24-18 LMAn registers

Register function	16-bit access registers		32-bit access registers	
	Name	Address	Name	Address
Control and status register:				
Control register	LMAnCTLL	<LMAn_base> + 80_H	LMAnCTL0W	<LMAn_base> + 180_H
	LMAnCTLH	<LMAn_base> + 84_H		
Status register	LMAnSTRL	<LMAn_base> + 88_H	LMAnSTR0W	<LMAn_base> + 188_H
	LMAnSTRH	<LMAn_base> + $8C_H$		
Status clear register	LMAnSTCL	<LMAn_base> + 90_H	LMAnSTC0W	<LMAn_base> + 190_H
	LMAnSTCH	<LMAn_base> + 94_H		
Compare register	LMAnCMPL	<LMAn_base> + 98_H	LMAnCMP0W	<LMAn_base> + 198_H
	LMAnCMPH	<LMAn_base> + $9C_H$		
Tx control register	LMAnTCTLL	<LMAn_base> + $D8_H$	LMAnTCTL0W	<LMAn_base> + $1D8_H$
	LMAnTCTLH	<LMAn_base> + DC_H		
Rx control register	LMAnRCTLL	<LMAn_base> + $F8_H$	LMAnRCTL0W	<LMAn_base> + $1F8_H$
	LMAnRCTLH	<LMAn_base> + FC_H		
Tx buffer register:				
Tx buffer register 01	LMAnTX01	<LMAn_base> + $C0_H$	LMAnTX00W	<LMAn_base> + $1C0_H$
Tx buffer register 23	LMAnTX23	<LMAn_base> + $C4_H$		
Tx buffer register 45	LMAnTX45	<LMAn_base> + $C8_H$	LMAnTX04W	<LMAn_base> + $1C8_H$
Tx buffer register 67	LMAnTX67	<LMAn_base> + CC_H		
Tx buffer register 89	LMAnTX89	<LMAn_base> + $D0_H$	LMAnTX08W	<LMAn_base> + $1D0_H$
Tx buffer register AB	LMAnTXAB	<LMAn_base> + $D4_H$		
Rx buffer register:				
Rx buffer register 01	LMAnRX01	<LMAn_base> + $E0_H$	LMAnRX00W	<LMAn_base> + $1E0_H$
Rx buffer register 23	LMAnRX23	<LMAn_base> + $E4_H$		
Rx buffer register 45	LMAnRX45	<LMAn_base> + $E8_H$	LMAnRX04W	<LMAn_base> + $1E8_H$
Rx buffer register 67	LMAnRX67	<LMAn_base> + EC_H		
Rx buffer register 89	LMAnRX89	<LMAn_base> + $F0_H$	LMAnRX08W	<LMAn_base> + $1F0_H$
Rx buffer register AB	LMAnRXAB	<LMAn_base> + $F4_H$		

<LMAn_base> The base addresses <LMAn_base> of the LMAn are defined in the first section of this chapter under the key word “Register addresses”.

Register access All registers are accessible in 16-bit width.
Writing to non-existing register bits is ignored, reading of these bits return always 0.

24.5.2 LMAn registers details

(1) LMAnCTLL - LMAn control register L

This register selects the LMAn operation modes.

Note This register can only be changed when the LMAn module is disabled (LMAnCTLH.LMAnPW = 0).

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + 80_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	LMAn MD[1:0]		LMAn ACSE	LMAn SCHE	LMAn AFE	LMAn ITMK	LMAn IRMK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-19 LMAnCTLL register contents (1/2)

Bit position	Bit name	Function
6 to 5	LMAn MD[1:0]	LMAn operation mode selection

MD[1:0]	Operation mode
00_B	UART through mode The LMAn is bypassed and the connected UART is used as a normal UART.
01_B	UART buffer mode The connected UART is used UART with buffer.
10_B	LIN master mode without break in header The connected UART is used as LIN master. Data transmission is continued even if data consistency errors within the LIN header (BF, SF) are detected. However, in case of an data consistency error in the PID data transmission is stopped.
11_B	LIN master mode with break in header The connected UART is used as LIN master. Data transmission is stopped if data consistency errors within the LIN header (BF, SF) are detected. In this case the status interrupt signal INTLMAnTIS is generated. Afterwards a new frame transaction with BF can be started.

Table 24-19 LMAnCTLL register contents (2/2)

Bit position	Bit name	Function
4	LMAn ACSE	Automatic checksum calculation enable 0: automatic checksum calculation disabled 1: automatic checksum calculation enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. If automatic checksum calculation is disabled, the • Rx checksum needs to be calculated by software and compared with the received checksum. • Tx checksum need to be calculated by software and to be set to the TX buffer prior starting data transmission. If automatic checksum calculation is enabled, the • Rx checksum is automatically calculated from the received data and compared with the received checksum. • Tx checksum is automatically calculated and set to the TX buffer upon start of a data transmission.
3	LMAn SCHE	Scheduler enable 0: scheduler disabled 1: scheduler enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. Before the scheduler is enabled, the scheduler counter, that is connected to LMAn, needs to be started.
2	LMAn AFE	Automatic frame start function enable 0: automatic frame start function disabled 1: automatic frame start function enabled In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0. If the automatic frame start function is disabled, a frame transmission is started by software, when LMAnTCTLL.LMAnTRQ is set to 1. If the automatic frame start function is enabled, a frame transmission is automatically started by the scheduler immediately after the interframe space, if LMAnTCTLL.LMAnTRQ = 1.
1	LMAn ITMK	Transmission interrupt request (INTLMAnTIT) mask 0: INTLMAnTIT not masked (INTLMAnTIT generated) 1: INTLMAnTIT masked (INTLMAnTIT is not generated) In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0.
0	LMAn IRMK	Reception interrupt request (INTLMAnTIR) mask 0: INTLMAnTIR not masked (INTLMAnTIR generated) 1: INTLMAnTIR masked (INTLMAnTIR is not generated) In UART through or UART buffer mode (LMAnMD[1:0] = $0x_B$) this bit must be set to 0.

(2) LMAnCTLH - LMAn control register H

This register enables/disables the LMAn operation.

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + 84_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn PW	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-20 LMAnCTLH register contents

Bit position	Bit name	Function
15	LMAn PW	LMAn enable 0: LMAn disabled (internal clock stopped) 1: LMAn enabled (internal clock operating) When LMAnPW is set to 0, all operations are stopped and LMAn is reset.

(3) LMAnSTRL - LMAn status register L

This register provides Rx process status indications.

Access This register can be read in 16-bit units.

Address <LMAn_base> + 88_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn SSB[2:0]			0	LMAn RXBE[11:00]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-21 LMAnSTRL register contents

Bit position	Bit name	Function
15 to 13	LMAn SSB[2:0]	Rx buffer mode status flags SSB[2:0] / Rx buffer mode 0_H / Idle state (no data was received) 5_H / Data has been received, but the Rx data length has not been reached. An Rx abort is necessary by LMAnRCTLH.LMAnRAB = 1. other / Abnormal operation has occurred. An Rx abort is necessary by LMAnRCTLH.LMAnRAB = 1. LMAnSSB[2:0] can be read for diagnostic purposes when a Rx process stalls.
11 to 0	LMAn RXBE[11:00]	Rx buffer error flag 0: no UART error detections of data in the Rx buffer 11 to 00 1: UART has detected an error of data in the Rx buffer 11 to 00 The bit number [11:00] corresponds to the Rx buffer number: 0: Rx byte 0 (LMAnRX01.LMAnRX00B[7:0] caused an error ... 11: Rx byte (LMAnRXAB.LMAnRX11B[7:0] caused an error Each bit remains at 1 until it is cleared by LMAnSTCL.LMAnCLRRXBE[11:00] = 1.

(4) LMAnSTRH - LMAn status register H

This register provides Rx process status indications.

Access This register can be read in 16-bit units.

Address <LMAn_base> + $8C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAn SST	LMAn SSR	LMAn TXEF	LMAn RXFF	LMAn ROVE	LMAn FCSE	LMAn SRF	LMAn PIE	LMAn BFE	LMAn SFE	LMAn SSL[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-22 LMAnSTRH register contents (1/3)

Bit position	Bit name	Function
15	LMAn SST	Tx status flag 0: no Tx request issued 1: Tx request issued LMAnSST is automatically set if a Tx request was set by LMAnTCTLL.LMAnTRQ= 1. If automatic frame start function in LIN mode is used (LMAnCTLL.LMAnAFE = 1), is automatically set upon automatic start of a new frame. This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • Tx process completion (frame transmission completed in LIN master mode)
14	LMAn SSR	Rx status flag 0: no Rx request issued 1: Rx request issued LMAnSSR is automatically set if a Rx request was set by LMAnRCTLL.LMAnRRQ= 1. This flag is not set in UART through or LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • Rx process completion
13	LMAn TXEF[a]	Tx buffer empty flag 0: data remaining in Tx buffer to be transmitted 1: Tx buffer empty: last Tx data transmitted This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLTXEF = 1

Table 24-22 LMAnSTRH register contents (2/3)

Bit position	Bit name	Function
12	LMAn RXFF[a]	Rx buffer full flag 0: data remaining to be received and to be stored in Rx buffer 1: Rx buffer full: last Rx data received This flag is not set in UART through mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLRXFF = 1
11	LMAn ROVE[a]	Rx buffer overflow flag 0: no Rx buffer overflow occurred 1: Rx buffer overflow occurred When an overflow occurs during data reception, the new data is not stored but discarded. This flag is not set in UART through or LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLROVE = 1
10	LMAn FCSE[a]	Checksum error flag 0: no checksum error occurred 1: checksum error occurred LMAnFCSE indicates the result of the checksum control during a LIN frame reception. An error is indicated, if the checksum, calculated from the received data, does not match the received checksum. This flag is only effective in LIN master mode with automatic checksum function enabled (LMAnCTLL.LMAnACSE = 1). It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLFCSE = 1
9	LMAn SRF[a]	Scheduler ready flag 0: no scheduler ready event occurred 1: scheduler ready event occurred A scheduler ready event occurs, when the value of the compare register LMAnCMPL.LMAnCMP[15:0] matches the value of the scheduler counter. In that case a status interrupt request INTLMAnTIS is also asserted. This flag is only effective in LIN master mode with scheduler function enabled (LMAnCTLL.LMAnSCHE = 1), while automatic frame start function remains disabled (LMAnCTLL.LMAnAFE = 0). It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLSRF = 1

Table 24-22 LMAnSTRH register contents (3/3)

Bit position	Bit name	Function
8	LMAn PIE[a]	Preparation incomplete error flag 0: Rx/Tx buffer preparation correct 1: Rx/Tx buffer preparation incorrect LMAnPIE is set, if a LIN master frame transfer is started, though the Rx respectively Tx buffer has not been prepared correctly. An incorrect buffer preparation is detected upon any of the following conditions: • LMAnTXEF = 1 (Tx buffer empty) • LMAnSTRH.LMAnRXFF = 1 (Rx buffer full) • LMAnSTRH.LMAnTCTLL.LMAnTLG[3:0] holds incorrect value This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLPIE = 1
7	LMAn BFE[a]	BF (Break Field) error flag 0: BF transmission successful 1: BF transmission failed LMAnBFE is set, if a framing, overrun or data consistency check error was detected during BF transmission at start of a LIN frame transmission. In case of an error detection the status interrupt request INTLMAnTIS is asserted. This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLBFE = 1
6	LMAn SFE[a]	SF (Sync Field) error flag 0: SF transmission successful 1: SF transmission failed LMAnSFE is set, if a framing, overrun or data consistency check error was detected during SF transmission at start of a LIN frame transmission. In case of an error detection the status interrupt request INTLMAnTIS is asserted. This flag is only effective in LIN master mode. It is cleared by • LMAnCTLH.LMAnPW = 0 • LMAnSTCH.LMAnCLSFE = 1
5 to 0	LMAn SSL[5:0]	LIN master mode status flag **LMAnSSL[5:0] / LIN master mode status** 0_H / Idle state (no operation) 19_H / No response from slave. An Tx abort is necessary by LMAnTCTLH.LMAnTAB = 1. other / Abnormal operation has occurred. An Tx abort is necessary by LMAnTCTLH.LMAnTAB = 1. LMAnSSL[2:0] can be read for diagnostic purposes when a LIN master transmission process stalls.

a) Before starting a data transmission process, these flags should be cleared by setting the corresponding status clear bit in LMAnSTC register to 1.

(5) LMAnSTCL - LMAn status clear register L

This register is used to clear the status and error bits of the LMAn status register L LMAnSTRL.

Access This register can be written in 16-bit units.
Reading this register returns an undefined value.

Address <LMAn_base> + 90_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	LMAn CLRXBE[11:00]											
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 24-23 LMAnSTCL register contents

Bit position	Bit name	Function
11 to 0	LMAn CLRXBE [11:00]	Clear Rx buffer error flags 0: writing 0 is ignored 1: writing 1 clears LMAnRXBE[11:00]

(6) LMAnSTCH - LMAn status clear register H

This register is used to clear the status and error bits of the LMAn status register L LMAnSTRH.

Access This register can be written in 16-bit units.
Reading this register returns an undefined value.

Address <LMAn_base> + 94_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	LMAnCLTXEF	LMAnCLRXFF	LMAnCLROVE	LMAnCLFCSE	LMAnCLSRF	LMAnCLPIE	LMAnCLBFE	LMAnCLSFE	0	0	0	0	0	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 24-24 LMAnSTCH register contents

Bit position	Bit name	Function
13	LMAn CL TXEF	Tx buffer empty flag 0: writing 0 is ignored 1: writing 1 clears LMAnTXEF
12	LMAn CL RXFF	Rx buffer full flag 0: writing 0 is ignored 1: writing 1 clears LMAnRXFF
11	LMAn CL ROVE	Rx buffer overflow flag 0: writing 0 is ignored 1: writing 1 clears LMAnROVE
10	LMAn CL FCSE	Checksum error flag 0: writing 0 is ignored 1: writing 1 clears LMAnFCSE
9	LMAn CL SRF	Scheduler ready flag 0: writing 0 is ignored 1: writing 1 clears LMAnSRF
8	LMAn CL PIE	Preparation incomplete error flag 0: writing 0 is ignored 1: writing 1 clears LMAnPIE
7	LMAn CL BFE	BF (Break Field) error flag 0: writing 0 is ignored 1: writing 1 clears LMAnBFE
6	LMAn CL SFE	SF (Sync Field) error flag 0: writing 0 is ignored 1: writing 1 clears LMAnSFE

(7) LMAnCMPL - LMAn compare register L

This register holds the lower 16 bit of scheduler comparison value.

Access This register can be read in 16-bit units.

Address <LMAn_base> + 98_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAnCMP[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-25 LMAnCMPL register contents

Bit position	Bit name	Function
15 to 0	LMAn CMP[15:00]	Current scheduler comparison value LMAnCMP[15:00] is loaded with the sum of the current free-running counter value CNATmCNT[15:0] and the frame slot length FRSL upon start of a LIN master frame.

(8) LMAnCMPH - LMAn compare register H

This register holds the upper 16 bit of scheduler comparison value.
These bits are always 0000_H.

Access This register can be read in 16-bit units.

Address <LMAn_base> + $9C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LMAnCMP[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 24-26 LMAnCMPH register contents

Bit position	Bit name	Function
15 to 0	LMAn CMP[31:16]	Current scheduler comparison value LMAnCMP[31:16] are always 0000_H.

(9) LMAnTCTLL - LMAn Tx control register L

This register controls the LMAn Tx buffer.

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + $D8_H$

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	LMAn SLEC	LMAn SLRT	LMAn FRQ	LMAn TRQ	LMAn TLG[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-27 LMAnTCTLL register contents (1/2)

Bit position	Bit name	Function
7	LMAn SLEC	Enhanced checksum control 0: classic checksum Only the data bytes are used for checksum calculation. 1: enhanced checksum Checksum is calculated from the data and the PID. LMAnSLEC is only effective, if automatic checksum calculation is enabled (LMAnCTLL.LMAnACSE = 1).
6	LMAn SLRT	LIN master mode operation control 0: Tx mode 1: Rx mode LMAnSLRT is only effective in LIN master mode.
5	LMAn FRQ	First Tx request control 0: LIN frame transaction starts with the scheduler ready event 1: LIN frame transaction starts immediately by requesting transmission (LMAnTRQ = 1), in case no Tx operation has been requested (LMAnSTRH.LMAnSST = 0). In case LMAnFRQ is set to 1 while LMAnSTRH.LMAnSST = 1, LIN frame transaction starts with the next scheduler event, and thus behaves as with LMAnFRQ = 0. Set LMAnFRQ = 1 together with LMAnTRQ = 1. LMAnFRQ is only effective in LIN master mode with using the scheduler (LMAnCTLL/LMAnSCHE = 1) and auto frame start enabled (LMAnCTLL.LMAnAFE = 1). In all other modes, LMAnFRQ shall be set to 0.
4	LMAn TRQ	Tx request control 0: Tx operation has started or has not been requested. 1: Tx operation request In LIN master mode LMAnTRQ = 1 triggers LIN frame transaction also in LIN master Rx mode (LMAnSLRT = 1. After setting LMAnTRQ = 1, LMAnTRQ returns automatically to 0 when transmission has started. Writing 0 to LMAnTRQ has no effect. In UART through (LMAnMD[1:0] = 00_B) this bit must be set to 0. **Caution:** It is prohibited to set LMAnTRQ = 1 during a pending transmission request (LMAnSTRH.LMAnSST = 1), except LMAn is operating in LIN master mode with scheduler and auto frame start (LMAnCTLL register bits LMAnSCHE = LMAnAFE = 1).

Table 24-27 LMAnTCTLL register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>3 to 0</td><td>LMAn TLG[3:0]</td><td>Tx buffer length specification
LMAnTLG[3:0] are only effective in UART buffer and LIN master mode.
• in UART buffer mode
<table>
<tr><th>LMAnTLG[3:0]</th><th>Tx buffer length</th></tr>
<tr><td>0</td><td>12 byte transmission</td></tr>
<tr><td>1 to 12</td><td>1 to 12 byte transmission in UART buffer mode</td></tr>
<tr><td>13 to 15</td><td>prohibited</td></tr>
</table>
• in LIN master mode
<table>
<tr><th>LMAnTLG[3:0]</th><th>Tx buffer length</th></tr>
<tr><td>0 to 1</td><td>prohibited</td></tr>
<tr><td>2 to 10</td><td>2 to 10 byte transmission in LIN master mode</td></tr>
<tr><td>11 to 15</td><td>prohibited</td></tr>
</table>
The value set to LMAnTLG[3:0] includes the PID and checksum bytes, thus a maximum of 8 data byte can be transmitted.
If any of the prohibited values are set to LMAnTLG[3:0], a preparation incomplete error is detected and indicated by LMAnSTRH.LMAnPIE = 1.</td></tr>
</table>

(10) LMAnTCTLH - LMAn Tx control register H

This register controls the Tx abort process.

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + DC_H

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LMAn TAB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-28 LMAnTCTLH register contents

Bit position	Bit name	Function
0	LMAn TAB	Tx abort request 0: reading returns always 0, writing 0 has no effect 1: writing 1 requests Tx abort If LMAnTAB is set to 1, Tx operation is stopped accordingly and the Tx status flag LMAnSTRH.LMAnSST is cleared. This bit has no effect in UART through mode. This bit is also effective during Rx process in LIN master mode.

Note If Tx is aborted by LMAnTAB = 1, the LMAn stops to send any further data to the UART. However ongoing transmissions are completed by the UART. The complete stop of transmissions can be confirmed by URTEnSTR0.URTEnSST = 0.

(11) LMAnRCTLL - LMAn Rx control register L

This register controls the LMAn Rx buffer.

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + $F8_H$

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	LMAn RRQ	LMAn RLG[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-29 LMAnRCTLL register contents

Bit position	Bit name	Function
4	LMAn RRQ	Rx request control 0: Rx operation has started or has not been requested. 1: Rx operation request After setting LMAnRRQ = 1, LMAnRRQ returns automatically to 0 when reception data storage to the Rx buffer has started. Writing 0 to LMAnRRQ has no effect. This bit is only effective in UART buffer mode. In all other modes this bit must not be set to 1.
3 to 0	LMAn RLG[3:0]	Rx buffer length specification LMAnRLG[3:0] are only effective in UART buffer mode. **LMAnRLG[3:0] / Rx buffer length** 0 / continuous data reception 1 to 12 / 1 to 12 byte reception to Rx buffer 13 to 15 / prohibited In continuous data reception (LMAnRLG[3:0] = 0) received data is stored to the Rx buffer continuously. After each 12 byte storage a reception interrupt INTLMAnTIR or status interrupt INTLMAnTIS request is generated. To stop continuous storage set LMAnRLG[3:0] to 1 to 12.

(12) LMAnRCTLH - LMAn Rx control register H

This register controls the Rx abort process.

Access This register can be read/written in 16-bit units.

Address <LMAn_base> + FC_H

Initial Value 0000_H. This register is initialized by any reset and by LMAnCTLH.LMAnPW = 0.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LMAn RAB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 24-30 LMAnRCTH register contents

Bit position	Bit name	Function
0	LMAn RAB	Rx abort request 0: reading returns always 0, writing 0 has no effect 1: writing 1 requests Rx abort If LMAnRAB is set to 1, Rx operation is stopped accordingly and the Rx status flag LMAnSTRH.LMAnSSR is cleared. This bit is only effective in UART buffer mode. In all other modes this bit must not be set to 1. This bit is also effective during Rx process in LIN master mode.

Note If Rx is aborted by LMAnRAB = 1, the LMAn stops to store any data in the Rx buffer. However ongoing reception by the UART will be completed.
The complete stop of reception can be confirmed by URTEnSTR0.URTEnSSR = 0.

Chapter 25 CAN Controller (FCN)

The microcontroller features on-chip CAN (Controller Area Network) controllers that complies with the CAN protocol as standardized in ISO 11898.

This chapter contains a generic description of the CAN Controller (FCN).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

25.1 V850E2/Fx4-H FCN Features

Instances

This microcontroller has following number of instances of the CAN Controller.

Table 25-1 Instances of FCN

CAN Controller	V850E2/FK4-H	V850E2/FL4-H
Instance	4	5
Name	FCN0 to FCN3	FCN0 to FCN4

Instances index n

Throughout this chapter, the instance of a CAN Controller is identified by the index "n" (n = 0 to 4), for example, FCNnGMCLCTL for the FCNn control register.

Message buffers index m

Throughout this chapter, the FCN message buffer registers are identified by “m” (for m refer to the table below), for example FCNnMmDAT4B for FCN instance n, message data byte 4 of message buffer register m. The number of message buffers for each instance of FCN is given in the following table:

Table 25-2 Message buffers of FCNn

FCNn instance	Number m of message buffers
FCN0	64
FCN1	64
FCN2	64
FCN3	128
FCN4	128

Register addresses

All CAN Controller register addresses are given as address offsets to the individual base address <FCNn_base>.
The <FCNn_base> address of each FCNn is given in the following table:

Table 25-3 Register base addresses <FCNn_base>

FCNn instance	<FCNn_base> address
FCN0	FF48 0000$_H$
FCN1	FF4A 0000$_H$
FCN2	FF4C 0000$_H$
FCN3	FF4E 0000$_H$
FCN4	FF50 0000$_H$

Clock supply

All CAN Controllers provide one clock input:

Table 25-4 FCNn clock supply

FCNn instance	FCNn clock	Connected to
FCN0	PCLK (CANCLK f_{CAN})	Clock Controller CKSCLK_113
FCN1	PCLK (CANCLK f_{CAN})	Clock Controller CKSCLK_113
FCN2	PCLK (CANCLK f_{CAN})	Clock Controller CKSCLK_113
FCN3	PCLK (CANCLK f_{CAN})	Clock Controller CKSCLK_115
FCN4	PCLK (CANCLK f_{CAN})	Clock Controller CKSCLK_115

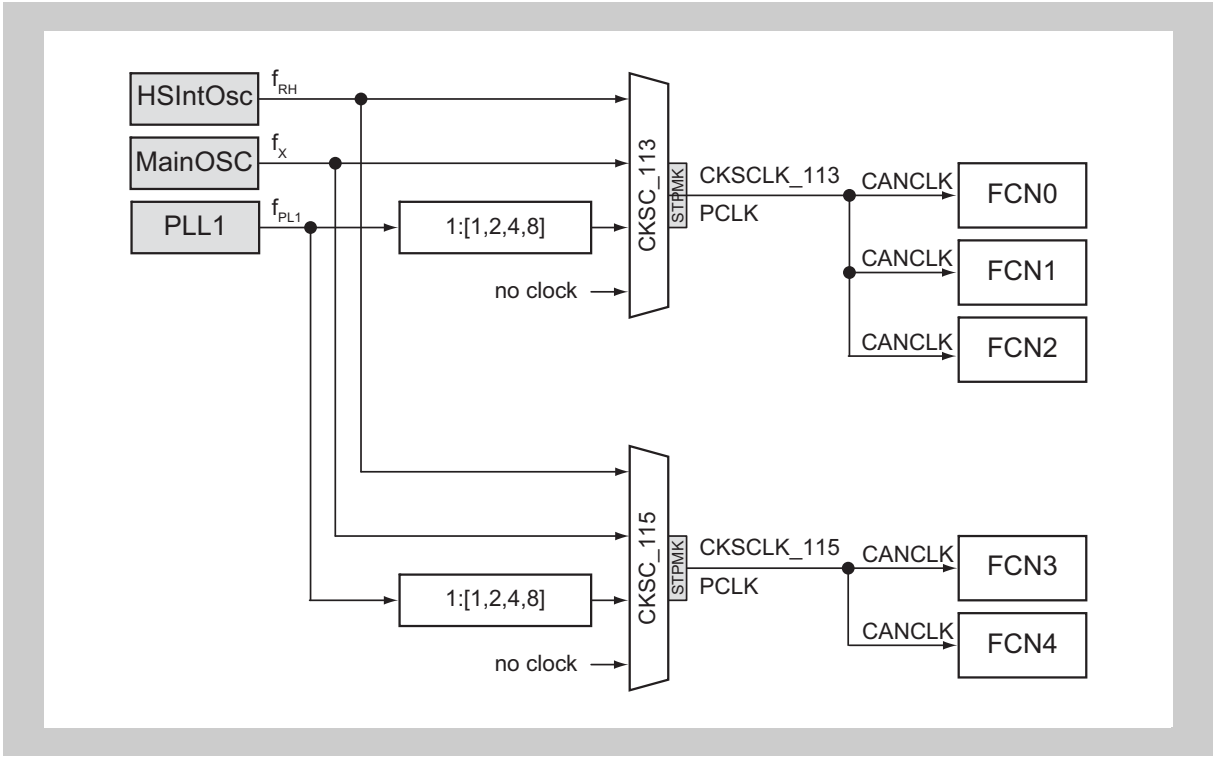


Figure 25-1 FCN clock supply

Interrupts and DMA

The CAN Controllers can generate the following interrupt and DMA requests:

Table 25-5 FCNn interrupt and DMA requests

FCNn signals	Function	Connected to
FCN0:		
INTC0ERR	Error indication	Interrupt Controller INTFCN0ERR
INTC0REC	Reception completion	Interrupt Controller INTFCN0REC
INTC0TRX	Transmission completion	Interrupt Controller INTFCN0TRX
INTC0WUP	Sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]
FCN1:		
INTC1ERR	Error indication	Interrupt Controller INTFCN1ERR
INTC1REC	Reception completion	Interrupt Controller INTFCN1REC
INTC1TRX	Transmission completion	Interrupt Controller INTFCN1TRX
INTC1WUP	Sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]
FCN2:		
INTC2ERR	Error indication	Interrupt Controller INTFCN2ERR
INTC2REC	Reception completion	Interrupt Controller INTFCN2REC
INTC2TRX	Transmission completion	Interrupt Controller INTFCN2TRX
INTC2WUP	Sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]
FCN3:		
INTC3ERR	Error indication	Interrupt Controller INTFCN3ERR
INTC3REC	Reception completion	Interrupt Controller INTFCN3REC DMA Controller trigger 74
INTC3TRX	Transmission completion	Interrupt Controller INTFCN3TRX
INTC3WUP	Sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]
FCN4:		
INTC4ERR	Error indication	Interrupt Controller INTFCN4ERR DMA Controller trigger 75
INTC4REC	Reception completion	Interrupt Controller INTFCN4REC DMA Controller trigger 74
INTC4TRX	Transmission completion	Interrupt Controller INTFCN4TRX DMA Controller trigger 76
INTC4WUP	Sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]

a) The Interrupt Controller input INTFCNWUP is a combination of the wake-up interrupts of the Diagnostic CAN Controller DCN0 and the CAN Controllers FCN0 to FCN4. Thus any of these wake-up interrupts asserts the interrupt request INTFCNWUP.

FCN H/W reset The CAN Controllers and their registers are initialized by the following reset signal:

Table 25-6 FCNn reset signal

FCNn	Reset signal
FCNn	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the CAN Controllers are listed in the table below.

Table 25-7 CAN Controllers I/O signals

FCNn signals	Function	Connected to
FCN0:		
CRXD0	CAN bus receive input	Port FCN0RX[a] or FCN1RX[b]
CTXD0	CAN bus transmit output	Port FCN0TX
FCN1:		
CRXD1	CAN bus receive input	Port FCN1RX[a]
CTXD1	CAN bus transmit output	Port FCN1TX
FCN2:		
CRXD2	CAN bus receive input	Port FCN2RX[a]
CTXD2	CAN bus transmit output	Port FCN2TX
FCN3:		
CRXD3	CAN bus receive input	Port FCN3RX[a]
CTXD3	CAN bus transmit output	Port FCN3TX
FCN4:		
CRXD4	CAN bus receive input	Port FCN4RX[a]
CTXD4	CAN bus transmit output	Port FCN4TX

a) These signals can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

b) Refer to *25.2 "FCN0 and FCN1 connection" on page 1893* for details.

Time stamp Following FCNn time stamp output signals can be internally connected to a capture input of the TimerArray Units.

Table 25-8 FCNn time stamp signals

FCNn signals	Function	Connected to
FCN0:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN0 TAUA0 TAUA0TTIN10 TAUJ0 TAUJ0TTIN1
FCN1:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN1 TAUA0 TAUA0TTIN11 TAUJ0 TAUJ0TTIN3
FCN2:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN12
FCN3:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN13
FCN4:		
TSOUT	CAN time stamp output	TAUA0 TAUA0TTIN14

Refer to the “Input Selection” sections of the respective “Timer Array Units” chapter for further details.

25.2 FCN0 and FCN1 connection

The FCN0 and FCN1 CAN Controllers have the option to be connected to the same CAN bus signals. This allow to operate the two CAN Controllers on the same CAN bus (FCN1 signals) thus allowing twice the number of message buffer support on this bus.

The following figure depicts the FCN0 and FCN1 connection scheme:

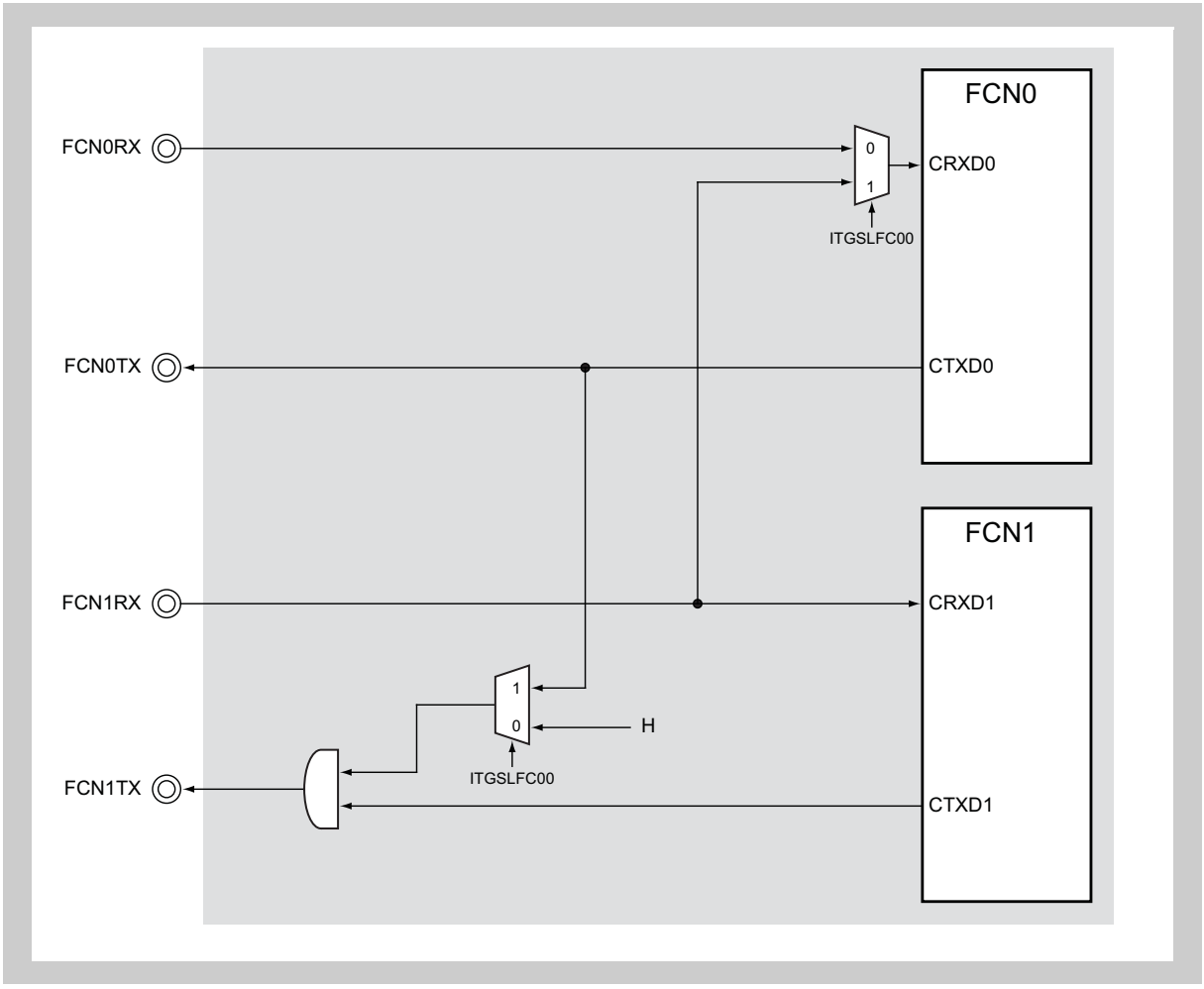


Figure 25-2 FCN0 and FCN1 connection scheme

(1) ITGSLFC0 - FCN0 signal connection selection register

This register selects the signals of FCN0.

Access This register can be read/written in 8-bit units.

Address FF77 2008$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ITGSLFC00
R	R	R	R	R	R	R	R/W

Table 25-9 ITGSLFC0 register contents

Bit position	Bit name	Function
0	ITGSLFC00	FCN0 signal seelction 0: use FCN0 bus signals (FCN0RX, FCN0TX) 1: use FNC1 bus signals (FCN1RX, FCN1TX) - combined operation

25.3 CAN baudrate and time quanta

When configuring the CAN Controller (FCN) one of the following combinations for

- the CAN Controller clock f_{CAN},
- the CAN baudrate and
- the number of time quantas (TQ) per data bit time (DBT)

must be used:

Table 25-10 CAN baudrate settings

CAN baudrate [kbps]	CAN Controller clock f_{CAN} [MHz]	TQs per DBT
1000	16	8
	24	8
	32	8
	40	≤ 10
500	16	≤ 16
	20	≤ 10
	24	≤ 12
	32	≤ 16
	40	≤ 20
250	16	≤ 16
	20	≤ 20
	24	≤ 24
	32	≤ 16
	40	≤ 20

Thus make sure to set the

- global clock selection register FCNnGMCSPRE
- module bit rate prescaler register FCNnCMBRPRS
- module bit rate register FCNnCMBTCTL

accordingly.

Note The table describes common combinations. For other combinations please contact your local Renesas engineering representative.

25.4 Features

- Compliant with ISO 11898 and tested according to ISO/DIS 16845 (CAN conformance test)
- Standard frame and extended frame transmission/reception enabled
- Transfer rate: 1 Mbps max. (if CAN Controller clock input ≥ 16 MHz, for 64 or more message buffers)
- 32, 64 or 128 message buffers per channel
  For the number of available message buffers for each CAN Controller refer to the key word "Message buffers index m" in the first section of this chapter.
- Receive/transmit history list function, with enable flag for each message buffer individually
- Automatic block transmission function
- Multi-buffer receive block function
- Mask setting of 8 patterns is possible for each channel, applicable for data and remote frames
- Data bit time, communication baudrate and sample point can be controlled by CAN channel bit-rate prescaler register (FCNnCMBRPRS) and bit rate register (FCNnCMBTCTL)
  - As an example the following sample-point configurations can be configured:
  - 66.7%, 70.0%, 75.0%, 80.0%, 81.3%, 85.0%, 87.5%
  - Baudrates in the range of 10 kbps up to 1000 kbps can be configured
- Enhanced features:
  - Each message buffer can be configured to operate as a transmit or a receive message buffer
  - A transmission request can be aborted by clearing the dedicated Transmit-Request flag of the concerned message buffer. Supported by Transmission Abort Interrupt, on successful abortion.
  - Automatic block transmission operation mode (ABT)
  - Time stamp function in collaboration with timers capture channels
  - Centralized data new flag register, collecting all data new flags to be read from one location

25.4.1 Overview of functions

Table 25-11 "Overview of functions" presents an overview of the CAN Controller functions.

Table 25-11 Overview of functions

Function	Details
Protocol	CAN protocol ISO 11898 (standard and extended frame transmission/reception)
Baudrate	Maximum 1 Mbps (minimum CAN Controller clock input = 16 MHz)
Data storage	Storing messages in the CAN RAM
Number of messages	• 32, 64 or 128 message buffers per channel For the number of available message buffers for each CAN Controller refer to the key word "Message buffers index m" in the first section of this chapter. • Each message buffer can be set to be either a transmit message buffer or a receive message buffer.
Message reception	• Unique ID can be set to each message buffer. • Mask setting of 8 patterns is possible for each channel, applicable for data and remote frames • A receive completion interrupt is generated each time a message is received and stored in a message buffer. • Two or more receive message buffers can be used as a FIFO receive buffer (multi-buffer receive block function). • Receive history list function, with enable flag for each message buffer individually • Centralized Data New Flag register
Message transmission	• Unique ID can be set to each message buffer. • Transmit completion interrupt for each message buffer • Transmit Abort interrupt and -flag for each message buffer (only one transmission of any buffer can be aborted at a time) • Message buffer number 0 to 7 (m = 32) / 15 (m = 64) / 31(m = 128) specified as the transmit message buffer can be set for automatic block transfer. Message transmission interval is programmable (automatic block transmission function (hereafter referred to as “ABT”)). • Transmission history list function, with enable flag for each message buffer individually
Remote frame processing	• Remote frame processing by transmit message buffer • Remote frame processing by receive message buffer, when applying one of the 8 masks
Time stamp function	• The time stamp function can be set for a message reception when a 16-bit timer is used in combination. • Time stamp capture trigger can be selected (SOF or EOF in a CAN message frame can be detected.).
Diagnostic function	• Readable error counters • “Valid protocol operation flag” for verification of bus connections • Receive-only mode • Single-shot mode • CAN protocol error type decoding • Self-test mode
Release from bus-off state	• Forced release from bus-off (by ignoring timing constraint) possible by software. • No automatic release from bus-off (software must re-enable).
Power save mode	• Sleep mode (can be woken up by CAN bus) • Stop mode (cannot be woken up by CAN bus)

25.4.2 Configuration

The CAN Controller is composed of the following four blocks.

- PBUS interface
  This functional block provides an PBUS interface and means of transmitting and receiving messages between the CAN Controller and the host CPU.
- MAC (Memory Access Controller)
  This functional block controls access to the CAN protocol layer and to the CAN RAM within the CAN Controller.
- CAN protocol layer
  This functional block is involved in the operation of the CAN protocol and its related settings.
- CAN RAM
  The CAN RAM is used to store message IDs, message data, etc.

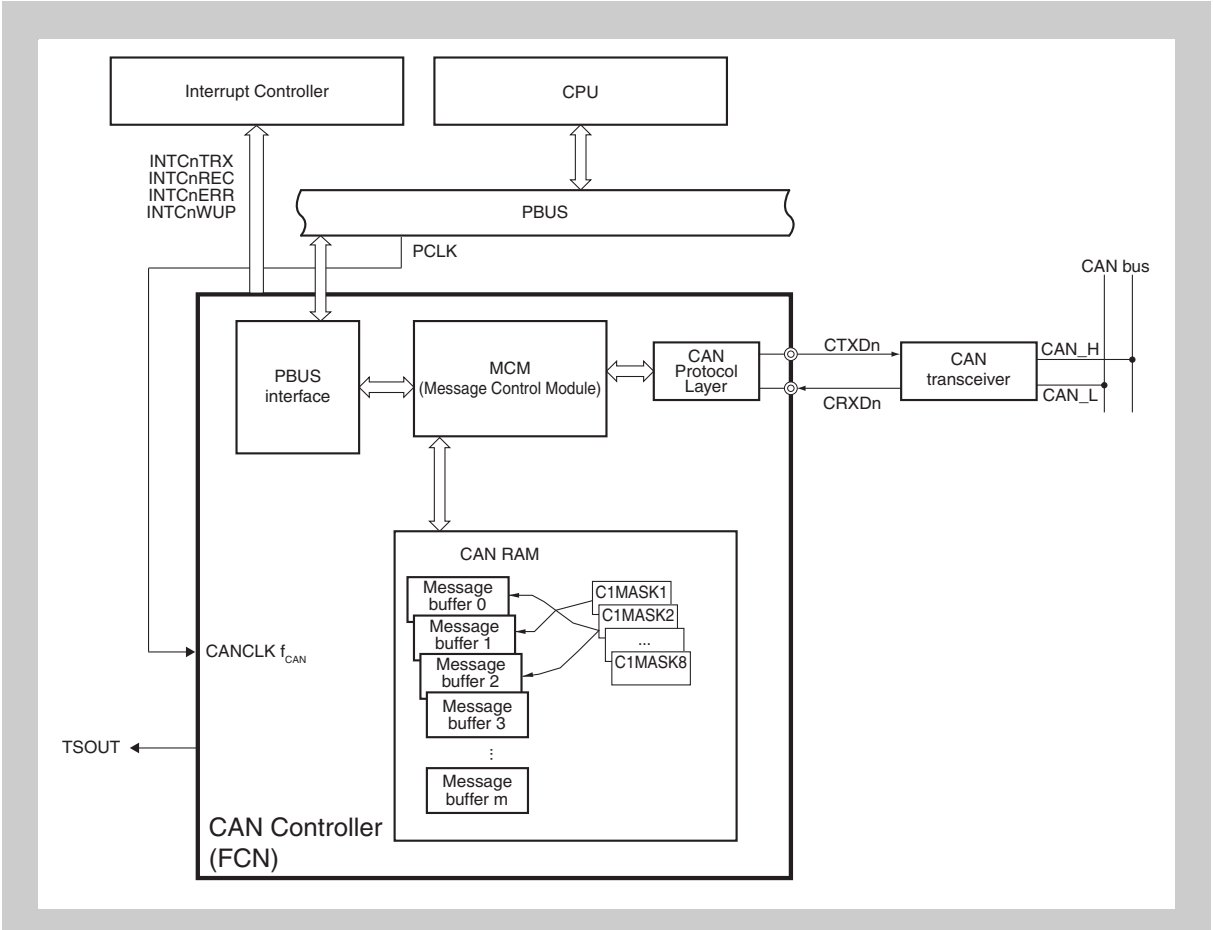


Figure 25-3 Block diagram of the CAN Controller

Note For the number m of available message buffers for each CAN Controller refer to the key word "Message buffers index m" in the first section of this chapter.

CAN RAM The CAN RAM is equipped with a memory checking module. If a CAN RAM error occurs during a software reset, the message buffer RAM read error detection bit FCNnGMCLCTL.FCNnGMCLECCF is set.
Check the CAN Controller functionality in such case.

25.5 Internal Registers of CAN Controller

25.5.1 CAN Controller configuration

Table 25-12 List of CAN Controller registers (1/2)

Item	Register Name
CAN Controller global registers	FCNn global control register (FCNnGMCLCTL)
	FCNn global clock selection register (FCNnGMCSPRE)
	FCNn global automatic block transmission control register (FCNnGMABCTL)
	FCNn global automatic block transmission delay setting register (FCNnGMADCTL)
	FCNn global Data New bit monitor registers (FCNnDNBMRX0 - FCNnDNBMRX3)
CAN channel registers	FCNn channel mask 1 registers (FCNnCMMKCTL01H, FCNnCMMKCTL02H, FCNnCMMKCTL01W)
	FCNn channel mask 2 registers (FCNnCMMKCTL03H, FCNnCMMKCTL04H, FCNnCMMKCTL03W)
	FCNn channel mask3 registers (FCNnCMMKCTL05H, FCNnCMMKCTL06H, FCNnCMMKCTL05W)
	FCNn channel mask 4 registers (FCNnCMMKCTL07H, FCNnCMMKCTL08H, FCNnCMMKCTL07W)
	FCNn channel mask 5 registers (FCNnCMMKCTL09H, FCNnCMMKCTL10H, FCNnCMMKCTL09W)
	FCNn channel mask 6 registers (FCNnCMMKCTL11H, FCNnCMMKCTL12H, FCNnCMMKCTL11W)
	FCNn channel mask registers (FCNnCMMKCTL13H, FCNnCMMKCTL14H, FCNnCMMKCTL13W)
	FCNn channel mask 8 registers (FCNnCMMKCTL15H, FCNnCMMKCTL16H, FCNnCMMKCTL15W)
	FCNn channel control register (FCNnCMCLCTL)
	FCNn channel last error information register (FCNnCMLCSTR)
	FCNn channel information register (FCNnCMLCSTR)
	FCNn channel error counter register (FCNnCMERCNT)
	FCNn channel last error code register (FCNnCMLCSTR)
	FCNn channel interrupt enable register (FCNnCMIECTL)
	FCNn channel interrupt status register (FCNnCMISCTL)
	FCNn channel bit rate prescaler and clock selector register (FCNnCMBRPRS)
	FCNn channel bit rate register (FCNnCMBTCTL)
	FCNn channel last in-pointer register (FCNnCMLISTR)
	FCNn channel receive history list register (FCNnCMRGRX)
	FCNn channel last out-pointer register (FCNnCMLOSTR)
	FCNn channel transmit history list register (FCNnCMTGTX)
	FCNn channel time stamp register (FCNnCMTSCTL)
	FCNn Controller message data byte 0 to 3 registers m (FCNnMmDAT0W, FCNnMmDAT0H, FCNnMmDAT2H, FCNnMmDAT0B, FCNnMmDAT1B, FCNnMmDAT2B, FCNnMmDAT3B)

Table 25-12 List of CAN Controller registers (2/2)

Item	Register Name
CAN Controller message buffer registers	FCNn message data byte 4 to 7 registers m (FCNnMmDAT4W, FCNnMmDAT4H, FCNnMmDAT6H, FCNnMmDAT4B, FCNnMmDAT5B, FCNnMmDAT6B, FCNnMmDAT7B)
	FCNn message data length register m (FCNnMmDTLGB)
	FCNn message configuration register m (FCNnMmSTRB)
	FCNn message ID registers m (FCNnMmMID0H, FCNnMmMID1H, FCNnMmMID0W)
	FCNn message control register m (FCNnMmCTL)

25.5.2 CAN Controller Registers Overview

Address offset All register addresses are given as offsets to the base address <FCNn_base>. The <FCNn_base> addresses of the registers are defined in the first section of this chapter under the keyword “Register addresses”.

(1) CAN Controller global and channel registers

Table 25-13 CAN Controller global and channel registers (1/2)

Address offset	Register name	Symbol	R/W	Access bit	After reset
0 0008$_H$	FCNn global clock selection register	FCNnGMCSPRE	R/W	8	0F$_H$
0 0020$_H$	FCNn global automatic block transmission delay register	FCNnGMADCTL	R/W	8	00$_H$
0 8000$_H$	FCNn global control register	FCNnGMCLCTL	R/W	16	00x0$_H$[a]
0 8018$_H$	FCNn global automatic block transmission register	FCNnGMABCTL	R/W	16	0000$_H$
1 00C0$_H$	FCNn global data new bit monitor register 0	FCNnDNBMRX0	R	32	0000 0000$_H$
1 00D0$_H$	FCNn global data new bit monitor register 1	FCNnDNBMRX1	R	32	0000 0000$_H$
1 00E0$_H$	FCNn global data new bit monitor register 2	FCNnDNBMRX2	R	32	0000 0000$_H$
1 00F0$_H$	FCNn global data new bit monitor register 3	FCNnDNBMRX3	R	32	0000 0000$_H$
0 8300$_H$	FCNn channel mask 1 register	FCNnCMMKCTL01H	R/W	16	0000$_H$
0 8308$_H$		FCNnCMMKCTL02H			
1 0300$_H$		FCNnCMMKCTL01W		32	0000 0000$_H$
0 8310$_H$	FCNn channel mask 2 register	FCNnCMMKCTL03H	R/W	16	0000$_H$
0 8318$_H$		FCNnCMMKCTL04H			
1 0310$_H$		FCNnCMMKCTL03W		32	0000 0000$_H$
0 8320$_H$	FCNn channel mask 3 register	FCNnCMMKCTL05H	R/W	16	0000$_H$
0 8328$_H$		FCNnCMMKCTL06H			
1 0320$_H$		FCNnCMMKCTL05W		32	0000 0000$_H$
0 8330$_H$	FCNn channel mask 4 register	FCNnCMMKCTL07H	R/W	16	0000$_H$
0 8338$_H$		FCNnCMMKCTL08H			
1 0330$_H$		FCNnCMMKCTL07W		32	0000 0000$_H$
0 8340$_H$	FCNn channel mask 5 register	FCNnCMMKCTL09H	R/W	16	0000$_H$
0 8348$_H$		FCNnCMMKCTL10H			
1 0340$_H$		FCNnCMMKCTL09W		32	0000 0000$_H$
0 8350$_H$	FCNn channel mask 6 register	FCNnCMMKCTL11H	R/W	16	0000$_H$
0 8358$_H$		FCNnCMMKCTL12H			
1 0350$_H$		FCNnCMMKCTL11W		32	0000 0000$_H$

Table 25-13 CAN Controller global and channel registers (2/2)

Address offset	Register name	Symbol	R/W	Access bit	After reset
0 8360$_H$	FCNn channel mask 7 register	FCNnCMMKCTL13H	R/W	16	0000$_H$
0 8368$_H$		FCNnCMMKCTL14H			
1 0360$_H$		FCNnCMMKCTL13W		32	0000 0000$_H$
0 8370$_H$	FCNn channel mask 8 register	FCNnCMMKCTL15H	R/W	16	0000$_H$
0 8378$_H$		FCNnCMMKCTL16H			
1 0370$_H$		FCNnCMMKCTL15W		32	0000 0000$_H$
0 0248$_H$	FCNn channel last error code register	FCNnCMLCSTR	R/W	8	00$_H$
0 024C$_H$	FCNn channel information register	FCNnCMINSTR	R	8	00$_H$
0 0268$_H$	FCNn channel bit-rate prescaler and clock selector register	FCNnCMBRPRS	R/W	8	FF$_H$
0 0278$_H$	FCNn channel last in-pointer register	FCNnCMLISTR	R	8	Undefined
0 0288$_H$	FCNn channel last out-pointer register	FCNnCMLOSTR	R	8	Undefined
0 8240$_H$	FCNn channel control register	FCNnCMCLCTL	R/W	16	0000$_H$
0 8250$_H$	FCNn channel error counter register	FCNnCMERCNT	R	16	0000$_H$
0 8258$_H$	FCNn channel interrupt enable register	FCNnCMIECTL	R/W	16	0000$_H$
0 8260$_H$	FCNn channel interrupt status register	FCNnCMISCTL	R/W	16	0000$_H$
0 8270$_H$	FCNn channel bit-rate register	FCNnCMBTCTL	R/W	16	370F$_H$
0 8280$_H$	FCNn channel receive history list register	FCNnCMRGRX	R/W	16	xx02$_H$
0 8290$_H$	FCNn channel transmit history list register	FCNnCMTGTX	R/W	16	xx02$_H$
0 8298$_H$	FCNn channel time stamp register	FCNnCMTSCTL	R/W	16	0000$_H$

a) Initial value depends on FCNnGMCLCTL.FCNnGMCLECCF, which indicates error detections when reading from message buffer RAM. Refer to the detailed description of the FCNnGMCLCTL register.

25.5.3 Register bit configuration

Table 25-14 CAN Controller global register bit configuration

Address offset	Symbol	Bit 7/ 15/23/ 31	Bit 6/ 14/22/ 30	Bit 5/ 13/21/ 29	Bit 4/ 12/20/ 28	Bit 3/ 11/19/ 27	Bit 2/ 10/18/ 26	Bit 1/ 9/17/ 25	Bit 0/ 8/16/ 24
$0\ 8000_H$	FCNnGMCLCTL (W)	0	0	FCNnGMCLCLMB		0	0	0	FCNnGMCLCLOM
		0	0	0	FCNnGMCLSESR	0	0	FCNnGMCLSESD	FCNnGMCLSEOM
	FCNnGMCLCTL (R)	0	0	FCNnGMCLECCF	FCNnGMCLSORF	0	0	FCNnGMCLESDE	FCNnGMCLPWOM
		FCN0GMCLSSMO	0	0	0	0	0	0	0
$0\ 0008_H$	FCNnGMCSPRE	0	0	0	0	FCNnGMCSPRSC[3:0]			
$0\ 8018_H$	FCNnGMABCTL (W)	0	0	0	0	0	0	0	FCNnGMABCLAT
		0	0	0	0	0	0	FCNnGMABSEAC	FCNnGMABSEAT
	FCNnGMABCTL (R)	0	0	0	0	0	0	FCNnGMABCLRF	FCNnGMABABTT
		0	0	0	0	0	0	0	0
$0\ 0020_H$	FCNnGMADCTL	0	0	0	0	FCNnGMADSSAD[3:0]			
$1\ 00C0_H$	FCNnDNBMRX0 (R)	FCNnDNBMSSDN[7:0]							
		FCNnDNBMSSDN[15:8]							
		FCNnDNBMSSDN[23:16]							
		FCNnDNBMSSDN[31:24]							
$1\ 00D0_H$	FCNnDNBMRX1 (R)[a]	FCNnDNBMSSDN[39:32]							
		FCNnDNBMSSDN[47:40]							
		FCNnDNBMSSDN[55:48]							
		FCNnDNBMSSDN[63:56]							
$1\ 00E0_H$	FCNnDNBMRX2 (R)[b]	FCNnDNBMSSDN[71:64]							
		FCNnDNBMSSDN[79:72]							
		FCNnDNBMSSDN[87:80]							
		FCNnDNBMSSDN[95:88]							
$1\ 00F0_H$	FCNnDNBMRX3 (R)[b]	FCNnDNBMSSDN[103:96]							
		FCNnDNBMSSDN[111:104]							
		FCNnDNBMSSDN[119:112]							
		FCNnDNBMSSDN[127:120]							

a) Only available with 64 (m = 0 to 63) and 128 message buffers (m = 0 to 127)
b) Only available with 128 message buffers (m = 0 to 127)

Table 25-15 CAN channel mask control 16-bit registers bit configuration

Address offset	Symbol	Bit 15	Bit 14	Bit 13	Bit 12 to 0
0 8300$_H$	FCNnCMMKCTL01H	FCNnCMMKSSID[15:0]			
0 8308$_H$	FCNnCMMKCTL02H	0	0	0	FCNnCMMKSSID[28:16]
0 8310$_H$	FCNnCMMKCTL03H	FCNnCMMKSSID[15:0]			
0 8318$_H$	FCNnCMMKCTL04H	0	0	0	FCNnCMMKSSID[28:16]
0 8320$_H$	FCNnCMMKCTL05H	FCNnCMMKSSID[15:0]			
0 8328$_H$	FCNnCMMKCTL06H	0	0	0	FCNnCMMKSSID[28:16]
0 8330$_H$	FCNnCMMKCTL07H	FCNnCMMKSSID[15:0]			
0 8338$_H$	FCNnCMMKCTL08H	0	0	0	FCNnCMMKSSID[28:16]
0 8340$_H$	FCNnCMMKCTL09H	FCNnCMMKSSID[15:0]			
0 8348$_H$	FCNnCMMKCTL10H	0	0	0	FCNnCMMKSSID[28:16]
0 8350$_H$	FCNnCMMKCTL11H	FCNnCMMKSSID[15:0]			
0 8358$_H$	FCNnCMMKCTL12H	0	0	0	FCNnCMMKSSID[28:16]
0 8360$_H$	FCNnCMMKCTL13H	FCNnCMMKSSID[15:0]			
0 8368$_H$	FCNnCMMKCTL14H	0	0	0	FCNnCMMKSSID[28:16]
0 8370$_H$	FCNnCMMKCTL15H	FCNnCMMKSSID[15:0]			
0 8378$_H$	FCNnCMMKCTL16H	0	0	0	FCNnCMMKSSID[28:16]

Table 25-16 CAN channel mask control 32-bit registers bit configuration (1/2)

Address offset	Symbol	Bit 31	Bit 30	Bit 29	Bit 28 to 0
1 0300$_H$	FCNnCMMKCTL01W	0	0	0	FCNnCMMKSSID[28:0]
1 0310$_H$	FCNnCMMKCTL03W	0	0	0	FCNnCMMKSSID[28:0]
1 0320$_H$	FCNnCMMKCTL05W	0	0	0	FCNnCMMKSSID[28:0]
1 0330$_H$	FCNnCMMKCTL07W	0	0	0	FCNnCMMKSSID[28:0]
1 0340$_H$	FCNnCMMKCTL09W	0	0	0	FCNnCMMKSSID[28:0]

Table 25-16 CAN channel mask control 32-bit registers bit configuration (2/2)

Address offset	Symbol	Bit 31	Bit 30	Bit 29	Bit 28 to 0
1 0350$_H$	FCNnCMMKCTL11W	0	0	0	FCNnCMMKSSID[28:0]
1 0360$_H$	FCNnCMMKCTL13W	0	0	0	FCNnCMMKSSID[28:0]
1 0370$_H$	FCNnCMMKCTL15W	0	0	0	FCNnCMMKSSID[28:0]

Table 25-17 CAN channel register bit configuration (1/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 8240$_H$	FCNnCMCLCTL (W)	0	FCNnCMCLCLAL	FCNnCMCLCLVL	FCNnCMCLCLPS[1:0]		FCNnCMCLCLOP[2:0]		
		FCNnCMCLSERC	FCNnCMCLCSEAL	0	FCNnCMSESEPS[1:0]		FCNnCMCSELOP[2:0]		
	FCNnCMCLCTL (R)	FCNnCMCLERCF	FCNnCMCLALBF	FCNnCMCLVALF	FCNnCMCLMDPF[1:0]		FCNnCMCLMDOF[2:0]		
		0	0	0	0	0	0	FCNnCMCLSSRS	FCNnCMCLSSTS
0 00248$_H$	FCNnCMLCSTR (W)	0	0	0	0	0	0	0	0
	FCNnCMLCSTR (R)	0	0	0	0	0	FCNnCMLCSSLC[2:0]		
0 024CH	FCNnCMINSTR	0	0	0	FCNnCMINBOFF	FCNnCMINSSTE[1:0]		FCNnCMINSSRE[1:0]	
0 8250$_H$	FCNnCMERCNT	FCNnCMERTECF[7:0]							
		FCNnCMERRPSF	FCNnCMERRECF[6:0]						
0 8258$_H$	FCNnCMIECTL (W)	0	FCNnCMIECLIE[6:0]						
		0	FCNnCMIESEIE[6:0]						
	FCNnCMIECTL (R)	0	FCNnCMIEINTF[6:0]						
		0	0	0	0	0	0	0	0
0 8260$_H$	FCNnCMISCTL (W)	0	FCNnCMISCLTS[6:0]						
		0	0	0	0	0	0	0	0
	FCNnCMISCTL (R)	0	FCNnCMISITSF[6:0]						
		0	0	0	0	0	0	0	0
0 0268$_H$	FCNnCMBRPRS	FCNnCMBRPRS[7:0]							
0 8270$_H$	FCNnCMBTCTL	0	0	0	0	FCNnCMBTS1LG[3:0]			
		0	0	FCNnCMBTJWLG[1:0]		0	FCNnCMBTS2LG[2:0]		
0 0278$_H$	FCNnCMLISTR	FCNnCMLISSLR[7:0]							

Table 25-17 CAN channel register bit configuration (2/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 8280$_H$	FCNnCM RGRX (W)	0	0	0	0	0	0	0	FCNnCMR GCLRV
		0	0	0	0	0	0	0	0
	FCNnCM RGRX (R)	0	0	0	0	0	0	FCNnCMR GSSPM	FCNnCMR GRVFF
		FCNnCMRGSSPT[7:0]							
0 0288$_H$	FCNnCM LOSTR	FCNnCMLOSTRSSLT[7:0]							
0 8290$_H$	FCNnCM TGTX (W)	0	0	0	0	0	0	0	FCNnCMT GCLTV
		0	0	0	0	0	0	0	0
	FCNnCM TGTX (R)	0	0	0	0	0	0	FCNnCMT GSSPM	FCNnCMT GTVFF
		FCNnCMTGSSPT[7:0]							
0 8298$_H$	FCNnCM TSCTL (W)	0	0	0	0	0	FCNnCMT SCLLK	FCNnCMT SCLSL	FCNnCMT SCLTS
		0	0	0	0	0	FCNnCMT SSELK	FCNnCMT SSESL	FCNnCMT SSETS
	FCNnCM TSCTL (R)	0	0	0	0	0	FCNnCMT SLOKE	FCNnCMT SSELE	FCNnCMT STSGE
		0	0	0	0	0	0	0	0

Table 25-18 Message buffer register bit configuration (1/2)

Address offset	Symbol	Bit 7/15/ 23/31	Bit 6/14/ 22/30	Bit 5/13/ 21/29	Bit 4/12/ 20/28	Bit 3/11/ 19/27	Bit 2/10/ 18/26	Bit 1/9/ 17/25	Bit 0/8/ 16/24
1 1000$_H$ + m x 40$_H$	FCNnMm DAT0W	FCNnMmSSD0[7:0]							
		FCNnMmSSD1[7:0]							
		FCNnMmSSD2[7:0]							
		FCNnMmSSD3[7:0]							
0 9000$_H$ + m x 40$_H$	FCNnMm DAT0H	FCNnMmSSD0[7:0]							
		FCNnMmSSD1[7:0]							
0 1000$_H$ + m x 40$_H$	FCNnMm DAT0B	FCNnMmSSD0[7:0]							
0 1004$_H$ + m x 40$_H$	FCNnMm DAT1B	FCNnMmSSD1[7:0]							
0 9008$_H$ + m x 40$_H$	FCNnMm DAT2H	FCNnMmSSD2[7:0]							
		FCNnMmSSD3[7:0]							
0 1008$_H$ + m x 40$_H$	FCNnMm DAT2B	FCNnMmSSD2[7:0]							
0 100C$_H$ + m x 40$_H$	FCNnMm DAT3B	FCNnMmSSD3[7:0]							

Table 25-18 Message buffer register bit configuration (2/2)

Address offset	Symbol	Bit 7/15/23/31	Bit 6/14/22/30	Bit 5/13/21/29	Bit 4/12/20/28	Bit 3/11/19/27	Bit 2/10/18/26	Bit 1/9/17/25	Bit 0/8/16/24
$1\ 1010_H$ + m x 40_H	FCNnMm DAT4W	FCNnMmSS4[7:0]							
		FCNnMmSSD5[7:0]							
		FCNnMmSSD6[7:0]							
		FCNnMmSSD7[7:0]							
$0\ 9010_H$ + m x 40_H	FCNnMm DAT4H	FCNnMmSSD4[7:0]							
		FCNnMmSSD5[7:0]							
$0\ 1010_H$ + m x 40_H	FCNnMm DAT4B	FCNnMmSSD4[7:0]							
$0\ 1014_H$ + m x 40_H	FCNnMm DAT5B	FCNnMmSSD5[7:0]							
$0\ 9018_H$ + m x 40_H	FCNnMm DAT6H	FCNnMmSSD6[7:0]							
		FCNnMmSSD7[7:0]							
$0\ 1018_H$ + m x 40_H	FCNnMm DAT6B	FCNnMmSSD6[7:0]							
$0\ 101C_H$ + m x 40_H	FCNnMm DAT7B	FCNnMmSSD7[7:0]							
$0\ 1020_H$ + m x 40_H	FCNnMm DTLGB	0				FCNnMmDTLG[3:0]			
$0\ 1024_H$ + m x 40_H	FCNnMm STRB	FCNnMm SSOW	FCNnMmSSMT[3:0]				FCNnMm SSRT	0	FCNnMm SSAM
$0\ 9028_H$ + m x 40_H	FCNnMm MID0H	FCNnMmSSID[7:0]							
		FCNnMmSSID[15:8]							
$0\ 9030_H$ + m x 40_H	FCNnMm MID1H	FCNnMmSSID[23:16]							
		FCNnMm SSIE	0	0	FCNnMmSSID[28:24]				
$1\ 1028_H$ + m x 40_H	FCNnMm MID0W	FCNnMmSSID[7:0]							
		FCNnMmSSID[15:8]							
		FCNnMmSSID[23:16]							
		FCNnMm SSIE	0	0	FCNnMmSSID[28:24]				
$0\ 9038_H$ + m x 40_H	FCNnMmCTL (W)	0	FCNnMm CLNH	0	FCNnMm CLMW	FCNnMm CLIE	FCNnMm CLDN	FCNnMm CLTR	FCNnMm CLRY
		0	FCNnMm SENH	0	0	FCNnMm SEIE	0	FCNnMm CSETR	FCNnMm SERY
	FCNnMmCTL (R)	0	FCNnMm NHMF	0	FCNnMm MOWF	FCNnMm IENF	FCNnMm DTNF	FCNnMm TRQF	FCNnMm RDYF
		0	0	FCNnMm MUCF	0	0	0	FCNnMm TCPF	0

25.6 Bit Set/Clear Function

The CAN Controller control registers include registers whose bits can be set or cleared via the CPU and via the CAN Controller. These register bits can not be changed directly by the CPU by any bit manipulation instructions, such as SET1, CLR1, and NOT1. Instead a special bit-set/bit-clear mechanism is used.

All registers where bit manlpulation operations are prohibited are organised in such a way that all bits allowed for changing by the CPU are located in the lower byte (RWx in the register layout below), while in the upper byte either no or read-only information is located (ROx in the register layout below).

The registers can be read in the usual way getting all 16 data bits in their current setting and as described in the register description.
For setting or clearing any of the lower 8 bits the following mechanism is implemented:

When writing 16-bit data to the register address

Bit clear

- each of the lower 8 data bits (CLx in the register layout below) indicates whether the corresponding register bit RWx should be
  - cleared, i.e. set to 0: if CLx = 1, the corresponding RWx is cleared to 0
  - remain unchanged: if CLx = 0, the corresponding RWx does not change

Bit set

- each of the upper 8 data bits (SEx in the register layout below) indicate whether the corresponding register bit should be
  - set, i.e. set to 1: if SEx = 1, the corresponding RWx is set to 1
  - remain unchanged: if SEx = 0, the corresponding RWx does not change

Register layout for read access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RO7	RO6	RO5	RO4	RO3	RO2	RO1	RO0	RW7	RW6	RW5	RW4	RW3	RW2	RW1	RW0
changing by the CPU not possible								bits for CPU manipulation via SE7-SE0 and CL7-CL0							

Register layout for write access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE7	SE6	SE5	SE4	SE3	SE2	SE1	SE0	CL7	CL6	CL5	CL4	CL3	CL2	CL1	CL0
SEx = 1 sets the corresponding RW7-RW0								CLx = 1 clears the corresponding RW7-RW0							

The following table denotes the operations applied to the RWx bits:

Table 25-19 Bit set/clear operation

CLx	SEx	Operation on RWx
0	0	no change of RWx
0	1	RWx set to 1
1	0	RWx cleared to 0
1	1	no change of RWx

Example Following an example.

The register with the content 1883_H shall be changed so, that

- bit 3 shall be set to 1: SE3 = 1

- bit 1 shall be cleared to 0: CL1 = 1

Register read before bit manipulations:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0	1	0	0	0	**0**	0	**1**	1
may hold any value, here 18_H								RW7-RW0: 83_H							

Register write access:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	**1**	0	0	0	0	0	0	0	0	0	**1**	0
SE3 = 1: 08_H								CL1 = 1: 02_H							

Register read after bit manipulations:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	0	0	0	1	0	0	0	**1**	0	**0**	1
may hold any value, here 18_H								RW7-RW0: 89_H							

25.7 Control Registers

25.7.1 CAN Controller global registers

(1) FCNnGMCLCTL - FCNn global control register

This register is used to control the operation of the CAN Controller.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8000_H$

Initial Value The initial value depends on the occurrence of errors in relation to a software reset:

- no CAN RAM error detection after CAN S/W reset: 0000_H
- no CAN RAM error detection while CAN S/W reset is ongoing: 0010_H
- CAN RAM error detection after CAN S/W reset: 0020_H
- CAN RAM error detection while CAN S/W reset is ongoing: 0030_H

(a) FCNnGMCLCTL read

15	14	13	12	11	10	9	8
FCNnGMCLSSMO	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	FCNnGMCLECCF	FCNnGMCLSORF	0	0	FCNnGMCLESDE	FCNnGMCLPWOM

FCNnGMCLSSMO	Bit enabling access to message buffer register, transmit/receive history registers
0	Write access and read access to the message buffer register and the transmit/receive history list registers is disabled.
1	Write access and read access to the message buffer register and the transmit/receive history list registers is enabled.

Cautions

1. While the FCNnGMCLCTL.FCNnGMCLSSMO is cleared (to 0), software access to all message buffers and message buffer registers (i.e. all registers with name prefix FCNnMm...), or registers related to transmit history or receive history (FCNnCMLOSTR, FCNnCMTGTX, FCNnCMLISTR, and FCNnCMRGRX) is disabled.
2. FCNnGMCLCTL.FCNnGMCLSSMO is read-only. Even if 1 is written while it is 0, its value does not change, and access to the message buffer registers, or registers related to transmit history or receive history remains disabled.

Note FCNnGMCLCTL.FCNnGMCLSSMO is cleared to 0 when the CAN Controller enters sleep/stop, or when the FCNnGMCLCTL.FCNnGMCLPWOM is cleared to 0.
FCNnGMCLSSMO is set to 1 when the CAN Controller sleep/stop mode is released, or when the FCNnGMCLCTL.FCNnGMCLPWOM is set to 1.

FCNnGMCLECCF	Message buffer RAM read error detect bit
0	Not detect error for reading from message buffer RAM.
1	Detect error for reading from message buffer RAM.

Notes

1. FCNnGMCLCTL.FCNnGMCLECCF is set (1) in case of detecting a memory error when reading from the message buffer RAM during the soft reset process. Once FCNnGMCLECCF is set (1), it keeps the level until it is cleared (0).
2. Although reading from message buffer RAM happens, when
   - reading a message buffer register,
   - reading a register that is related to receive and transmit history lists,
   - sending (transmitting) a message or
   - receiving a message,

   FCNnGMCLECCF is only evaluated during soft reset is ongoing. During any other operation, FCNnGMCLECCF is not updated.
3. It is impossible to clear FCNnGMCLECCF (0) during FCNnGMCLCTL.FCNnGMCLSORF is set (1) (soft reset is ongoing).

FCNnGMCLSORF	Soft reset execution status bit
0	No soft reset
1	Soft reset is ongoing

Notes

1. While a soft reset is ongoing (FCNnGMCLCTL.FCNnGMCLSORF is set (1)), it is impossible to set FCNnGMCLCTL.FCNnGMCLPWOM and FCNnGMCLCTL.FCNnGMCLESDE.
   It is possible to set start a software reset by FCNnGMCLCTL.FCNnGMCLSESR = 1 during FCNnGMCLCTL.FCNnGMCLPWOM bit is clear (0).
2. When FCNnGMCLCTL.FCNnGMCLSORF is set (1), the initialization of message buffer RAM starts. It is possible to detect error during initializing message buffer RAM, if FCNnGMCLCTL.FCNnGMCLECCF is cleared before setting FCNnGMCLSORF.
3. When FCNnGMCLCTL.FCNnGMCLSORF is set (1) again in the condition that is is already set (1), the soft reset procedure does not restart, but continues.
4. After release of the H/W reset FCNnGMCLCTL.FCNnGMCLSORF is set (1) automatically and initialization of message buffer RAM starts.
5. It is impossible that clearing FCNnGMCLCTL.FCNnGMCLPWOM (0) and setting FCNnGMCLCTL.FCNnGMCLSORF (1) are done at the same time.

6. If a hardware RESET occurs during FCNnGMCLCTL.FCNnGMCLSORF = 1, then the soft reset procedure is stopped (aborted), and the hardware RESET starts.

FCNnGMCLESDE	Bit enabling forced shut down
0	Forced shut down disabled.
1	Forced shut down of FCNnGMCLCTL.FCNnGMCLPWOM bit = 0 enabled.

Caution To request a forced shut down, FCNnGMCLCTL.FCNnGMCLPWOM must be cleared to 0 in a subsequent, immediately following access after FCNnGMCLCTL.FCNnGMCLESDE has been set to 1. If any access to another register (including reading the FCNnGMCLCTL register) is executed without clearing FCNnGMCLPWOM immediately after FCNnGMCLESDE has been set to 1, FCNnGMCLESDE is forcibly cleared to 0, and the forced shut down request is invalid.

FCNnGMCLPWOM	Global operation mode bit
0	CAN Controller is disabled.
1	CAN Controller is enabled to operate.

Caution FCNnGMCLCTL.FCNnGMCLPWOM can be cleared only in the initialization mode or immediately after FCNnGMCLCTL.FCNnGMCLESDE is set (forced shutdown).

(b) FCNnGMCLCTL write

15	14	13	12	11	10	9	8
0	0	0	FCNnGMCLSESR	0	0	FCNnGMCLSESD	FCNnGMCLSEOM
7	6	5	4	3	2	1	0
0	0	FCNnGMCLCLMB	0	0	0	0	FCNnGMCLCLOM

FCNnGMCLSESR	Software reset start
0	No changes.
1	Start soft reset.

FCNnGMCLSESD	FCNnGMCLSESD bit setting
0	No change in FCNnGMCLSESD bit.
1	FCNnGMCLESDE bit set to 1.

FCNnGMCLSEOM	FCNnGMCLCLOM	FCNnGMCLPWOM bit setting
0	1	FCNnGMCLCTL.FCNnGMCLPWOM bit cleared to 0.
1	0	FCNnGMCLCTL.FCNnGMCLPWOM bit set to 1.
Other than above		No change of FCNnGMCLCTL.FCNnGMCLPWOM bit.

Caution Set FCNnGMCLCTL.FCNnGMCLPWOM and FCNnGMCLCTL.FCNnGMCLESDE bit always separately.

FCNnGMCLCLMB	FCNnGMCLCTL.FCNnGMCLECCF bit clear
0	No change in FCNnGMCLCTL.FCNnGMCLECCF bit.
1	FCNnGMCLCTL.FCNnGMCLECCF bit cleared to 0.

(2) FCNnGMCSPRE - FCNn global clock selection register

This register is used to select the CAN channel clock.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0008_H

Initial Value $0F_H$

7	6	5	4	3	2	1	0
0	0	0	0	FCNnGMCSPRSC[3:0]			

FCNnGMCSPRSC[3:0]	CAN channel clock (f_{CANCHN})
0000_B	$f_{CAN}/1$
0001_B	$f_{CAN}/2$
0010_B	$f_{CAN}/3$
0011_B	$f_{CAN}/4$
0100_B	$f_{CAN}/5$
0101_B	$f_{CAN}/6$
0110_B	$f_{CAN}/7$
0111_B	$f_{CAN}/8$
1000_B	$f_{CAN}/9$
1001_B	$f_{CAN}/10$
1010_B	$f_{CAN}/11$
1011_B	$f_{CAN}/12$
1100_B	$f_{CAN}/13$
1101_B	$f_{CAN}/14$
1110_B	$f_{CAN}/15$
1111_B	$f_{CAN}/16$ (default value)

Note f_{CAN} = clock supplied to CAN Controller.

Caution Setting of the channel clock, CAN baudrate and time quantas (TQs) per data bit time (DBT) is allowed only in certain combinations.
Refer to the section *"CAN baudrate and time quanta"* above in this chapter.

(3) FCNnGMABCTL - FCNn global automatic block transmission control register

This register is used to control the automatic block transmission (ABT) operation.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8018_H$

Initial Value 0000_H

(a) FCNnGMABCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnGMABCLRF	FCNnGMABABTT

FCNnGMABCLRF	Automatic block transmission engine clear status bit
0	Clearing the automatic transmission engine is completed.
1	The automatic transmission engine is being cleared.

Notes

1. Start automatic transmission engine clearance by FCNnGMABCTL.FCNnGMABCLAT = 1 while FCNnGMABCTL.FCNnGMABCLRF = 0.
   The operation is not guaranteed if FCNnGMABCLRF is set to 1 while FCNnGMABCLRF = 1.

FCNnGMABABTT	Automatic block transmission status bit
0	Automatic block transmission is stopped.
1	Automatic block transmission is under execution.

(b) FCNnGMABCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	FCNnGMABSEAC	FCNnGMABSEAT
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnGMABCLAT

Note When the automatic block transmission engine is cleared by setting FCNnGMABCTL.FCNnGMABSEAC to 1, FCNnGMABCLRF is automatically set, and cleared to 0 as soon as the requested clearing processing is completed.

Cautions

1. Before changing the normal operation mode with ABT to the initialization mode, be sure to set the FCNnGMABCTL register to the default value (0000_H) and confirm the FCNnGMABCTL register is surely initialized to the default value (0000_H).
2. Do not start automatic block transmission in the initialization mode. If automatic block transmission is started in the initialization mode, the operation is not guaranteed after the CAN Controller has entered the normal operation mode with ABT.
3. Do not start automatic block transmission while FCNnCMCLCTL.FCNnCMCLSSTS is set to 1 (transmission in progress). Confirm FCNnCMCLSSTS = 0 directly in advance before starting automatic block transmission.

FCNnGMABSEAC	Automatic block transmission engine clear request bit
0	The automatic block transmission engine is in idle status or under operation.
1	Request to clear the automatic block transmission engine. After the automatic block transmission engine has been cleared, automatic block transmission is started from message buffer 0 by setting the FCNnGMABCTL.FCNnGMABABTT = 1.

FCNnGMABSEAT	FCNnGMABCLAT	Automatic block transmission start bit
0	1	Request to stop automatic block transmission.
1	0	Request to start automatic block transmission.
Other than above		No change of FCNnGMABCTL.FCNnGMABABTT.

(4) FCNnGMADCTL - FCNn global automatic block transmission delay register

This register is used to set the interval at which the data of the message buffer assigned to ABT is to be transmitted in the normal operation mode with ABT.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0020_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	FCNnGMADSSAD[3:0]			

FCNnGMADSSAD[3:0]	Data frame interval during automatic block transmission in DBT[a]
0000_B	0 DBT (default value)
0001_B	2^5 DBT
0010_B	2^6 DBT
0011_B	2^7 DBT
0100_B	2^8 DBT
0101_B	2^9 DBT
0110_B	2^{10} DBT
0111_B	2^{11} DBT
1000_B	2^{12} DBT
Other than above	Setting prohibited

a) Unit: Data bit time (DBT)

Cautions

1. Do not change the contents of the FCNnGMADCTL register while FCNnGMABCTL.FCNnGMABCLRF = 1 (clearing of ABT in progress).
2. The timing at which the ABT message is actually transmitted onto the CAN bus differs depending on the status of transmission from the other station or how a request to transmit a message other than an ABT message is made.

(5) FCNnDNBMRXk - FCNn data new bit monitor registers (k = 0 to 3)

These registers are used to read Data New Flags globally for several message buffers at a time.

Access These registers can be read in 32-bit units.

Address FCNnDNBMRX0: <FCNn_base> + 1 $00C0_H$
Following register is available only with m = 64 or 128 message buffers
FCNnDNBMRX1: <FCNn_base> + 1 $00D0_H$
Following registers are available only with m = 128 message buffers:
FCNnDNBMRX2: <FCNn_base> + 1 $00E0_H$
FCNnDNBMRX3: <FCNn_base> + 1 $00F0_H$

Initial Value $0000\ 0000_H$

FCNnDNBMRX0:

31 ... 0
FCNnDNBMSSDN[31:00]

FCNnDNBMRX1 (for m = 64 or 128 message buffers only):

31 ... 0
FCNnDNBMSSDN[63:32]

FCNnDNBMRX2 (for m = 128 message buffers only):

31 ... 0
FCNnDNBMSSDN[95:64]

FCNnDNBMRX3 (for m = 128 message buffers only):

31 ... 0
FCNnDNBMSSDN[127:96]

FCNnDNBMSSDN[31:0]	Message buffer data new bit
0	No remote or data frame has been stored into the message buffer.
1	A remote or data frame has been stored into the message buffer.

25.7.2 CAN channel registers

(1) FCNnCMMKCTLaH - FCNn channel mask control register

These registers are used to extend the number of receivable messages into the same message buffer by masking part of the identifier (ID) comparison of a message and invalidating the ID of the masked part.

Two 16-bit registers FCNnCMMKCTLaH (a = 01 to 16) can also be accessed via a single 32-bit access to the registers FCNnCMMKCTLaW (a = 01, 03, 05, 07, 09, 11, 13, 15).

Access The FCNnCMMKCTLaH registers can be read/written in 16-bit units.
The FCNnCMMKCTLaW registers can be read/written in 32-bit units.

Address FCnCMMKCTL01H: <FCNn_base> + $0\ 8300_H$
FCnCMMKCTL02H: <FCNn_base> + $0\ 8308_H$
FCnCMMKCTL03H: <FCNn_base> + $0\ 8310_H$
FCnCMMKCTL04H: <FCNn_base> + $0\ 8318_H$
FCnCMMKCTL05H: <FCNn_base> + $0\ 8320_H$
FCnCMMKCTL06H: <FCNn_base> + $0\ 8328_H$
FCnCMMKCTL07H: <FCNn_base> + $0\ 8330_H$
FCnCMMKCTL08H: <FCNn_base> + $0\ 8338_H$
FCnCMMKCTL09H: <FCNn_base> + $0\ 8340_H$
FCnCMMKCTL10H: <FCNn_base> + $0\ 8348_H$
FCnCMMKCTL11H: <FCNn_base> + $0\ 8350_H$
FCnCMMKCTL12H: <FCNn_base> + $0\ 8358_H$
FCnCMMKCTL13H: <FCNn_base> + $0\ 8360_H$
FCnCMMKCTL14H: <FCNn_base> + $0\ 8368_H$
FCnCMMKCTL15H: <FCNn_base> + $0\ 8370_H$
FCnCMMKCTL16H: <FCNn_base> + $0\ 8378_H$

FCnCMMKCTL01W: <FCNn_base> + $1\ 0300_H$
FCnCMMKCTL03W: <FCNn_base> + $1\ 0310_H$
FCnCMMKCTL05W: <FCNn_base> + $1\ 0320_H$
FCnCMMKCTL07W: <FCNn_base> + $1\ 0330_H$
FCnCMMKCTL09W: <FCNn_base> + $1\ 0340_H$
FCnCMMKCTL11W: <FCNn_base> + $1\ 0350_H$
FCnCMMKCTL13W: <FCNn_base> + $1\ 0360_H$
FCnCMMKCTL15W: <FCNn_base> + $1\ 0370_H$

Initial Value 0000_H

(a) FCNnCMMKCTLaH (a = 01, 03, 05, 07, 09, 11, 13, 15)

15	14	13	12	11	10	9	8
FCNnCMMKSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnCMMKSSID[7:0]							

(b) FCNnCMMKCTLaH (a = 02, 04, 06, 08, 10, 12, 14, 16)

15	14	13	12	11	10	9	8
0	0	0	FCNnCMMKSSID[28:24]				

7	6	5	4	3	2	1	0
FCNnCMMKSSID[23:16]							

(c) FCNnCMMKCTLaW (a = 01, 03, 05, 07, 09, 11, 13, 15)

31	30	29	28	27	26	25	24
0	0	0	FCNnCMMKSSID[28:24]				

23	22	21	20	19	18	17	16
FCNnCMMKSSID[23:16]							

15	14	13	12	11	10	9	8
FCNnCMMKSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnCMMKSSID[7:0]							

FCNnCMMKSSID[i] [a]	Mask pattern setting of ID bit
0	The ID bit **i** of the message buffer m set by FCNnMmSSID[**i**] are compared with the ID bits of the received message frame.
1	The ID bit **i** of the message buffer m set by FCNnMmSSID[**i**] are not compared with the ID bits of the received message frame (they are masked).

a) i = [28:0]

Note Masking is always defined by an ID length of 29 bits. If a mask is assigned to a message with a standard ID, FCNnCMMKSSID[17:0] are ignored. Therefore, only FCNnCMMKSSID[28:18] of the received ID are masked. The same mask can be used for both the standard and extended IDs.

(2) FCNnCMCLCTL - FCNn channel control register

This register is used to control the operation mode of the CAN Controller.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8240_H

Initial Value 0000_H

(a) FCNnCMCLCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	FCNnCM CLSSRS	FCNnCM CLSSTS

7	6	5	4	3	2	1	0
FCNnCM CLERCF	FCNnCM CLALBF	FCNnCM CLVALF	FCNnCM CLMDPF[1:0]		FCNnCM CLMDOF[2:0]		

FCNnCMCLSSRS	Reception status bit
0	Reception is stopped.
1	Reception is in progress.

Notes

1. FCNnCMCLSSRS is set to 1 under the following conditions (timing)
   - The SOF bit of a receive frame is detected
   - On occurrence of arbitration loss during a transmit frame
2. FCNnCMCLSSRS is cleared to 0 under the following conditions (timing)
   - When a recessive level is detected at the second bit of the interframe space
   - On transition to the initialization mode at the first bit of the interframe space

FCNnCMCLSSTS	Transmission status bit
0	Transmission is stopped.
1	Transmission is in progress.

Notes

1. FCNnCMCLSSTS is set to 1 under the following conditions (timing)
   - The SOF bit of a transmit frame is detected
2. FCNnCMCLSSTS is cleared to 0 under the following conditions (timing)
   - During transition to bus-off state
   - On occurrence of arbitration loss in transmit frame
   - On detection of recessive level at the second bit of the interframe space
   - On transition to the initialization mode at the first bit of the interframe space

FCNnCMCLERCF	Error counter clear bit
0	The FCNnCMERCNT and FCNnCMLCSTR registers are not cleared in the initialization mode.
1	The FCNnCMERCNT and FCNnCMLCSTR registers are cleared in the initialization mode.

Caution The error counter register FCNnCMERCNT and the information register FCNnCMINSTR are cleared by FCNnCMCLERCF bit only following conditions:

- During Bus-Off state, while in initialization mode.
- After starting the CAN Controller (FCNnGMCLPWOM is set at FCNnGMCLPWOM=0), while in initialization mode.
- After aborting all transmit request according to *Figure 25-27 “Transmission abort processing (except normal operation mode with ABT)” on page 1999* or *Figure 25-28 “Transmission abort processing (normal operation mode with ABT) - Repeat option for aborted message” on page 2000* , while in initialization mode.

Notes
1. When the FCNnCMERCNT and FCNnCMLCSTR registers have been cleared, FCNnCMCLERCF is also cleared to 0 automatically.
2. FCNnCMCLERCF can be set to 1 at the same time as a request to change the initialization mode to an operation mode is made.
3. FCNnCMCLERCF is read-only in the CAN Controller sleep or stop mode.
4. The error counters can also be cleared by shut down or forced shut down of the CAN controller.

FCNnCMCLALBF	Bit to set operation in case of arbitration loss
0	Re-transmission is not executed in case of an arbitration loss in the single-shot mode.
1	Re-transmission is executed in case of an arbitration loss in the single-shot mode.

Note FCNnCMCLALBF is valid only in the single-shot mode.

FCNnCMCLVALF	Valid receive message frame detection bit
0	A valid message frame has not been received since FCNnCMCLVALF was last cleared to 0.
1	A valid message frame has been received since FCNnCMCLVALF was last cleared to 0.

Notes
1. Detection of a valid receive message frame is not dependent upon storage in the receive message buffer (data frame) or transmit message buffer (remote frame).
2. Clear FCNnCMCLVALF (0) before changing the initialization mode to an operation mode.

3. If only two CAN nodes are connected to the CAN bus with one transmitting a message frame in the normal mode and the other in the receive-only mode, FCNnCMCLVALF is not set to 1 before the transmitting node enters the error passive state, because in receive-only mode no acknowledge is generated.
4. To clear FCNnCMCLVALF, set FCNnCMCLCLVL to 1 first and confirm that FCNnCMCLVALF is cleared. If it is not cleared, perform clearing processing again.

FCNnCMCLMDPF[1:0]	Power save mode
00_B	No power save mode is selected.
01_B	CAN Controller sleep mode
10_B	Setting prohibited
11_B	CAN Controller stop mode

Cautions

1. Transition to and from the CAN Controller stop mode must be made via sleep mode. A request for direct transition to and from the stop mode is ignored.
2. The FCNnGMCLSSMO flag of FCNnGMCLCTL must be checked after releasing a power save mode, prior to access the message buffers again.
3. CAN Controller sleep mode requests are kept pending, until cancelled by software or entered on appropriate bus condition (bus idle). Software can check the actual status by reading FCNnCMCLMDPF[1:0].

Note In case that the CAN bus is blocked on dominant level, so that the CAN Controller could not synchronize since initialization mode was left, the sleep mode can be reached nevertheless. In this case however, the wake up from sleep mode will happen after the first recessive to dominant edge, after a synchronization was successful. Waking up by software is possible in any case.

FCNnCMCLMDOF[2:0]	Operation mode
000_B	No operation mode is selected (CAN Controller is in the initialization mode).
001_B	Normal operation mode
010_B	Normal operation mode with automatic block transmission function (normal operation mode with ABT)
011_B	Receive-only mode
100_B	Single-shot mode
101_B	Self-test mode
Other than above	Setting prohibited

Caution Transit to initialization mode or power saving modes may take some time. Be sure to verify the success of mode change by reading the values, before proceeding.

Note FCNnCMCLMDOF[2:0] are read-only in the CAN Controller sleep or stop mode.

(b) FCNnCMCLCTL write

15	14	13	12	11	10	9	8
FCNnCMCLSERC	FCNnCMCLSEAL	0	FCNnCMSESEPS[1:0]		FCNnCMCLSEOP[2:0]		
7	6	5	4	3	2	1	0
0	FCNnCMCLCLAL	FCNnCMCLCLVL	FCNnCMSECLPS[1:0]		FCNnCMCLCLOP[2:0]		

FCNnCMCLSERC	Setting of FCNnCMCLERCF bit
1	FCNnCMCLERCF is set to 1.
Other than above	FCNnCMCLERCF is not changed.

FCNnCMCLSEAL	FCNnCMCLCLAL	Setting of FCNnCMCLALBF bit
0	1	FCNnCMCLALBF is cleared to 0.
1	0	FCNnCMCLALBF is set to 1.
Other than above		FCNnCMCLALBF is not changed.

FCNnCMCLCLVL	Setting of FCNnCMCLVALF bit
0	FCNnCMCLVALF is not changed.
1	FCNnCMCLVALF is cleared to 0.

FCNnCMSESEPS0	FCNnCMSECLPS0	Setting of FCNnCMCLMDPF0 bit
0	1	FCNnCMCLMDPF0 is cleared to 0.
1	0	FCNnCMCLMDPF0 is set to 1.
Other than above		FCNnCMCLMDPF0 is not changed.

FCNnCMSESEPS1	FCNnCMSECLPS1	Setting of FCNnCMCLMDPF1 bit
0	1	FCNnCMCLMDPF1 is cleared to 0.
1	0	FCNnCMCLMDPF1 is set to 1.
Other than above		FCNnCMCLMDPF1 is not changed.

FCNnCMCLSEOP0	FCNnCMCLCLOP0	Setting of FCNnCMCLMDOF0 bit
0	1	FCNnCMCLMDOF0 is cleared to 0.
1	0	FCNnCMCLMDOF0 is set to 1.
Other than above		FCNnCMCLMDOF0 is not changed.

FCNnCMCLSEOP1	FCNnCMCLCLOP1	Setting of FCNnCMCLMDOF1 bit
0	1	FCNnCMCLMDOF1 is cleared to 0.
1	0	FCNnCMCLMDOF1 is set to 1.
Other than above		FCNnCMCLMDOF1 is not changed.

FCNnCMCLSEOP2	FCNnCMCLCLOP2	Setting of FCNnCMCLMDOF2 bit
0	1	FCNnCMCLMDOF2 is cleared to 0.
1	0	FCNnCMCLMDOF2 is set to 1.
Other than above		FCNnCMCLMDOF2 is not changed.

Caution When setting initialization mode while reception is ongoing in an operation mode, a last reception may occur, which sets the data new flag of a message box. However, the transition back to an operation mode also clears the Receive History List. Therefore, reaching the initialization mode must be confirmed by software by reading back the operation mode. Before restarting an operation mode, all set data new flags of all active receive message boxes must be cleared, before activating an operation mode again.

(3) FCNnCMLCSTR - FCNn channel last error information register

This register provides the error information of the CAN protocol.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0248_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	FCN0CMLCSSLC[2:0]		

Notes

1. The contents of the FCNnCMLCSTR register are not cleared when the CAN Controller changes from an operation mode to the initialization mode.
2. If an attempt is made to write a value other than 00_H to the FCNnCMLCSTR register by software, the access is ignored.

FCN0CMLCSSLC[2:0]	Last CAN protocol error information
000_B	No error
001_B	Stuff error
010_B	Form error
011_B	ACK error
100_B	Bit error. (The CAN Controller tried to transmit a recessive-level bit as part of a transmit message (except the arbitration field), but the value on the CAN bus is a dominant-level bit.)
101_B	Bit error. (The CAN Controller tried to transmit a dominant-level bit as part of a transmit message, ACK bit, error frame, or overload frame, but the value on the CAN bus is a recessive-level bit.)
110_B	CRC error
111_B	Undefined

(4) FCNnCMINSTR - FCNn channel information register

This register indicates the status of the CAN Controller.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + 0 024C$_H$

Initial Value 00$_H$

7	6	5	4	3	2	1	0
0	0	0	FCNnCM INBOFF	FCNnCM INSSTE[1:0]		FCNnCM INSSRE[1:0]	

FCNnCMINBOFF	Bus-off state bit
0	Not bus-off state (transmit error counter ≤ 255). (The value of the transmit counter is less than 256.)
1	Bus-off state (transmit error counter > 255). (The value of the transmit counter is 256 or more.)

FCNnCMINSSTE[1:0]	Transmission error counter status bit
00$_B$	The value of the transmission error counter is less than that of the warning level (< 96).
01$_B$	The value of the transmission error counter is in the range of the warning level (96 to 127).
10$_B$	Undefined
11$_B$	The value of the transmission error counter is in the range of the error passive or bus-off status (≥ 128).

FCNnCMINSSRE[1:0]	Reception error counter status bit
00$_B$	The value of the reception error counter is less than that of the warning level (< 96).
01$_B$	The value of the reception error counter is in the range of the warning level (96 to 127).
10$_B$	Undefined
11$_B$	The value of the reception error counter is in the error passive range (≥ 128).

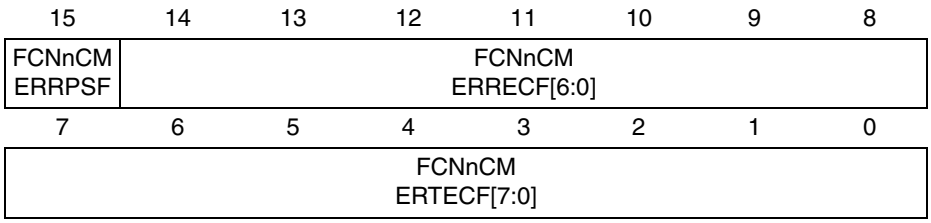
(5) FCNnCMERCNT - FCNn channel error counter register

This register indicates the count value of the transmission/reception error counter.

Access This register is read-only in 16-bit units.

Address <FCNn_base> + $0\ 8250_H$

Initial Value 0000_H



FCNnCMERRPSF	Reception error passive status bit
0	The reception error counter is not in the error passive range (< 128)
1	The reception error counter is in the error passive range (≥ 128)

FCNnCMERRECF[6:0]	Reception error counter bit
0 to 127	Number of reception errors. These bits reflect the status of the reception error counter. The number of errors is defined by the CAN protocol.

Note FCNnCMERRECF[6:0] are invalid in the reception error passive state (FCNnCMINSTR.FCNnCMINSSRE[1:0] = 11_B).

FCNnCMERTECF[7:0]	Transmission error counter bit
0 to 255	Number of transmission errors. These bits reflect the status of the transmission error counter. The number of errors is defined by the CAN protocol.

Note FCNnCMERTECF[7:0] are invalid in the bus-off state (FCNnCMINSTR.FCNnCMINBOFF = 1).

(6) FCNnCMIECTL - FCNn channel interrupt enable register

This register is used to enable or disable the interrupts of the CAN Controller.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8258_H$

Initial Value 0000_H

(a) FCNnCMIECTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMIEINTF[6:0]						

FCNnCMIEINTF[6:0]	CAN Controller interrupt enable bit
0	Output of the interrupt corresponding to interrupt status register FCNnCMISCTL is disabled.
1	Output of the interrupt corresponding to interrupt status register FCNnCMISCTL is enabled.

(b) FCNnCMIECTL write

15	14	13	12	11	10	9	8
0	FCNnCMIESEIE[6:0]						
7	6	5	4	3	2	1	0
0	FCNnCMIECLIE[6:0]						

FCNnCMIESEIE[6:0]	FCNnCMIECLIE[6:0]	Setting of FCNnCMIEINTF[6:0] bit
0	1	FCNnCMIEINTF[6:0] bit is cleared to 0.
1	0	FCNnCMIEINTF[6:0] bit is set to 1.
Other than above		FCNnCMIEINTF[6:0] bit is not changed.

(7) FCNnCMISCTL - FCNn channel interrupt status register

This register indicates the interrupt status of the CAN Controller.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8260_H$

Initial Value 0000_H

(a) FCNnCMISCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMISITSF[6:0]						

FCNnCMISITSF[6:0]	CAN Controller interrupt status bit
0	No related interrupt source event is pending
1	A related interrupt source event is pending

Interrupt status bit	Related interrupt source event
FCNnCMISITSF6	CAN Controller transmission abort interrupt status bit
FCNnCMISITSF5	Wakeup interrupt from CAN Controller sleep mode [a]
FCNnCMISITSF4	Arbitration loss interrupt
FCNnCMISITSF3	CAN protocol error interrupt
FCNnCMISITSF2	CAN error status interrupt
FCNnCMISITSF1	Interrupt on completion of reception of valid message frame to message buffer m
FCNnCMISITSF0	Interrupt on normal completion of transmission of message frame from message buffer m

a) FCNnCMISITSF5 is set only when the CAN Controller is woken up from the CAN Controller sleep mode by a CAN bus operation. It is not set when the sleep mode has been released by software.

(b) FCNnCMISCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	FCNnCMISITSF[6:0]						

FCNnCMISITSF[6:0]	Clearing of FCNnCMISITSF[6:0]
0	FCNnCMISITSF[6:0] bits are not changed
1	FCNnCMISITSF[6:0] bits are cleared to 0

Caution Clear the status bit of this register by software, when the confirmation of each status is necessary in the interrupt processing, because these bits are not cleared automatically.

(8) FCNnCMBRPRS - FCNn channel bit rate prescaler register

This register is used to select the CAN protocol layer clock (f_{TQ}). The communication baudrate is set to the FCNnCMBTCTL register.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + $0\ 0268_H$

Initial Value FF_H

7	6	5	4	3	2	1	0
FCNnCMBRPRS[7:0]							

FCNnCMBRPRS	CAN protocol layer clock (f_{TQ})
0	$f_{CANCHN}/1$
1	$f_{CANCHN}/2$
n	$f_{CANCHN}/(n+1)$
:	:
255	$f_{CANCHN}/256$ (default value)

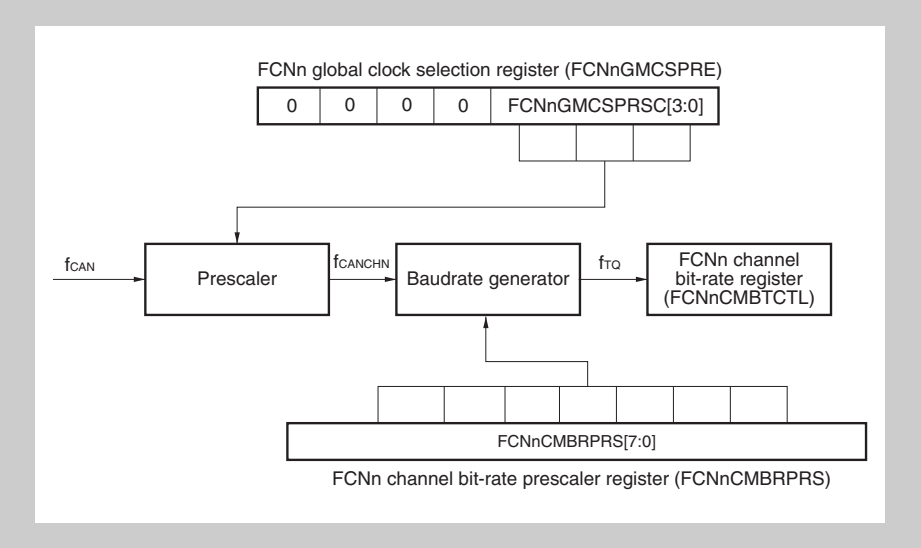


Figure 25-4 CAN Controller clocks

Note
f_{CAN}: clock supplied to CAN Controller
f_{CANCHN}: CAN channel clock
f_{TQ}: CAN protocol layer clock

Cautions
1. FCNnCMBRPRS can be write-accessed only in the initialization mode.
2. Setting of the channel clock, CAN baudrate and time quantas (TQs) per data bit time (DBT) is allowed only in certain combinations.
   Refer to the section *"CAN baudrate and time quanta"* above in this chapter.

(9) FCNnCMBTCTL - FCNn channel bit rate register

This register is used to control the data bit time of the communication baudrate.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8270_H$

Initial Value $0370F_H$

15	14	13	12	11	10	9	8
0	0	FCNnCM BTJWLG[1:0]		0	FCNnCM BTS2LG[2:0]		
7	6	5	4	3	2	1	0
0	0	0	0	FCNnCMBTS1LG[3:0]			

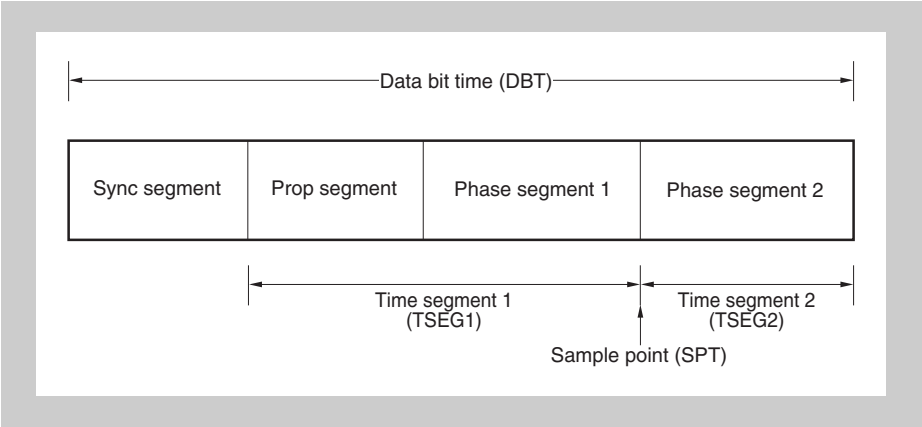


Figure 25-5 Data bit time

FCNnCMBTJWLG[1:0]	Length of synchronization jump width (SJW)
00_B	$1T_Q$
01_B	$2T_Q$
10_B	$3T_Q$
11_B	$4T_Q$ (default value)

FCNnCMBTS2LG[2:0]	Length of time segment 2 (TSEG2)
000_B	$1T_Q$
001_B	$2T_Q$
010_B	$3T_Q$
011_B	$4T_Q$
100_B	$5T_Q$
101_B	$6T_Q$
110_B	$7T_Q$
111_B	$8T_Q$ (default value)

FCNnCMBTS1LG[3:0]	Length of time segment 1(TSEG1)
0000_B	Setting prohibited
0001_B	2 T_Q[a]
0010_B	3 T_Q[a]
0011_B	4 T_Q
0100_B	5 T_Q
0101_B	6 T_Q
0110_B	7 T_Q
0111_B	8 T_Q
1000_B	9 T_Q
1001_B	10 T_Q
1010_B	11 T_Q
1011_B	12 T_Q
1100_B	13 T_Q
1101_B	14 T_Q
110_B	15 T_Q
1111_B	16 T_Q (default value)

a) This setting must not be made when the FCNnCMBRPRS register = 00_H

Note $T_Q = 1/f_{TQ}$ (f_{TQ}: CAN protocol layer clock)

Caution Setting of the CAN channel clock, CAN baudrate and time quantas (TQs) per data bit time (DBT) is allowed only in certain combinations.
Refer to the section *"CAN baudrate and time quanta"* above in this chapter.

(10) FCNnCMLISTR - FCNn channel last in-pointer register

This register indicates the number of the message buffer in which a data frame or a remote frame was last stored.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + 0 0278_H

Initial Value Undefined.

7	6	5	4	3	2	1	0
FCNnCMLISSLT[7:0]							

FCNnCMLISSLT[7:0]	Last in-pointer register
0 to 31 [a] 0 to 63 [b] 0 to 127 [c]	When the FCNnCMLISTR register is read, the contents of the element indexed by the last in-pointer (FCNnCMLISSLT[7:0]) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame was last stored.

a) On 32 message buffer CAN Controller.
b) On 64 message buffer CAN Controller.
c) On 128 message buffer CAN Controller.

Note The read value of FCNnCMLISTR is undefined if a data frame or a remote frame has never been stored in the message buffer. If FCNnCMRGRX.FCNnCMRGSSPM is set to 1 after the CAN Controller has changed from the initialization mode to an operation mode, therefore, the read value of FCNnCMLISTR is undefined.

(11) FCNnCMRGRX - FCNn channel receive history list register

This register is used to read the receive history list (RHL).

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8280_H

Initial Value $xx02_H$

(a) FCNnCMRGRX read

15	14	13	12	11	10	9	8
FCNnCMRGSSPT[7:0]							
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnCM RGSSPM	FCNnCM RGRVFF

FCNnCMRGSSPT[7:0]	Receive history list read pointer
0 to 31 [a] 0 to 63 [b] 0 to 127 [c]	When FCNnCMRGRX is read, the contents of the element indexed by the receive history list get pointer (FCNnCMRGRX.FCNnCMRGSSPT) of the receive history list are read. These contents indicate the number of the message buffer in which a data frame or a remote frame has been stored.

a) On 32 message buffer CAN Controller.
b) On 64 message buffer CAN Controller.
c) On 128 message buffer CAN Controller.

FCNnCMRGSSPM[a]	Receive history list pointer match
0	The receive history list has at least one message buffer number that has not been read.
1	The receive history list has no message buffer numbers that have not been read.

a) The read value of FCNnCMRGSSPT[7:0] is invalid when FCNnCMRGSSPM = 1.

FCNnCMRGRVFF[a]	Receive history list overflow bit [b]
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers in which a new data frame or remote frame has been received and stored are recorded to the receive history list (the receive history list has a vacant element).
1	At least **(i)** entries have been stored since the host processor has serviced the RHL last time (i.e. read FCNnCMRGRX). The first **(i-1)** entries are sequentially stored while the last entry can have been overwritten whenever newly received message is stored, because all buffer numbers are stored at position **(i)**, when FCNnCMRGRVFF is set. Thus the sequence of receptions can not be recovered completely now.

a) If FCNnCMRGRVFF is set, FCNnCMRGSSPM is no longer cleared on message storage, but FCNnCMRGSSPM is still set, if all entries of FCNnCMRGRX have been read by software.
b) i = 23 on 32 message buffer CAN Controller;
i = 47 on 64 message buffer CAN Controller;
i = 95 on 128 message buffer CAN Controller.

(b) FCNnCMRGRX write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnCMRGCLRV

FCNnCMRGCLRV	Clearing of FCNnCMRGRVFF bit
0	FCNnCMRGRVFF bit is not changed.
1	FCNnCMRGRVFF bit is cleared to 0.

(12) FCNnCMLOSTR - FCNn channel last out-pointer register

This register indicates the number of the message buffer, from which a data frame or a remote frame was transmitted last.

Access This register is read-only in 8-bit units.

Address <FCNn_base> + $0\ 0288_H$

Initial Value Undefined

7	6	5	4	3	2	1	0
FCNnCMLOSTR[7:0]							

FCNnCMLOSTR[7:0]	Last out-pointer of transmit history list
0 to 31 [a] 0 to 63 [b] 0 to 127 [c]	When the FCNnCMLOSTR register is read, the contents of the element indexed by the last out-pointer (FCNnCMLOSTR[7:0]) of the receive history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

a) On 32 message buffer CAN Controller.
b) On 64 message buffer CAN Controller.
c) On 128 message buffer CAN Controller.

Note The value read from the FCNnCMLOSTR register is undefined if a data frame or remote frame has never been transmitted from a message buffer. If FCNnCMTGTX.FCNnCMTGSSPM is set to 1 after the CAN Controller has changed from the initialization mode to an operation mode, therefore, the read value of the FCNnCMLOSTR register is undefined.

(13) FCNnCMTGTX - FCNn channel transmit history list register

This register is used to read the transmit history list (THL).

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 8290_H

Initial Value $xx02_H$

(a) FCNnCMTGTX read

15	14	13	12	11	10	9	8
FCNnCMTGSSPT[7:0]							
7	6	5	4	3	2	1	0
0	0	0	0	0	0	FCNnCMTGSSPM	FCNnCMTGTVFF

FCNnCMTGSSPT[7:0]	Transmit history list read pointer
0 to 31 [a] 0 to 63 [b] 0 to 127 [c]	When the FCNnCMTGTX register is read, the contents of the element indexed by the read pointer (FCNnCMTGSSPT[7:0]) of the transmit history list are read. These contents indicate the number of the message buffer to which a data frame or a remote frame was transmitted last.

a) On 32 message buffer CAN Controller.
b) On 64 message buffer CAN Controller.
c) On 128 message buffer CAN Controller.

FCNnCMTGSSPM[a]	Transmit history pointer match
0	The transmit history list has at least one message buffer number that has not been read.
1	The transmit history list has no message buffer numbers that have not been read.

a) The read value of FCNnCMTGSSPT[7:0] is invalid when the FCNnCMTGSSPM = 1.

FCNnCMTGTVFF[a]	Transmit history list overflow bit [b]
0	All the message buffer numbers that have not been read are preserved. All the numbers of the message buffers to which a new data frame or remote frame has been transmitted are recorded to the transmit history list (the transmit history list has a vacant element).
1	At least **(i)** entries have been stored since the host processor has serviced the THL last time (i.e. read FCNnCMTGTX). The first **(i-1)** entries are sequentially stored while the last entry can have been overwritten whenever newly received message is stored, because all buffer numbers are stored at position **(i)**, when FCNnCMTGTVFF is set. Thus the sequence of receptions can not be recovered completely now.

a) If FCNnCMTGTVFF is set, FCNnCMTGSSPM is no longer cleared on message transmission, but FCNnCMTGSSPM is still set, if all entries of FCNnCMTGTX are read by software.

b) i = 7 on 32 message buffer CAN Controller
i = 15 on 64 message buffer CAN Controller;
i = 31 on 128 message buffer CAN Controller.

Note Transmission from message buffers ...

- 0 to 8 (for 32 message buffer CAN Controller)
- 0 to 16 (for 64 message buffer CAN Controller)
- 0 to 32 (for 128 message buffer CAN Controller)

... is not recorded to the transmit history list in the normal operation mode with ABT.

(b) FCNnCMTGTX write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	FCNnCMTGCLTV

FCNnCMTGCLTV	Setting of FCNnCMTGTVFF bit
0	FCNnCMTGTVFF bit is not changed
1	FCNnCMTGTVFF bit is cleared to 0

(14) FCNnCMTSCTL - FCNn channel time stamp register

This register is used to control the time stamp function.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + $0\ 8298_H$

Initial Value 0000_H

(a) FCNnCMTSCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	FCNnCMTSLOKE	FCNnCMTSSELE	FCNnCMTSTSGE

Note The lock function of the time stamp function must not be used when the CAN Controller is in the normal operation mode with ABT.

FCNnCMTSLOKE	Time stamp lock function enable bit
0	Time stamp lock function stopped. The TSOUT signal is toggled each time the selected time stamp capture event occurs.
1	Time stamp lock function enabled. The TSOUT signal is toggled each time the selected time stamp capture event occurs. However, the TSOUT output signal is locked when a data frame has been correctly received to message buffer 0 [a].

a) FCNnCMTSTSGE is automatically cleared to 0.

FCNnCMTSSELE	Time stamp capture event selection bit
0	The time capture event is SOF.
1	The time stamp capture event is the last bit of EOF.

FCNnCMTSTSGE	TSOUT operation setting bit
0	TSOUT toggle operation is disabled.
1	TSOUT toggle operation is enabled.

(b) FCNnCMTSCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	FCNnCMTSSELK	FCNnCMTSSES	FCNnCMTSSETS
7	6	5	4	3	2	1	0
0	0	0	0	0	FCNnCMTSCLLK	FCNnCMTSCLSL	FCNnCMTSCLTS

FCNnCMTSSELK	FCNnCMTSCLLK	Setting of FCNnCMTSLOKE bit
0	1	FCNnCMTSLOKE is cleared to 0.
1	0	FCNnCMTSLOKE is set to 1.
Other than above		FCNnCMTSLOKE is not changed.

FCNnCMTSSES	FCNnCMTSCLSL	Setting of FCNnCMTSSELE bit
0	1	FCNnCMTSSELE is cleared to 0.
1	0	FCNnCMTSSELE is set to 1.
Other than above		FCNnCMTSSELE is not changed.

FCNnCMTSSETS	FCNnCMTSCLTS	Setting of FCNnCMTSTSGE bit
0	1	FCNnCMTSTSGE is cleared to 0.
1	0	FCNnCMTSTSGE is set to 1.
Other than above		FCNnCMTSTSGE is not changed.

25.7.3 Message buffer registers

(1) FCNnMmDATxB/H/W, FCNn message data byte registers

These registers are used to store the data of a transmit/receive message.

Access The FCNnMmDATxW registers can be read/written in 32-bit units.
The FCNnMmDATxH registers can be read/written in 16-bit units.
The FCNnMmDATxB registers can be read/written in 8-bit units.

Address FCNnMmDAT0B: <FCNn_base> + 0 1000_H + m x 40_H
FCNnMmDAT1B: <FCNn_base> + 0 1004_H + m x 40_H
FCNnMmDAT2B: <FCNn_base> + 0 1008_H + m x 40_H
FCNnMmDAT3B: <FCNn_base> + 0 $100C_H$ + m x 40_H
FCNnMmDAT4B: <FCNn_base> + 0 1010_H + m x 40_H
FCNnMmDAT5B: <FCNn_base> + 0 1014_H + m x 40_H
FCNnMmDAT6B: <FCNn_base> + 0 1018_H + m x 40_H
FCNnMmDAT7B: <FCNn_base> + 0 $101C_H$ + m x 40_H

FCNnMmDAT0H: <FCNn_base> + 0 9000_H + m x 40_H
FCNnMmDAT2H: <FCNn_base> + 0 9008_H + m x 40_H
FCNnMmDAT4H: <FCNn_base> + 0 9010_H + m x 40_H
FCNnMmDAT6H: <FCNn_base> + 0 9018_H + m x 40_H

FCNnMmDAT0W: <FCNn_base> + 1 1000_H + m x 40_H
FCNnMmDAT4W: <FCNn_base> + 1 1010_H + m x 40_H

Initial Value Undefined.

(a) FCNnCMmDATxB (x = 0 to 7)

7	6	5	4	3	2	1	0
FCNnMmSSD0[7:0], FCNnMmSSD1[7:0], FCNnMmSSD2[7:0], FCNnMmSSD3[7:0], FCNnMmSSD4[7:0], FCNnMmSSD5[7:0], FCNnMmSSD6[7:0], FCNnMmSSD7[7:0]							

(b) FCNnCMmDATyH (y = 0, 2, 4, 6)

15	14	13	12	11	10	9	8
FCNnMmSSD0[7:0], FCNnMmSSD2[7:0], FCNnMmSSD4[7:0], FCNnMmSSD6[7:0]							

7	6	5	4	3	2	1	0
FCNnMmSSD1[7:0], FCNnMmSSD3[7:0], FCNnMmSSD5[7:0], FCNnMmSSD7[7:0]							

(c) FCNnCMmDATzW (z = 0, 4)

31	30	29	28	27	26	25	24
FCNnMmSSD0[7:0], FCNnMmSSD4[7:0]							

23	22	21	20	19	18	17	16
FCNnMmSSD1[7:0], FCNnMmSSD5[7:0]							

15	14	13	12	11	10	9	8
FCNnMmSSD2[7:0], FCNnMmSSD6[7:0]							

7	6	5	4	3	2	1	0
FCNnMmSSD3[7:0], FCNnMmSSD7[7:0]							

(2) FCNnMmDTLGB - FCNn message data length register m

This register is used to set the number of bytes of the data field of a message buffer.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0 1020_H + m x 40_H

Initial Value Undefined.

7	6	5	4	3	2	1	0
0	0	0	0	FCNnMmDTLG[3:0]			

FCNnMmDTLG[3:0]	Data length of transmit/receive message
0000_B	0 bytes
0001_B	1 byte
0010_B	2 bytes
0001_B	3 bytes
0100_B	4 bytes
0101_B	5 bytes
0110_B	6 bytes
0111_B	7 bytes
1000_B	8 bytes
1001_B	Setting prohibited (If these bits are set during transmission, 8-byte data is transmitted regardless of the set DLC value when a data frame is transmitted. However, the DLC actually transmitted to the CAN bus is the DLC value set to this register.)Note
1010_B	
1001_B	
1100_B	
1101_B	
1110_B	
1111_B	

Note The data and DLC value actually transmitted to CAN bus are as follows.

Type of transmit frame	Length of transmit data	DLC transmitted
Data frame	Number of bytes specified by DLC (However, 8 bytes if DLC ≥ 8)	FCNnMmDTLGB.FCNnMmDTLG[3:0] bits
Remote frame	0 bytes	

Cautions

1. Be sure to set bits 7 to 4 to 0000_B.
2. Receive data is stored in as many FCNnMmDATxB register as the number of bytes (however, the upper limit is 8) corresponding to DLC of the received frame. The FCNnMmDATxB register in which no data is stored is undefined.
3. On reception, FCNnMmDTLGB is updated according to the received frame.

(3) FCNnMmSTRB - FCNn message configuration register m

This register is used to specify the type of the message buffer and to set a mask.

Access This register can be read/written in 8-bit units.

Address <FCNn_base> + 0 1024_H + m x 40_H

Initial Value 00_H

7	6	5	4	3	2	1	0
FCNnMm SSOW	FCNnMm SSMT[3:0]				FCNnMm SSRT	0	FCNnMm SSAM

FCNnMmSSOW	Overwrite control bit
0	The message buffer that has already received a data frame[a] is not overwritten by a newly received data frame. The newly received data frame is discarded.
1	The message buffer that has already received a data frame[a] is overwritten by a newly received data frame.

a) The “message buffer that has already received a data frame” is a receive message buffer whose FCNnMmCTL.FCNnMmDTNF bit has been set to 1.

Note A remote frame is received and stored, regardless of the setting of FCNnMmCTL.FCNnMmSSOW and FCNnMmCTL.FCNnMmDTNF. A remote frame that satisfies the other conditions

- either:
  ID matches exactly, FCNnMmSTRB.FCNnMmSSRT = 0, FCNnMmCTL.FCNnMmTRQF = 0, FCNnMmSSMT = 0000_B
- or:
  ID matches with or without mask, FCNnMmSTRB.FCNnMmSSRT = 1, FCNnMmSSMT <> 0000_B

is always received and stored in the corresponding message buffer (interrupt generated, FCNnMmDTNF flag set, FCNnMmDTLGB.FCNnMmDTLG[3:0] updated, and recorded to the receive history list).

FCNnMmSSRT	Remote frame request bit
0	Transmit / receive a data frame.
1	Transmit / receive a remote frame.

FCNnMmSTRB.FCNnMmSSRT specifies the type of message frame that is transmitted or received from/to a message buffer.

Notes

1. If the message buffer is defined as a transmit message buffer, and a remote frame shall be received into it, the FCNnMmSSRT bit must be cleared.
2. Even if a valid remote frame has been received in a transmit message buffer, the FCNnMmSSRT bit of the transmit message buffer that has received the frame remains cleared to 0.

3. Even if a remote frame whose ID matches has been received from the CAN bus, if the FCNnMmSSRT bit of a transmit message buffer is set to 1 (to transmit a remote frame), that remote frame is not stored in this transmit message buffer.
4. If the message buffer is defined as a receive message buffer, the FCNnMmSSRT bit must be set, in order to receive remote frames instead of data frames.

FCNnMmSSMT[3:0]	Message buffer type setting bit
0000_B	Transmit message buffer
0001_B	Receive message buffer (no mask setting)
0010_B	Receive message buffer (mask 1 set)
0011_B	Receive message buffer (mask 2 set)
0100_B	Receive message buffer (mask 3 set)
0101_B	Receive message buffer (mask 4 set)
0110_B	Receive message buffer (mask 5 set)
0111_B	Receive message buffer (mask 6 set)
1000_B	Receive message buffer (mask 7 set)
1001_B	Receive message buffer (mask 8 set)
Other than above	Setting prohibited

Note The setting of FCNnMmSSMT is also valid to select masks in conjunction with remote frame reception. To receive remote frames in receive message buffers, the flag FCNnMmSSRT of the message buffer must be set.

FCNnMmSSAM	Message buffer assignment bit
0	Message buffer not used.
1	Message buffer used.

Caution Be sure to write 0 to bits 2 and 1.

(4) FCNnMmMID0H, FCNnMmMID1H, FCNnMmMID0W - FCNn message ID register m

These registers are used to set an identifier (ID).

Access FCNnMmMID0H, FCNnMmMID1H can be read/written in 16-bit units.
FCNnMmMID0W can be read/written in 32-bit units.

Address FCNnMmMID0H: <FCNn_base> + 0 9028_H + m x 40_H
FCNnMmMID1H: <FCNn_base> + 0 9030_H + m x 40_H

FCNnMmMID0W: <FCNn_base> + 1 1028_H + m x 40_H

Initial Value Undefined.

(a) FCNnMmMID0H

15	14	13	12	11	10	9	8
FCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnMmSSID[7:0]							

(b) FCNnMmMID1H

15	14	13	12	11	10	9	8
FCNnMmSSIE	0	0	FCNnMmSSID[28:24]				

7	6	5	4	3	2	1	0
FCNnMmSSID[23:16]							

(c) FCNnCMmMID0W

31	30	29	28	27	26	25	24
FCNnMmSSIE	0	0	FCNnMmSSID[28:24]				

23	22	21	20	19	18	17	16
FCNnMmSSID[23:16]							

15	14	13	12	11	10	9	8
FCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
FCNnMmSSID[7:0]							

FCNnMmSSIE	Format mode specification bit
0	Standard format mode (FCNnMmSSID[28:18]: 11 bits, FCNnMmSSID[17:0] are not used)
1	Extended format mode (FCNnMmSSID[28:0]: 29 bits)

FCNnMmSSID[28:0]	Message ID
FCNnMmSSID[28:18]	Standard ID value of 11 bits (when FCNnMmSSIE = 0)
FCNnMmSSID[28:0]	Extended ID value of 29 bits (when FCNnMmSSIE = 1)

Cautions

1. Be sure to write 0 to bits 14 and 13 of FCNnMmMID1H, respectively bits 30 and 29 of FCNnMmMID0W register.
2. Be sure to align the ID value according to the given bit positions into this registers. Note that for standard ID, the ID value must be shifted to fit into FCNnMmSSID[28:18] bit positions.

(5) FCNnMmCTL - FCNn message control register m

This register is used to control the operation of the message buffer.

Access This register can be read/written in 16-bit units.

Address <FCNn_base> + 0 9038_H + m x 40_H

Initial Value 0000_H

(a) FCNnMmCTL read

15	14	13	12	11	10	9	8
0	0	FCNnMm MUCF	0	0	0	FCNnMm TCPF	0
7	6	5	4	3	2	1	0
0	FCNnMm NHMF	0	FCNnMm MOWF	FCNnMm IENF	FCNnMm DTNF	FCNnMm TRQF	FCNnMm RDYF

FCNnMmNHMF	History Mask flag [a]
0	Updates of receive/transmit history list registers FCNnCMRGRX/FCNnCMRGTX are not masked.
1	Updates of receive/transmit history list registers FCNnCMRGRX/FCNnCMRGTX are masked.

a) When masked, the history lists are not updated upon reception or transmission completion activity on this message buffer.

FCNnMmMUCF[a]	Bit indicating that message buffer data is being updated
0	The CAN Controller is not updating the message buffer (reception and storage).
1	The CAN Controller is updating the message buffer (reception and storage).

a) FCNnMmMUCF is undefined until the first reception and storage is performed.

FCNnMmTCPF[a]	Transmission completely finished flag
0	Transmission failed.[b] [c]
1	Transmission (including ABT) completely finished.

a) FCNnMmTCPF is cleared if FCNnMmRDYF is changed or FCNnMmTRQF is set.
b) This indicates a successful transmission abort, if this was requested by the application by clearing the FCNnMmTRQF flag.
c) FCNnMmTCPF is not cleared, if FCNnMmTRQF is set by the ABT operation.

FCNnMmMOWF	Message buffer overwrite status bit
0	The message buffer is not overwritten by a newly received data or remote frame.
1	The message buffer is overwritten by a newly received data or remote frame.

FCNnMmIENF	Message buffer interrupt request enable bit
0	Receive message buffer: Valid message reception completion interrupt disabled. Transmit message buffer: Normal message transmission completion interrupt disabled.
1	Receive message buffer: Valid message reception completion interrupt enabled. Transmit message buffer: Normal message transmission completion interrupt enabled.

FCNnMmDTNF	Message buffer data update bit
0	No new data frame or remote frame has been stored in the message buffer.
1	A new data frame or remote frame has been stored in the message buffer.

FCNnMmTRQF	Message buffer transmission request bit
0	No message frame transmitting request that is pending or being transmitted is in the message buffer.
1	The message buffer is holding transmission of a message frame pending or is transmitting a message frame.

FCNnMmRDYF	Message buffer ready bit
0	The message buffer can be written by software. The CAN Controller cannot write to the message buffer.
1	Writing the message buffer by software is ignored (except a write access to the FCNnMmRDYF, FCNnMmTRQF, FCNnMmDTNF, and FCNnMmMOWF). The CAN Controller can write to the message buffer.

(b) FCNnMmCTL write

15	14	13	12	11	10	9	8
0	FCNnMm SENH	0	0	FCNnMm SEIE	FCNnMm SEDN	FCNnMm SETR	FCNnMm SERY
7	6	5	4	3	2	1	0
0	FCNnMm CLNH	0	FCNnMm CLMW	FCNnMm CLIE	FCNnMm CLDN	FCNnMm CLTR	FCNnMm CLRY

FCNnMmSENH	FCNnMmCLNH	Setting of FCNnMmNHMF bit
0	1	FCNnMmNHMF is cleared to 0.
1	0	FCNnMmNHMF is set to 1.
Other than above		FCNnMmNHMF is not changed.

FCNnMmCLMW	Setting of FCNnMmMOWF bit
0	FCNnMmMOWF is not changed.
1	FCNnMmMOWF is cleared to 0.

FCNnMmSEIE	FCNnMmCLIE	Setting of FCNnMmIENF bit
0	1	IE is cleared to 0.
1	0	IE is set to 1.
Other than above		IE is not changed.

FCNnMmSEDN	FCNnMmCLDN	Setting of FCNnMmDTNF bit
0	1	FCNnMmDTNF is cleared to 0.
1	0	FCNnMmDTNF is set to 1.
Other than above		FCNnMmDTNF is not changed.

Note If FCNnMmDTNF is cleared by the finish of ID field receiving, the message buffer participates in the search to store the receiving frame.

FCNnMmSETR	FCNnMmCLTR	Setting of FCNnMmTRQF bit
0	1	FCNnMmTRQF is cleared to 0.
1	0	FCNnMmTRQF is set to 1.
Other than above		FCNnMmTRQF is not changed.

FCNnMmSERY	FCNnMmCLRY	Setting of FCNnMmRDYF bit
0	1	FCNnMmRDYF is cleared to 0.
1	0	FCNnMmRDYF is set to 1.
Other than above		FCNnMmRDYF is not changed.

Cautions

1. Set FCNnMmIENF and FCNnMmRDYF always separately.
2. Do not set FCNnMmDTNF to 1 by software. Be sure to write 0 to bit 10.
3. Do not set FCNnMmTRQF and FCNnMmRDYF to 1 at the same time. Set FCNnMmRDYF = 1 before setting FCNnMmTRQF = 1.
4. Do not clear FCNnMmRDYF to “0” during message transmission. Follow the transmission abort process about clearing FCNnMmRDYF for redefinition of the message buffer.
5. Clearing of FCNnMmRDYF may take some time, depending on activity of the CAN Controller. Repeat the clearing access, until reading of FCNnMmRDYF confirms that the bit is cleared.
6. Be sure that FCNnMmRDYF is cleared before writing to the other message buffer registers, by checking the status of FCNnMmRDYF.

25.8 CAN Controller Initialization

25.8.1 Initialization of CAN Controller

Before the CAN Controller operation is enabled, the CAN channel clock T_{CANCH} needs to be determined by setting FCNnGMCSPRE.FCNnGMCSPRSC[3:0] by software.
Do not change the setting of the CAN channel clock T_{CANCH} after CAN Controller operation is enabled.

The CAN Controller is enabled by setting FCNnGMCLCTL.FCNnGMCLPWOM.

For the procedure of initializing the CAN Controller, refer to *25.16 “Operation of the CAN Controller” on page 1987* .

25.8.2 Initialization of message buffer

After the CAN Controller is enabled, the message buffers contain undefined values. A minimum initialization for all the message buffers, even for those not used in the application, is necessary before switching the CAN Controller from the initialization mode to one of the operation modes.

- Clear FCNnMmRDYF, FCNnMmTRQF, and FCNnMmDTNF of the FCNnMmCTL registers to 0.
- Clear all FCNnMmSTRB.FCNnMmSSAM to 0.

25.8.3 Redefinition of message buffer

Redefining a message buffer means changing the ID and control information of the message buffer while a message is being received or transmitted, without affecting other transmission/reception operations.

(1) To redefine message buffer in initialization mode

Place the CAN Controller in the initialization mode once and then change the ID and control information of the message buffer in the initialization mode. After changing the ID and control information, set the CAN Controller to an operation mode.

(2) To redefine message buffer during reception

Perform redefinition as shown in *Figure 25-20 “Message buffer redefinition during reception”*.

(3) To redefine message buffer during transmission

To rewrite the contents of a transmit message buffer to which a transmission request has been set, perform transmission abort processing (see and *2 “Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)” on page 1968*). Confirm that transmission has been aborted or completed, and then redefine the message buffer. After redefining the transmit message buffer, set a transmission request using the procedure described below. When setting a transmission request to a message buffer that has been redefined without aborting the transmission in progress, however, the 1-bit wait time is not necessary.

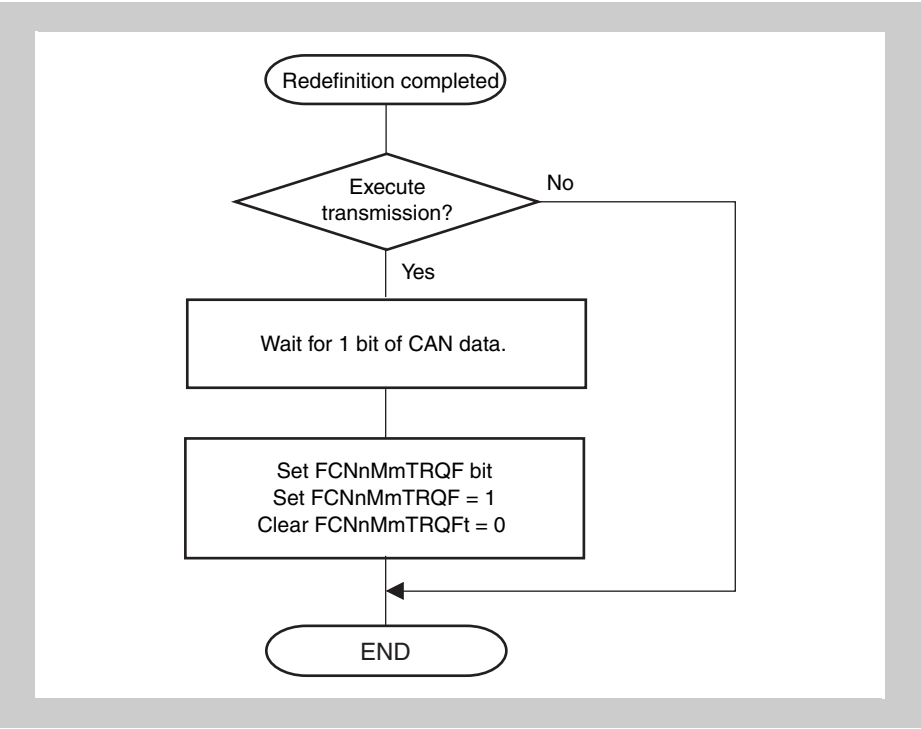


Figure 25-6 Setting transmission request (FCNnMmCTL.FCNnMmTRQF) to transmit message buffer after redefinition

Cautions

1. When a message is received, reception filtering is performed in accordance with the ID and mask set to each receive message buffer. If the procedure in *Figure 25-20 “Message buffer redefinition during reception” on page 1991* is not observed, the contents of the message buffer after it has been redefined may contradict the result of reception (result of reception filtering). If this happens, check that the ID and IDE received first and stored in the message buffer following redefinition are those stored after the message buffer has been redefined. If no ID and IDE are stored after redefinition, redefine the message buffer again.
2. When a message is transmitted, the transmission priority is checked in accordance with the ID, IDE, and FCNnMmSTRB.FCNnMmSSRT set to each transmit message buffer to which a transmission request was set. The transmit message buffer having the highest priority is selected for transmission. If the procedure in *Figure 25-6 “Setting transmission request (FCNnMmCTL.FCNnMmTRQF) to transmit message buffer after redefinition” on page 1951* is not observed, a message with an ID not having the highest priority may be transmitted after redefinition.

25.8.4 Transition from initialization mode to operation mode

The CAN Controller can be switched to the following operation modes.

- Normal operation mode
- Normal operation mode with ABT
- Receive-only mode
- Single-shot mode
- Self-test mode

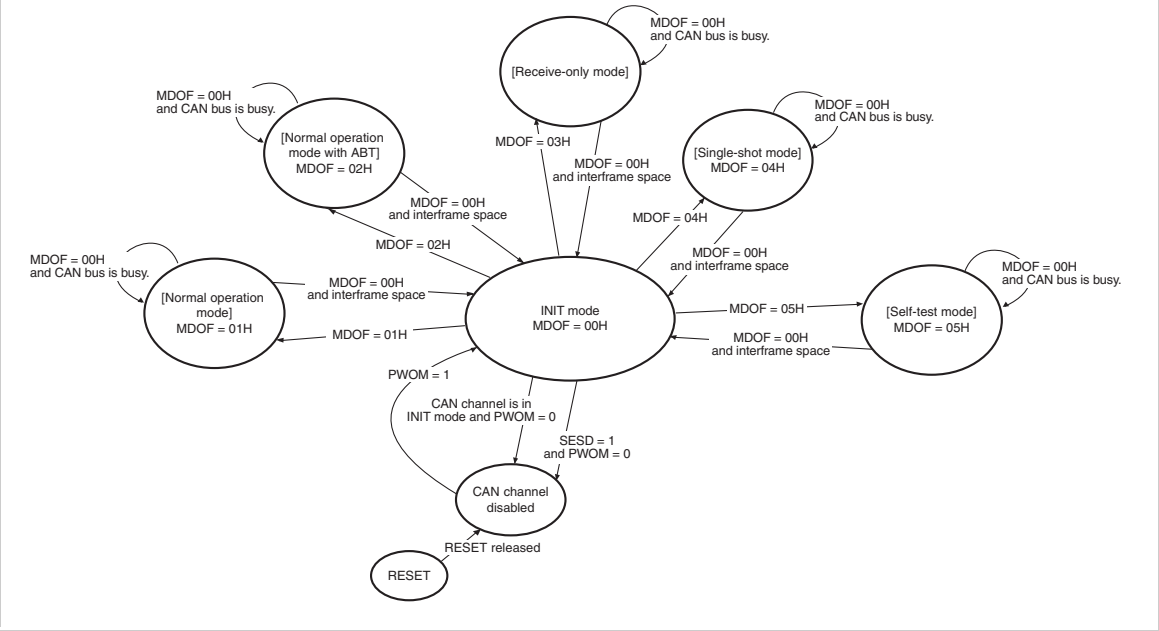


Figure 25-7 Transition to operation modes

Note In the figure above following abbreviations are used:

- MDOF = FCNnCMCLCTL.FCNnCMCLMDOF[2:0]
- PWOM = FCNnGMCLCTL.FCNnGMCLPWOM
- SESD = FCNnGMCLCTL.FCNnGMCLSESD

The transition from the initialization mode to an operation mode is controlled by the bit string FCNnCM.FCNnCMCLMDOF[2:0].

Changing from one operation mode into another requires shifting to the initialization mode in between. Do not change one operation mode to another directly; otherwise the operation will not be guaranteed.

Requests for transition from an operation mode to the initialization mode are held pending when the CAN bus is not in the interframe space (i.e., frame reception or transmission is in progress), and the CAN Controller enters the initialization mode at the first bit in the interframe space (the values of the FCNnCMCLCTL.FCNnCMCLMDOF[2:0] are changed to $000_{B)}$. After issuing a request to change the mode to the initialization mode, read FCNnCMCLCTL.FCNnCMCLMDOF[2:0] until their value becomes 000_B to confirm that the channel has entered the initialization mode (see *Figure 25-17 “Re-initialization without Software Reset function” on page 1988*).

25.9 Message Reception

25.9.1 Message reception

In all the operation modes, the complete message buffer area is analyzed to find a suitable buffer to store a newly received message. All message buffers satisfying the following conditions are included in that evaluation (RX-search process).

- Used as a message buffer
  (FCNnMmSTRB.FCNnMmSSAM = 1.)
- Set as a receive message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B to 1001_B.)
- Ready for reception
  (FCNnMmCTL.FCNnMmRDYF = 1.)

When two or more message buffers of the CAN Controller are found to be able to receive a message, the message is stored according to the priority explained below. The message is always stored in the message buffer with the highest priority, not in a message buffer with a low priority. For example, when an unmasked receive message buffer and a receive message buffer linked to mask 1 have the same ID, the received message is not stored in the message buffer linked to mask 1, even if that message buffer has not received a message and a message has already been received in the unmasked receive message buffer. In other words, when a condition has been set in two or more message buffers with different priorities, the message buffer with the highest priority always stores the message; the message is not stored in message buffers with a lower priority. This also applies when the message buffer with the highest priority is unable to store a message (i.e., when FCNnMmCTL.FCNnMmDTNF = 1 indicating that a message has already been received, but rewriting is disabled because FCNnMmSTRB.FCNnMmSSOW = 0). In this case, the message is not actually stored in the candidate message buffer with the highest priority, but neither is it stored in a message buffer with a lower priority.

Table 25-20 MBRB priorities

Priority	Storing condition if same ID is set	
1 (high)	Unmasked message buffer	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
2	Message buffer linked to mask 1	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
3	Message buffer linked to mask 2	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOW = 1
...	...	
9 (low)	Message buffer linked to mask 8	FCNnMmDTNF = 0
		FCNnMmDTNF = 1 and FCNnMmSSOWt = 1

25.9.2 Receive data read

To keep data consistency when reading CAN Controller message buffers, perform the data reading according to *Figure 25-34 “Reception via interrupt (using FCNnCMLISTR register)” on page 2007* to *Figure 25-37 “Reception via software polling” on page 2011* .

During message reception, the CAN Controller sets FCNnMmCTL.FCNnMmDTNF two times: at the beginning of the storage process of data to the message buffer, and again at the end of this storage process. During this storage process, FCNnMmCTL.FCNnMmMUCF of the message buffer is set (refer to *Figure 25-8 “FCNnMmCTL.FCNnMmDTNF and FCNnMmCTL.FCNnMmMUCF bit setting period (for standard ID format)”*).

The receive history list is also updated just before the storage process. In addition, during storage process (FCNnMmCTL.FCNnMmMUCF = 1), FCNnMmCTL.FCNnMmRDYF of the message buffer is locked to avoid any coincidental data write by CPU. Note that the storage process may be disturbed (delayed) when the CPU accesses the message buffer.

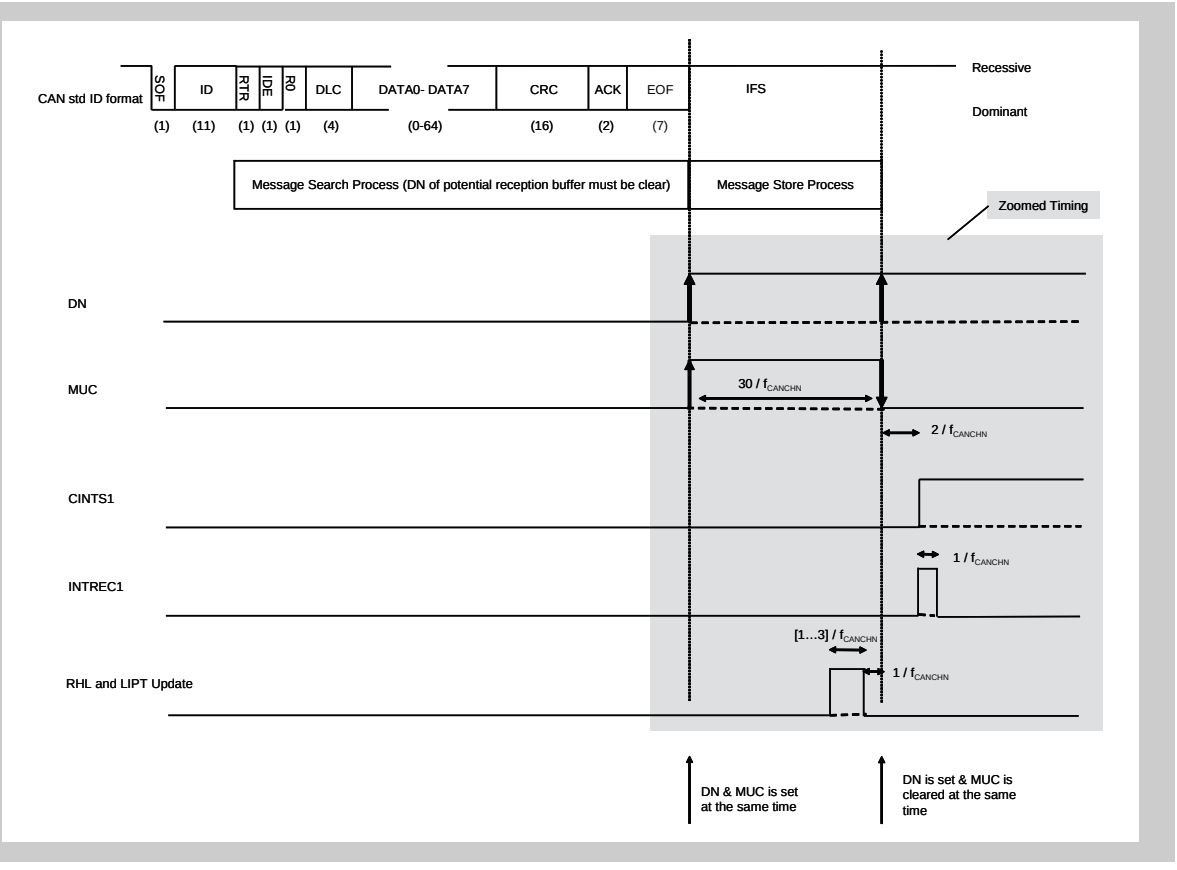


Figure 25-8 FCNnMmCTL.FCNnMmDTNF and FCNnMmCTL.FCNnMmMUCF bit setting period (for standard ID format)

Note If a message shall be stored in a message buffer, the FCNnMmDTNF bit of this buffer must be cleared before the Message Search Process is started, i.e., right after the ID of the frame is on the bus. In worst case, this happens 15 CAN bits after EOF of the previous frame. Consider to use more than one Message Buffer for reception of a frame, if CAN frames are appearing back-to-back on the bus and none shall be lost.

25.9.3 Receive history list function

The receive history list (RHL) function records in the receive history list the number of the receive message buffer in which each data frame or remote frame was received and stored. The RHL consists of storage elements equivalent to up to 23 messages (on 32 message buffer CAN Controller), 47 messages (on 64 message buffer CAN Controller) or up to 95 messages (on 128 message buffer CAN Controller), the last in-message pointer FCNnCMLISSLR[7:0] with the corresponding FCNnCMLISTR register and the receive history list get pointer FCNnCMRGSSPT with the corresponding FCNnCMRGRX register.

The RHL is undefined immediately after the transition of the CAN Controller from the initialization mode to one of the operation modes.

The FCNnCMLISTR register holds the contents of the RHL element indicated by the value of the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer minus 1. By reading the FCNnCMLISTR register, therefore, the number of the message buffer that received and stored a data frame or remote frame first can be checked. The FCNnCMLISSLR[7:0] pointer is utilized as a write pointer that indicates to what part of the RHL a message buffer number is recorded. Any time a data frame or remote frame is received and stored, the corresponding message buffer number is recorded to the RHL element indicated by the FCNnCMLISSLR[7:0] pointer. Each time recording to the RHL has been completed, the FCNnCMLISSLR[7:0] pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

For message buffers, where the flag FCNnMmCTL.FCNnMmNHMF is set, no entry in the history lists is recorded.

The FCNnCMRGRX.FCNnCMRGSSPT pointer is utilized as a read pointer that reads a recorded message buffer number from the RHL. This pointer indicates the first RHL element that the CPU has not read yet. By reading the FCNnCMRGRX register by software, the number of a message buffer that has received and stored a data frame or remote frame can be read. Each time a message buffer number is read from the FCNnCMRGRX register, the FCNnCMRGSSPT pointer is automatically incremented.

If the value of the FCNnCMRGRX.FCNnCMRGSSPT pointer matches the value of the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer, FCNnCMRGRX.FCNnCMRGSSPM (receive history list pointer match) is set to 1. This indicates that no message buffer number that has not been read remains in the RHL. If a new message buffer number is recorded, the FCNnCMLISSLR[7:0] pointer is incremented and because its value no longer matches the value of the FCNnCMRGSSPT pointer, FCNnCMRGSSPM is cleared. In other words, the numbers of the unread message buffers exist in the RHL.

If the FCNnCMLISTR.FCNnCMLISSLR[7:0] pointer is incremented and matches the value of the FCNnCMRGRX.FCNnCMRGSSPT pointer minus 1, FCNnCMRGRX.FCNnCMRGRVFF (receive history list overflow) is set to 1. This indicates that the RHL is full of numbers of message buffers that have not been read. When further message reception and storing occur, the last recorded message buffer number is overwritten by the number of the message buffer that received and stored the newly received message. In this case, after FCNnCMRGRVFF has been set (1), the recorded message buffer numbers in the RHL do not completely reflect the chronological order. However messages itself are not lost and can be located by CPU search in message buffer memory with the help of FCNnMmCTL.FCNnMmDTNF, or by reading the global registers FCNnDNBMRX[3:0].

Caution If the history list is in the overflow condition (FCNnCMRGRX.FCNnCMRGRVFF is set), reading the history list contents is still possible, until the history list is empty (indicated by FCNnCMRGRX.FCNnCMRGSSPM flag set). Nevertheless, the history list remains in the overflow condition, until FCNnCMRGRVFF is cleared by software. If FCNnCMRGRVFF is not cleared, the FCNnCMRGSSPM flag will also not be updated (cleared) upon a message storage of newly received frame. This may lead to the situation, that FCNnCMRGSSPM indicates an empty history list, although a reception has taken place, while the history list is in the overflow state (FCNnCMRGRVFF and FCNnCMRGSSPM are set).

As long as the RHL still has free entries, the sequence of occurrence is maintained. If more receptions occur without reading the RHL by the host processor, complete sequence of receptions can not be recovered.

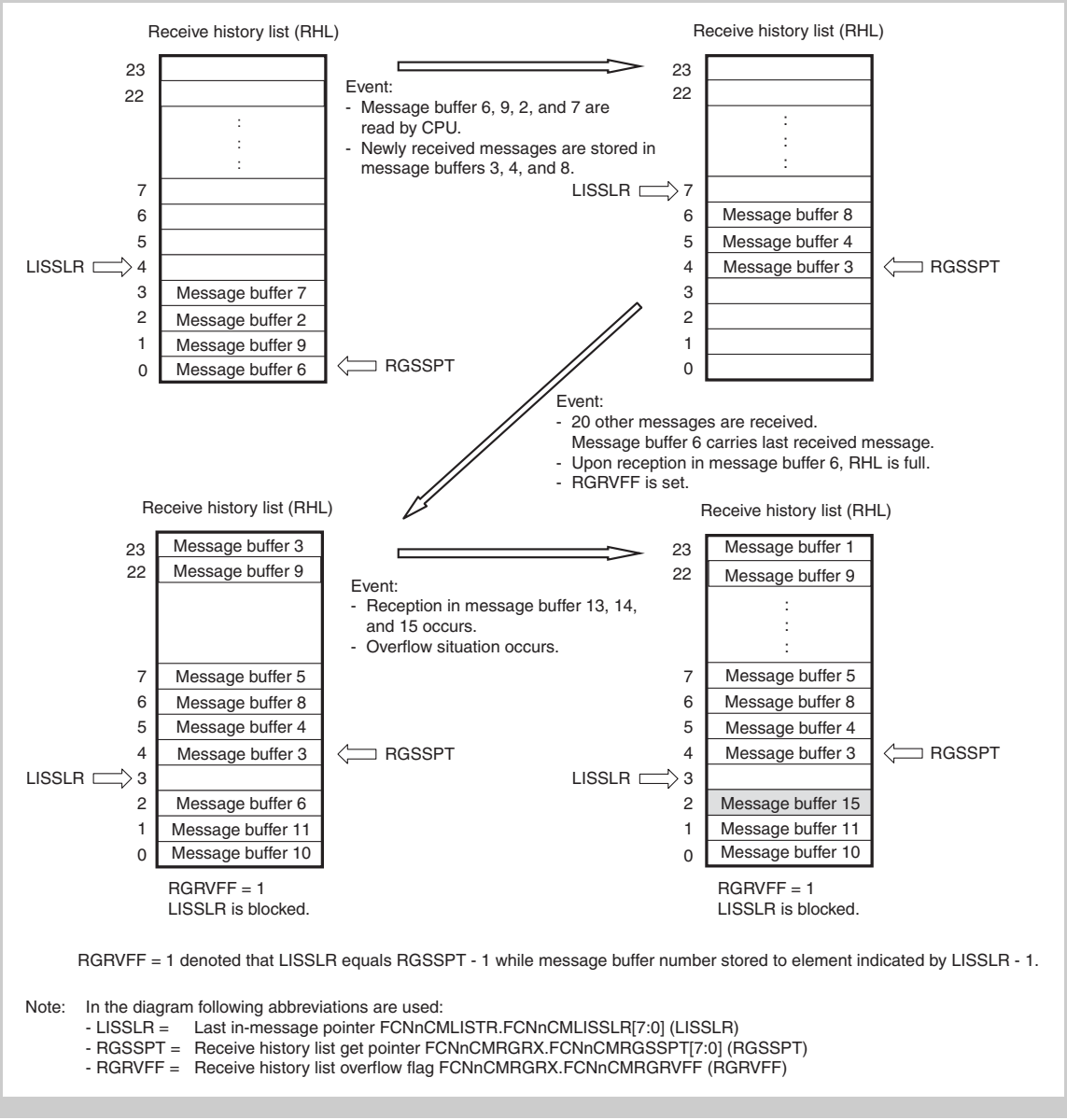


Figure 25-9 Receive history list

25.9.4 Mask function

For any message buffer, which is used for reception, the assignment to one of eight global reception masks (or no mask) can be selected.

By using the mask function, the message ID comparison can be reduced by masked bits, herewith allowing the reception of several different IDs into one buffer.

While the mask function is in effect, an identifier bit that is defined to be 1 by a mask in the received message is not compared with the corresponding identifier bit in the message buffer.

However, this comparison is performed for any bit whose value is defined as 0 by the mask.

For example, let us assume that all messages that have a standard-format ID, in which bits ID27 to ID25 are 0 and bits ID24 and ID22 are 1, are to be stored in message buffer 14. The procedure for this example is shown below.

(1) Identifier to be stored in message buffer

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

(2) Identifier to be configured in message buffer 14 (example) (using FCNnCM14MID0W register)

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
x	0	0	0	1	x	1	x	x	x	x
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
x	x	x	x	x	x	x	x	x	x	x
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
x	x	x	x	x	x	x				

Notes

1. ID with the ID27 to ID25 bits cleared to 0 and the ID24 and ID22 bits set to 1 is registered (initialized) to message buffer 14. Other bits of ID can be set to any value (x), because they are going to be masked.
2. Message buffer 14 is set as a standard format identifier that is linked to mask 1 (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0010_B).

Mask setting for CAN Controller 1 (mask 1) (example)
(using CAN1 address mask 1 register FCNnCMMKCTL01W)

FNCnCMMKSSID[..]

ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21	ID20	ID19	ID18
1	0	0	0	0	1	0	1	1	1	1
ID17	ID16	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
1	1	1	1	1	1	1	1	1	1	1
ID6	ID5	ID4	ID3	ID2	ID1	ID0				
1	1	1	1	1	1	1				

1: Not compared (masked)
0: Compared

FCNnCMMKSSID[27:24] and FCNnCMMKSSID[22] are cleared to 0, and FCNnCMMKSSID[28], FCNnCMMKSSID[23], and FCNnCMMKSSID[21:0] are set to 1.

25.9.5 Multi buffer receive block function

The multi buffer receive block (MBRB) function is used to store a block of data in two or more message buffers sequentially with no CPU interaction, by setting the same ID to two or more message buffers with the same message buffer type. These message buffers can be allocated anywhere in the message buffer memory, they do not even have to follow each other adjacently.

Suppose, for example, the same message buffer type is set to 10 message buffers, message buffers 10 to 19, and the same ID is set to each message buffer. If the first message whose ID matches an ID of the message buffers is received, it is stored in message buffer 10. At this point, FCNnMmCTL.FCNnMmDTNF of message buffer 10 is set, prohibiting overwriting the message buffer when subsequent messages are received.

When the next message with a matching ID is received, it is received and stored in message buffer 11. Each time a message with a matching ID is received, it is sequentially (in the ascending order) stored in message buffers 12, 13, and so on. Even when a data block consisting of multiple messages is received, the messages can be stored and received without overwriting the previously received matching-ID data.

Whether a data block has been received and stored can be checked by setting FCNnMmCTL.FCNnMmIENF of each message buffer. For example, if a data block consists of k messages, k message buffers are initialized for reception of the data block. FCNnMmIENF in message buffers 0 to (k-2) is cleared to 0 (interrupts disabled), and FCNnMmIENF in message buffer k-1 is set to 1 (interrupts enabled). In this case, a reception completion interrupt occurs when a message has been received and stored in message buffer k-1, indicating that MBRB has become full. Alternatively, by clearing FCNnMmIENF of message buffers 0 to (k-3) and setting FCNnMmIENF of message buffer k-2, a warning that MBRB is about to overflow can be issued.

The basic conditions of storing receive data in each message buffer for the MBRB are the same as the conditions of storing data in a single message buffer.

Cautions

1. MBRB can be configured for each of the same message buffer types. Therefore, even if a message buffer of another MBRB whose ID matches but whose message buffer type is different has a vacancy, the received message is not stored in that message buffer, but instead discarded.
2. MBRB does not have a ring buffer structure. Therefore, after a message is stored in the message buffer having the highest number in the MBRB configuration, a newly received message will not be stored in the message buffer having the lowest message buffer number.
3. MBRB operates based on the reception and storage conditions; there are no settings dedicated to MBRB, such as function enable bits. By setting the same message buffer type and ID to two or more message buffers, MBRB is automatically configured.
4. With MBRB, “matching ID” means “matching ID after mask”. Even if the ID set to each message buffer is not the same, if the ID that is masked by the mask register matches, it is considered a matching ID and the buffer that has this ID is treated as the storage destination of a message.
5. The priority between MBRBs is mentioned in the table *Table 25-20 “MBRB priorities”*.

25.9.6 Remote frame reception

In all the operation modes, when a remote frame is received, the message buffer that is to store the remote frame is searched from all the message buffers satisfying the following conditions (1 and 2, condition 1 has priority on reception acceptance). If condition 1 is not fulfilled, the remaining message buffers are scanned, whether condition 2 could be fulfilled.

- Condition 1:
  Set as a transmit message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B)
  - Used as a message buffer
    (FCNnMmSTRB.FCNnMmSSAM = 1.)
  - Ready for reception
    (FCNnMmCTL.FCNnMmRDYF = 1.)
  - Set to data frame message type
    (FCNnMmSTRB.FCNnMmSSRT = 0.)
  - Transmission request is not set.
    (FCNnMmCTL.FCNnMmTRQF = 0.)
- Condition 2:
  Set as a receive message buffer
  (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B ... 1001_B)
  - Used as a message buffer
    (FCNnMmSTRB.FCNnMmSSAM = 1.)
  - Ready for reception
    (FCNnMmCTL.FCNnMmRDYF = 1.)
  - Set to remote frame message type
    (FCNnMmSTRB.FCNnMmSSRT = 1.)
  - Buffer is ready to store a message
    (FCNnMmCTL.FCNnMmDTNF = 0, or FCNnMmSTRB.FCNnMmSSOW = 1 with FCNnMmCTL.FCNnMmDTNF = 1).

Upon acceptance of a remote frame, the following actions are executed if the ID of the received remote frame matches the ID of a message buffer that satisfies the above conditions.

- The FCNnMmDTLG[3:0] bit string in the FCNnMmDTLGB register store the received DLC value.
- The FCNnMmDAT0B to FCNnMmDAT7B registers in the data area are not updated (data before reception is saved).
- FCNnMmCTL.FCNnMmDTNF is set to 1.
- FCNnCMISCTL.FCNnCMISITSF1 is set to 1 (if FCNnMmCTL.FCNnMmIENF of the message buffer that receives and stores the frame is set to 1).
- The receive completion interrupt (INTCnREC) is output (if FCNnMmCTL. FCNnMmIENF of the message buffer that receives and stores the frame is set to 1 and if FCNnCMIECTL.FCNnCMIESEIE1 is set to 1).
- The message buffer number is recorded in the receive history list, if the flag FCNnMmCTL.FCNnMmNHMF is not set.

Caution When a transmit message buffer is found for receiving and storing a remote frame, overwrite control by FCNnMmSTRB.FCNnMmSSOW of the message buffer and FCNnMmCTL.FCNnMmDTNF are not checked. The setting of FCNnMmSSOW is ignored, and FCNnMmDTNF is set in any case.

Notes

1. If more than one transmit message buffer has the same ID and the ID of the received remote frame matches that ID, the remote frame is stored in the transmit message buffer with the lowest message buffer number.
2. If transmit and receive message buffers are found, which could receive a remote frame matching with its ID, either masked or unmasked, the remote frame is stored in the transmit message buffer.
3. If several receive message buffers would match for reception for a remote frame, the reception priority is identical as for a data frame.
4. If a receive message buffer is found to match for a remote frame reception, and selected for storage, but this receive message buffer does not allow the storage, because FCNnMmDTNF is set, and FCNnMmSSOW is not set, the remote frame is not stored at all.

25.10 Message Transmission

25.10.1 Message transmission

A message buffer with its FCNnMmCTL.FCNnMmTRQF bit set to 1 participates in the search for the most high-prioritized message when the following conditions are fulfilled. This behavior is valid for all operational modes.

- Used as a message buffer (FCNnMmSTRB.FCNnMmSSAM = 1.)
- Set as a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B.)
- Ready for transmission (FCNnMmCTL.FCNnMmRDYF = 1.)

The CAN bus is a multi-master communication system. In a system like this, the priority of message transmission is determined based on message identifiers (IDs). To facilitate transmission processing by software when there are several messages awaiting transmission, the CAN Controller uses hardware to check the ID of the message with the highest priority and automatically identifies that message. This eliminates the need for software-based priority control.

Transmission priority is controlled by the identifier (ID).

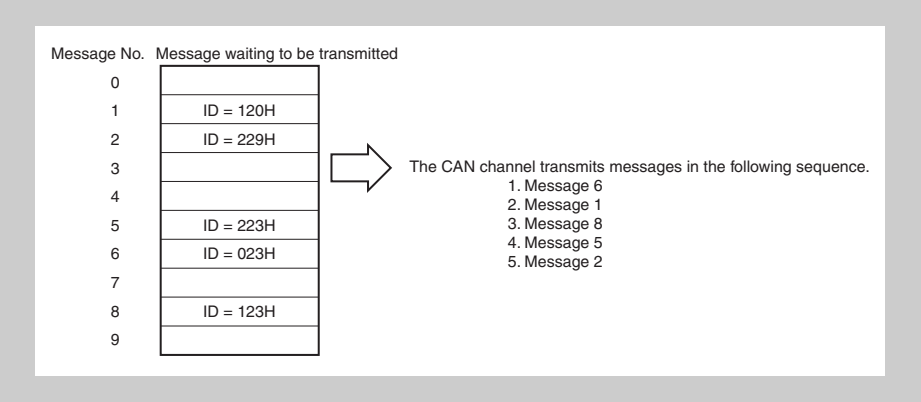


Figure 25-10 Message processing example

After the transmit message search, the transmit message with the highest priority of the transmit message buffers that have a pending transmission request (message buffers with the FCNnMmCTL.FCNnMmTRQF bit set to 1 in advance) is transmitted.

If a new transmission request is set, the transmit message buffer with the new transmission request is compared with the transmit message buffer with a pending transmission request. If the new transmission request has a higher priority, it is transmitted, unless transmission of a message with a low priority has already started. If transmission of a message with a low priority has already started, however, the new transmission request is transmitted later. To solve this priority inversion effect, the software can perform a transmission abort request for the lower priority message. The highest priority is determined according to the following rules.

Priority	Conditions	Description
1 (high)	Value of first 11 bits of ID [ID28 to ID18]:	The message frame with the lowest value represented by the first 11 bits of the ID is transmitted first. If the value of an 11-bit standard ID is equal to or smaller than the first 11 bits of a 29-bit extended ID, the 11-bit standard ID has a higher priority than a message frame with a 29-bit extended ID.
2	Frame type	A data frame with an 11-bit standard ID (FCNnMmSTRB.FCNnMmSSRT cleared to 0) has a higher priority than a remote frame with a standard ID and a message frame with an extended ID.
3	ID type	A message frame with a standard ID (message buffer identifier register FCNnMmMID... bit FCNnMmSSIE is cleared to 0) has a higher priority than a message frame with an extended ID.
4	Value of lower 18 bits of ID [ID17 to ID0]:	If one or more transmission-pending extended ID message frame has equal values in the first 11 bits of the ID and the same frame type (equal FCNnMmSTRB.FCNnMmSSRT bit values), the message frame with the lowest value in the lower 18 bits of its extended ID is transmitted first.
5 (low)	Message buffer number	If two or more message buffers request transmission of message frames with the same ID, the message from the message buffer with the lowest message buffer number is transmitted first.

Notes 1. If the automatic block transmission request bit FCNnGMABCTL.FCNnGMABABTT is set to 1 in the normal operation mode with ABT, FCNnMmCTL.FCNnMmTRQF is set to 1 only for one message buffer in the ABT message buffer group.

If the ABT mode was triggered by FCNnGMABCTL.FCNnGMABSEAT = 1, one FCNnMmCTL.FCNnMmTRQF is set to 1 in the ABT area (buffer 0 through 7). Beyond this transmit request, the application can request transmissions (set FCNnMmTRQF to 1) for other TX-message buffers that do not belong to the ABT area. In that case an interval arbitration process (TX-search) evaluates all TX-message buffers with FCNnMmTRQF set to 1 and chooses the message buffer that contains the highest prioritized identifier for the next transmission. If there are 2 or more identifiers that have the highest priority (i.e. identical identifiers), the message located at the lowest message buffer number is transmitted at first.

Upon successful transmission of a message frame, the following operations are performed.

- The FCNnMmCTL.FCNnMmTRQF flag of the corresponding transmit message buffer is automatically cleared to 0.
- The transmission completion status bit FCNnCMISCTL.FCNnCMISITSF0 is set to 1 (if the interrupt enable bit FCNnMmIENF of the corresponding transmit message buffer is set to 1).
- An interrupt request signal INTCnTRX is output (if FCNnCMIECTL.FCNnCMIESEIE0 is set to 1 and if the interrupt enable bit FCNnMmIENF of the corresponding transmit message buffer is set to 1).

2. When changing the contents of a transmit buffer, the FCNnMmCTL.FCNnMmRDYF flag of this buffer must be cleared before updating the buffer contents. As during internal transfer actions, the FCNnMmRDYF flag may be locked temporarily, the status of FCNnMmRDYF must be checked by software, after changing it.

25.10.2 Transmit history list function

The transmit history list (THL) function records in the transmit history list the number of the transmit message buffer from which data or remote frames have been were sent. The THL consists of storage elements equivalent to up to 7 messages (on 32 message buffer CAN Controller), 15 messages (on 64 message buffer CAN Controller) or up to 31 messages (on 128 message buffer CAN Controller), the last out-message pointer FCNnCMLOSTR[7:0] with the corresponding FCNnCMLOSTR register, and the transmit history list get pointer FCNnCMTGSSPT[7:0] with the corresponding FCNnCMTGTX register.

The THL is undefined immediately after the transition of the CAN Controller from the initialization mode to one of the operation modes.

The FCNnCMLOSTR register holds the contents of the THL element indicated by the value of the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer minus 1. By reading the FCNnCMLOSTR register, therefore, the number of the message buffer that transmitted a data frame or remote frame first can be checked. The FCNnCMLOSTR[7:0] pointer is utilized as a write pointer that indicates to what part of the THL a message buffer number is recorded. Any time a data frame or remote frame is transmitted, the corresponding message buffer number is recorded to the THL element indicated by the FCNnCMLOSTR[7:0] pointer. Each time recording to the THL has been completed, the FCNnCMLOSTR[7:0] pointer is automatically incremented. In this way, the number of the message buffer that has received and stored a frame will be recorded chronologically.

For message buffers, where the flag FCNnMmCTL.FCNnMmNHMF is set, no entry in the history lists is recorded.

The FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer is utilized as a read pointer that reads a recorded message buffer number from the THL. This pointer indicates the first THL element that the CPU has not yet read. By reading the FCNnCMTGTX register by software, the number of a message buffer that has completed transmission can be read. Each time a message buffer number is read from the FCNnCMTGTX register, the FCNnCMTGSSPT[7:0] pointer is automatically incremented.

If the value of the FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer matches the value of the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer, FCNnCMTGTX.FCNnCMTGSSPM (transmit history list pointer match) is set to 1. This indicates that no message buffer numbers that have not been read remain in the THL. If a new message buffer number is recorded, the FCNnCMLOSTR[7:0] pointer is incremented and because its value no longer matches the value of the FCNnCMTGSSPT[7:0] pointer, FCNnCMTGSSPM is cleared. In other words, the numbers of the unread message buffers exist in the THL.

If the FCNnCMLOSTR.FCNnCMLOSTR[7:0] pointer is incremented and matches the value of the FCNnCMTGTX.FCNnCMTGSSPT[7:0] pointer minus 1, FCNnCMTGTX.FCNnCMTGTVFF (transmit history list overflow) is set to 1. This indicates that the THL is full of message buffer numbers that have not been read. If a new message is received and stored, the message buffer number recorded last is overwritten by the message buffer number that transmitted its message afterwards. In this case, after FCNnCMTGTVFF has been set (1), therefore, the recorded message buffer numbers in the THL do not completely reflect the chronological order. However the other transmitted messages can be found by a CPU search applied to all transmit message buffers unless the CPU has not overwritten a transmit object in one of these buffers beforehand. In total up to six transmission completions can occur without overflowing the THL.

Caution If the history list is in the overflow condition (FCNnCMTGTX.FCNnCMTGTVFF is set), reading the history list contents is still possible, until the history list is empty (indicated by FCNnCMTGTX.FCNnCMTGSSPM flag set). Nevertheless, the history list remains in the overflow condition, until FCNnCMTGTVFF is cleared by software. If FCNnCMTGTVFF is not cleared, the FCNnCMTGTX.FCNnCMTGSSPM flag will also not be updated (cleared) upon successful transmission of a new message. This may lead to the situation, that FCNnCMTGSSPM indicates an empty history list, although a successful transmission has taken place, while the history list is in the overflow state (FCNnCMTGTVFF and FCNnCMTGSSPM are set).

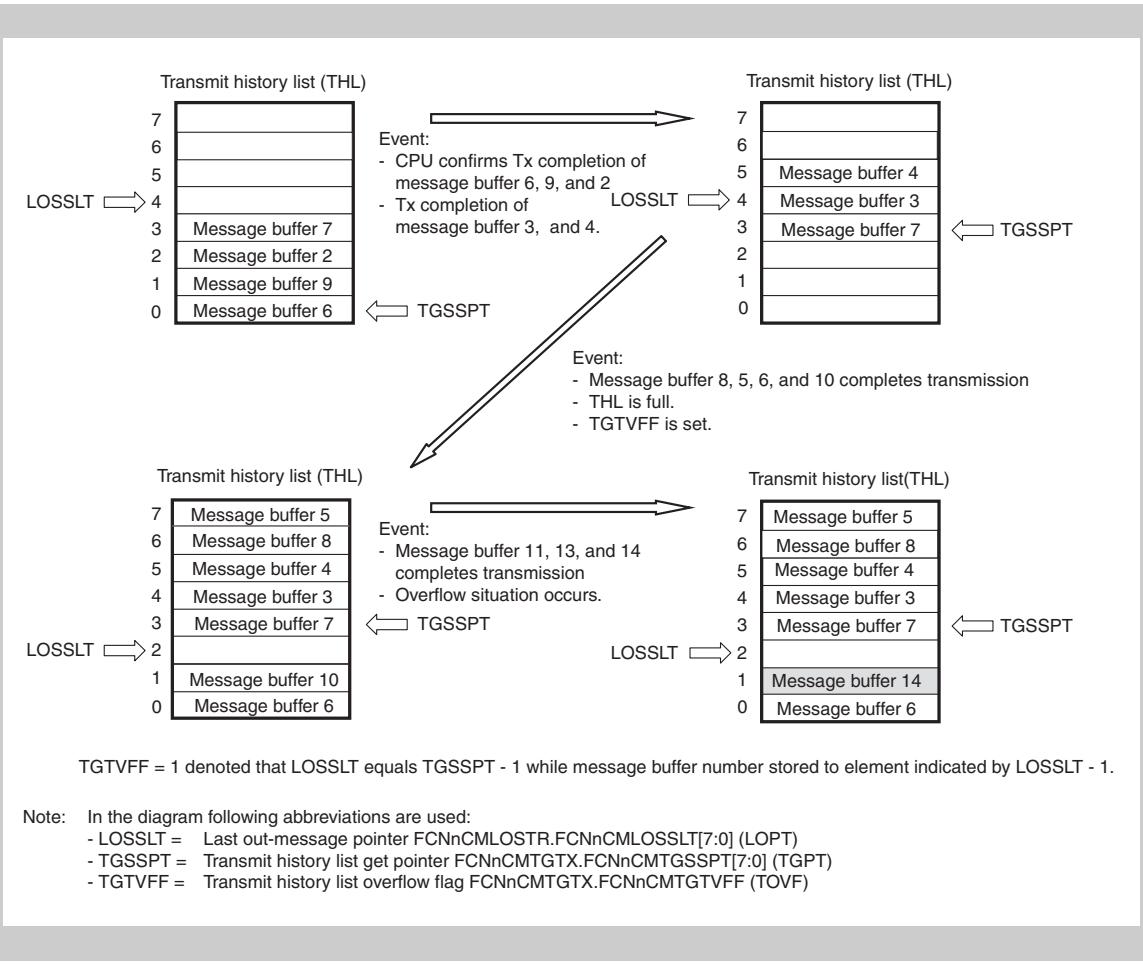


Figure 25-11 Transmit history list

25.10.3 Automatic block transmission (ABT)

The automatic block transmission (ABT) function is used to transmit two or more data frames successively with no CPU interaction. The maximum number of transmit message buffers assigned to the ABT function is 8 (for 32 message buffer CAN Controller), 16 (for 64 message buffer CAN Controller) or 32 (for 128 message buffer CAN Controller), always located in the lowest message buffers.

By setting FCNnCM.FCNnCMCLMDOF[2:0] to 010_B, “normal operation mode with automatic block transmission function” (hereafter referred to as ABT mode) can be selected.

To issue an ABT transmission request, define the message buffers by software first. Set FCNnMmSTRB.FCNnMmSSAM = 1 in all the message buffers used for ABT, and define all the buffers as transmit message buffers by setting the FCNnMmSTRB.FCNnMmSSMT[3:0] bits to 0000_B. Be sure to set the same ID for the message buffers for ABT even when that ID is being used for all the message buffers. To use two or more IDs, set the ID of each message buffer by using the FCNnMmMID0H and FCNnMmMID1H or FCNnMmMID0W registers. Set the CAN Controller message buffer data bytes registers before issuing a transmission request for the ABT function.

After initialization of message buffers for ABT is finished, FCNnMmCTL.FCNnMmRDYF needs to be set to 1. In the ABT mode, FCNnMmCTL.FCNnMmTRQF does not have to be manipulated by software.

After the data for the ABT message buffers has been prepared, set FCNnGMABCTL.FCNnGMABSEAT = 1. Automatic block transmission is then started. When ABT is started, FCNnMmCTL.FCNnMmTRQF in the first message buffer (message buffer 0) is automatically set to 1. After transmission of the data of message buffer 0 is finished, the FCNnMmTRQF of the next message buffer, message buffer 1, is set automatically. In this way, transmission is executed successively.

A delay time can be inserted by program in the interval in which the transmission request FCNnMmCTL.FCNnMmTRQF is automatically set while successive transmission is being executed. The delay time to be inserted is defined by the FCNnGMADCTL register. The unit of the delay time is DBT (data bit time). DBT depends on the setting of the FCNnCMBRPRS and FCNnCMBTCTL registers.

Among transmit objects within the ABT-area, the priority of the transmission ID is not evaluated. The data of message buffers 0 to 7 are sequentially transmitted. When transmission of the data frame from message buffer 7 has been completed, FCNnGMABCTL.FCNnGMABABTT is automatically cleared to 0 and the ABT operation is finished.

If FCNnMmCTL.FCNnMmRDYF of an ABT message buffer is cleared during ABT, no data frame is transmitted from that buffer, ABT is stopped, and FCNnGMABCTL.FCNnGMABABTT is cleared. After that, transmission can be resumed from the message buffer where ABT stopped, by setting FCNnMmRDYF and FCNnGMABABTT to 1 by software. To not resume transmission from the message buffer where ABT stopped, the internal ABT engine can be reset by setting the FCNnGMABCTL.FCNnGMABCLRF bit to 1 while ABT mode is stopped and FCNnGMABABTT is cleared to 0. In this case, transmission is started from message buffer 0 if FCNnGMABCTL.FCNnGMABSEAC is cleared to 0 and then FFCNnGMABABTT is set to 1.

An interrupt can be used to check if data frames have been transmitted from all the message buffers for ABT. To do so, FCNnMmCTL.FCNnMmIENF of each message buffer except the last message buffer needs to be cleared (0).

If a transmit message buffer other than those used by the ABT function is assigned to a transmit message buffer, the message to be transmitted next is determined by the priority of the transmission ID of the ABT message buffer whose transmission is currently held pending and the transmission ID of the message buffers other than those used by the ABT function.

Transmission of a data frame from an ABT message buffer is not recorded in the transmit history list (THL).

Cautions

1. Set FCNnGMABCTL.FCNnGMABSEAC = 1 while FCNnGMABCTL.FCNnGMABABTT is cleared to 0 in order to resume ABT operation at buffer No. 0. If FCNnGMABSEAC is set to 1 while FCNnGMABABTT is set to 1, the subsequent operation is not guaranteed.
2. If the automatic block transmission engine is cleared by setting FCNnGMABCTL.FCNnGMABSEAC = 1, FCNnGMABSEAC is automatically cleared immediately after the processing of the clearing request is completed.
3. Do not trigger automatic block transmission in the initialization mode. If FCNnGMABCTL.FCNnGMABSEAT is set in the initialization mode, the proper operation is not guaranteed after the mode is changed from the initialization mode to the ABT mode.
4. Do not set FCNnMmCTL.FCNnMmTRQF of the ABT message buffers to 1 by software in the normal operation mode with ABT. Otherwise, the operation is not guaranteed.
5. The FCNnGMADCTL register is used to set the delay time that is inserted in the period from completion of the preceding ABT message to setting of FCNnMmCTL.FCNnMmTRQF for the next ABT message when the transmission requests are set in the order of message numbers for each message for ABT that is successively transmitted in the ABT mode. The timing at which the messages are actually transmitted onto the CAN bus varies depending on the status of transmission from other stations and the status of the setting of the transmission request for messages other than the ABT messages.
6. If a transmission request is made for a message other than an ABT message and if no delay time is inserted in the interval in which transmission requests for ABT are automatically set (FCNnGMADCTL = 00_H), messages other than ABT messages may be transmitted not depending on their priority compared to the priority of the ABT message.
7. Do not clear FCNnMmCTL.FCNnMmRDYF to 0 when FCNnGMABCTL.FCNnGMABABTT = 1.
8. If a message is received from another node while normal operation mode with ABT is active, the TX-message from the ABT-area may be transmitted with delay of one frame although FCNnGMADCTL register was set up with 00_H.

25.10.4 Transmission abort process

(1) Transmission abort process except for in normal operation mode with automatic block transmission (ABT)

The user can clear FCNnMmCTL.FCNnMmTRQF to 0 to abort a transmission request. FCNnMmTRQF will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using FCNnCMCLCTL.FCNnCMCLSSTS and the FCNnCMTGTX register, or the FCNnMmCTL.FCNnMmTCPF flag, which indicate the transmission status on the CAN bus (for details, refer to the processing in *Figure 25-27 “Transmission abort processing (except normal operation mode with ABT)” on page 1999*).

(2) Transmission abort process except for ABT transmission in normal operation mode with automatic block transmission (ABT)

The user can clear FCNnGMABCTL.FCNnGMABABTT to 0 to abort a transmission request. After checking FCNnGMABABTT = 0, clear FCNnMmCTL.FCNnMmTRQF to 0. FCNnMmTRQF will be cleared immediately if the abort was successful. Whether the transmission was successfully aborted or not can be checked using FCNnCMCLCTL.FCNnCMCLSSTS and the FCNnCMTGTX register, or the FCNnMmCTL.FCNnMmTCPF flag, which indicate the transmission status on the CAN bus (for details, refer to the processing in *Figure 25-28 “Transmission abort processing (normal operation mode with ABT) - Repeat option for aborted message” on page 2000*).

(3) Transmission abort process for ABT transmission in normal operation mode with automatic block transmission (ABT)

To abort ABT that is already started, clear FCNnGMABCTL.FCNnGMABCLAT to 0. In this case, FCNnGMABCTL.FCNnGMABABTT remains 1 if an ABT message is currently being transmitted and until the transmission is completed (successfully or not), and is cleared to 0 as soon as transmission is finished. This aborts ABT.

If the last transmission (before ABT) was successful, the normal operation mode with ABT is left with the internal ABT pointer pointing to the next message buffer to be transmitted.

In the case of an erroneous transmission, the position of the internal ABT pointer depends on the status of FCNnMmCTL.FCNnMmTRQF in the last transmitted message buffer. If FCNnMmTRQF is set to 1 when clearing FCNnGMABCTL.FCNnGMABABTT is requested, the internal ABT pointer points to the last transmitted message buffer (for details, refer to the process in *Figure 25-29 “Transmission abort processing (normal operation mode with ABT) - No repetition option for aborted message” on page 2001*). If FCNnMmTRQF is cleared to 0 when clearing FCNnGMABABTT is requested, the internal ABT pointer is incremented (+1) and points to the next message buffer in the ABT area (for details, refer to the process in *Figure 25-30 “ABT transmission request abort processing (normal operation mode with ABT)” on page 2002*).

Caution Be sure to abort ABT by clearing FCNnGMABCTL.FCNnGMABCLAT to 0. The operation is not guaranteed if aborting transmission is requested by clearing FCNnMmCTL.FCNnMmRDYF.

When the normal operation mode with ABT is resumed after ABT has been aborted and FCNnGMABCTL.FCNnGMABSEAT is set to 1, the next ABT message buffer to be transmitted can be determined from the following table.

Status of FCNnMmCTL. FCNnMmTRQF of ABT message buffer	Abort after successful transmission	Abort after erroneous transmission
Set (1)	Next message buffer in the ABT area[a]	Same message buffer in the ABT area
Cleared (0)	Next message buffer in the ABT area[a]	Next message buffer in the ABT area[a]

a) The above resumption operation can be performed only if a message buffer ready for ABT exists in the ABT area. For example, an abort request that is issued while ABT of highest ABT message buffer is in progress is regarded as completion of ABT, rather than abort, if transmission of this message buffer has been successfully completed, even if FCNnGMABCTL.FCNnGMABABTT is cleared to 0. If FCNnMmCTL.FCNnMmRDYF in the next message buffer in the ABT area is cleared to 0, the internal ABT pointer is retained, but the resumption operation is not performed even if FCNnGMABABTT is set to 1, and ABT ends immediately.

25.10.5 Remote frame transmission

Remote frames can be transmitted only from transmit message buffers. Set whether a data frame or remote frame is transmitted via FCNnMmSTRB.FCNnMmSSRT. Setting FCNnMmSSRT = 1 sets remote frame transmission.

25.11 Power Saving Modes

25.11.1 CAN Controller sleep mode

The CAN Controller sleep mode can be used to set the CAN Controller to stand-by mode in order to reduce power consumption. The CAN Controller can enter the sleep mode from all operation modes. Release of the sleep mode returns the CAN Controller to exactly the same operation mode from which the sleep mode was entered.

In the sleep mode, the CAN Controller does not transmit messages, even when transmission requests are issued or pending.

(1) Entering CAN Controller sleep mode

The CPU issues a CAN Controller sleep mode transition request by setting FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B.

This transition request is acknowledged only under the following conditions.

1. The CAN Controller is already in one of the following operation modes
   – Normal operation mode
   – Normal operation mode with ABT
   – Receive-only mode
   – Single-shot mode
   – Self-test mode
   – CAN Controller stop mode in all the above operation modes
2. The CAN bus state is bus idle (the 4th bit in the interframe space is recessive).
   If the CAN bus is fixed to dominant, the request for transition to the sleep mode is held pending. Also the transition from stop mode to sleep mode is independent of the CAN bus state.
3. No transmission request is pending.

Note If a sleep mode request is pending, and at the same time a message is received in a message box, the sleep mode request is not cancelled, but is executed right after message storage has been finished. This may result in CAN Controller being in sleep mode, while the CPU would execute the RX interrupt routine. Therefore, the interrupt routine must check the access to the message buffers as well as reception history list registers by using the FCNnGMCLSSMO flag, if sleep mode is used.

If any one of the conditions mentioned above is not met, the CAN Controller will operate as follows.

- If the sleep mode is requested from the initialization mode, the sleep mode transition request is ignored and the CAN Controller remains in the initialization mode.
- If the CAN bus state is not bus idle (i.e., the CAN bus state is either transmitting or receiving) when the sleep mode is requested in one of the operation modes, immediate transition to the sleep mode is not possible. In this case, the sleep mode transition request is held pending until the CAN bus state becomes bus idle (the 4th bit in the interframe space is recessive). In the time from the sleep mode request to successful transition, FCNnCMCLCTL.FCNnCMCLMDPF[1:0] remain 00_B. When the channel has entered the sleep mode, the FCNnCMCLMDPF[1:0] bits are set to 01_B.

- If a request for transition to the initialization mode and a request for transition to the sleep mode are made at the same time while the CAN Controller is in one of the operation modes, the request for the initialization mode is enabled. The CAN Controller enters the initialization mode at a predetermined timing. At this time, the sleep mode request is not held pending and is ignored.
- Even when initialization mode and sleep mode are not requested simultaneously (i.e the first request has not been granted while the second request is made), the request for initialization has priority over the sleep mode request. The sleep mode request is cancelled when the initialization mode is requested. When a pending request for initialization mode is present, a subsequent request for Sleep mode request is cancelled right at the point in time where it was submitted.

(2) Status in CAN Controller sleep mode

The CAN Controller is in the following state after it enters the sleep mode:

- The CAN channel clock T_{CANCH} is stopped and the power consumption is minimized.
- The function to detect the falling edge of the CAN Controller reception pin (CRXDn) remains in effect to wake up the CAN Controller from the CAN bus.
- To wake up the CAN Controller from the CPU, data can be set to FCNnCMCLCTL.FCNnCMCLMDPF[1:0], but nothing can be written to other CAN Controller registers or bits.
- The CAN Controller registers can be read, except for the FCNnCMLISTR, FCNnCMRGRX, FCNnCMLOSTR, and FCNnCMTGTX registers.
- The CAN Controller message buffer registers cannot be written or read.
- FCNnGMCLCTL.FCNnGMCLSSMO is cleared.
- The registers FCNnDNBMRX cannot be read.
- A request for transition to the initialization mode is not acknowledged and is ignored.

(3) Releasing CAN Controller sleep mode

The CAN Controller sleep mode is released by the following events:

- When the CPU sets 00_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0]
- A falling edge at the CAN Controller reception pin CRXDn (i.e. the CAN bus level shifts from recessive to dominant)

Caution Even if the falling edge belongs to the SOF of a receive message, this message will not be received and stored. If the CPU has turned off the clock supply to the CAN Controller while the CAN Controller was in sleep mode, even subsequently the sleep mode will not be released and FCNnCMCLMDPF[1:0] will remain 01_B unless the clock to the CAN Controller is supplied again. In addition to this, the receive message will not be received after that.

After releasing the sleep mode, the CAN Controller returns to the operation mode from which the sleep mode was requested and FCNnCMCLCTL.FCNnCMCLCTL[1:0] must be reset by software to 00_B. If the CAN Controller sleep mode is released by a change in the CAN bus state, FCNnCMISCTL.FCNnCMISITSF5 is set to 1, regardless of FCNnCMIECTL.FCNnCMIEINTF[6:0]. After the CAN Controller is released from the sleep mode, it participates in the CAN bus again by automatically detecting 11 consecutive recessive-level bits on the CAN bus. The user application has to wait until FCNnGMCLCTL.FCNnGMCLSSMO = 1, before accessing message buffers again.

When a request for transition to the initialization mode is made while the CAN Controller is in the sleep mode, that request is ignored; the CAN Controller has to be released from sleep mode by software first before entering the initialization mode.

Cautions

1. Be aware that the release of sleep mode by CAN bus event, and thus the wake up interrupt may happen at any time, even right after requesting sleep mode, if a CAN bus event occurs.
2. Always reset the FCNnCMCLCTL.FCNnCMCLMDPF[1:0] bits to 00_B, when waking up from sleep mode, before accessing any other registers of the CAN Controller.
3. Always clear the interrupt flag FCNnCMISCTL.FCNnCMISITSF5, when waking up from sleep mode.

25.11.2 CAN Controller stop mode

The CAN Controller stop mode can be used to set the CAN Controller to stand-by mode to reduce power consumption. The CAN Controller can enter the stop mode only from the sleep mode. Release of the stop mode puts the CAN Controller in the sleep mode.

The stop mode can only be released (entering sleep mode) by setting 01_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0] and not by a change in the CAN bus state. No message is transmitted even when transmission requests are issued or pending.

(1) Entering CAN Controller stop mode

A stop mode transition request is issued by setting 11_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0].

A stop mode request is only acknowledged when the CAN Controller is in the sleep mode. In all other modes, the request is ignored.

Caution To set the CAN Controller to the stop mode, it must be in the sleep mode first. To confirm that the CAN Controller is in the sleep mode, check that the FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B, and then request the stop mode. If a bus change occurs at the CAN Controller reception pin CRXDn while this process is being performed, the sleep mode is automatically released. In this case, the stop mode transition request cannot be acknowledged.

(2) Status in CAN Controller stop mode

The CAN Controller is in the following state after it enters the stop mode.

- The CAN channel clock T_{CANCH} is stopped and the power consumption is minimized.
- To wake up the CAN Controller from the CPU, data can be set to FCNnCMCLCTL.FCNnCMCLMDPF[1:0], but nothing can be written to other CAN Controller registers or bits.
- The CAN Controller registers can be read, except for the FCNnCMLISTR, FCNnCMRGRX, FCNnCMLOSTR, and FCNnCMTGTX registers.
- The CAN Controller message buffer registers cannot be written or read.
- FCNnGMCLCTL.FCNnGMCLSSMO is cleared.
- The registers FCNnDNBMRX cannot be read.
- An initialization mode transition request is not acknowledged and is ignored.

(3) Releasing CAN Controller stop mode

The stop mode can only be released by writing 01_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0]. After releasing the stop mode, the CAN Controller enters the sleep mode.

When the initialization mode is requested while the CAN Controller is in the stop mode, that request is ignored; the CPU has to release the stop mode and subsequently sleep mode before entering the initialization mode. It is impossible to enter the other operation mode directly from the stop mode not entering the sleep mode, that request is ignored.

25.11.3 Example of using power saving modes

In some application systems, it may be necessary to place the CPU in a power saving mode to reduce the power consumption. By using the power saving mode specific to the CAN Controller and the power saving mode specific to the CPU in combination, the CPU can be woken up from the power saving status by the CAN bus.

Here is an example for using the power saving modes.

- First, put the CAN Controller in the sleep mode (FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 01_B).
  After successfully confirming this state by reading back the sleep mode status, put the CPU in the power saving mode. Disable interrupts for the CPU, while processing additional tasks after the CAN Controller is in sleep mode, to avoid that the CAN Controller wakeup interrupt is acknowledged. If an edge transition from recessive to dominant is detected at the CAN Controller reception pin CRXDn in this status, FCNnCMISCTL.FCNnCMISITSF5 in the CAN Controller is set to 1. If FCNnCNIECTL.FCNnCMIEINT5 is set to 1, a wakeup interrupt (INTCnWUP) is generated.
  The CAN Controller is automatically released from sleep mode (FCNnCMCLMDPF[1:0] = 00_B) and returns to normal operation mode.
- The CPU, in response to INTCnWUP, can release its own power saving mode and return to normal operation mode.

  To further reduce the power consumption of the CPU, the internal clock - including that of the CAN Controller - may be stopped. In this case, the operating clock supplied to the CAN Controller is stopped after the CAN Controller has been put in sleep mode. Then the CPU enters a power saving mode in which the clock supplied to the CPU is stopped.
- If an edge transition from recessive to dominant is detected at the CAN Controller reception pin CRXDn in this status, the CAN Controller can set FCNnCMISCTL.FCNnCMISITSF5 to 1 and generate the wakeup interrupt INTCnWUP even if it is not supplied with the clock.
- The other functions, however, do not operate, because clock supply to the CAN Controller is stopped, which remains in sleep mode.
- The CPU, in response to INTCnWUP,
  - releases its power saving mode,
  - resumes supply of the internal clocks - including the clock to the CAN Controller - after the oscillation stabilization time has elapsed, and
  - starts instruction execution.
- The CAN Controller is immediately released from the sleep mode when clock supply is resumed, and returns to the normal operation mode (FCNnCMCLCTL.FCNnCMCLMDPF[1:0] = 00_B).

25.12 Interrupt Function

The CAN Controller provides 6 different interrupt sources.

The occurrence of these interrupt sources is stored in interrupt status registers. Four separate interrupt request signals are generated from the six interrupt sources. When an interrupt request signal that corresponds to two or more interrupt sources is generated, the interrupt sources can be identified by using an interrupt status register. After an interrupt source has occurred, the corresponding interrupt status bit must be cleared to 0 by software.

Table 25-21 List of CAN Controller interrupt sources

No.	Interrupt status bit FCNnCMISCTL.	Interrupt enable bit FCNnCMIECTL. [a]	Interrupt request signal	Interrupt source description
1	FCNnCMISITSF0	FCNnCMIESEIE0	INTCnTRX	Message frame successfully transmitted from message buffer m
2	FCNnCMISITSF1	FCNnCMIESEIE1	INTCnREC	Valid message frame reception in message buffer m
3	FCNnCMISITSF2	FCNnCMIESEIE2	INTCnERR	CAN Controller error state interrupt • This interrupt is generated when the transmission/reception error counter is at the warning level, or in the error passive or bus-off state.
4	FCNnCMISITSF3	FCNnCMIESEIE3		CAN Controller protocol error interrupt • This interrupt is generated when a stuff error, form error, ACK error, bit error, or CRC error occurs.
5	FCNnCMISITSF4	FCNnCMIESEIE4		CAN Controller arbitration loss interrupt
6	FCNnCMISITSF5	FCNnCMIESEIE5	INTCnWUP	CAN Controller wakeup interrupt from CAN Controller sleep mode • This interrupt is generated when the CAN Controller is woken up from the sleep mode because a falling edge is detected at the CAN Controller reception pin (CAN bus transition from recessive to dominant).
7	FCNnCMISITSF6	FCNnCMIESEIE6		CAN Controller transmit abort interrupt status • This interrupt is generated when the abortion of a transmission was successful (aborted message was not sent).

a) The message buffer interrupt enable bit FCNnMmCTL.FCNnMmIENF of the corresponding message buffer has to be set to 1 for that message buffer to participate in the interrupt generation process.

25.13 Diagnosis Functions and Special Operational Modes

The CAN Controller provides a receive-only mode, single-shot mode, and self-test mode to support CAN bus diagnosis functions or the operation of special CAN communication methods.

25.13.1 Receive-only mode

The receive-only mode is used to monitor receive messages without causing any interference on the CAN bus and can be used for CAN bus analysis nodes.

For example, this mode can be used for automatic baud-rate detection. The baudrate in the CAN Controller is changed until “valid reception” is detected, so that the baudrates are matching (“valid reception” means a message frame has been received in the CAN protocol layer without occurrence of an error and with an appropriate ACK between nodes connected to the CAN bus). A valid reception does not require message frames to be stored in a receive message buffer (data frames) or transmit message buffer (remote frames). The event of valid reception is indicated by setting FCNnCMCLCTL.FCNnCMCLVALF = 1.

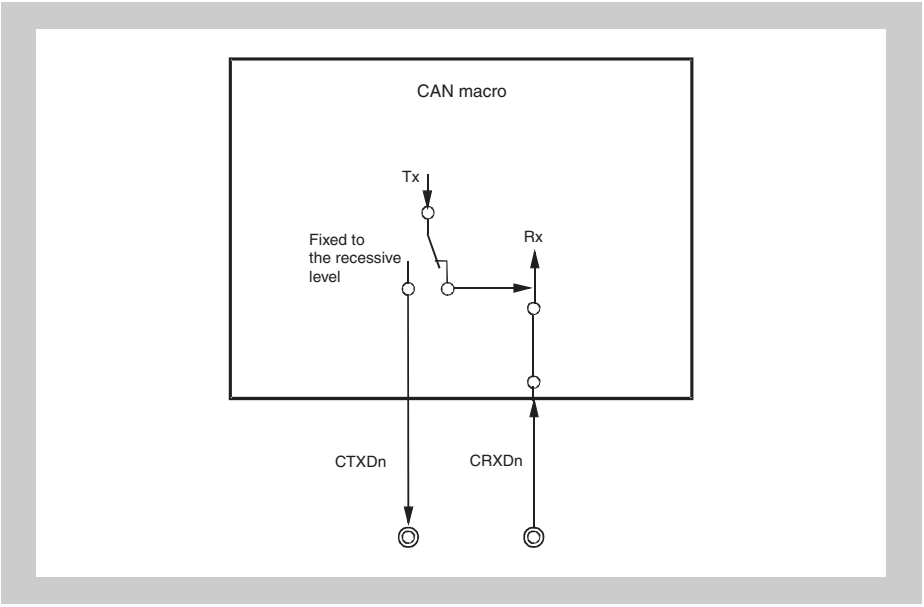


Figure 25-12 CAN Controller terminal connection in receive-only mode

In the receive-only mode, no message frames can be transmitted from the CAN Controller to the CAN bus. Transmit requests issued for message buffers defined as transmit message buffers are held pending.

In the receive-only mode, the CAN Controller transmission pin CTXDn in the CAN Controller is fixed to the recessive level. Therefore, no active error flag can be transmitted from the CAN Controller to the CAN bus even when a CAN bus error is detected while receiving a message frame. Since no transmission can be issued from the CAN Controller, the transmission error counter the FCNnCMERCNT.TEC7 to FCNnCMERCNT.TEC0 bits are never updated. Therefore, a CAN Controller in the receive-only mode does not enter the bus-off state.

Furthermore, in the receive-only mode ACK is not returned to the CAN bus in this mode upon the valid reception of a message frame. Internally, the local node recognizes that it has transmitted ACK. An overload frame cannot be transmitted to the CAN bus.

Caution If only two CAN nodes are connected to the CAN bus and one of them is operating in the receive-only mode, there is no ACK on the CAN bus. Due to the missing ACK, the transmitting node will transmit an active error flag, and repeat transmitting a message frame. The transmitting node becomes error passive after transmitting the message frame 16 times (assuming that the error counter was 0 in the beginning and no other errors have occurred). After the message frame for the 17th time is transmitted, the transmitting node generates a passive error flag. The receiving node in the receive-only mode detects the first valid message frame at this point, and the FCNnCMCLCTL.FCNnCMCLVALF bit is set to 1 for the first time.

25.13.2 Single-shot mode

In the single-shot mode, automatic re-transmission as defined in the CAN protocol is switched off. (According to the CAN protocol, a message frame transmission that has been aborted by either arbitration loss or error occurrence has to be repeated without control by software.) All other behavior of single shot mode is identical to normal operation mode. Features of single shot mode can not be used in combination with normal mode with ABT.

The single-shot mode disables the re-transmission of an aborted message frame transmission according to the setting of FCNnCMCLCTL.FCNnCMCLALBF. When FCNnCMCLALBF is cleared to 0, re-transmission upon arbitration loss and upon error occurrence is disabled. If FCNnCMCLALBF is set to 1, re-transmission upon error occurrence is disabled, but re-transmission upon arbitration loss is enabled. As a consequence, FCNnMmCTL.FCNnMmTRQF in a message buffer defined as a transmit message buffer is cleared to 0 by the following events:

- Successful transmission of the message frame
- Arbitration loss while sending the message frame
- Error occurrence while sending the message frame

The events arbitration loss and error occurrence can be distinguished by checking FCNnCMISCTL.FCNnCMISITSF4 and FCNnCMISCTL.FCNnCMISITSF3 respectively, and the type of the error can be identified by reading FCNnCMLCSTR.FCN0CMLCSSLC[2:0].

Upon successful transmission of the message frame, the transmit completion interrupt bit FCNnCMISCTL.FCNnCMISITSF0 is set to 1. If FCNnCMIECTL.FCNnCMIEINTF0 is set to 1 at this time, an interrupt request signal is output.

The single-shot mode can be used when emulating time-triggered communication methods (e.g., TTCAN level 1).

Caution FCNnCMCLCTL.FCNnCMCLALBF is only valid in single-shot mode. It does not influence the operation of re-transmission upon arbitration loss in the other operation modes.

25.13.3 Self-test mode

In the self-test mode, message frame transmission and message frame reception can be tested without connecting the CAN node to the CAN bus or without affecting the CAN bus.

In the self-test mode, the CAN Controller is completely disconnected from the CAN bus, but transmission and reception are internally looped back. The CAN Controller transmission pin CTXDn is fixed to the recessive level.

If the falling edge on the CAN Controller reception pin CRXDn is detected after the CAN Controller has entered the CAN Controller sleep mode from the self-test mode, however, the CAN Controller is released from sleep mode in the same manner as the other operation modes. To keep the CAN Controller in the sleep mode, use the CAN Controller reception pin CRXDn as a port pin.

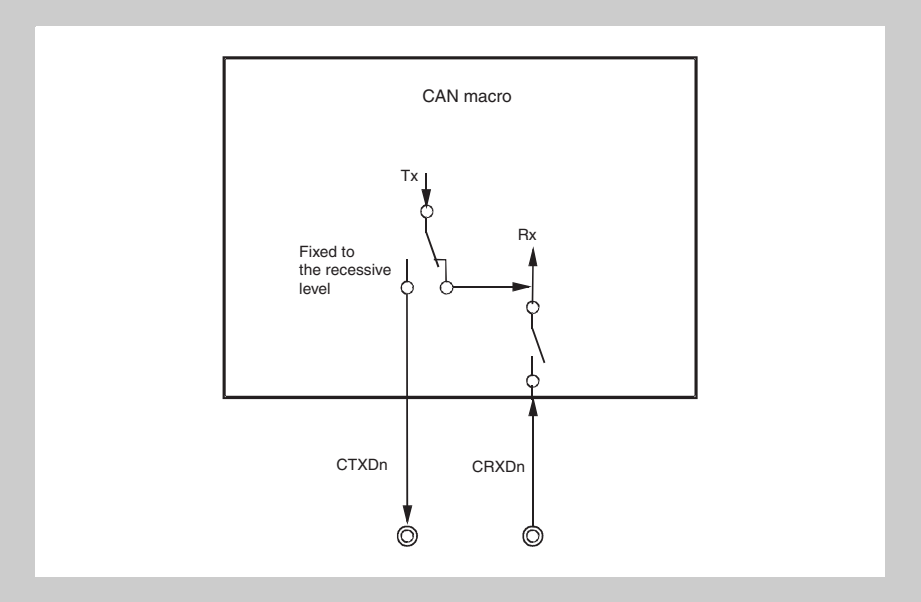


Figure 25-13 CAN Controller terminal connection in self-test mode

25.13.4 Receive/transmit operation in each operation mode

The following table shows outline of the receive/transmit operation in each operation mode.

Table 25-22 Outline of the receive/transmit in each operation mode

Operation mode	Transmission of data/ remote frame	Transmission of ACK	Transmission of error/ overload frame	Transmission retry	Automatic block transmission (ABT)	Set of FCNnCMCLVALF bit	Store data to message buffer
Initialization mode	No	No	No	No	No	No	No
Normal operation mode	Yes	Yes	Yes	Yes	No	Yes	Yes
Normal operation mode with ABT	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Receive only mode	No	No	No	No	No	Yes	Yes
Single-shot mode	Yes	Yes	Yes	No[a]	No	Yes	Yes
Self-test mode	Yes[b]	Yes[b]	Yes[b]	Yes[b]	No	Yes[b]	Yes[b]

a) When the arbitration lost occurs, control of re-transmission is possible by FCNnCMCLCTL.FCNnCMCLALBF.
b) Each signals are not generated to outside, but generated into the CAN Controller.

25.14 Time Stamp Function

CAN is an asynchronous, serial protocol. All nodes connected to the CAN bus have a local, autonomous clock. As a consequence, the clocks of the nodes have no relation (i.e., the clocks are asynchronous and may have different frequencies).

In some applications, however, a common time base over the network (= global time base) is needed. In order to build up a global time base, a time stamp function is used. The essential mechanism of a time stamp function is the capture of timer values triggered by signals on the CAN bus.

25.14.1 Time stamp function

The CAN Controller supports the capturing of timer values triggered by a specific frame. An on-chip 16-bit capture timer unit of the microcontroller is used in addition to the CAN Controller. The 16-bit capture timer unit captures the timer value according to a trigger signal (TSOUT) for capturing that is output when a data frame is received from the CAN Controller. The CPU can retrieve the time of occurrence of the capture event, i.e., the time stamp of the message received from the CAN bus, by reading the captured value. The TSOUT signal can be selected from the following two event sources and is specified by FCNnCMTSCTL.FCNnCMTSSELE.

- SOF event (start of frame)
  (FCNnCMTSCTL.FCNnCMTSSELE = 0)
- EOF event (last bit of end of frame)
  (FCNnCMTSCTL.FCNnCMTSSELE = 1)

The TSOUT signal is enabled by setting FCNnCMTSCTL.FCNnCMTSTSGE = 1.

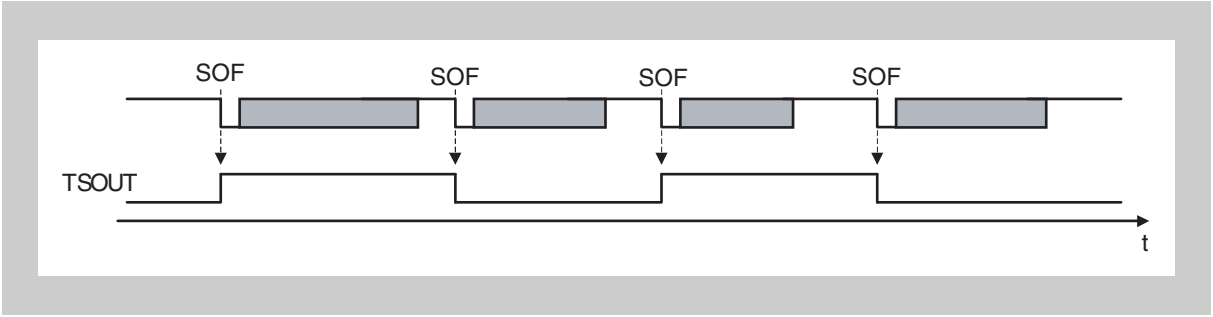


Figure 25-14 Timing diagram of capture signal TSOUT

The TSOUT signal toggles its level upon occurrence of the selected event during data frame reception (in *Figure 25-14 "Timing diagram of capture signal TSOUT"*, the SOF is used as the trigger event source). To capture a timer value by using the TSOUT signal, the capture timer unit must detect the capture signal at both the rising edge and falling edge.

This time stamp function is controlled by the FCNnCMTSLOKE bit of the FCNnCMTSCTL register. When FCNnCMTSLOKE is cleared to 0, the TSOUT signal toggles upon occurrence of the selected event. If FCNnCMTSLOKE is set to 1, the TSOUT signal toggles upon occurrence of the selected event, but the toggle is stopped as FCNnCMTSCTL.FCNnCMTSTSGE is automatically cleared to 0 as soon as the message storing to the message buffer 0 starts. This suppresses the subsequent toggle occurrence by the TSOUT signal, so

that the time stamp value toggled last (= captured last) can be saved as the time stamp value of the time at which the data frame was received in message buffer 0.

Caution The time stamp function using the FCNnCMTSLOKE bit stops toggle of the TSOUT signal by receiving a data frame in message buffer 0. Toggle of the TSOUT signal does not stop when a data frame is received in a message buffer other than message buffer 0.
A data frame cannot be received in message buffer 0 when the CAN Controller is in the normal operation mode with ABT, because message buffer 0 must be set as a transmit message buffer.
In this operation mode, therefore, the function to stop toggle of the TSOUT signal by the FCNnCMTSLOKE bit cannot be used.

25.15 Baudrate Settings

25.15.1 Baudrate setting conditions

Make sure that the settings are within the range of limit values for ensuring correct operation of the CAN Controller, as follows.

- 5 TQ ≤ SPT (sampling point) ≤ 17 TQ

  SPT = TSEG1 + 1
- 8 TQ ≤ DBT (data bit time) ≤ 25 TQ

  DBT = TSEG1 + TSEG2 + 1 TQ = TSEG2 + SPT
- 1 TQ ≤ SJW (synchronization jump width) ≤ 4 TQ

  SJW ≤ DBT – SPT
- 4 ≤ TSEG1 ≤ 16
- 1 ≤ TSEG2 ≤ 8

Notes
1. TQ = $1/f_{TQ}$ (f_{TQ}: CAN protocol layer clock)
2. The values TSEG1, TSEG2 and SJW are defined by following register bits:
   TSEG1 = FCNnCMBTCTL.FCNnCMBTS1LG[3:0] + 1
   TSEG2 = FCNnCMBTCTL.FCNnCMBTS2LG[2:0] + 1
   SJW = FCNnCMBTCTL.FCNnCMBTJWLG[1:0] + 1

Table 25-23 "Settable bit rate combinations" shows the combinations of bit rates that satisfy the above conditions.

Table 25-23 Settable bit rate combinations (1/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
25	1	8	8	8	1111	111	68.0
24	1	7	8	8	1110	111	66.7
24	1	9	7	7	1111	110	70.8
23	1	6	8	8	1101	111	65.2
23	1	8	7	7	1110	110	69.6
23	1	10	6	6	1111	101	73.9
22	1	5	8	8	1100	111	63.6
22	1	7	7	7	1101	110	68.2
22	1	9	6	6	1110	101	72.7
22	1	11	5	5	1111	100	77.3
21	1	4	8	8	1011	111	61.9
21	1	6	7	7	1100	110	66.7
21	1	8	6	6	1101	101	71.4
21	1	10	5	5	1110	100	76.2
21	1	12	4	4	1111	011	81.0
20	1	3	8	8	1010	111	60.0
20	1	5	7	7	1011	110	65.0
20	1	7	6	6	1100	101	70.0
20	1	9	5	5	1101	100	75.0
20	1	11	4	4	1110	011	80.0
20	1	13	3	3	1111	010	85.0
19	1	2	8	8	1001	111	57.9
19	1	4	7	7	1010	110	63.2
19	1	6	6	6	1011	101	68.4
19	1	8	5	5	1100	100	73.7
19	1	10	4	4	1101	011	78.9
19	1	12	3	3	1110	010	84.2
19	1	14	2	2	1111	001	89.5
18	1	1	8	8	1000	111	55.6
18	1	3	7	7	1001	110	61.1
18	1	5	6	6	1010	101	66.7
18	1	7	5	5	1011	100	72.2
18	1	9	4	4	1100	011	77.8
18	1	11	3	3	1101	010	83.3
18	1	13	2	2	1110	001	88.9
18	1	15	1	1	1111	000	94.4
17	1	2	7	7	1000	110	58.8

Table 25-23 Settable bit rate combinations (2/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
17	1	4	6	6	1001	101	64.7
17	1	6	5	5	1010	100	70.6
17	1	8	4	4	1011	011	76.5
17	1	10	3	3	1100	010	82.4
17	1	12	2	2	1101	001	88.2
17	1	14	1	1	1110	000	94.1
16	1	1	7	7	0111	110	56.3
16	1	3	6	6	1000	101	62.5
16	1	5	5	5	1001	100	68.8
16	1	7	4	4	1010	011	75.0
16	1	9	3	3	1011	010	81.3
16	1	11	2	2	1100	001	87.5
16	1	13	1	1	1101	000	93.8
15	1	2	6	6	0111	101	60.0
15	1	4	5	5	1000	100	66.7
15	1	6	4	4	1001	011	73.3
15	1	8	3	3	1010	010	80.0
15	1	10	2	2	1011	001	86.7
15	1	12	1	1	1100	000	93.3
14	1	1	6	6	0110	101	57.1
14	1	3	5	5	0111	100	64.3
14	1	5	4	4	1000	011	71.4
14	1	7	3	3	1001	010	78.6
14	1	9	2	2	1010	001	85.7
14	1	11	1	1	1011	000	92.9
13	1	2	5	5	0110	100	61.5
13	1	4	4	4	0111	011	69.2
13	1	6	3	3	1000	010	76.9
13	1	8	2	2	1001	001	84.6
13	1	10	1	1	1010	000	92.3
12	1	1	5	5	0101	100	58.3
12	1	3	4	4	0110	011	66.7
12	1	5	3	3	0111	010	75.0

Table 25-23 Settable bit rate combinations (3/3)

Valid bit rate setting					FCNnCMBTCTL register setting value		Sampling point (unit %)
DBT length	SYNC SEGMENT	PROP SEGMENT	PHASE SEGMENT1	PHASE SEGMENT2	FCNnCMBTS1LG[3:0]	FCNnCMBTS2LG[2:0]	
12	1	7	2	2	1000	001	83.3
12	1	9	1	1	1001	000	91.7
11	1	2	4	4	0101	011	63.6
11	1	4	3	3	0110	010	72.7
11	1	6	2	2	0111	001	81.8
11	1	8	1	1	1000	000	90.9
10	1	1	4	4	0100	011	60.0
10	1	3	3	3	0101	010	70.0
10	1	5	2	2	0110	001	80.0
10	1	7	1	1	0111	000	90.0
9	1	2	3	3	0100	010	66.7
9	1	4	2	2	0101	001	77.8
9	1	6	1	1	0110	000	88.9
8	1	1	3	3	0011	010	62.5
8	1	3	2	2	0100	001	75.0
8	1	5	1	1	0101	000	87.5
7[a]	1	2	2	2	0011	001	71.4
7[a]	1	4	1	1	0100	000	85.7
6[a]	1	1	2	2	0010	001	66.7
6[a]	1	3	1	1	0011	000	83.3
5[a]	1	2	1	1	0010	000	80.0
4[a]	1	1	1	1	0001	000	75.0

a) Setting with a DBT value of 7 or less is valid only when the value of the FCNnCMBRPRS register is other than 00_H.

Caution The values in *Table 25-23 "Settable bit rate combinations"* do not guarantee the operation of the network. Thoroughly check the effect on the network, taking into consideration oscillation errors and delays of the CAN bus and CAN transceiver.

25.15.2 Clock prescaler and baudrate generator settings

In order to achieve the desired baudrate, time quanta, respectively data bit rate, combinations, described in the section above, the clock prescaler and the baudrate generator must be set up properly.

The diagram belows shows the clocks and how they are generated.

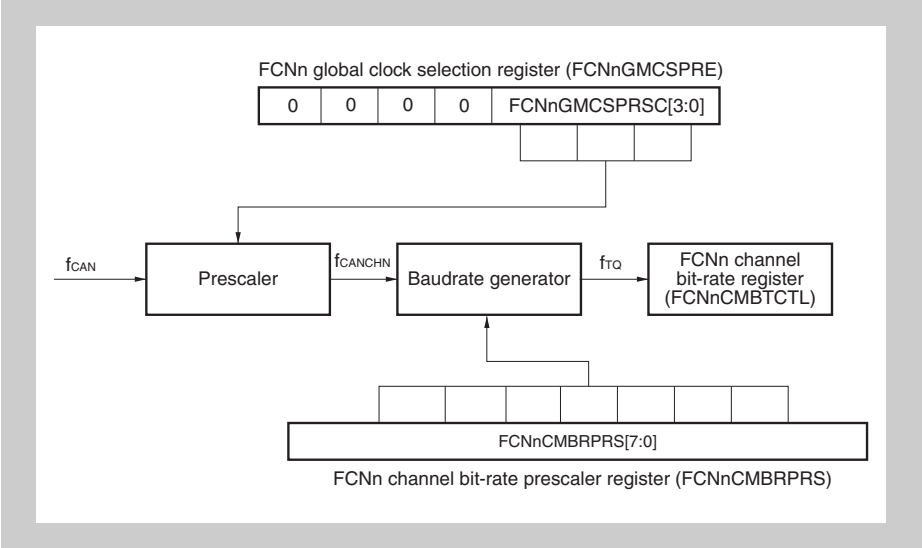


Figure 25-15 CAN Controller clocks

The setting of the global clock selection register bits FCNnGMCSPRE.FCNnCSPRSC[3:0] for the desired CAN channel clock f_{CANCHN} is calculated as follows:

$$FCNnGMCSPRSC = \frac{f_{CAN}}{f_{CANCHN}} - 1$$

f_{CAN}: clock supplied to CAN Controller (unit: Hz)

f_{CANCHN}: CAN channel clock (unit: Hz)

Following conditions must be met:

- $0 \leq FCNnCSPRSC \leq 15$
- FCNnCSPRSC defines an integer value with no remainder.

The setting of the channel bit rate prescaler register FCNnCMBRPRS for the desired data bit time DBT and the baudrate is calculated as follows:

$$FCNnCMBRPRS = \frac{f_{CANCHN}}{DBT \bullet BRT} - 1$$

f_{CANCHN}: CAN channel clock (unit: Hz)

DBT: Data bit time (unit: 1/bit)

BRT: Baudrate (unit: bit/s)

If the baudrate is given, f_{CANCHN} and DBT must be selected such, that following conditions are met:

- $0 \leq FCNnCMBRPRS \leq 15$
- FCNnCMBRPRS defines an integer value with no remainder.
- $8 \leq DBT \leq 25$
- The restrictions in *Table 25-23 "Settable bit rate combinations" on page 1982* have to be obeyed.
- Restrictions of DBT selection and baudrate depending on clock f_{CAN} have to be obeyed. Refer to the section *"CAN baudrate and time quanta"* above in this chapter.

25.16 Operation of the CAN Controller

The processing procedure for showing in this chapter is recommended processing procedure to operate CAN Controller.

Develop the program referring to recommended processing procedure in this chapter.

25.16.1 Initialization

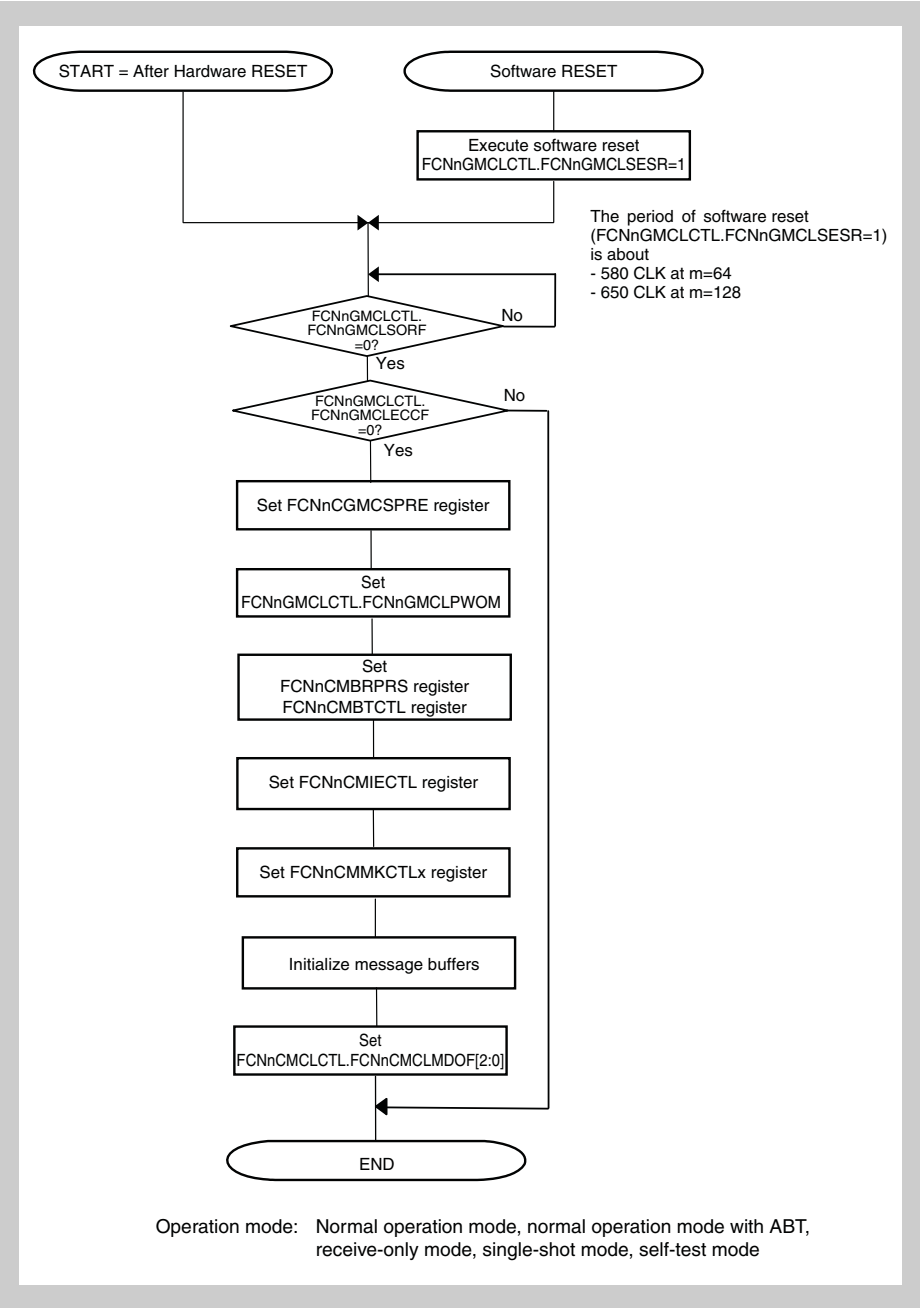


Figure 25-16 Initialization

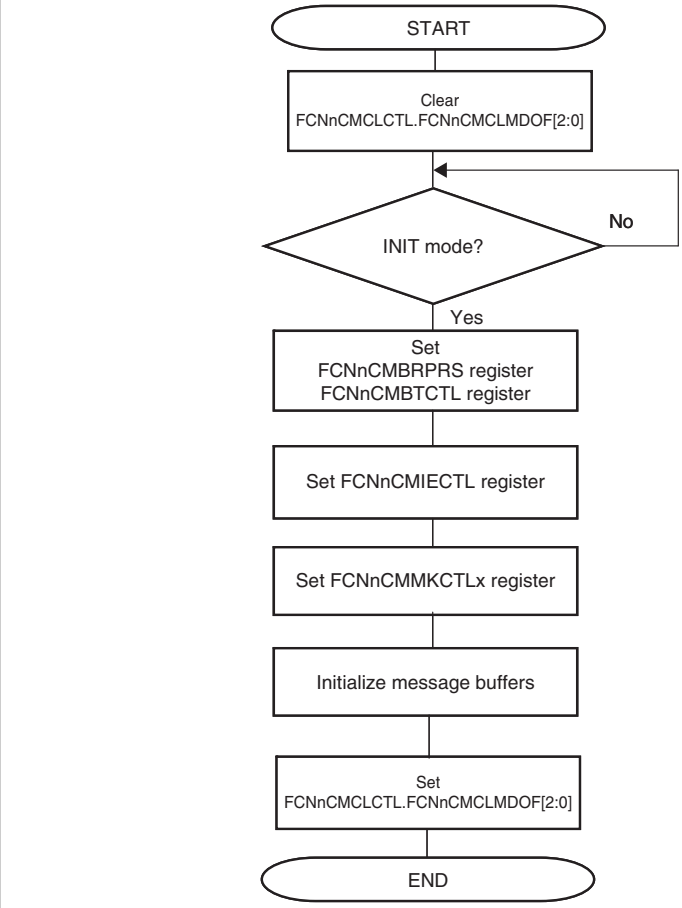


Figure 25-17 Re-initialization without Software Reset function

Caution When clearing the error counter in Re-initialization, set FCNnCMCLERCF bit only following conditions:

- After starting the CAN Controller (FCNnGMCLPWOM is set at FCNnGMCLPWOM=0), while in initialization mode.
- After aborting all transmit request according to *Figure 25-27 “Transmission abort processing (except normal operation mode with ABT)” on page 1999* or *Figure 25-28 “Transmission abort processing (normal operation mode with ABT) - Repeat option for aborted message” on page 2000* , while in initialization mode.

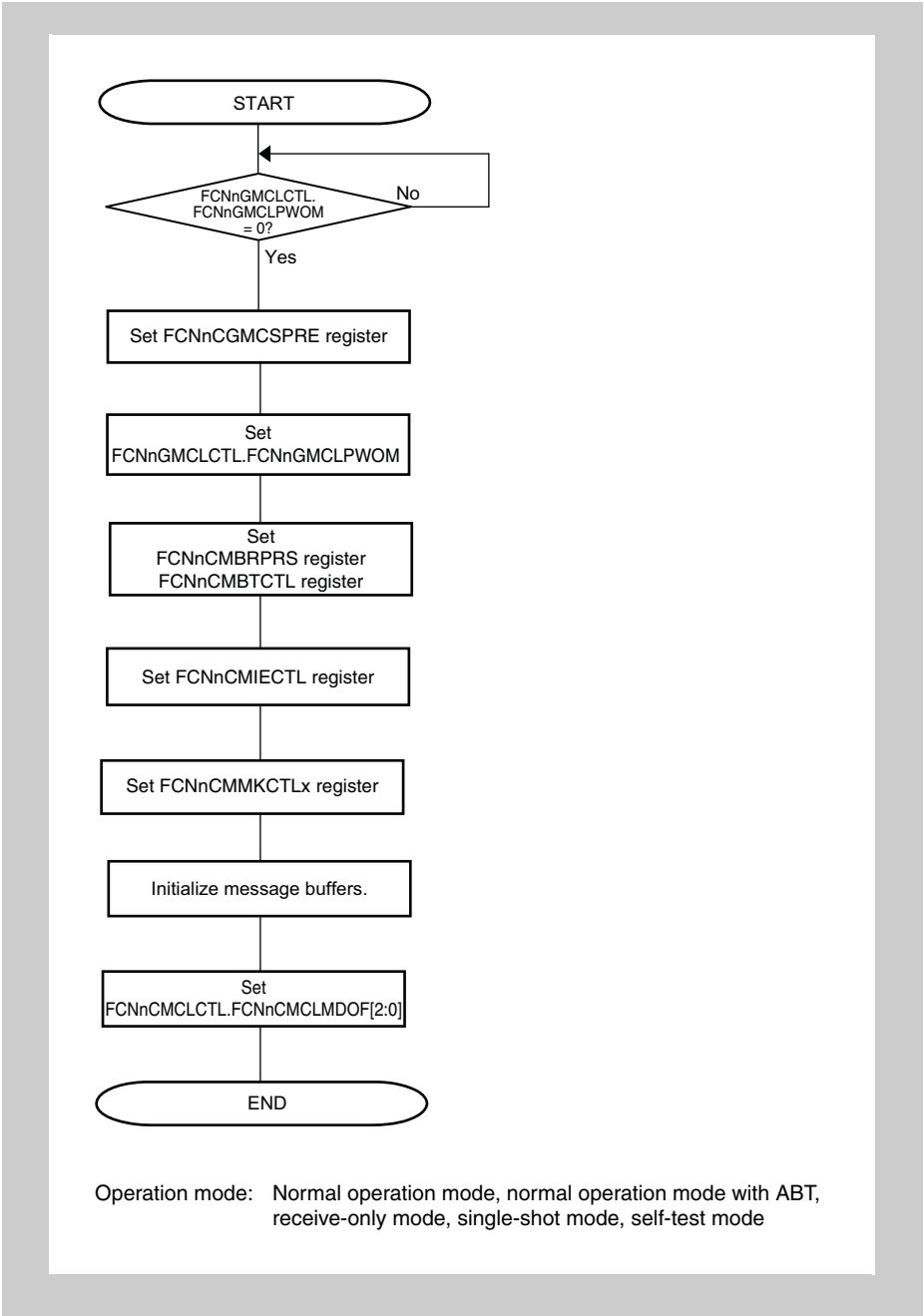


Figure 25-18 Re-initialization with Software Reset function

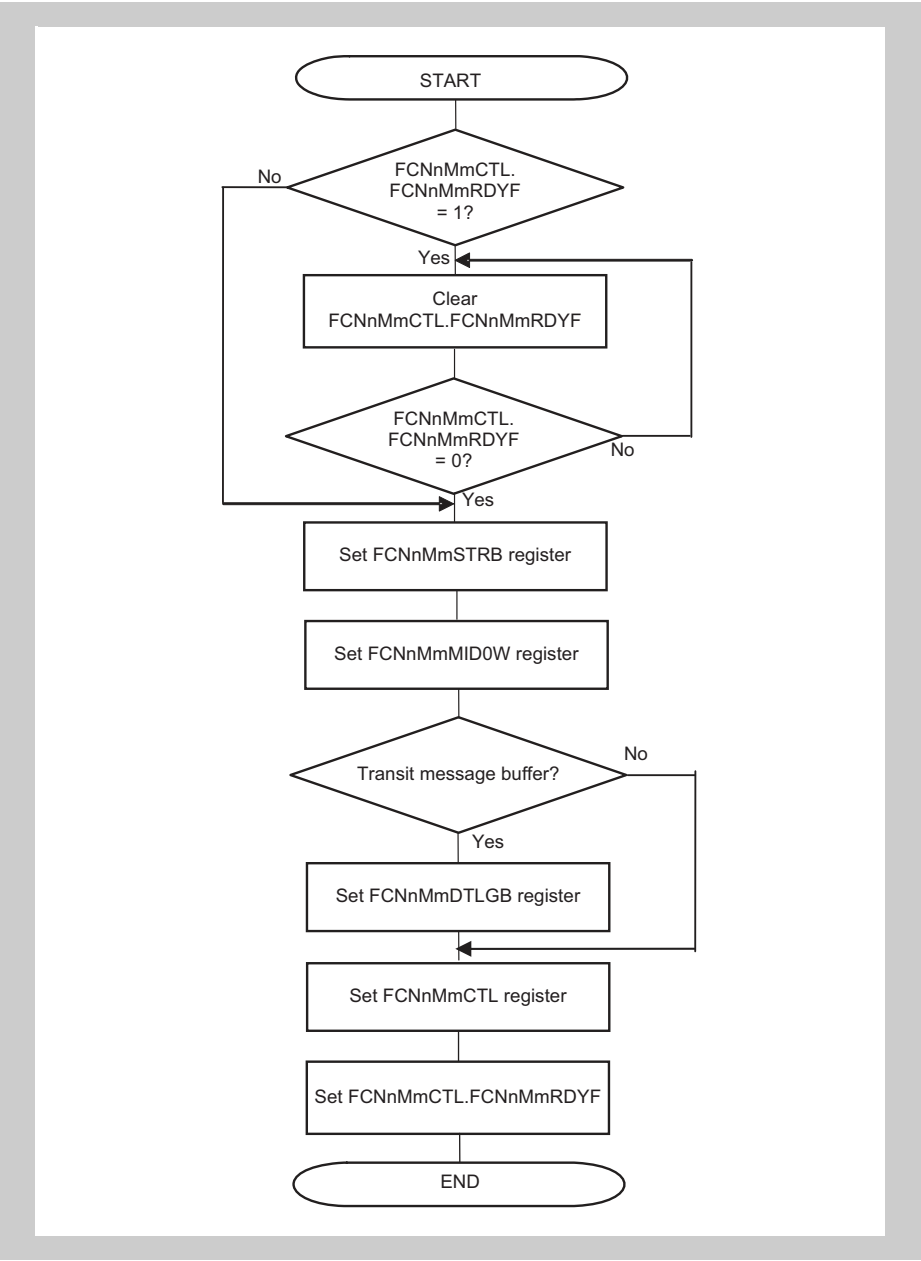


Figure 25-19 Message buffer initialization

Cautions

1. Before a message buffer is initialized, FCNnMmCTL.FCNnMmRDYF must be cleared.
2. Make the following settings for message buffers not used by the application.
   - Clear FCNnMmRDYF, FCNnMmTRQF, and FCNnMmDTNF bits of the FCNnMmCTL register to 0.
   - Clear FCNnMmSTRB.FCNnMmSSAM to 0.

Figure 25-20 "Message buffer redefinition during reception" shows the processing for a receive message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0001_B to 1000_B).

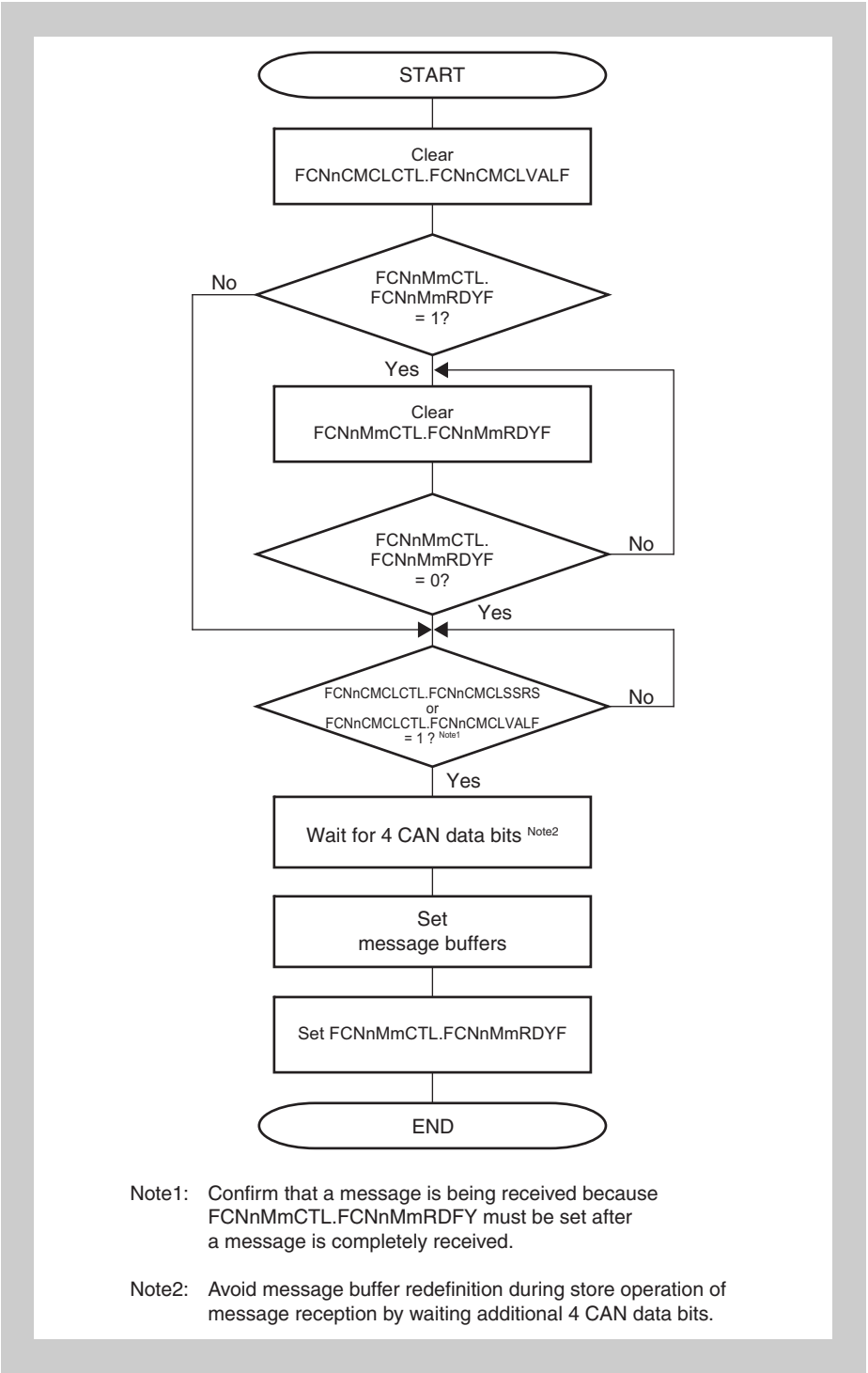


Figure 25-20 Message buffer redefinition during reception

Figure 25-21 "Message buffer redefinition during transmission" shows the processing for a transmit message buffer during transmission (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B).

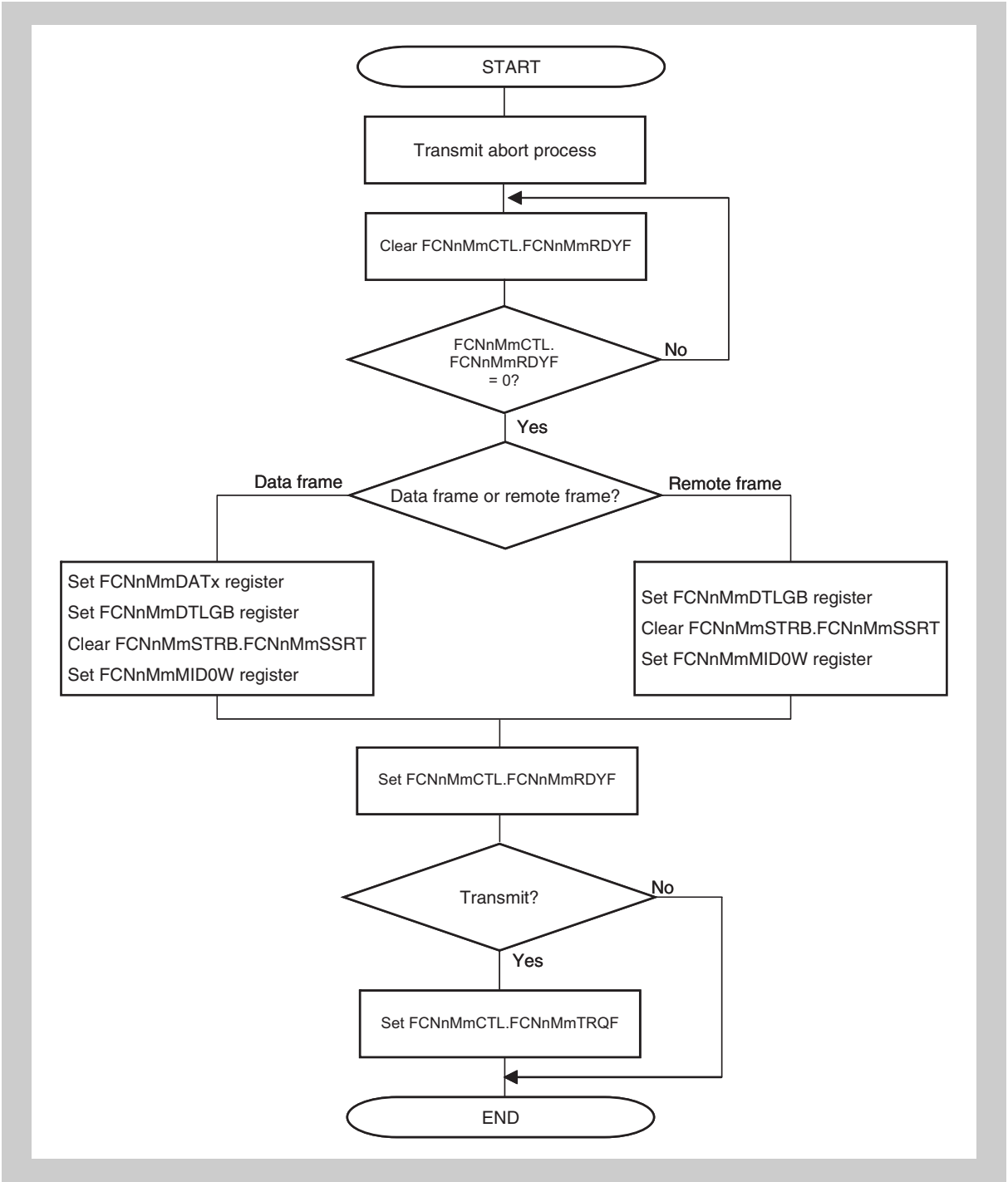


Figure 25-21 Message buffer redefinition during transmission

25.16.2 Message transmission

Figure 25-22 "Message transmit processing" shows the processing for a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B).

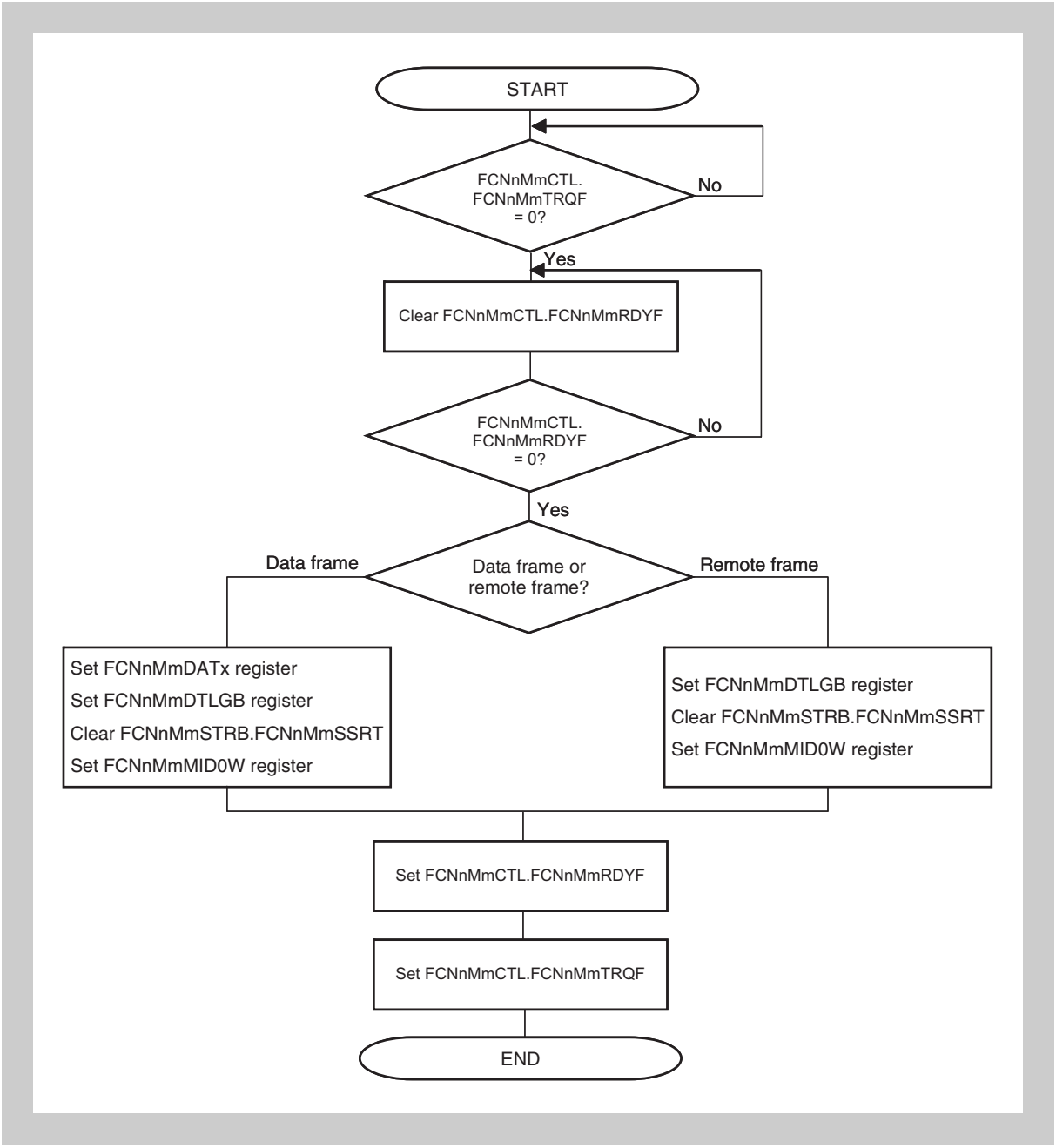


Figure 25-22 Message transmit processing

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Figure 25-23 "ABT message transmit processing" shows the processing for a transmit message buffer (FCNnMmSTRB.FCNnMmSSMT[3:0] = 0000_B)

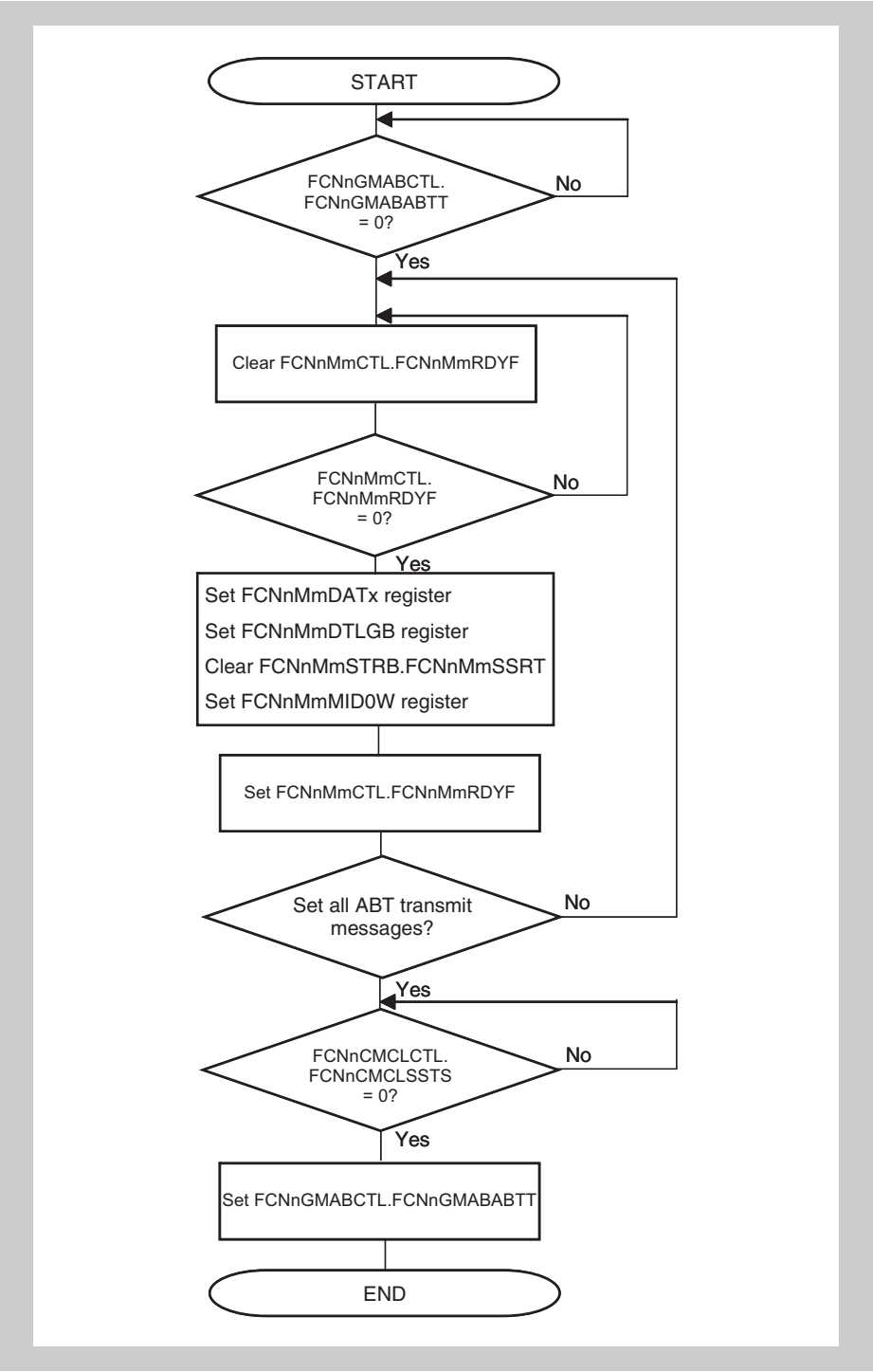


Figure 25-23 ABT message transmit processing

Note This processing (normal operation mode with ABT) can only be applied to message buffers usable with ABT mode. For message buffers other than the ABT message buffers, see *Figure 25-22 "Message transmit processing" on page 1993* .

Caution FCNnGMABCTL.FCNnGMABSEAT should be set to 1 after FCNnCMCLCTL.FCNnCMCLSSTS is cleared to 0. Checking FCNnCMCLCTL.FCNnCMCLSSTS and setting FCNnGMABCTL.FCNnGMABSEAT = 1 must be processed consecutively.

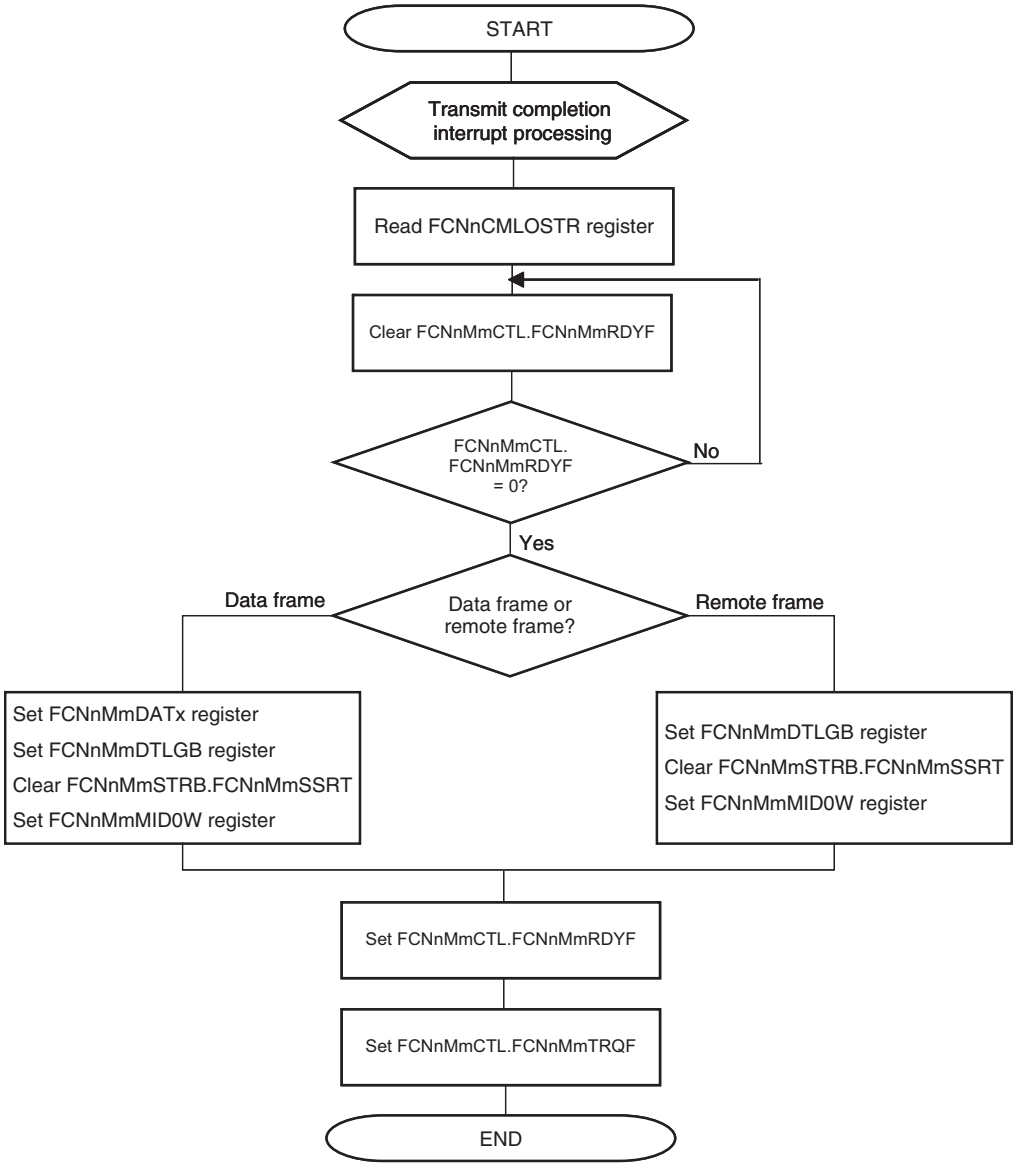


Figure 25-24 Transmission via interrupt (using FCNnCMLOSTR register)

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Note Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
It is recommended to cancel any sleep mode requests, before processing TX interrupts.

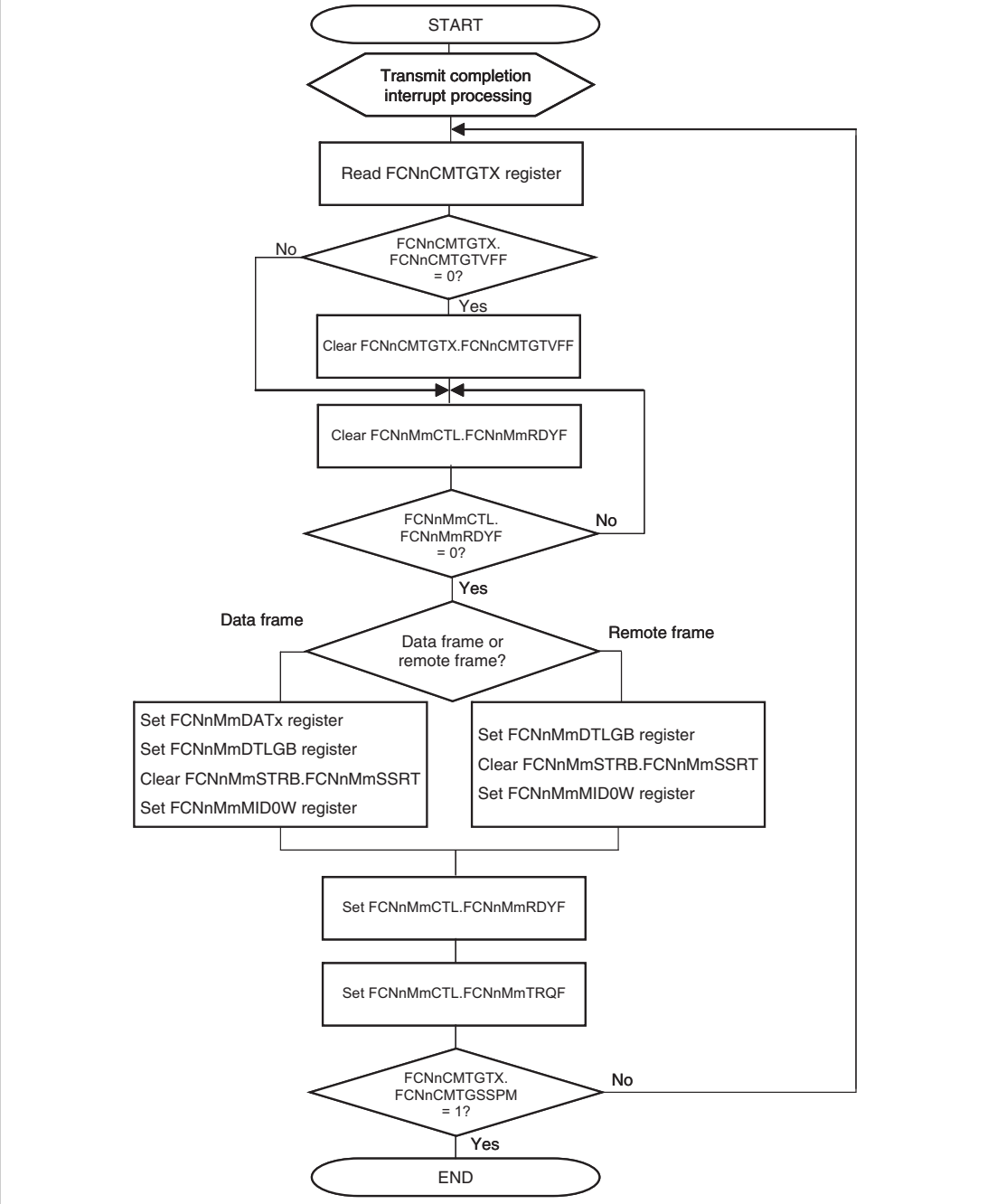


Figure 25-25 Transmission via interrupt (using FCNnCMTGTX register)

Cautions 1. FCNnMmCTL.FCNnMmTRQF should be set after

FCNnMmCTL.FCNnMmRDYF is set.

2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing TX interrupts.
2. If FCNnCMTGTX.FCNnCMTGTVFF was set once, the transmit history list is inconsistent. Consider to scan all configured transmit buffers for completed transmissions.

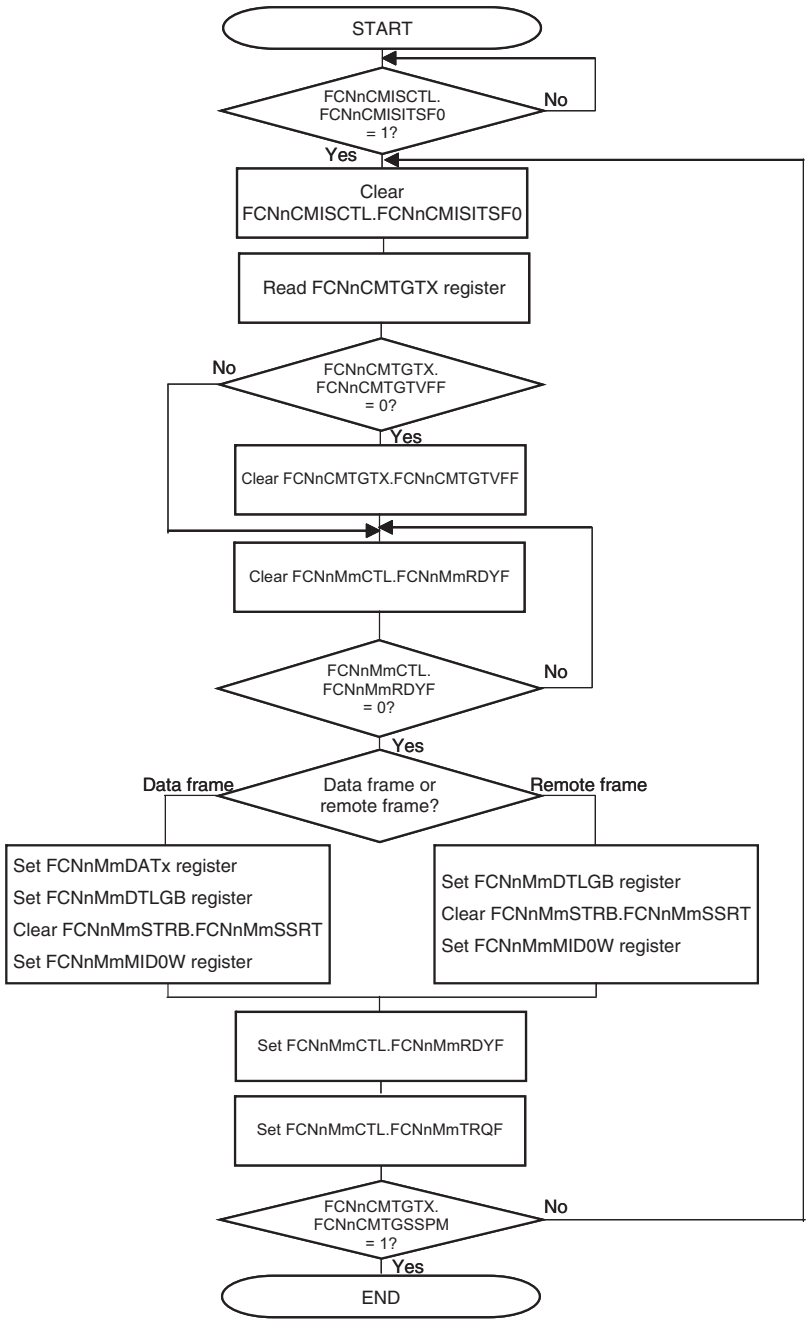


Figure 25-26 Transmission via software polling

Cautions

1. FCNnMmCTL.FCNnMmTRQF should be set after FCNnMmCTL.FCNnMmRDYF is set.
2. FCNnMmCTL.FCNnMmRDYF and FCNnMmCTL.FCNnMmTRQF should not be set at the same time.

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the polling routine, in order to check the access to the message buffers as well as TX history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
2. If FCNnCMTGTX.FCNnCMTGTVFF was set once, the transmit history list is inconsistent. Consider to scan all configured transmit buffers for completed transmissions.

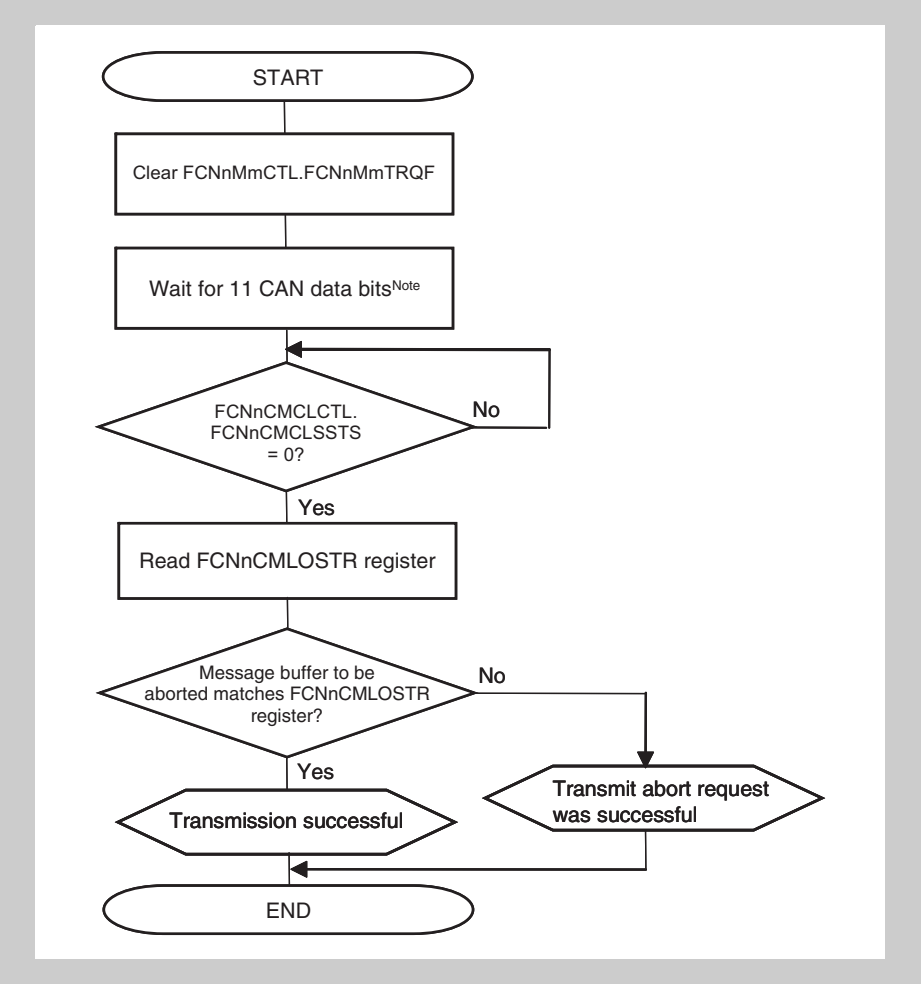


Figure 25-27 Transmission abort processing (except normal operation mode with ABT)

Note There is a possibility of starting the transmission without being aborted even if FCNnMmCTL.FCNnMmTRQF is cleared, because the transmission request to protocol layer might already been accepted between 11 bits, total of interframe space (3 bits) and suspend transmission (8 bits).

Cautions

1. Clear FCNnMmCTL.FCNnMmTRQF for aborting transmission request, not FCNnMmCTL.FCNnMmRDYF.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. FCNnCMCLCTL.FCNnCMCLSSTS can be periodically checked by a user application or can be checked after the transmit completion interrupt.

4. Do not execute any new transmission request including in the other message buffers while transmission abort processing is in progress.

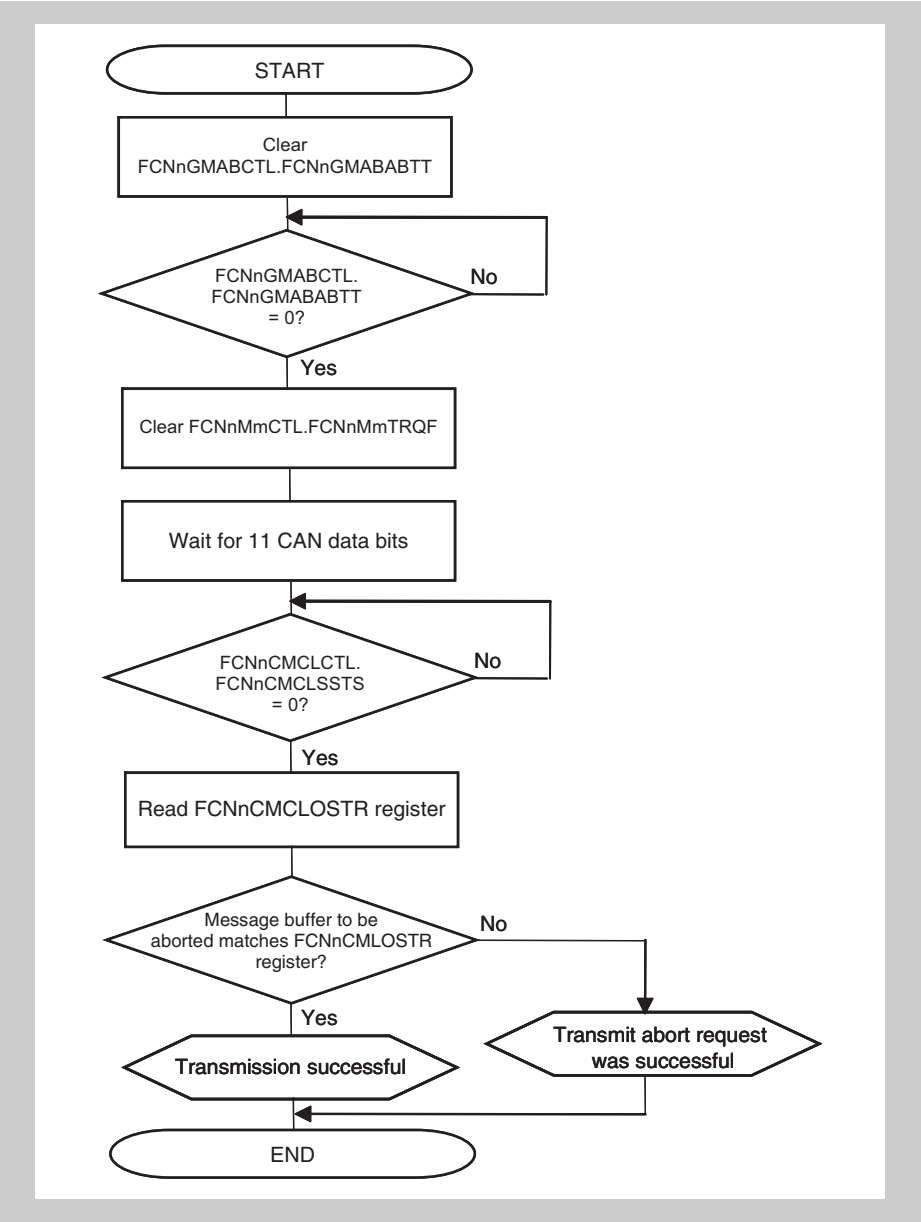


Figure 25-28 Transmission abort processing (normal operation mode with ABT) - Repeat option for aborted message

Cautions

1. Clear FCNnMmCTL.FCNnMmTRQF for aborting transmission request, not FCNnMmCTL.FCNnMmRDYF.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. FCNnCMCLCTL.FCNnCMCLSSTS can be periodically checked by a user application or can be checked after the transmit completion interrupt.
4. Do not execute any new transmission request including in the other message buffers while transmission abort processing is in progress.

Figure 25-29 "Transmission abort processing (normal operation mode with ABT) - No repetition option for aborted message" shows the processing to skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.

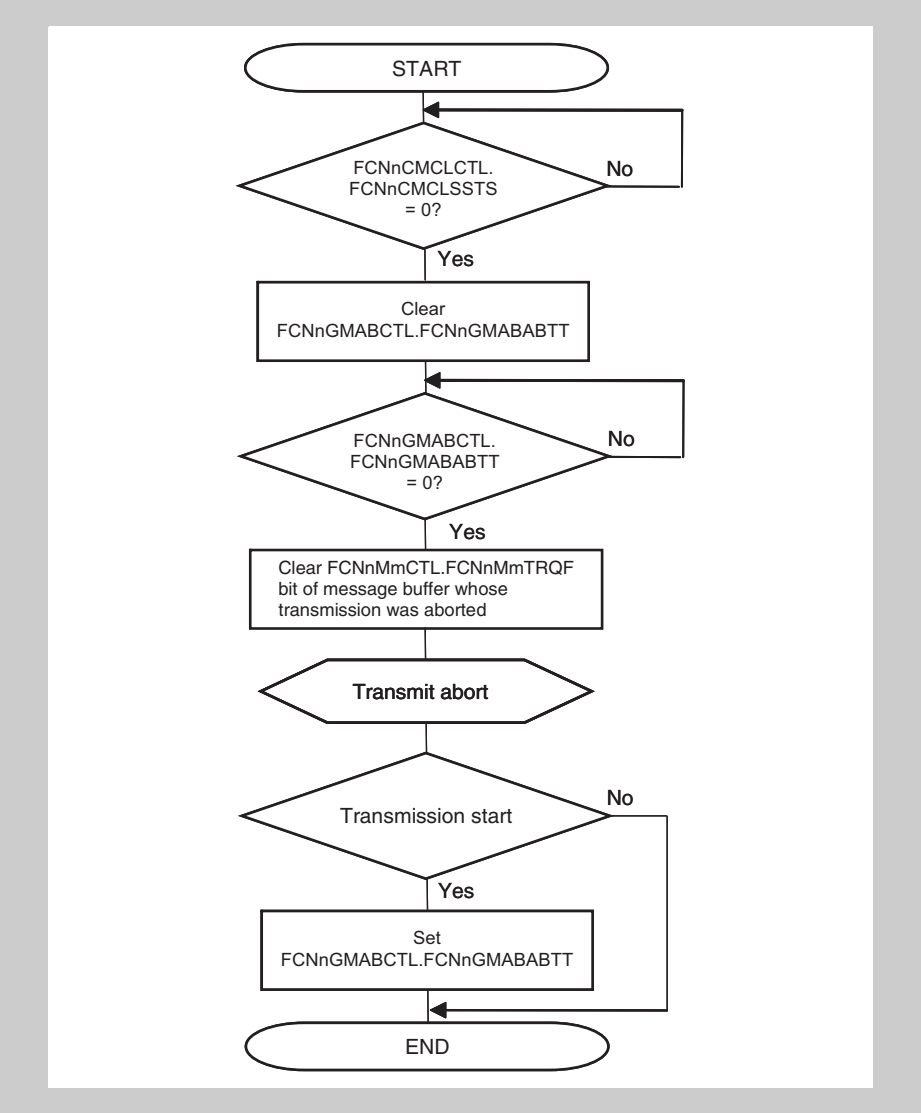


Figure 25-29 Transmission abort processing (normal operation mode with ABT) - No repetition option for aborted message

Cautions

1. Do not set any transmission requests while ABT transmission abort processing is in progress.
2. Make a CAN Controller sleep/stop mode transition request after FCNnGMABCTL.FCNnGMABABTT is cleared (after ABT mode is aborted) following the procedure shown in *Figure 25-29 "Transmission abort processing (normal operation mode with ABT) - No repetition option for aborted message"* or *Figure 25-30 "ABT transmission request abort processing (normal operation mode with ABT)"*. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in *Figure 25-27 "Transmission abort processing (except normal operation mode with ABT)" on page 1999* .

Figure 25-30 "ABT transmission request abort processing (normal operation mode with ABT)" shows the processing to not skip resumption of transmitting a message that was stopped when transmission of an ABT message buffer was aborted.

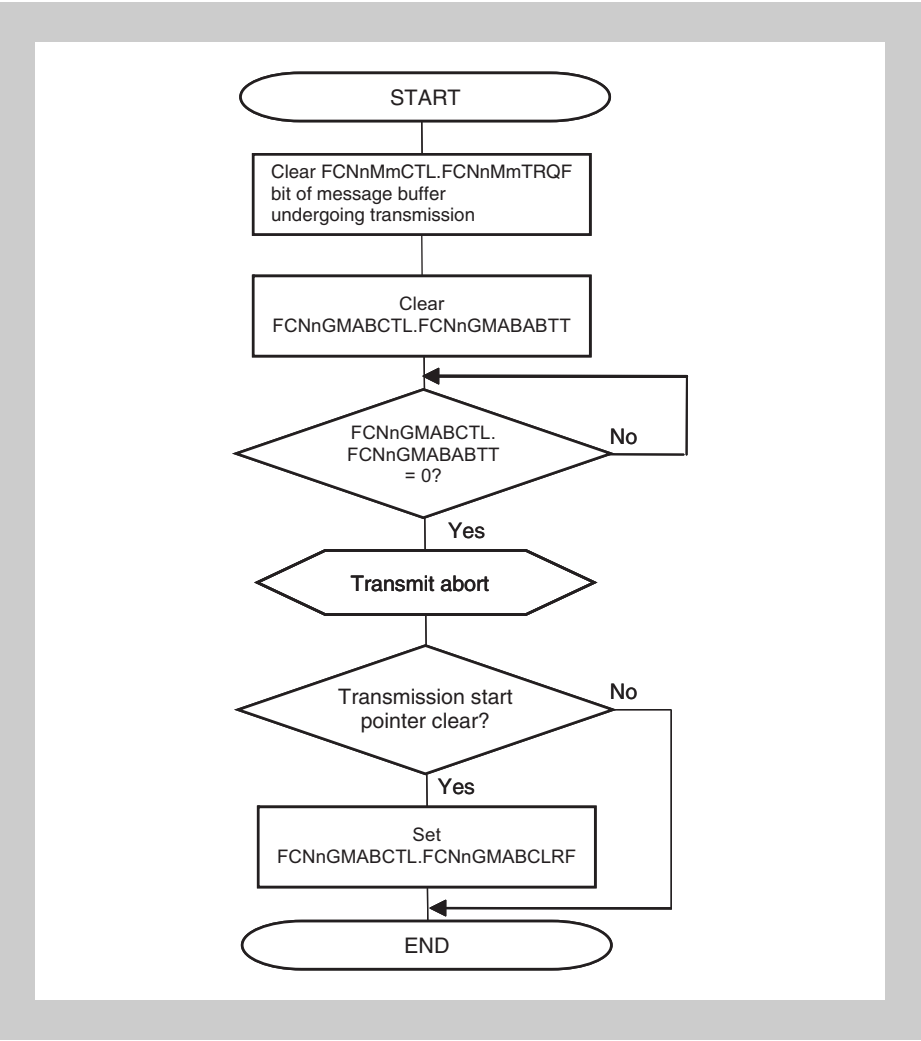


Figure 25-30 ABT transmission request abort processing (normal operation mode with ABT)

Cautions

1. Do not set any transmission requests while ABT transmission abort processing is in progress.
2. Make a CAN Controller sleep/stop mode request after FCNnGMABCTL.FCNnGMABABTT is cleared (after ABT mode is stopped) following the procedure shown in *Figure 25-29 "Transmission abort processing (normal operation mode with ABT) - No repetition option for aborted message"* or *Figure 25-30 "ABT transmission request abort processing (normal operation mode with ABT)"*. When clearing a transmission request in an area other than the ABT area, follow the procedure shown in *Figure 25-27 "Transmission abort processing (except normal operation mode with ABT)" on page 1999* .

Figure 25-31 "ABT transmission request abort processing (normal operation mode with ABT) with transmit abort interrupt flag" shows the processing on ABT mode, when using the Transmit Abort functionality (Transmit Abort Flag). The box "Transmission Abort Success" represents the checking of the transmission abort success by checking the FCNnMmTCPF flag within the ABT message buffers.

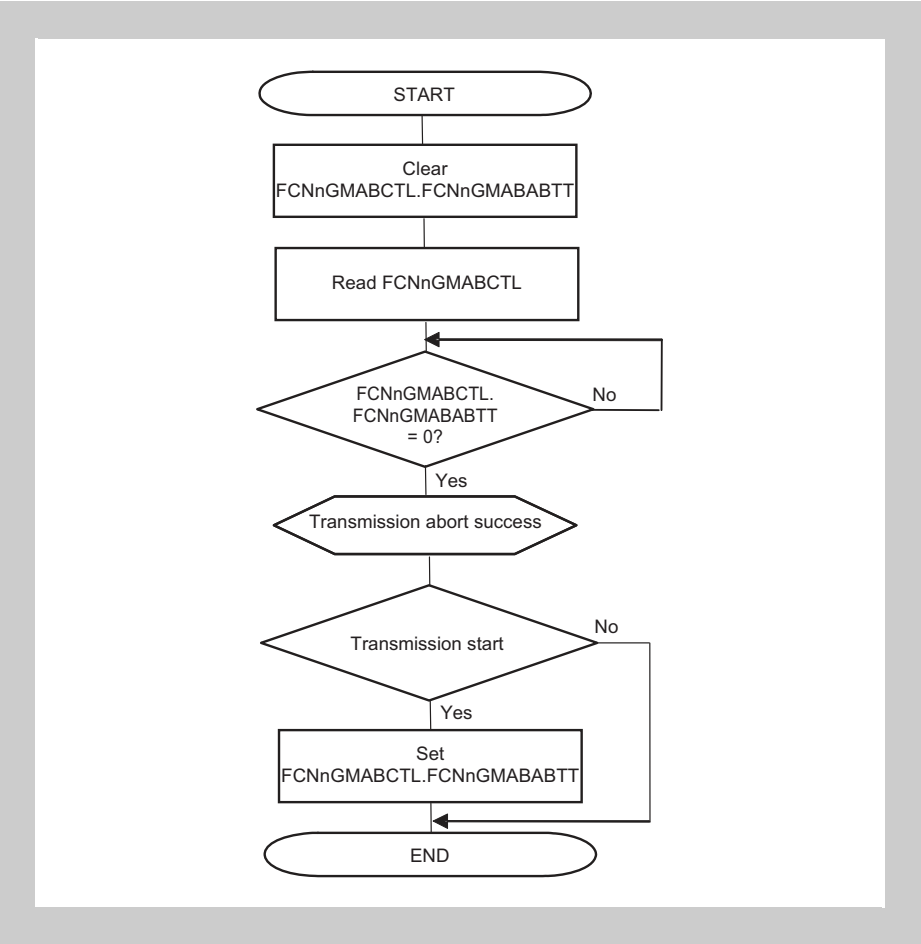


Figure 25-31 ABT transmission request abort processing (normal operation mode with ABT) with transmit abort interrupt flag

Cautions

1. Do not set any transmission requests while ABT transmission abort processing is in progress.
2. Issue a CAN sleep mode/CAN stop mode request only, after ABTTRG is cleared (and after ABT mode has stopped) following the procedure shown above. When clearing a transmission request in an area other than the ABT area, follow the procedures shown in flowcharts of aborting transmit requests in operation modes except ABT mode.

Note There is the case that all ABT is transmitted completely even if transmission abort interrupt is occurred. Then it is possible to know which message is finished transmission.

Figure 25-32 "Transmission request abort processing (except normal operation mode with ABT) with transmit abort interrupt and transmission completely finished flag" shows the processing when using the Transmit Abort functionality (Transmit Abort Interrupt).

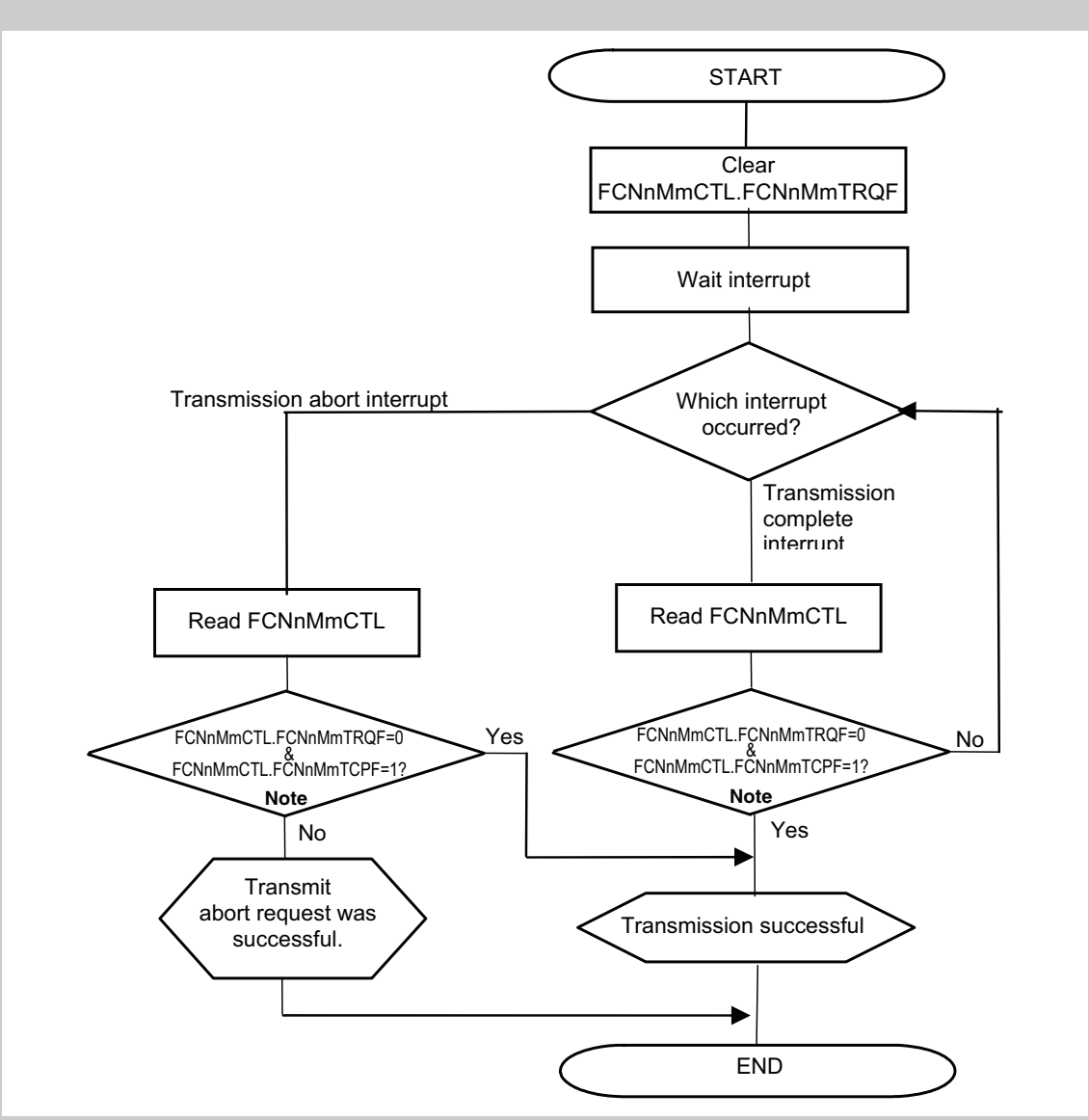


Figure 25-32 Transmission request abort processing (except normal operation mode with ABT) with transmit abort interrupt and transmission completely finished flag

Note Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.

Cautions

1. Execute transmission request abort processing by clearing the TRQ bit, not the RDY bit.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. Do not execute another new transmission request in this or the other message buffers, while transmission abort processing is in progress.

4. It is prohibited to clear the transmit request flags of other messages, while transmission abort is in progress.
5. If a new transmission request is executed for a message buffer within 400 clocks of the CAN Controller, after transmission abort process and before IFS (Inter-Frame Space), that message might be transmitted in the next following transmission, although its ID priority was low.

Note Even if the TRQ bit is cleared, there is the possibility that the transmission still starts in the following 11 bits of interframe space (3bit) and suspend transmission (8bit), because the transmission request has already been received by protocol layer.

Figure 25-33 "Transmission request abort processing (except normal operation mode with ABT) with transmission completely finished flag" shows the processing when using the Transmit Abort functionality (Transmission Completely Finished Flag FCNnMmTCPF).

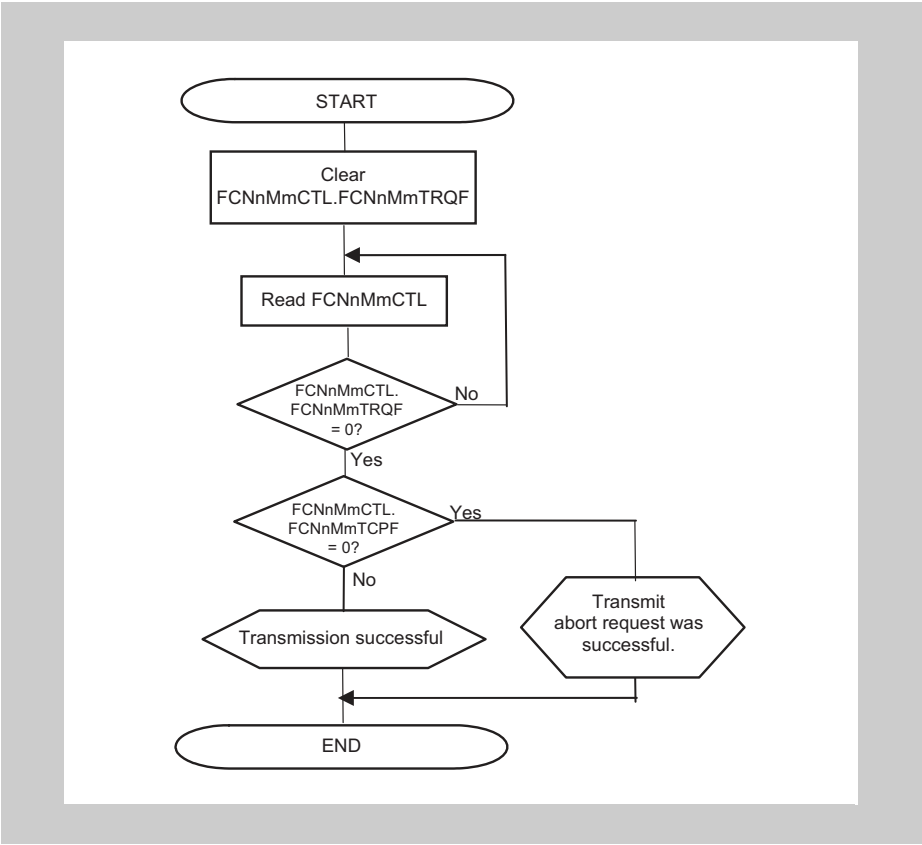


Figure 25-33 Transmission request abort processing (except normal operation mode with ABT) with transmission completely finished flag

Cautions

1. Execute transmission request abort processing by clearing the TRQ bit, not the RDY bit.
2. Before making a sleep mode transition request, confirm that there is no transmission request left using this processing.
3. Do not execute another new transmission request in this or the other message buffers, while transmission abort processing is in progress.
4. It is prohibited to clear the transmit request flags of other messages, while transmission abort is in progress.
5. If a new transmission request is executed for a message buffer within 400 T_{CANCH} of the CAN Controller, after transmission abort process and before IFS (Inter-Frame Space), that message might be transmitted in the next following transmission, although its ID priority was low.

Note Even if the TRQ bit is cleared, there is the possibility that the transmission still starts in the following 11 bits of interframe space (3bit) and suspend transmission (8bit), because the transmission request has already been received by protocol layer.

25.16.3 Message reception

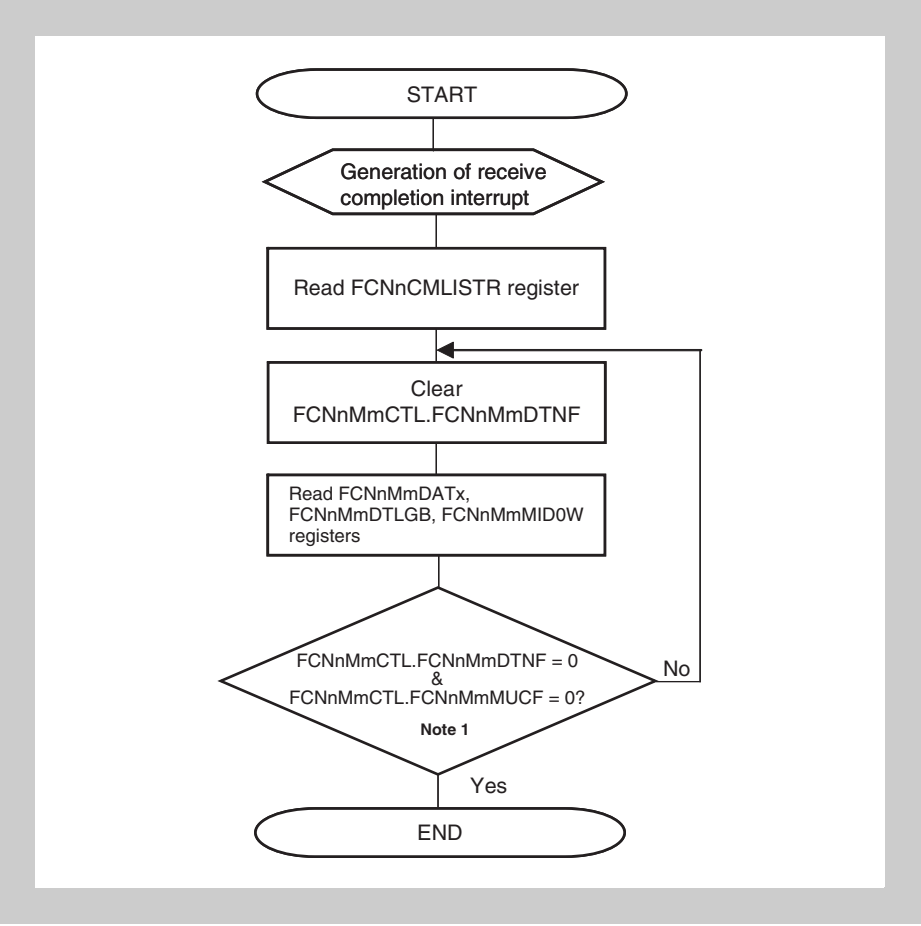


Figure 25-34 Reception via interrupt (using FCNnCMLISTR register)

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing RX interrupts.

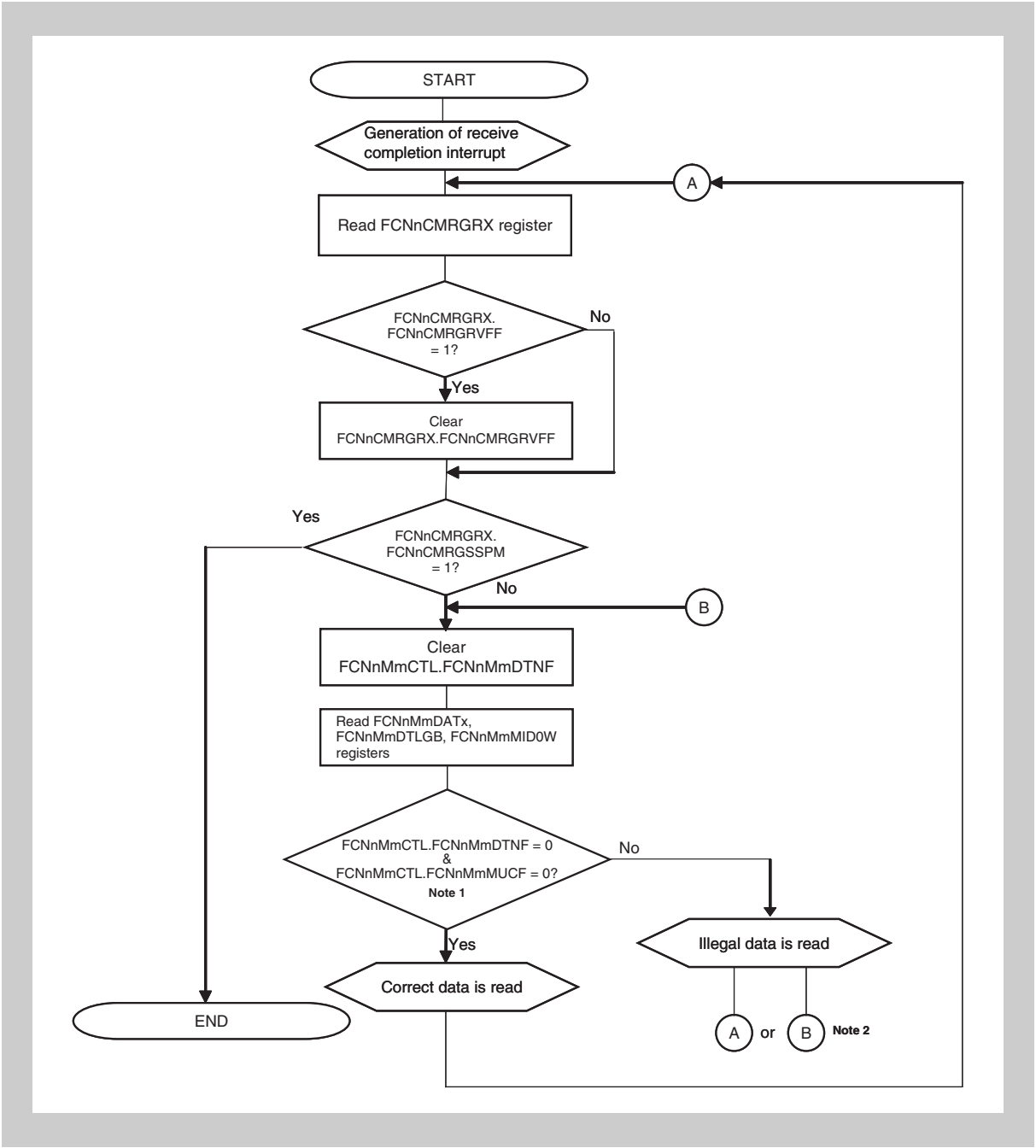


Figure 25-35 Reception via interrupt (using FCNnCMRGRX register)

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Depending of the processing target of the application, two ways are possible:
   – Way A: The message is not processed within this pass, but with the next pass, depending on the timing this can happen latest with the next Receive Interrupt.
   Other messages will be processed earlier.
   – Way B: The message is processed within this pass, the loop waits on this message.
   Other messages will be processed later.

3. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
It is recommended to cancel any sleep mode requests, before processing RX interrupts.
4. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.

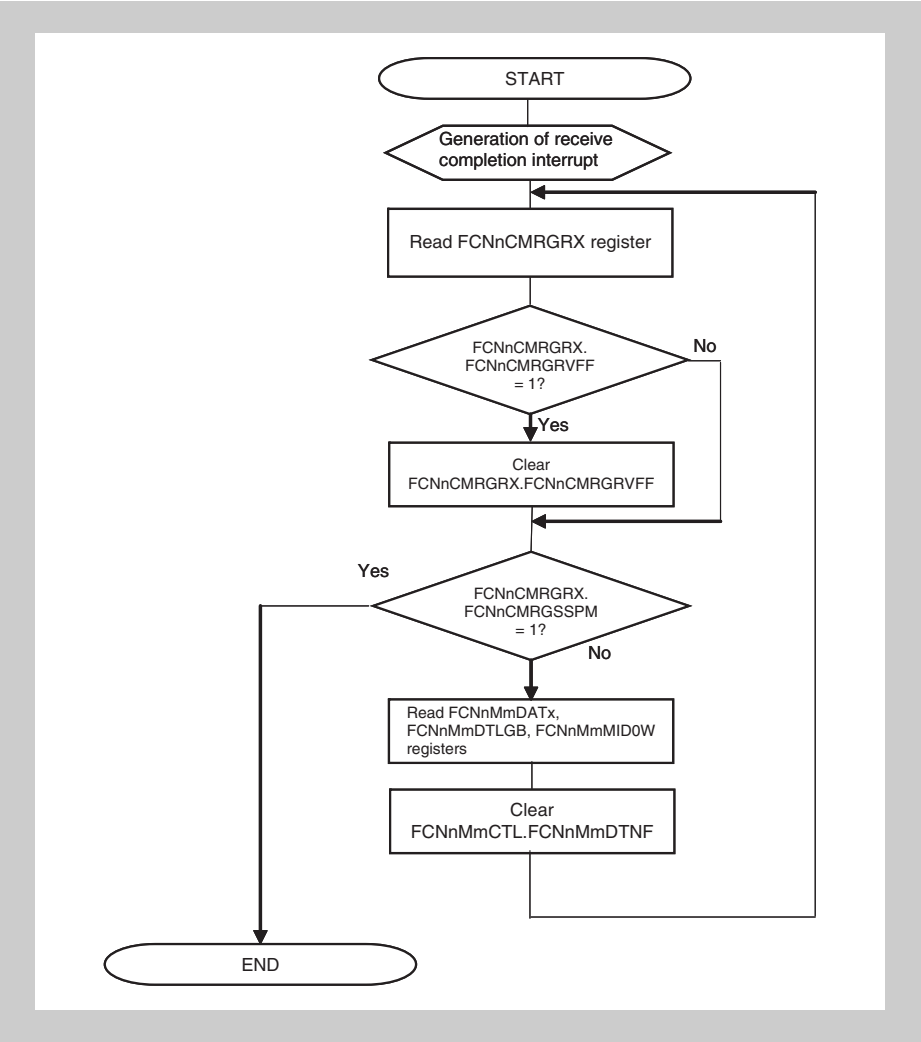


Figure 25-36 Reception via interrupt (using FCNnCMRGRX register), alternative way

Notes

1. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the interrupt routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
   It is recommended to cancel any sleep mode requests, before processing RX interrupts.
2. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.
3. This flow will not provide most recently received data for the application. However, due to less effort on processing, it reduces interrupt load.
4. The overwrite function (FCNnMmSTRB.FCNnMmSSOW=1) must not be used with this flow - data inconsistency could occur.
5. It can be used alternatively to *Figure 25-35 “Reception via interrupt (using FCNnCMRGRX register)” on page 2008* .

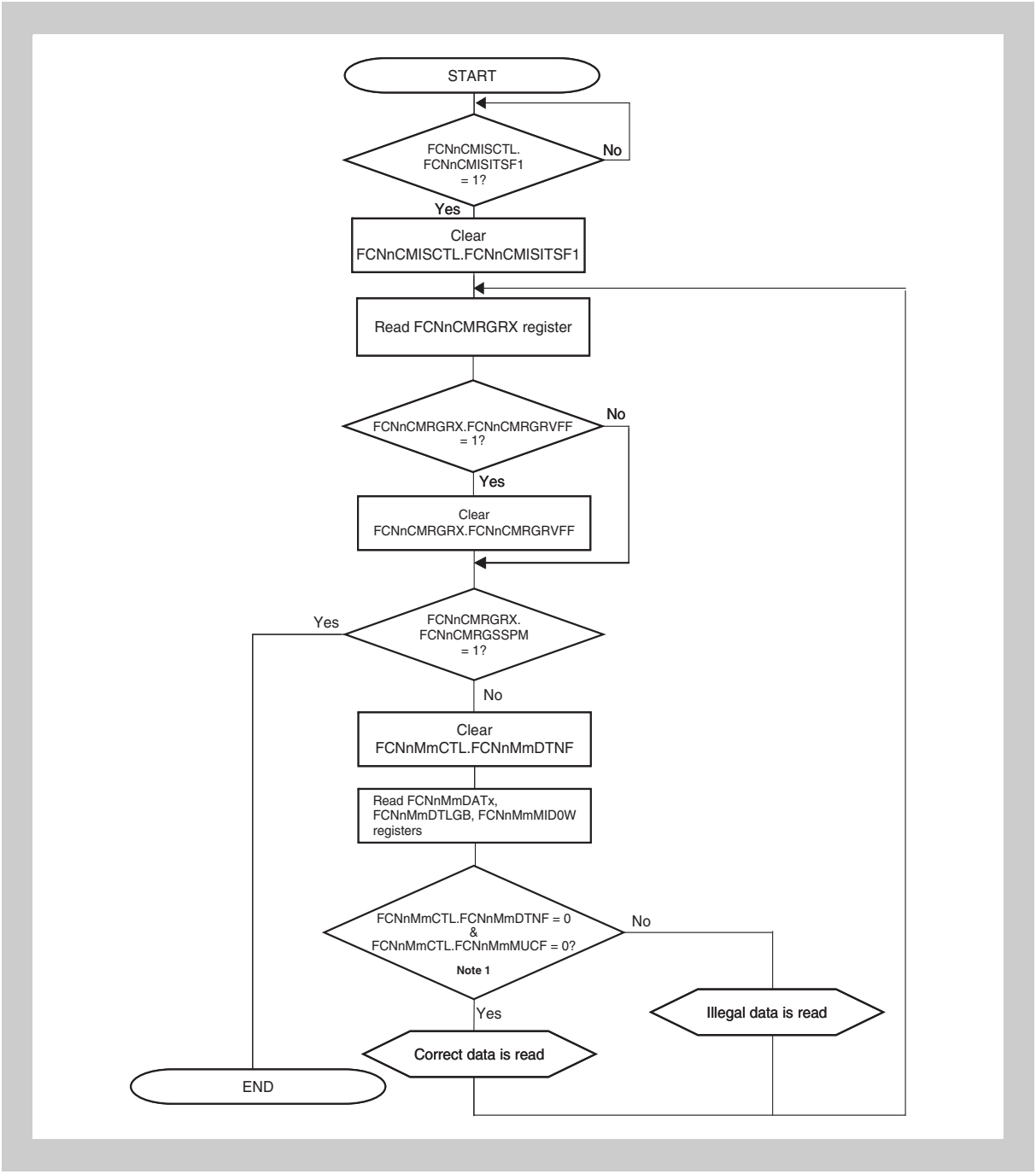


Figure 25-37 Reception via software polling

Notes

1. Check FCNnMmCTL.FCNnMmMUCF and FCNnMmCTL.FCNnMmDTNF bits using one read access.
2. Also check the FCNnGMCLSSMO flag at the beginning and at the end of the polling routine, in order to check the access to the message buffers as well as reception history list registers, in case a pending sleep mode had been executed. If FCNnGMCLSSMO is detected to be cleared at any check, the actions and results of the processing have to be discarded and processed again, after FCNnGMCLSSMO is set again.
3. If FCNnCMRGRX.FCNnCMRGRVFF was set once, the receive history list is inconsistent. Consider to scan all configured receive buffers for receptions.

25.16.4 Power save modes

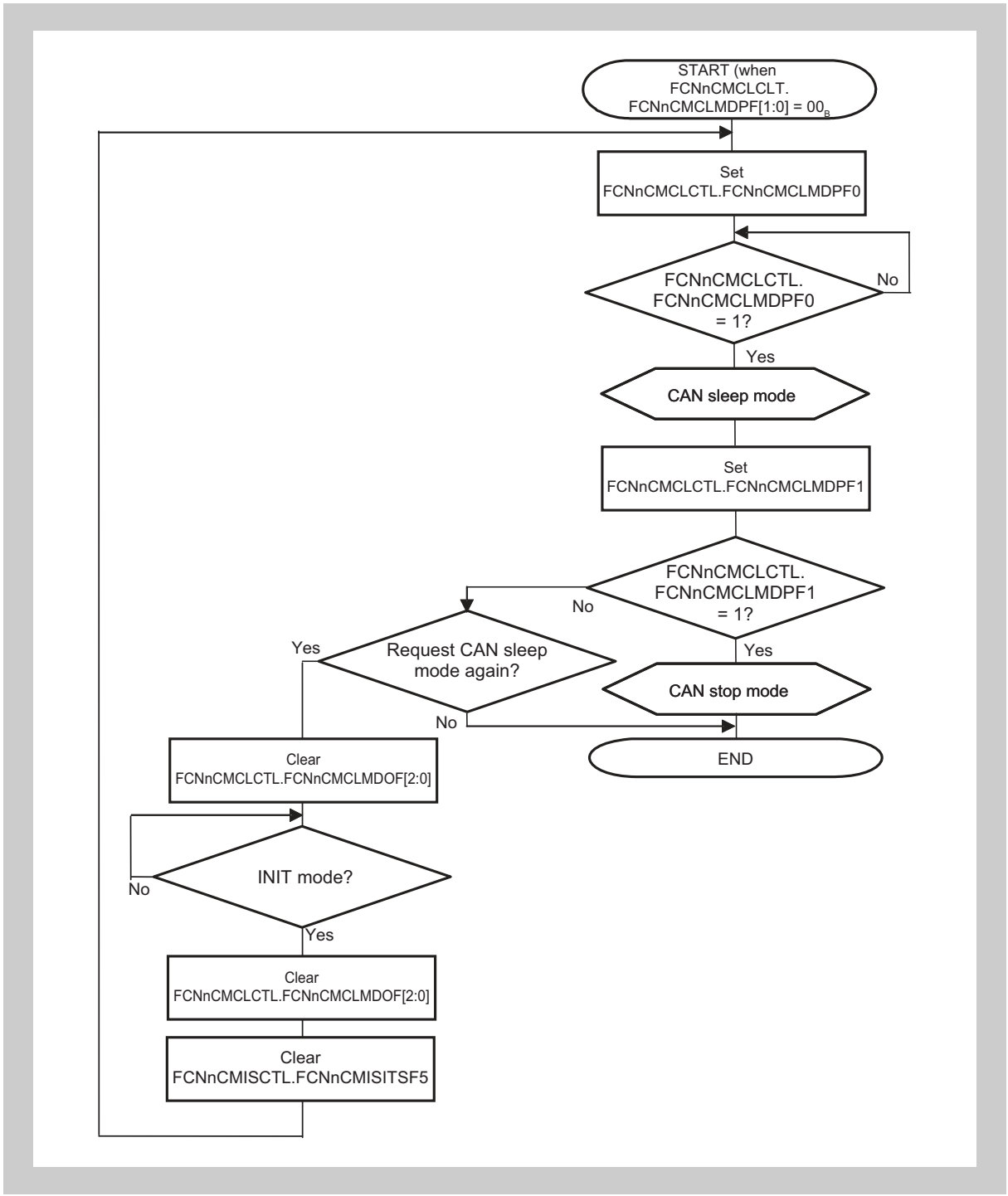


Figure 25-38 Setting CAN Controller sleep mode/stop mode

Caution To abort transmission before making a request for the CAN Controller sleep mode, perform processing according to previously given flowcharts.

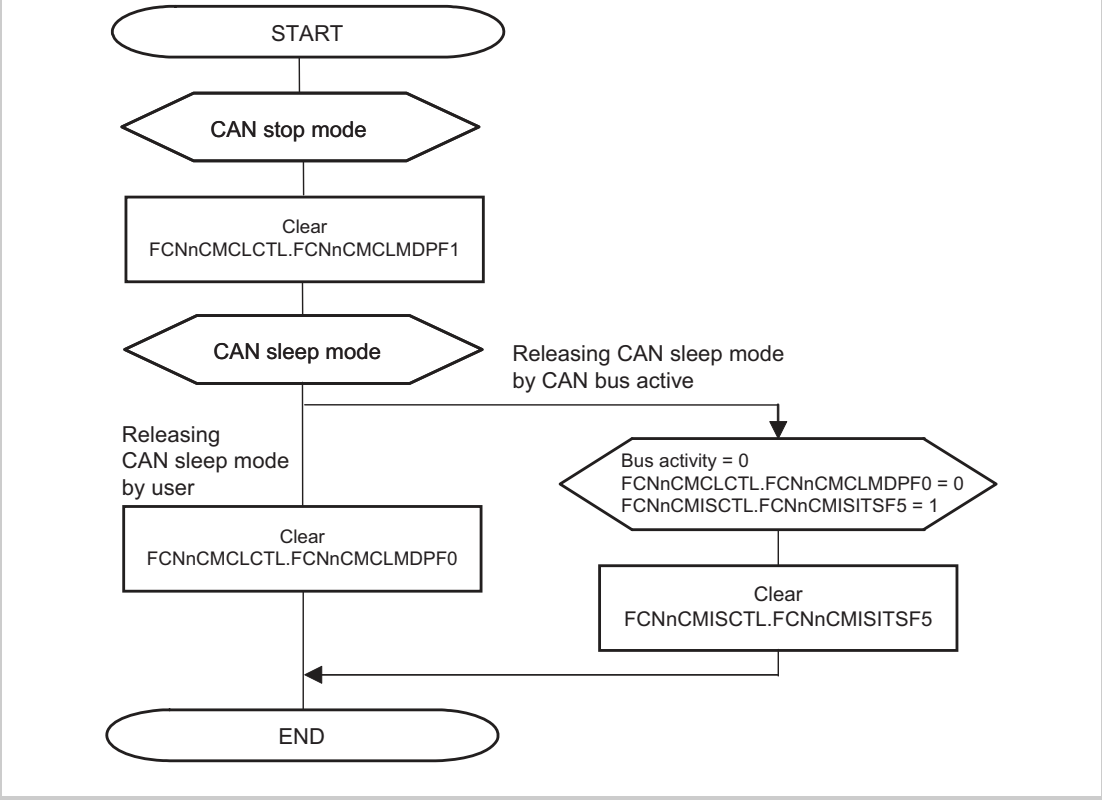


Figure 25-39 Clear CAN Controller sleep/stop mode

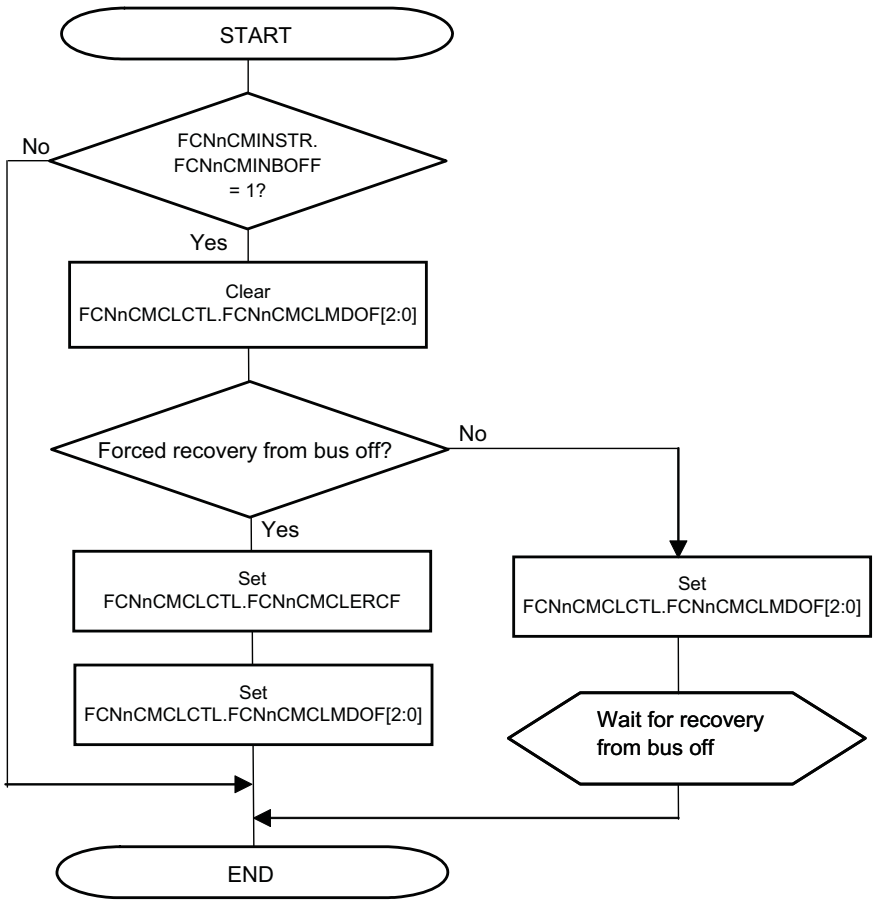


Figure 25-40 Bus-off recovery (except normal operation mode with ABT)

Caution When the transmission from the initialization mode to any operation modes is requested to execute bus-off recovery sequence again in the bus-off recovery sequence, reception error counter is cleared.
Therefore it is necessary to detect 11 consecutive recessive-level bits 128 times on the bus again.

Note Operation mode: Normal operation mode, normal operation mode with ABT, receive-only mode, singleshot mode, self-test mode.

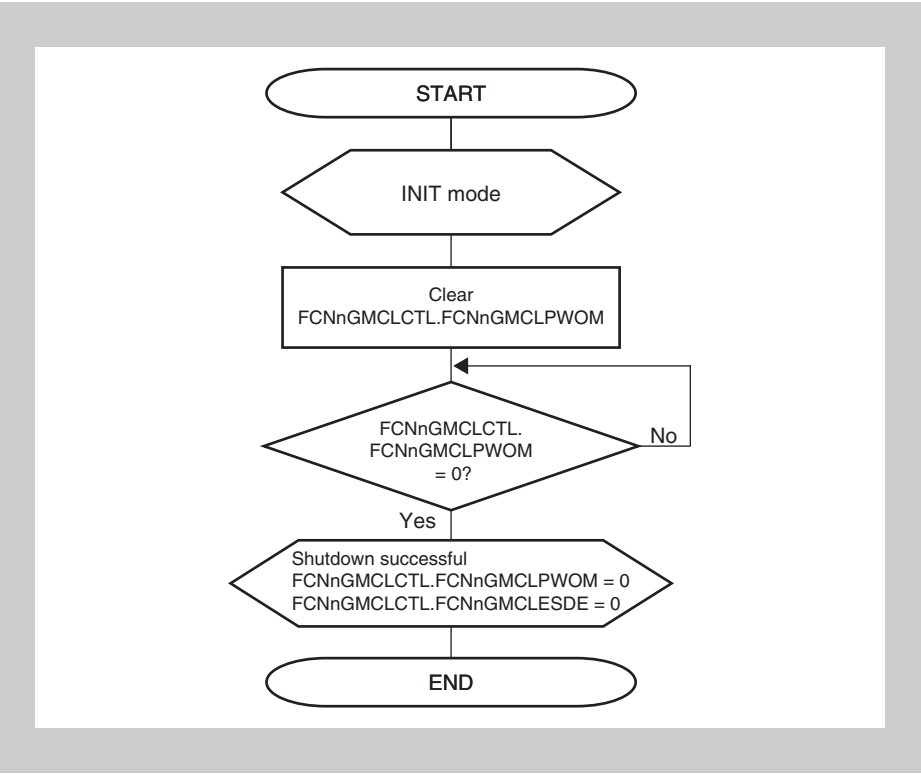


Figure 25-41 Normal shutdown process

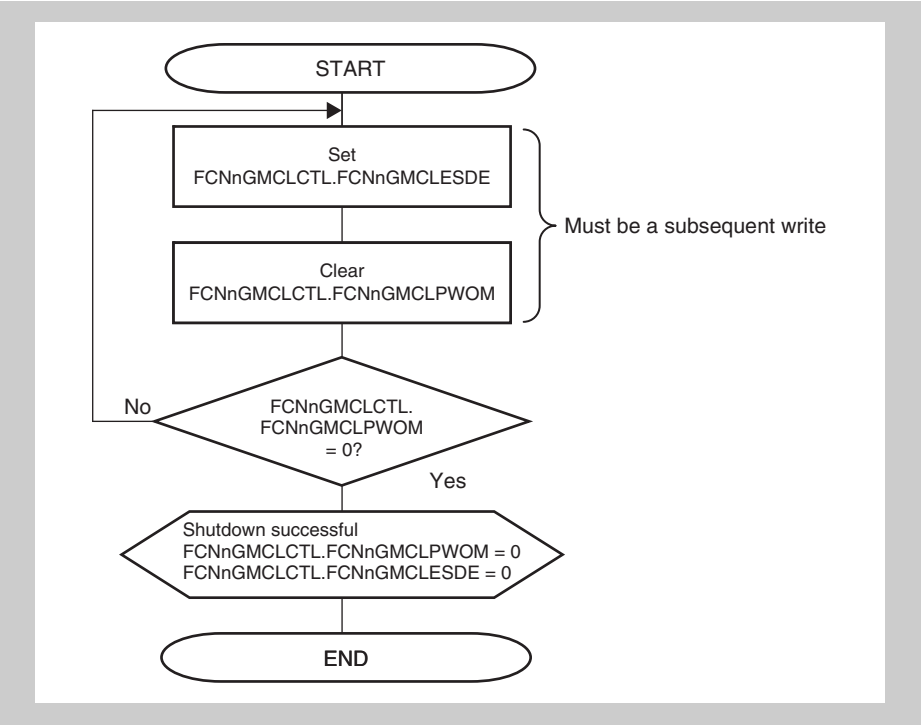


Figure 25-42 Forced shutdown process

Caution Do not read- or write-access any registers by software between setting the FCNnGMCLESDE bit and clearing the FCNnGMCLPWOM bit.

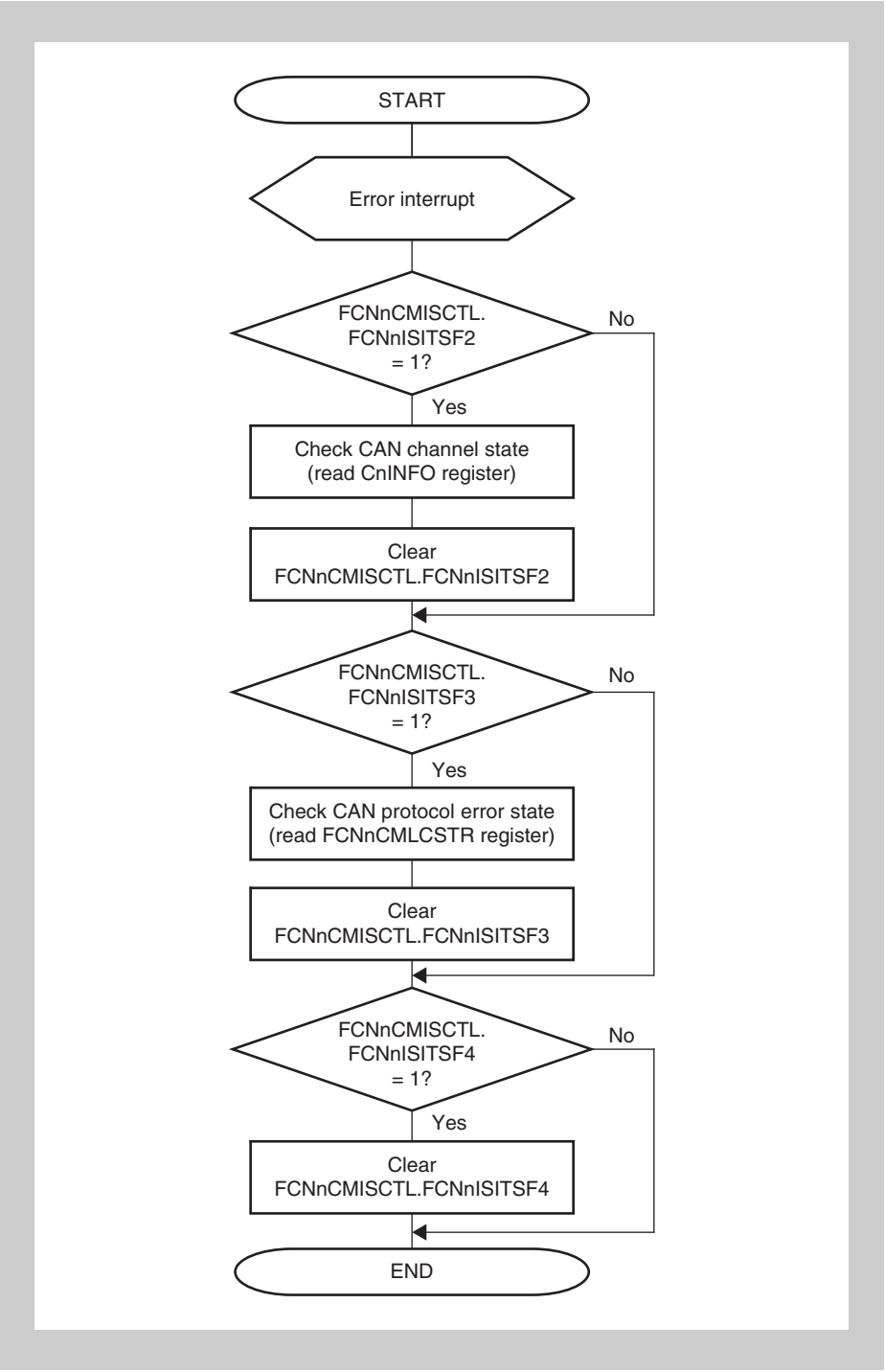


Figure 25-43 Error handling

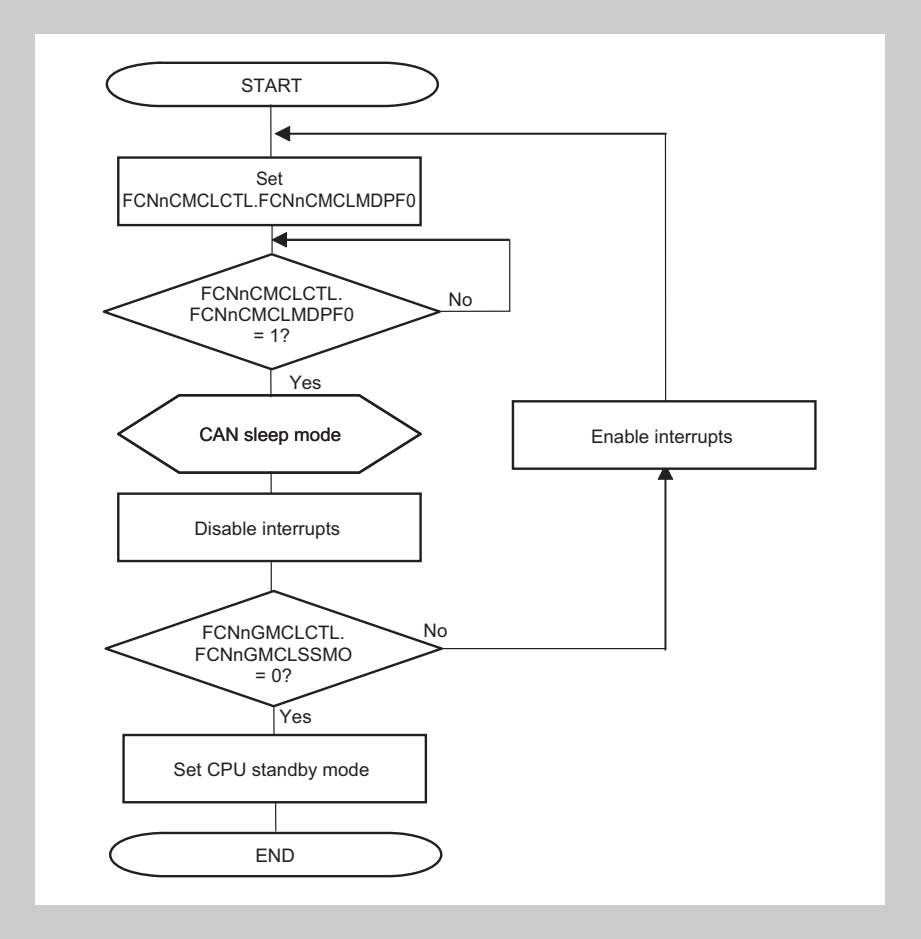


Figure 25-44 Setting CPU stand-by (from CAN Controller sleep mode)

Notes

1. Before the CPU is set in the CPU standby mode, please check if the CAN Controller sleep mode has been reached.
   However, after check of the sleep mode, until the CPU is set in the CPU standby mode, the sleep mode may be cancelled by wakeup from CAN bus.
2. There is a possibility, that between the check of FCNnGMCLSSMO = 0 and setting of the CPU standby mode a wake up condition on the CAN bus occurs. In that case the CAN Controller releases the SLEEP mode, the FCNnCMISITSF5 bit is set and if enabled the wake up interrupt will be generated.

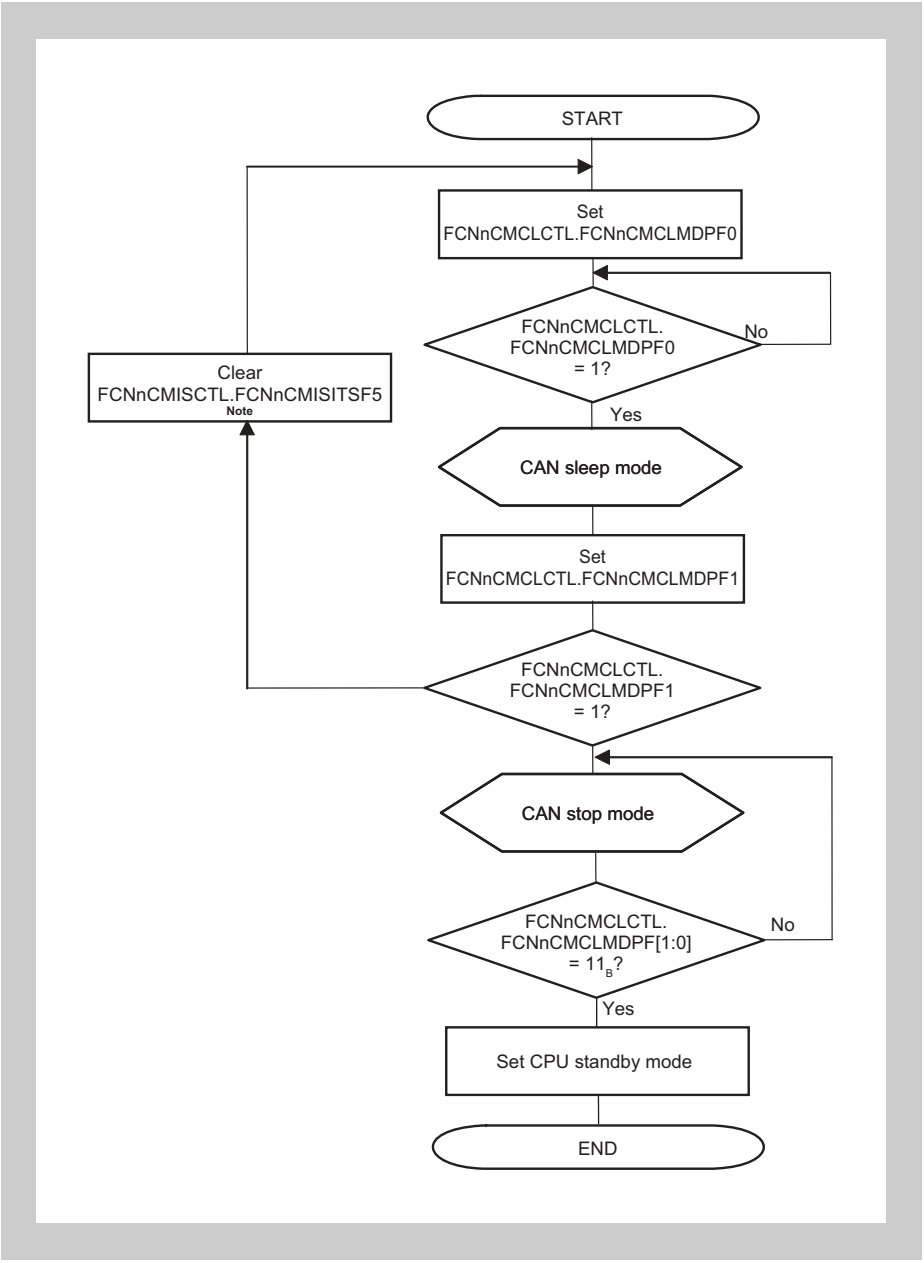


Figure 25-45 Setting CPU stand-by (from CAN Controller stop mode)

Caution The CAN Controller stop mode can only be released by setting 01_B to FCNnCMCLCTL.FCNnCMCLMDPF[1:0] and not by a change in the CAN Controller bus state.

Chapter 26 Diagnostic CAN Controller (DCN)

The microcontroller features on-chip CAN (Controller Area Network) controllers with diagnostic features that complies with the CAN protocol as standardized in ISO 11898.

This chapter contains a generic description of the Diagnostic CAN Controller (DCN).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

26.1 V850E2/Fx4-H DCN Features

Instances

This microcontroller has the following number of instances of the Diagnostic CAN Controller.

Table 26-1 Instances of DCN

Diagnostic CAN Controller	
Instance	1
Name	DCN0

Instances index n

Throughout this chapter, the instance of a Diagnostic CAN Controller is identified by the index "n" (n = 0), for example, DCNnGMCLCTL for the DCNn control register.

Message buffers index m

Throughout this chapter, the DCN message buffer registers are identified by “m” (for m refer to the table below), for example DCNnMmDAT4B for DCN instance n, message data byte 4 of message buffer register m. The number of message buffers for each instance of DCN is given in the following table:

Table 26-2 Message buffers of DCNn

DCNn instance	Number m of message buffers
DCN0	128

Register addresses

All Diagnostic CAN Controller register addresses are given as address offsets to the individual base address <DCNn_base>.
The <DCNn_base> address of each DCNn is given in the following table:

Table 26-3 Register base addresses <DCNn_base>

DCNn instance	<DCNn_base> address
DCN0	FF52 0000_H

Clock supply All Diagnostic CAN Controllers provide one clock input:

Table 26-4 DCNn clock supply

DCNn instance	DCNnn clock	Connected to
DCN0	PCLK	Clock Controller CKSCLK_115

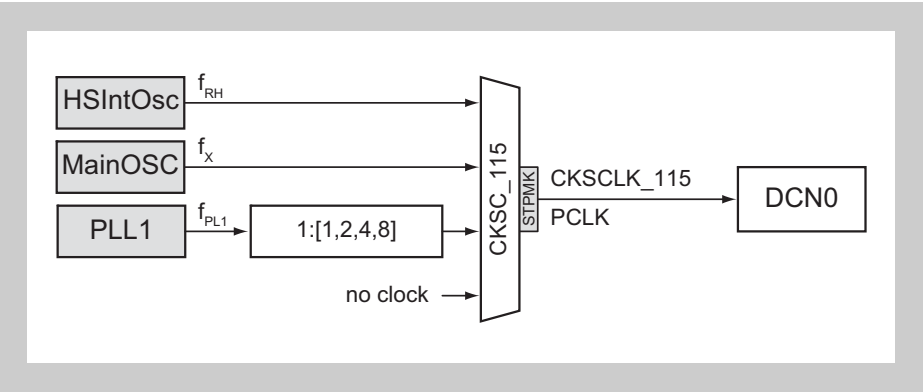


Figure 26-1 DCN clock supply

Interrupts and DMA The Diagnostic CAN Controller can generate the following interrupt and DMA/DTS requests:

Table 26-5 DCNn interrupt and DMA/DTS requests

DCNn signals	Function	Connected to
INTC0ERR	DCN0 error indication	Interrupt Controller INTDCN0ERR
INTC0REC	DCN0 reception completion	Interrupt Controller INTDCN0REC DMA Controller trigger 84 DTS Controller trigger 15
INTC0TRX	DCN0 transmission completion	Interrupt Controller INTDCN0TRX
INTC0WUP	DCN0 sleep wake-up / transmission abortion	Interrupt Controller INTFCNWUP[a]

a) The Interrupt Controller input INTFCNWUP is a combination of the wake-up interrupts of the Diagnostic CAN Controller DCN0 and the CAN Controllers FCN0 to FCN4. Thus any of these wake-up interrupts asserts the interrupt request INTFCNWUP.

DCN H/W reset The Diagnostic CAN Controllers and their registers are initialized by the following reset signal:

Table 26-6 DCNn reset signal

DCNn	Reset signal
DCN0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the Diagnostic CAN Controller are listed in the table below.

Table 26-7 CAN Controllers I/O signals

DCNn signals	Function	Connected to
CRXD0	DCN0 CAN bus receive input	Port FCN5RX[a]
CTXD0	DCN0 CAN bus transmit output	Port FCN5TX

a) This signal can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

26.2 Introduction

The DCN (Diagnostic CAN Controller) module is in first place an ordinary single channel CAN Controller module with 128 message buffers. In difference to the single channel CAN Controller module the DCN module features two CAN channels. The channel, which can be operated as an ordinary CAN interface, is called diagnostic channel (DIAG_CH).

- For specific diagnostic purposes of the application a second CAN interface is available. This second channel has only limited operational modes because its main purpose is to listen only to the data traffic of one of the other single channel CAN Controller modules. This channel is called listen-only channel (RXONLY_CH). It provides the mirror function, which transfers the messages from the source CAN bus (monitored channel) to the diagnostic channel from where these messages are automatically sent onto the CAN bus. Furthermore, it provides the mirror mode with the Transfer ID Filter function (with TIF). This mode can transmit particular ID only from the source CAN channel to destination channel automatically.

The mirror function as part of a diagnosis concept including other CAN channels is shown in *Figure 26-2 “Diagnosis concept using DCN and 5 other CAN channels”*.

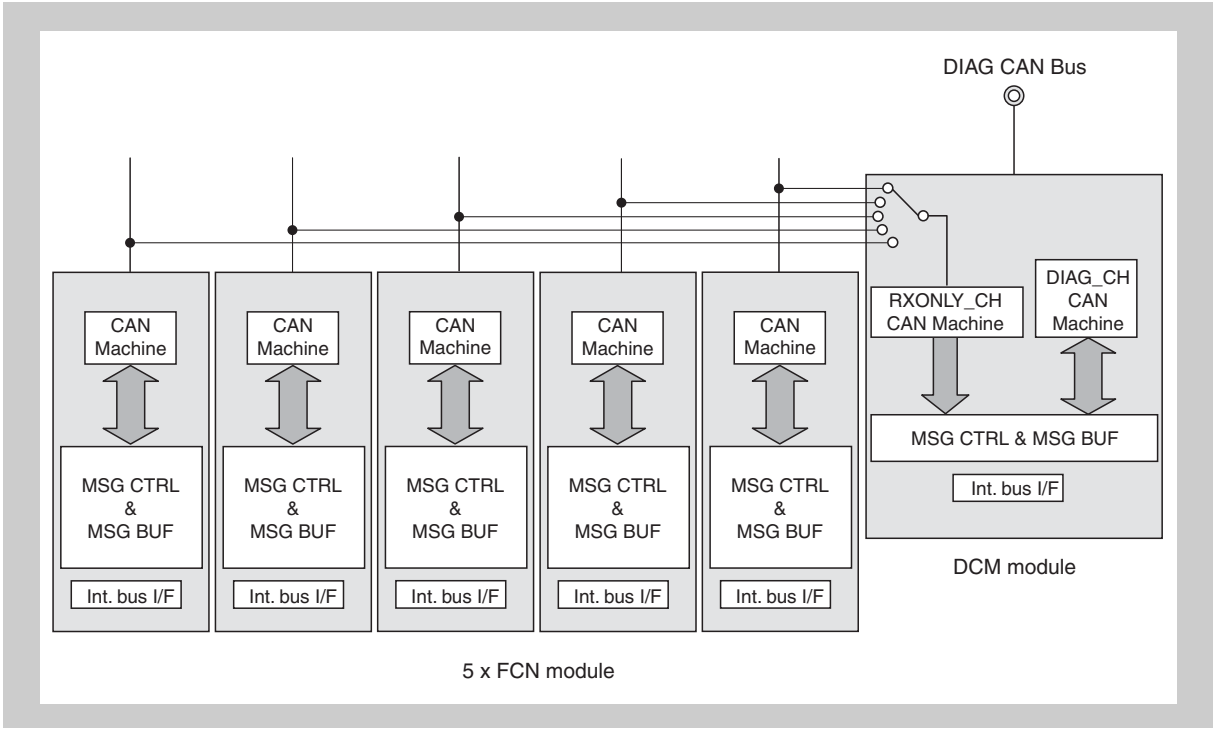


Figure 26-2 Diagnosis concept using DCN and 5 other CAN channels

The RXONLY_CH and the DIAG_CH have independent control registers for power save and operational modes. The combination of both affects the message buffer allocation for the RXONLY_CH. When RXONLY_CH is in power save mode STOP or in operational mode INIT, the upper 16 message buffer are assigned to the DIAG_CH. This offers the opportunity to use the DCN module as an ordinary CAN channel with 128 message buffers when monitoring of other CAN channels is not necessary.

26.3 Overview of Functions

Table 26-8 Outline of DCN module functions

Feature	Details
Protocol	Active support of extended frame format (international standard ISO11898)
Channels	RXONLY_CH, DIAG_CH
Baud rate	Max. 1Mbit/s @ $f_{CAN} \geq 16$ MHz (min. 16 CANCLK/bit)
Message Storage	RAM area with shared access (Accessing entities: CPU, RXONLY_CH CAN, DIAG_CH CAN) The DCN module features 128 message buffers. The upper 16 buffer are assigned to the RXONLY_CH depending on operational and power save mode.
Message Organisation	Each message buffer can be initialised either to be a transmit message buffer or a receive message buffer. A group of message buffers assigned as receive buffer can form a multi-buffer receive block. A group of message buffers assigned as transmit buffer can be used for automatic block transfer.
Masks (DIAG_CH only)	The DIAG_CH features 8 masks. Each mask can be assigned to each message buffer or a receive message buffer. There is no difference between global and local masks within one CAN module. Mirror mode does not provide any masks. All messages are copied to DIAG_CH. Mirror mode with TIF of the RXONLY_CH provides 8 reference transfer ID registers (DCNnTIDRTXx) and 1 mask transfer ID register (DCNnTIDMTXx) for filtering. Only message matching the filter criteria are stored in the upper 16 message buffers. Only those messages stored here participate in the Mirror Mode.
Application Support for Message Handling	The DIAG_CH provides a fast mechanism to find receive message buffers with updated content (Reception History List). The DIAG_CH provides a fast mechanism to find the transmit message buffers, from which a message frame has been sent (Transmission History List). The RXONLY_CH does not provide these history lists.
Remote Frame Support	Remote frame handling by message buffer defined for transmit and receive.
Time Stamp Functions [a] (DIAG_CH only)	Time stamp upon message reception Trigger for time stamp capture selectable (SOF or EOF detection in a CAN message frame).
Diagnostics	The DCN module features a mirror function. Messages received by RXONLY-CH are automatically sent on the DIAG_CH. The RXONLY-CH input can be switched to any of the other CAN channels of the device. Readable error counters Valid protocol activity flag for verification of bus connection “Receive-only Mode” CAN protocol error type decoding “Self-test Mode” (DIAG_CH only)
Power Save Modes	Sleep Mode: Wake up from CAN bus Stop Mode: No wake up from CAN bus The RXONLY_CH CAN I/F module and the DIAG_CH CAN I/F module can be configured independently to all these power save modes respectively. The whole module consumes minimum power when both CAN I/F modules are in such modes simultaneously.

a) The architecture of the DCN module is prepared for the integration of a TTCAN module, but not all DCN module derivatives are equipped with a TTCAN Module.

26.4 Architecture

The DCN module is connected to a host CPU and provides two CAN bus interfaces. The DIAG_CH CAN I/F module is dedicated for a particular CAN bus, the diagnostic CAN bus. The RXONLY_CH CAN I/F module can be connected to several CAN buses via a programmable selector. The selector is located within the DCN module and provides connectivity for up to 8 sources.

The figure below shows an example of DCN module with 8 inputs for other CAN buses and 128 message buffers.

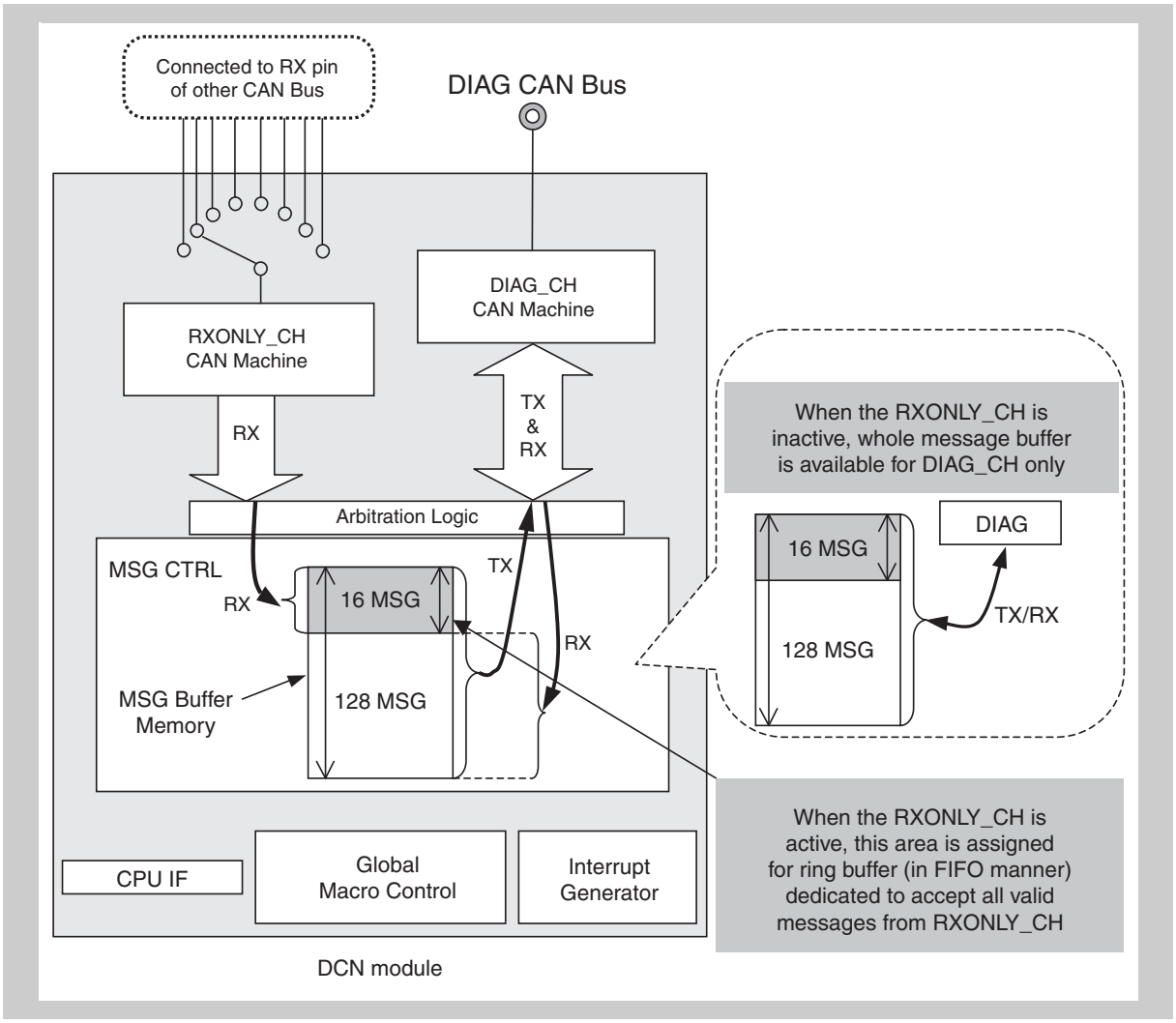


Figure 26-3 Architecture of DCN module

The DCN module contains several sub-blocks that are described in the following chapters.

26.4.1 CPU interface

The CPU I/F is the interface of the entire module to the bus system of the microcontroller. The CPU I/F enables the CPU to access the control and status registers of all sub-blocks within the module directly. The addresses of the control and status registers are mapped into the memory map of the microcomputer system.

The interrupt signals from the CAN Interrupt Generator to the Interrupt Controller of the microcomputer system are routed via the CPU I/F.

The module clock f_{CAN} is connected to a peripheral module clock of the clock generator of the microcomputer system.

26.4.2 Global module control

The Global Module Control sub-block contains the logic that enables or disables the entire module.
As well the module clock f_{CANMOD} is set here.

26.4.3 CAN interrupt generator

In the CAN interrupt generator all interrupt request signals from the CAN I/F channels are collected. Appropriated signal shaping is applied to those interrupt request signals for a proper connection to the interrupt request signal inputs of the interrupt controller in the microcomputer system (i.e. pulse generation).

26.4.4 Message control (MSG Ctrl)

The MSG Ctrl sub-block provides the memory for the message buffers and its control.

The upper 16 message buffers have no static assignment to RXONLY_CH or DIAG_CH. The assignment of the message buffers depends on the status of RXONLY_CH. The basic functions of these message buffers are:

- When assigned to the RXONLY_CH:
  - Receive all data frames from RXONLY_CH without any acceptance filtering when Mirror mode or RECONLY mode is selected. In case of the Mirror mode with TIF, data/remote frames are filtered by mask register and reference ID registers.
  - Receive all remote frames from RXONLY_CH without any acceptance filtering.
  - Behave as ring buffer in FIFO manner. Storage of received messages is calculated with help of LIPT of RXONLY_CH.
  - Automatically send all messages to DIAG_CH in FIFO manner. The TX-search state machine of the DIAG_CH takes care to send these message on the diagnostic bus.
    Any TRQ by a FIFO candidate needs to start a TX-search, which always starts at message buffer #0. The TX-search always must be applied to both areas, the DIAG_CH message buffers and the FIFO.

- When assigned to the DIAG_CH:
  - Send data frames to DIAG_CH
  - Receive data frames from DIAG_CH
  - Receive remote frames from DIAG_CH
  - Send remote frames to DIAG_CH
  - Each message buffer contains its own identifier. Therefore a message buffer can be reserved to receive a data frame unambiguously or to transmit only a particular message frame.
  - A mask can be assigned to a message buffer in case the reception of a group of data frames is required.
  - Furthermore, message buffers can be grouped to build multi-buffer receive blocks (MBRB).

In other words, the complete message buffer memory (128 buffers) are fully assigned to DIAG_CH with all related features of an ordinary CAN Controller channel.

The lower message buffers (#0 to #111) are always assigned to the DIAG_CH.

26.4.5 Arbitration logic

The Arbitration Logic sub-block controls the access of the RXONLY_CH and the DIAG_CH to the MSG Ctrl sub-block.

26.4.6 RXONLY_CH CAN machine

The RXONLY_CH CAN machine sub-block contains logic of the CAN protocol transfer layer, which is dedicated to monitor a CAN BUS chosen by the selector. It never transmits any dominant bit on the monitored bus. The RXONLY_CH is only connected to the monitored CAN channel via the RX-input terminal.

26.4.7 DIAG_CH CAN machine

The DIAG_CH CAN machine sub-block contains all logic for the CAN Protocol transfer layer. All protocol activities required for the reception of data frames, the transmission of message frames and the CAN protocol error management are executed automatically.

For transmission the DIAG_CH CAN machine uses always all 128 message buffers. However the storage of received messages depends on the state of the RXONLY_CH CAN machine. If the RXONLY_CH CAN machine is ‘in power save mode’ (DCNnCRCLMDOF = STOP) or in initialisation mode (DCNnCRCLMDPF = INIT), all message buffer are assigned to the DIAG_CH. When RXONLY_CH CAN machine is online or in sleep mode while the operating mode ‘RXONLY’ or ‘Mirror Mode’ is selected, only the lower 112 message buffer (#0 - #111) are used to store received messages from the DIAG_CH CAN machine.

Note When preparing for mirror mode while the upper 16 message buffers are still assigned to DIAG_CH, the application needs to clear all pending DCNnMmTRQF bits of these buffers at first. Then, all DCNnMmRDYF of these buffers need to be cleared as well. Then all these buffer need to be configured as receive message buffers before the initialisation state for the RXONLY_CH is released; the buffers are assigned to RXONLY_CH.

The acceptance filtering process, which decides whether a received data frame has to be stored into one of the assigned message buffers, is executed in the CAN module. The acceptance filtering process manages the storage into message buffers with and without an assigned mask, into multi-buffer receive blocks, and it resolves ambiguous storage situations.

In case more than one of the assigned message buffers are defined as transmit message buffer, more than one transmit request may be pending when the CAN module is able to get access to the CAN bus. The CAN module automatically detects the pending transmit request with the highest priority, which is evaluated by specific rules, and processes the transmission.

26.5 Module Initialisation and Control

Module initialisation is defined as the software processes, which are necessary to use the DCN module after a RESET.

Global DCN module switch-off by software is performed by clearing the DCNnGMCLCTL.DCNnGMCLPWOM bit (0).

After those 2 transitions the global module is disabled. When setting DCNnGMCLSESR, a software reset can be performed. The CPU can check the status of the global module by reading the DCNnGMCLPWOM bit in the DCNnGMCLCTL register.

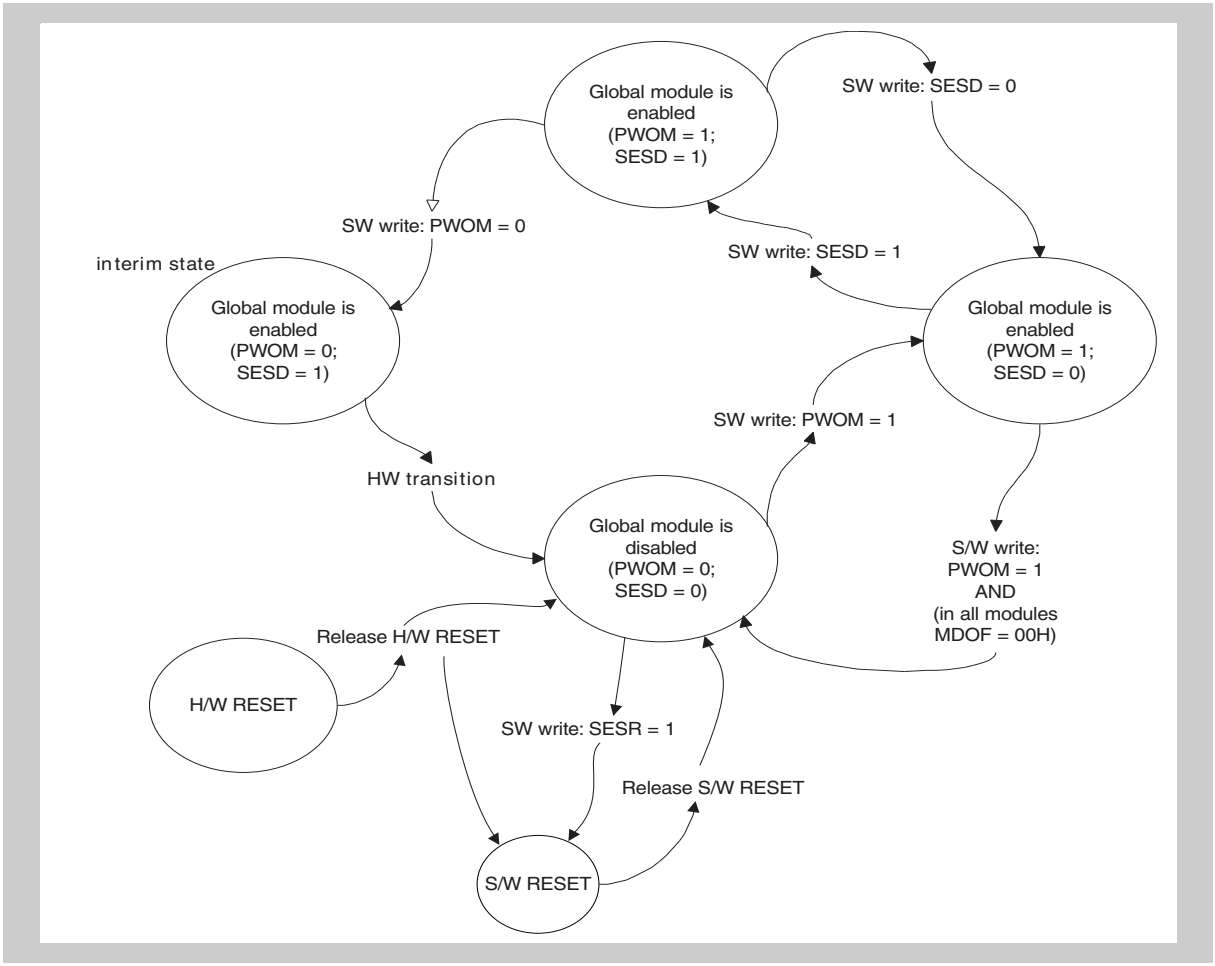


Figure 26-4 State transitions of global module switching

Note In the figure above following abbreviations are used:

- MDOF = DCNnCMCLCTL.DCNnCMCLMDOF[2:0]
- PWOM = DCNnGMCLCTL.DCNnGMCLPWOM
- SESD = DCNnGMCLCTL.DCNnGMCLSESD
- SESR = DCNnGMCTL.DCNnGMCLSESR

26.5.1 Global Module initialisation and control

(1) Global module disabled (DCNnGMCLCTL.DCNnGMCLPWOM = 0)

The global module is disabled after a H/W or S/W reset or a global DCN module switch-off by software. The CPU can check the status of the module by reading the DCNnGMCLCTL.DCNnGMCLPWOM bit.

- Characteristic of global module disabled (DCNnGMCLPWOM = 0):
  - CAN I/F channel output signal CANnTX is fixed to recessive level (logical 1)
  - Read and write access to the global module registers DCNnGMCLCTL, DCNnGMCSPRE, DCNnGMABCTL, DCNnGMADCTL is possible.
  - No read access and no write access to the message buffers.
  - CAN modules are disabled.
  - No write access to the CAN module register (DCNnCMMKCTLx, DCNnCMCLCTL, DCNnCMLCSTR, DCNnCMINSTR, DCNnCMERCNT, DCNnCMIECTL, DCNnCMISCTL, DCNnCMBRPRS, DCNnCMBTCTL, DCNnCMLISTR, DCNnCMRGRX, DCNnCMLOSTR, DCNnCMTGTX, DCNnCMTSCTL, DCNnCRCLCTL, DCNnCRLCSTR, DCNnCRERCNT, DCNnCRIECTL, DCNnCRISCTL, DCNnCRBRPRS, DCNnCRBTCTL, DCNnCRLISTR, DCNnCRBSSTR). Accidental write accesses to those registers are ignored by the hardware.

(2) Initialisations before enabling the global module

The application software has to determine all settings for the module clock f_{CANMOD} in the DCNnGMCSPRE register. The clock settings for the module clock f_{CANMOD} must not be changed after the global module is enabled.

(3) Global module enabled

The global module is switched on by setting the DCNnGMCLCTL.DCNnGMCLPWOM bit to “1”.

- Characteristics of global module enabled (DCNnGMCLPWOM = 1):
  - CAN I/F channels are enabled
  - Both CAN modules are in INIT mode
  - The CAN module register DCNnCMCLCTL, DCNnCMLCSTR, DCNnCMINSTR, DCNnCMERCNT, DCNnCMIECTL, DCNnCMISCTL, DCNnCMBRPRS, DCNnCMBTCTL, DCNnCRCLCTL, DCNnCRLCSTR, DCNnCRERCNT, DCNnCRIECTL, DCNnCRISCTL, DCNnCRBRPRS, DCNnCRBTCTL, DCNnCRBSSTR contain their initial values upon switching on the global module.
    The CAN module register DCNnCMLISTR, DCNnCMRGRX, DCNnCMLOSTR, DCNnCMTGTX, DCNnCRLISTR, DCNnCRLOSTR contain undefined values.
  - Write access to the global module register DCNnGMCLCTL is ignored.
  - Read and write access to all message buffers is possible.
  - Read and write access to the CAN module registers (DCNnCMMKCTLx, DCNnCMCLCTL, DCNnCMLCSTR, DCNnCMERCNT, DCNnCMIECTL, DCNnCMISCTL, DCNnCMBRPRS, DCNnCMTSCTL, DCNnCRCLCTL, DCNnCRLCSTR, DCNnCRIECTL, DCNnCRISCTL, DCNnCRBRPRS, DCNnCRBTCTL,DCNnCRBSSTR) is possible.
    Read access to the CAN module registers DCNnCMLISTR,

DCNnCMRGRX, DCNnCMLOSTR, DCNnCMTGTX, DCNnCRLISTR is possible.

– The CAN module register DCNnTIDRTXx, DCNnTIDMTXx contain their initial values upon switching on the global module.

(4) Switch-off the global module

The global module must not be switched off while one or more CAN modules are processing a message frame. A sudden interruption of a message frame transmission or reception will cause an error in other nodes connected to the same CAN bus.

Before switching off the global module the application software has to switch all CAN modules in INIT mode. Clearing the DCNnGMCLPWOM bit in the DCNnGMCLCTL register will put a switch-off request to the global module. The switch-off request is discarded in case one or more of the CAN modules are not properly set to INIT mode.

Some applications may require an immediate switch-off of the entire module in an emergency case, regardless of generating disturbance to the other nodes connected to the CAN bus. For such applications the DCNnGMCLSESD bit in DCNnGNCLCTL register can be set (1) to allow an immediate global module switch-off without switching the CAN modules to a defined state (i.e. INIT mode) beforehand.

When using the ‘forced shut down’ function by setting DCNnGMCLSESD bit (1) the subsequent access to the module has to be the instruction to clear DCNnGMCLPWOM bit (0). In case setting DCNnGMCLSESD bit (1) is not directly followed by clearing DCNnGMCLPWOM bit (0), the DCNnGMCLSESD bit is cleared (0) automatically (i.e. the ‘forced shut down’ function is disabled). Also a read access to the DCNnGMCLCTL register between setting of DCNnGMCLSESD bit (1) and clearing DCNnGMCLPWOM bit (0) will lead to an automatic clear (0) of DCNnGMCLSESD bit.

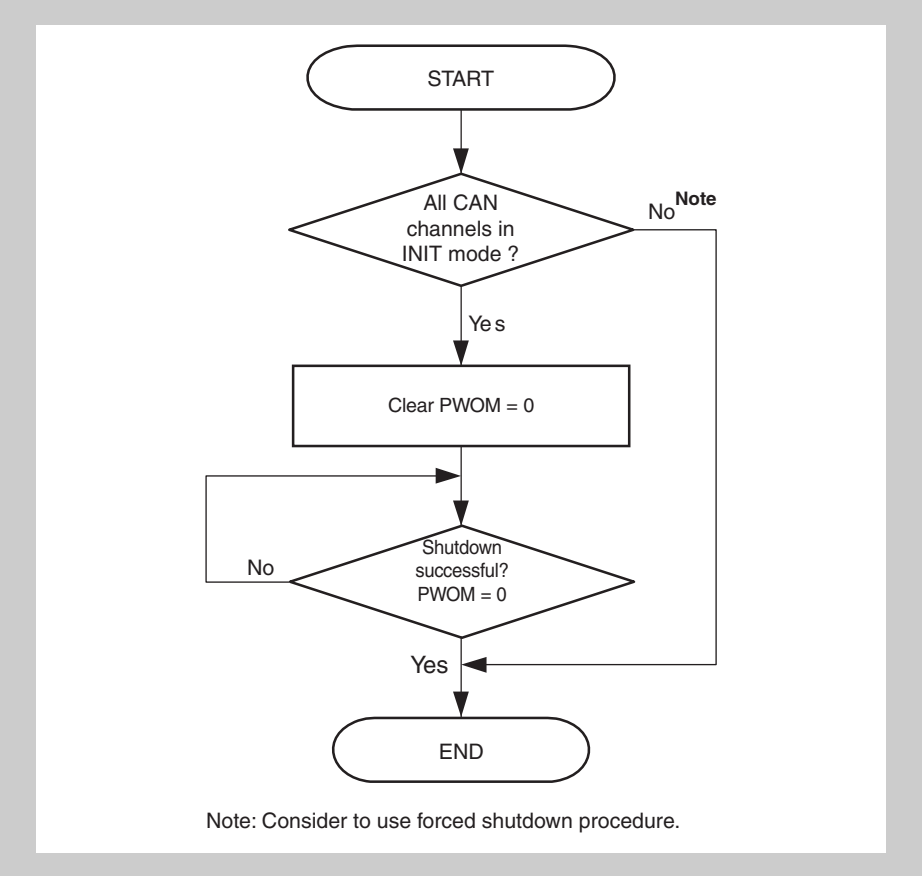


Figure 26-5 Shutdown process (normal shutdown)

Note In the figure above following abbreviations are used:

- PWOM = DCNnGMCLCTL.DCNnGMCLPWOM

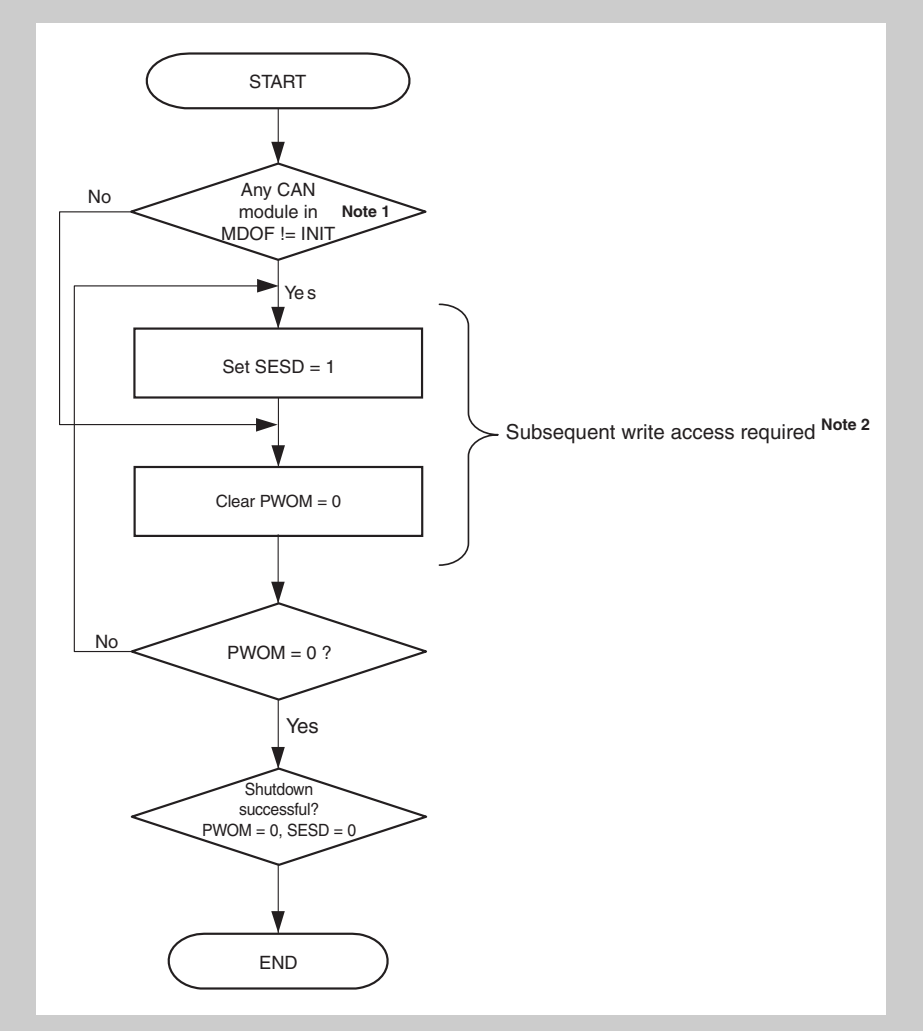


Figure 26-6 Shutdown process (forced shutdown)

Note In the figure above following abbreviations are used:

- MDOF = DCNnCMCLCTL.DCNnCMCLMDOF[2:0]
- PWOM = DCNnGMCLCTL.DCNnGMCLPWOM
- SESD = DCNnGMCLCTL.DCNnGMCLSESD

Note 1. OPMODE:

- Normal operation mode
- Normal operation mode with ABT
- Receive only mode
- Single shot mode
- Self test mode

2. Ensure that no CPU read or write access to any register happen between DCNnGMCLSESD = 1 and DCNnGMCLPWOM = 0.

(5) Flowchart of the initialisation

The following flowcharts describe the basic initialisation and in addition the re-initialisation of the CAN module. Note that the configuration for the baud rate of RXONLY_CH should have the same configuration as the monitored CAN channel.

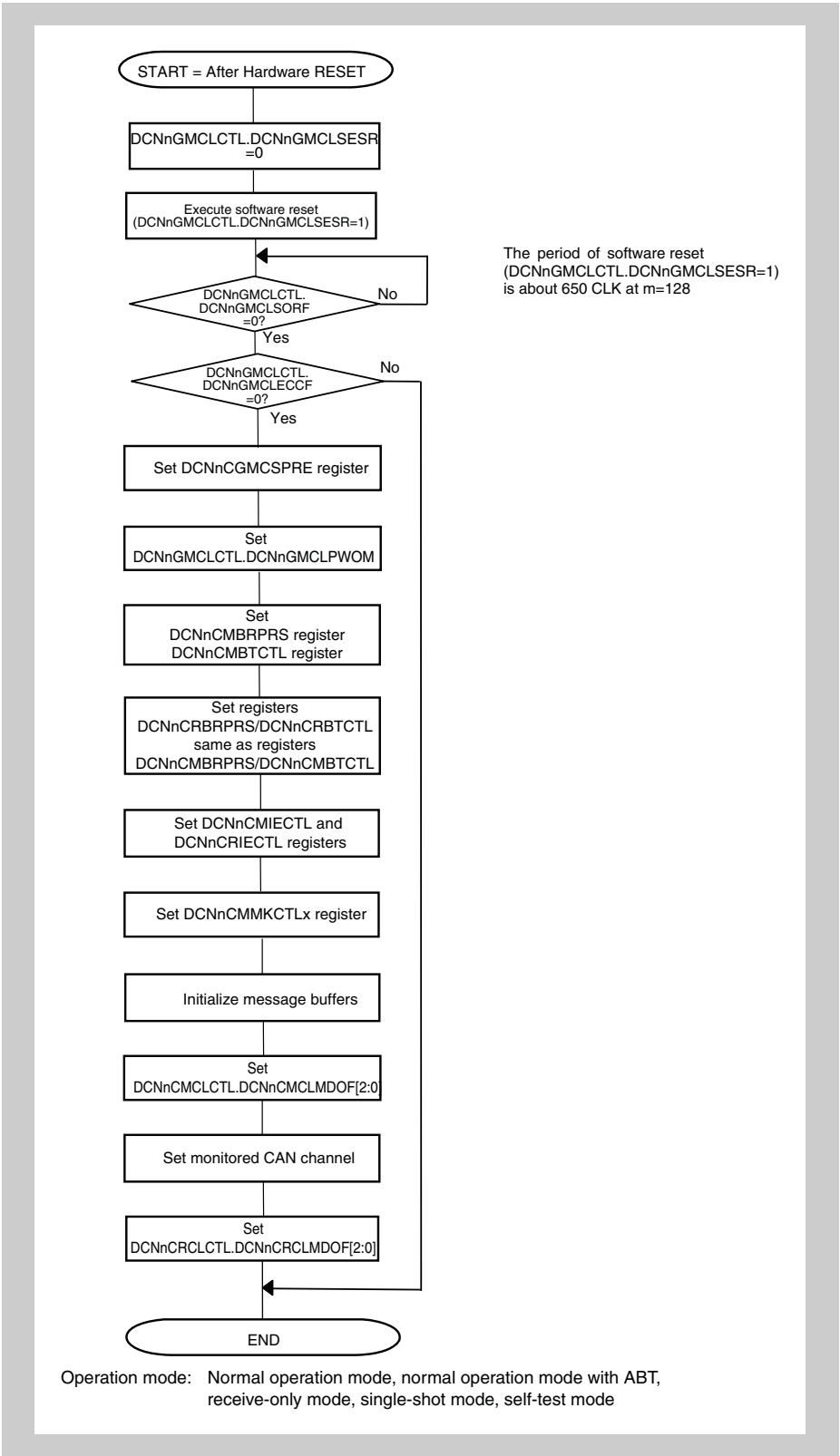


Figure 26-7 DCN module initialisation

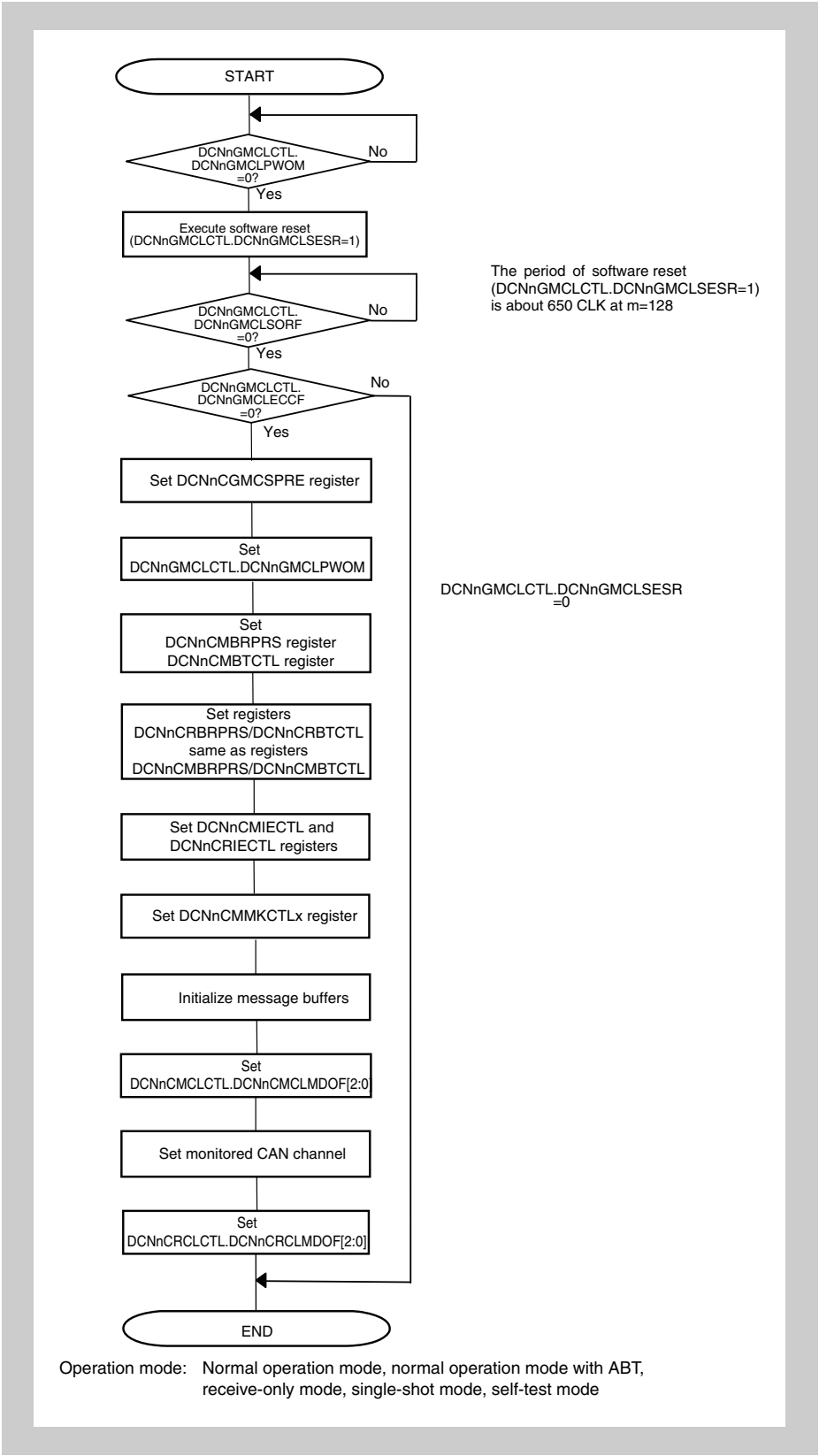


Figure 26-8 Reinitialisation with software reset

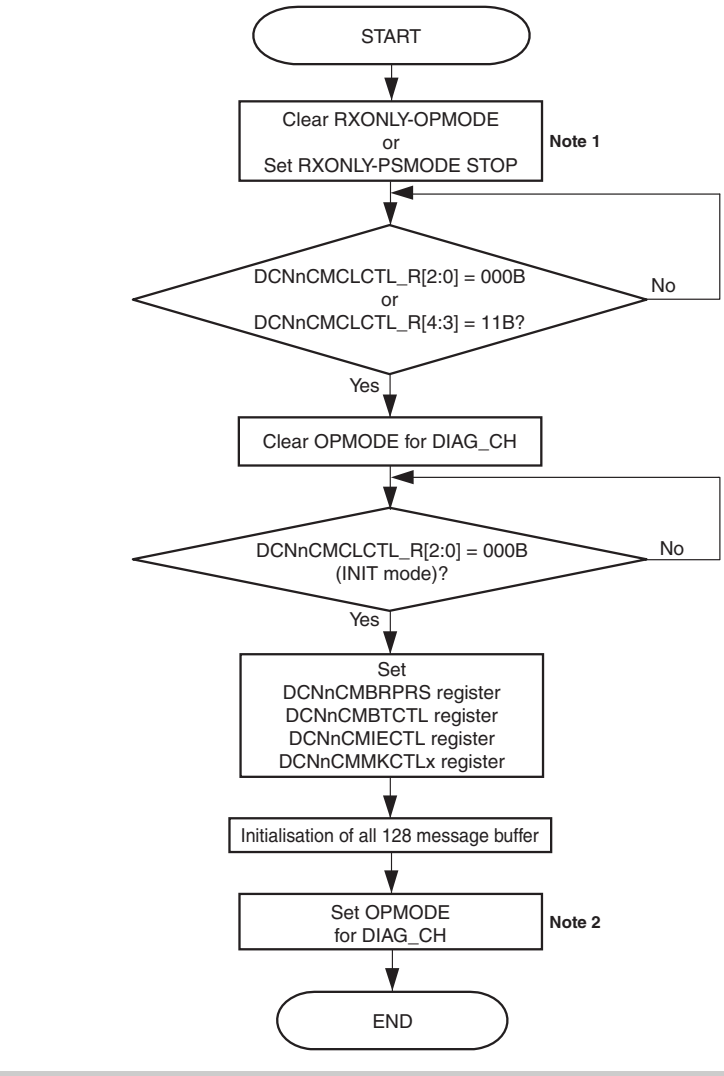


Figure 26-9 Re-initialisation of the DIAG_CH CAN module using 128 buffers without software reset

Note

1. OPMODE for RXONLY_CH - Initialization mode
   PSMODE for RXONLY_CH - STOP mode
2. OPMODE:
   - Normal operation mode
   - Normal operation mode with ABT
   - Receive only mode
   - Single shot mode
   - Self test mode

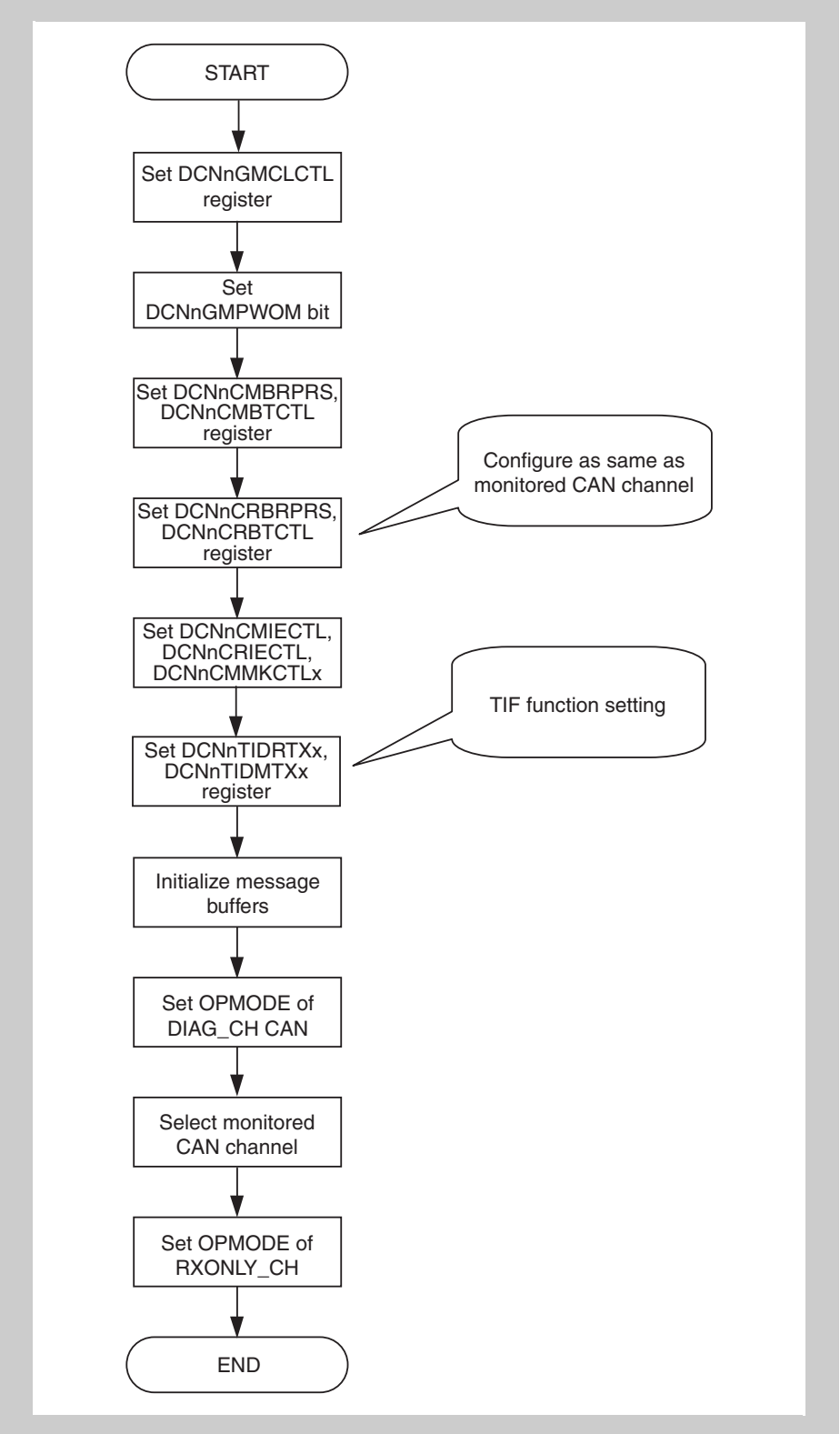


Figure 26-10 Re-initialisation of the DIAG_CH CAN module using TX ID filter function without software reset

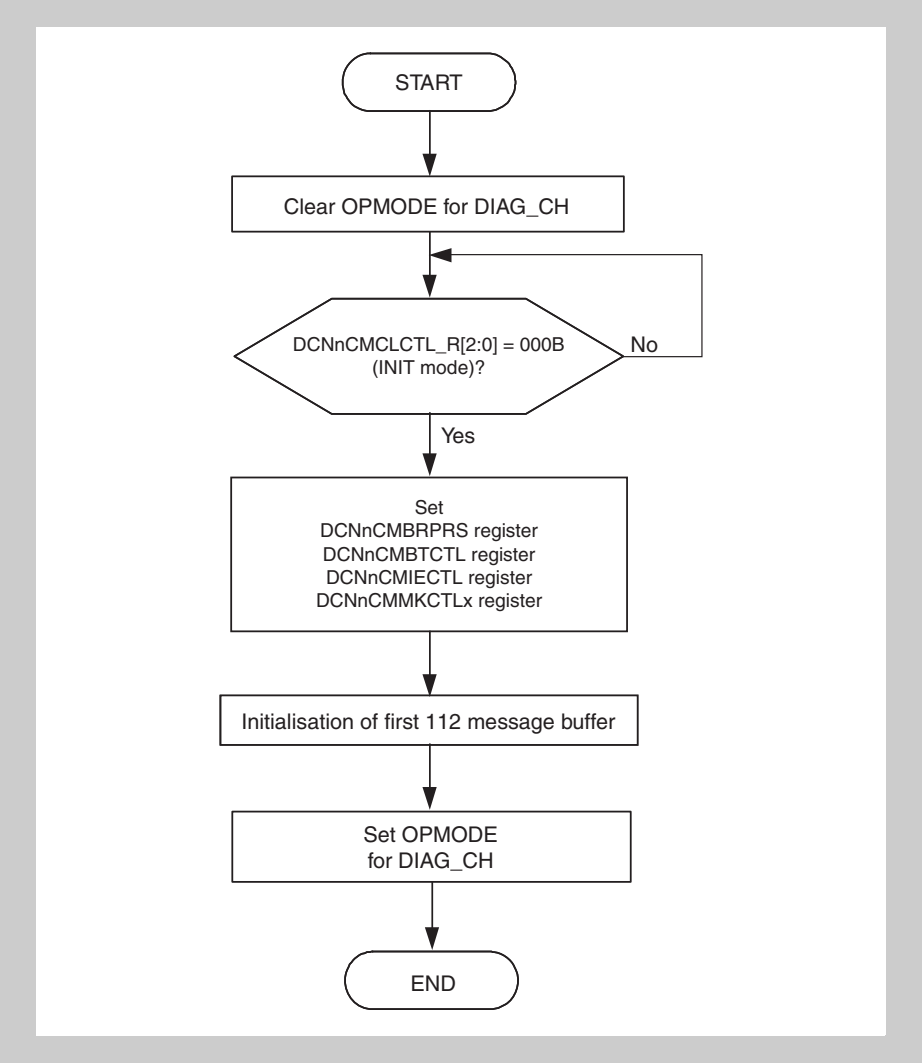


Figure 26-11 Re-initialisation of the DIAG_CH CAN Module using first 112 buffers only without software reset

Note OPMODE:

- Normal operation mode
- Normal operation mode with ABT
- Receive only mode
- Single shot mode
- Self test mode

26.5.2 Message buffer initialisation and configuration

After a hardware or software reset, all message buffers are initialized automatically.

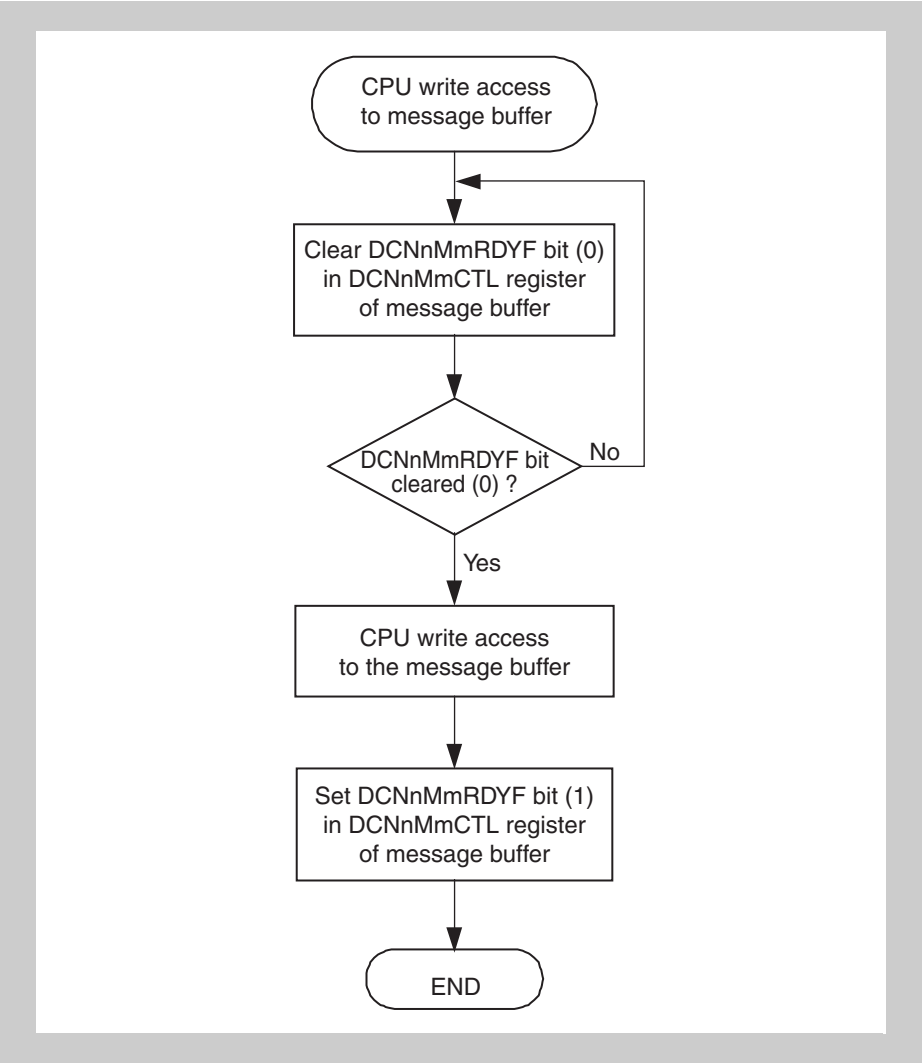


Figure 26-12 CPU write access to a message buffer

While DCNnRDYF bit is cleared (0) the message buffer is not accessed by the assigned CAN I/F channel (CAN module and/or TTCAN module) and/or bridge module (RXONLY_CH). Thus no message frame can be received in the message buffer or transmitted from the message buffer. Whenever the CPU wants to write into the message buffer, the DCNnRDYF bit must be cleared (0).

Once the CPU has finished the write access to the message buffer it must set the DCNnRDYF bit (1) to allow CAN protocol processing to the particular message buffer by the assigned CAN I/F channel.

CPU write accesses to a message buffer with DCNnRDYF bit set (1) are ignored.

For the RXONLY_CH the semaphore handling via DCNnRDYF bit is identical to the rule mentioned above. However, the application is strongly advised to set all DCNnRDYF bits (1) for the upper 16 message buffers in order to provide the maximum depth for the ring buffer in mirror mode. As well it is absolutely mandatory not to write to the DCNnRDYF bit (clear or set the DCNnRDYF bit)

to any of the upper 16 message buffers while the RXONLY_CH is operated in mirror mode or mirror mode with TIF. This will terminate the mirror mode or mirror mode with TIF or cause an erroneous sequence of monitored messages on the diagnostic CAN bus.

26.5.3 Message buffer to CAN I/F channel assignment

The area of upper 16 message buffers can be assigned to the DIAG_CH or RXONLY_CH while the rest of the message buffers is statically assigned to the DIAG_CH as shown in the figures below.

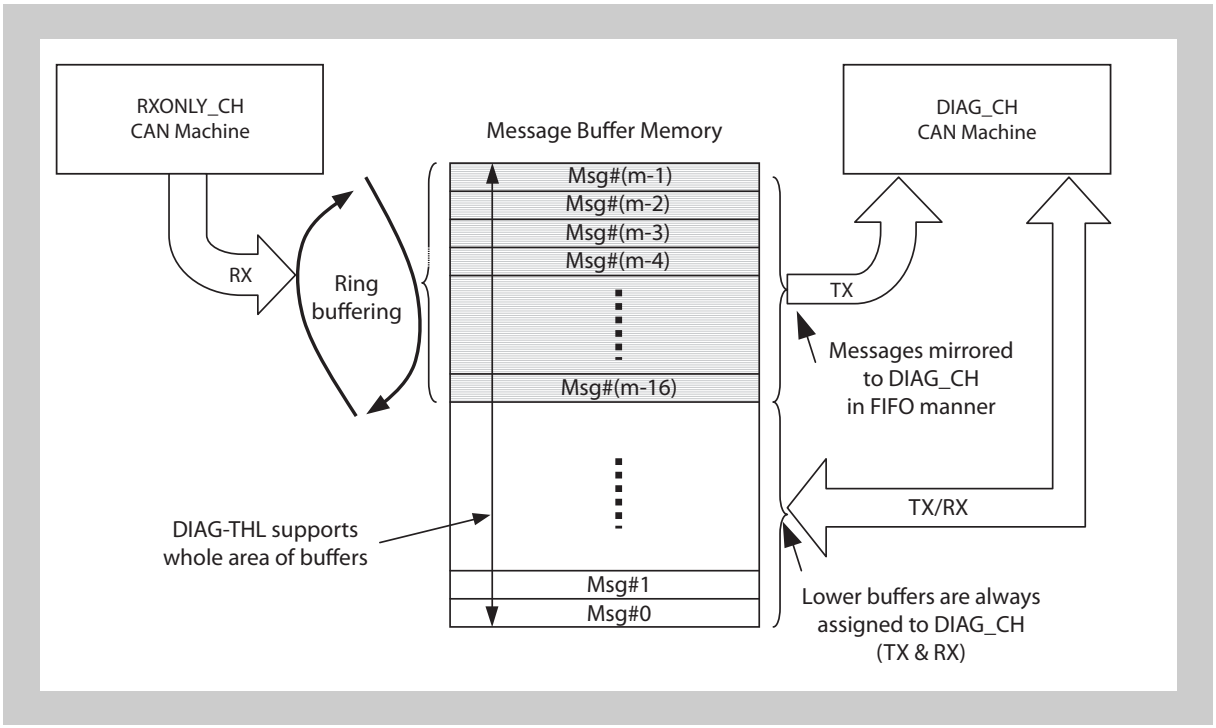


Figure 26-13 Operation in mirror or receive-only mode of RXONLY_CH

The assignment of the upper 16 message buffers depends on the operation mode and power save mode of RXONLY-CH CAN machine.

Basically, these buffers are assigned to the RXONLY_CH while the RXONLY_CH is operational mode (i.e. mirror mode / rec-only mode). And additionally, it must be assigned to the RXONLY_CH while the RXONLY_CH is in SLEEP mode because the wake-up event on the RXONLY_CH is unpredictable. In STOP mode these buffers are assigned to the DIAG_CH in order to provide a mode of operation where the DIAG_CH uses all 128 buffers while the RXONLY_CH uses minimum power. The user needs to configure the upper 16 message buffers before changing assignment (i.e. leaving initialisation or STOP mode of the RXONLY_CH).

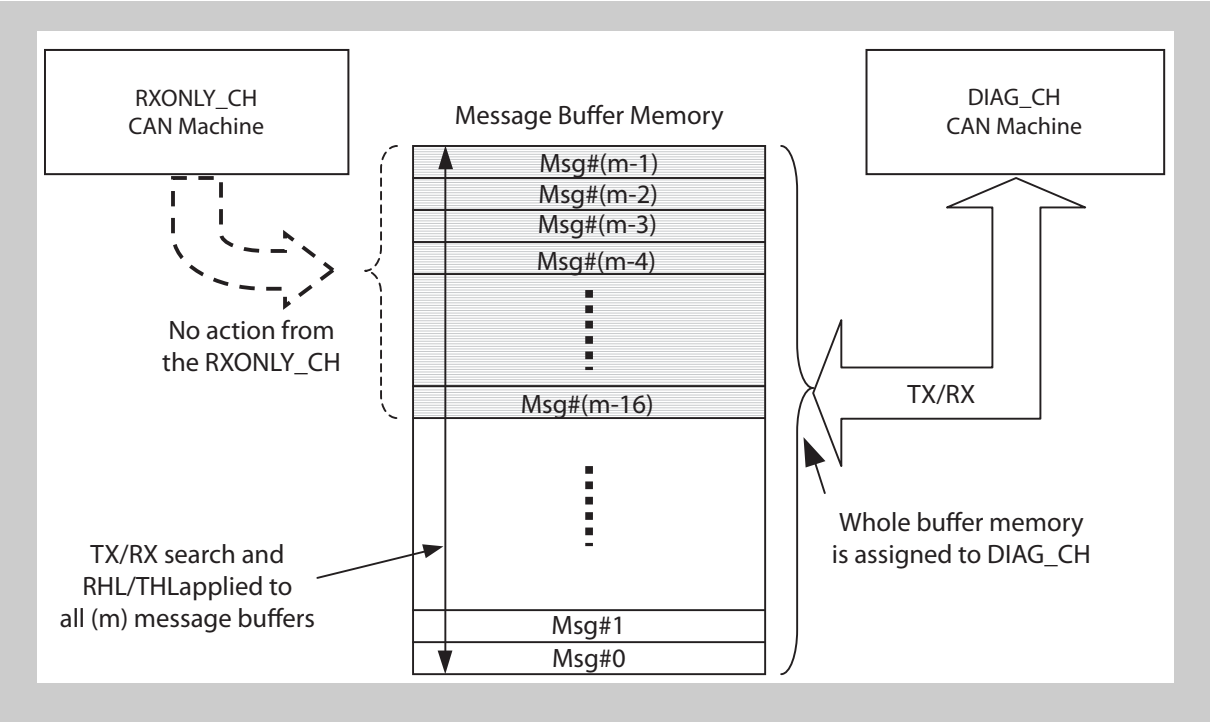


Figure 26-14 Operation during INIT or STOP of RXONLY_CH

The assignment of message buffers versus operational and power-save modes is shown in the table below.

Table 26-9 Upper 16 message buffer assignment versus operational and power save modes

Operation mode / power save mode	MIRROR mode	MIRROR mode with TIF	REC-ONLY mode	INIT mode
NO PWR-SAVE	RXONLY	RXONLY	RXONLY	DIAG
SLEEP	RXONLY	RXONLY	RXONLY	
STOP	DIAG	DIAG	DIAG	

When the assignment of the upper 16 message buffers switches, the application should configure all buffers beforehand as follows.

(1) Configuration before switching to Mirror Mode

For mirror mode operation all upper 16 message buffers shall be used as a ring buffer working in FIFO manner. Thus, the application has to prepare these buffers as TX-buffers beforehand as follows. Note that OWS setting is invalid (not necessary) because the non-overwriting of message buffers with DN = 1 is built in feature of operational modes of the RXONLY_CH.

Bit name	Configuration
DCNnMmCTL.DCNnMmMOWF	Clear
DCNnMmCTL.DCNnMmIENF	Depends on application
DCNnMmCTL.DCNnMmDTNF	Clear
DCNnMmCTL.DCNnMmTRQF	Clear
DCNnMmSTRB.DCNnMmMA[2:0]	001_B
DCNnMmSTRB.DCNnMmSSMT[3:0]	000_B
DCNnMmSTRB.DCNnMmSSRT	Don’t care
DCNnMmSTRB.DCNnMmSSOW	Don’t care
DCNnMmSTRB.DCNnMmSSIE	Don’t care
DCNnMmMIDx.DCNnMmSSID[28:0]	Don’t care
DCNnMmDTLGB.DCNnMmDTLG[3:0]	Don’t care
DCNnMmDATx.DCNnMmSSD	Don’t care

The flow of operations in order to change the assignment of message buffers to a CAN-channel is shown in the figure below. At first the application needs to apply RDY = 0 for the upper 16 message buffer or set the DIAG_CH into INIT mode before the new configuration can be written to the buffers.

Once the new configuration is set, the DIAG_CH followed by RXONLY_CH is put into its designated operational and power save mode. Finally the upper 16 message buffer are re-enabled by setting the DCNnRDYF bit (1).

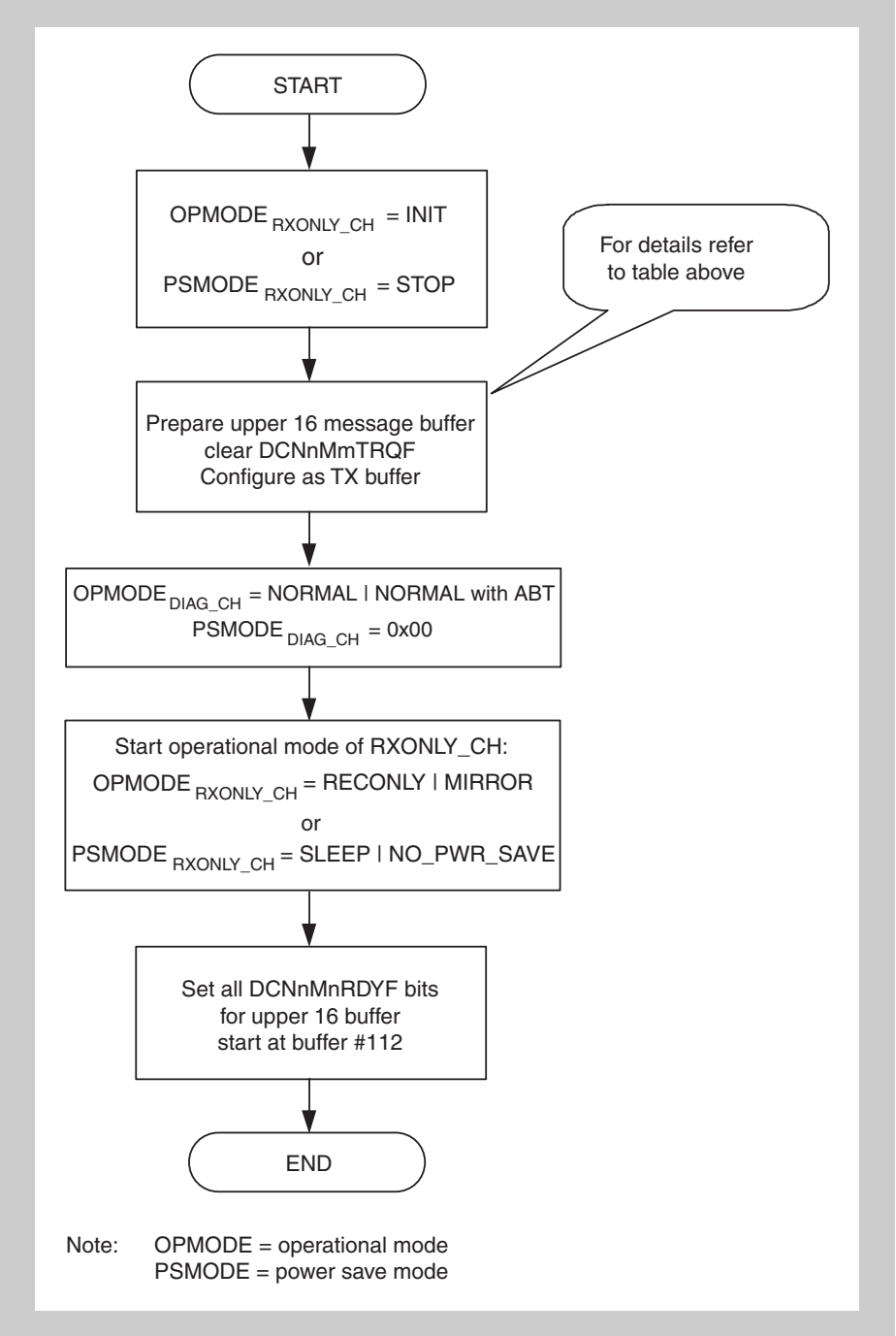


Figure 26-15 Set up of mirror mode, mirror mode with TIF or receive-only mode for RXONLY_CH

(2) Configuration before switching to DIAG side

All upper 16 message buffers can be assigned to the DIAG_CH in order to use them as normal TX/RX message buffers by the application. It is not necessary to apply a particular configuration for these buffers before the assignment is linked again to the DIAG_CH. However, it is recommended that the application clears DCNnMmRDYF and DCNnMnTRQF bits before assigning the buffers to the DIAG_CH or puts the DIAG_CH into operation mode INIT or power save mode STOP before changing the assignment. Then, at least clear the DCNnMmRDYF bits for the upper 16 message buffers before putting the DIAG_CH to normal mode. This will ensure proper message handling during the transition.

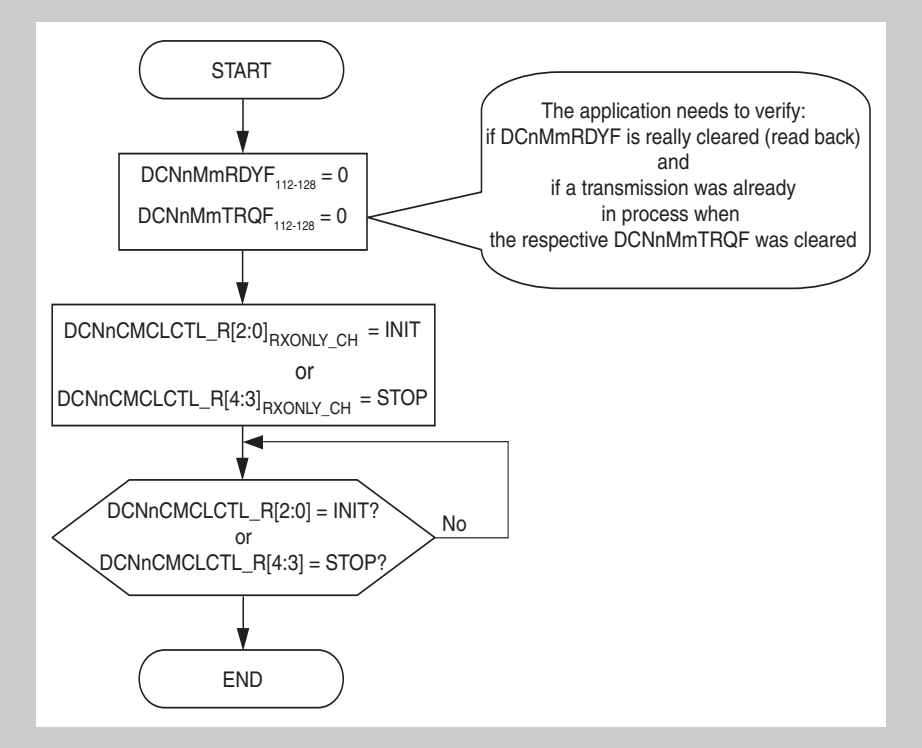


Figure 26-16 Cancellation of mirror mode or receive-only mode for RXONLY_CH

When clearing the DCNnMmTRQF in the upper message buffer a transmission from one of these buffers may still be in progress. In order to ensure correct message handling (i.e. transmission history list), it is recommended to wait for the first transmission completion interrupt of the DIAG_CH before resuming the transition procedure.

26.5.4 DCN module initialisation and control

After a RESET the registers DCNnGMCLCTL, DCNnGMCSPRE, DCNnGMABCTL, DCNnCMMKCTLx, DCNnCMCLCTL, DCNnCMLCSTR, DCNnCMERCNT, DCNnCMIECTL, DCNnCMISCTL, DCNnCMBRPRS, DCNnCMBTCTl, DCNnCMTSCTL, DCNnCRCLCTL, DCNnCRLCSTR, DCNnCRERCNT, DCNnCRIECTL, DCNnCRISCTL, DCNnCRBRPRS, DCNnCRBTCTl, DCNnCRLISTR, DCNnCRBSSTR of the CAN module in the CAN I/F channels contain their initial value.

Before switching the CAN module into an operational mode the registers (DCNnCMMKCTLx, DCNnCMCLCTL, DCNnCMIECTL, DCNnCMBRPRS, DCNnCMBTCTl, DCNnCMTSCTL, DCNnCRCLCTL, DCNnCRIECTL, DCNnCRBRPRS, DCNnCRBTCTl) must be initialised according the requirements of the application. Further, the DCNnCMLCSTR and DCNnCRLCSTR registers must be set to its initial value (00H).

26.5.5 CAN bit time programming

After proper setting of the CAN bit time, the CAN module can be released from INIT mode into one of its operational modes.

26.5.6 Transitions for operational modes of DIAG_CH

The DIAG_CH CAN machine can be switched the following operational modes:

“Normal Operating Mode”

“Normal Operating Mode with Automatic Block Transmission”

“Receive-only Mode”

“Single-shot Mode”

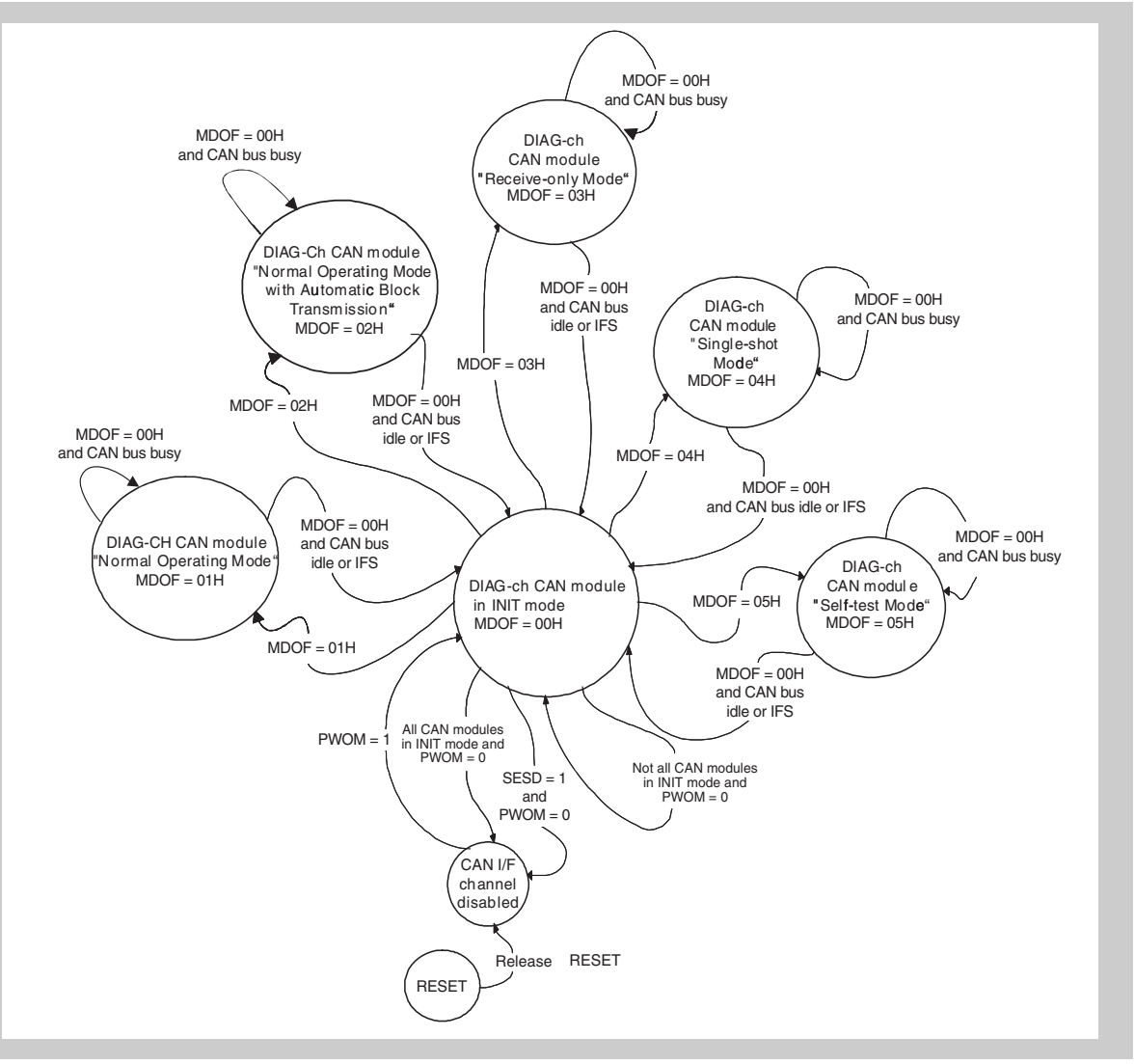


Figure 26-17 Transitions for operational modes of the DIAG_CH CAN module

Note In the figure above following abbreviations are used:

- MDOF = DCNnCMCLCTL.DCNnCMCLMDOF[2:0]
- PWOM = DCNnGMCLCTL.DCNnGMCLPWOM
- SESD = DCNnGMCLCTL.DCNnGMCLSESD

The transitions from INIT mode to the operational modes is controlled by the bit-string DCNnCMCLMDOF[2:0] in the DCNnCMCLCTL register.

Changing from one operational mode into another operational mode requires to transit into INIT mode intermittently. The CAN module refuses CPU attempts to change from one operational mode into another operational mode directly.

Transition requests from the operational modes to the INIT mode are not directly accepted by the CAN module when the CAN bus is not idle (i.e. frame reception or transmission is ongoing), but it has to be kept until when the CAN module detects the first bit of intermission. As soon the above mentioned condition is detected, the transition from the operational mode to the INIT mode is executed and the bit-string DCNnCMCLMDOF[2:0] value changes to 00H. The CPU has to confirm the proper transition into INIT mode by reading the DCNnCMCLMDOF[2:0] bit-string until DCNnCMCLMDOF[2:0] = 00H.

When a successful receive interrupt is detected for a specific CAN module that already entered INIT mode, this interrupt was generated by a reception process that coincided with the request of INIT mode by the CPU. The received message that caused this interrupt was still stored before INIT mode was becoming valid.

26.5.7 Transition for operational modes of RXONLY_CH

The RXONLY-H CAN machine can be switched the following operational modes:

- “Mirror Mode”
- “Mirror Mode with Transfer ID Filter function (with TIF)”
- “Receive-only Mode”

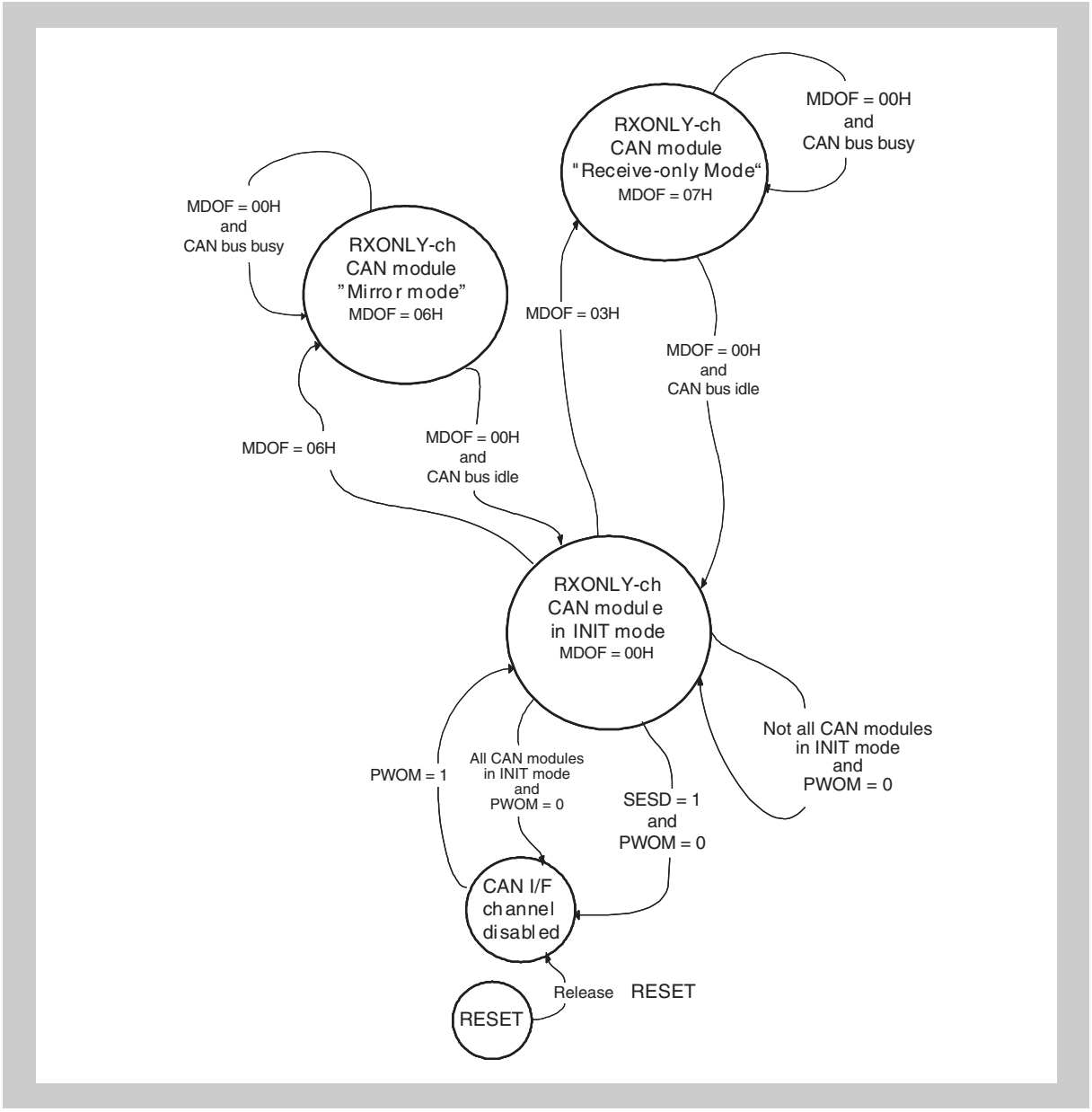


Figure 26-18 Transitions for operational modes of the RXONLY_CH CAN module

Note In the figure above following abbreviations are used:

- MDOF = DCNnCMCLCTL.DCNnCMCLMDOF[2:0]
- PWOM = DCNnGMCLCTL.DCNnGMCLPWOM
- SESD = DCNnGMCLCTL.DCNnGMCLSESD

The transitions from INIT mode to the operational modes is controlled by the bit-string DCNnCMCLMDOF[2:0] in the DCNnCMCLCTL register.

The rules for changing from one to another operational mode are the same as for the DIAG_CH.

26.6 Module Interrupts

The DCN module provides 13 different interrupt source events. The occurrence of those interrupt source events is stored in interrupt status registers. Four separate interrupt request signals are generated from the 13 source events. The signals are routed to the interrupt controller in the microcomputer system. In case the interrupt controller in the microcomputer system does not provide a sufficient number of interrupt request signal inputs, the 4 interrupt request signals of a CAN module can be grouped (i.e. logical OR). With help of the interrupt status register the user can determine the actual interrupt source event for a particular interrupt. After determination of the interrupt source event the user must clear the corresponding interrupt status bit.

Table 26-10 List of all module interrupt sources (1/2)

#	Interrupt status bit		Interrupt enable bit		Interrupt request signal	Interrupt source description
	Name	Register	Name	Register		
1	CINTS0	DCNn CMISCLT	DCNn CMIE INTF0[a]	DCNn CMIECTL	INTTRX	CAN module interrupt status bit for interrupt event 'Message frame successfully transmitted from message buffer m' by the DIAG_CH. The interrupt is also provided for mirrored messages.
2	CINTS1	DCNn CMISCLT	DCNn CMIE INTF1[a]	DCNn CMIECTL	INTREC	CAN module interrupt status bit for interrupt event 'Valid message frame reception in message buffer m' from the DIAG_CH.
3	CINTS1	DCNn CRISCLT	DCNn CRIE INTF1[a]	DCNn CRIECTL		CAN module interrupt status bit for interrupt event 'Valid message frame reception in one of the upper 16 message buffers from the RXONLY_CH.
4	CINTS2	DCNn CMISCLT	DCNn CMIE INTF2	DCNn CMIECTL	INTERR	CAN module error state interrupt status of the DIAG_CH
5	CINTS2	DCNn CRISCLT	DCNn CRIE INTF2	DCNn CRIECTL		CAN module error state interrupt status of the RXONLY_CH
6	CINTS3	DCNn CMISCLT	DCNn CMIE INTF3	DCNn CMIECTL		CAN module protocol error interrupt status of the DIAG_CH
7	CINTS3	DCNn CRISCLT	DCNn CRIE INTF3	DCNn CRIECTL		CAN module protocol error interrupt status of the RXONLY_CH
8	CINTS4	DCNn CMISCLT	DCNn CMIE INTF4	DCNn CMIECTL		CAN module arbitration loss interrupt status of the DIAG_CH
9	CINTS5	DCNn CMISCLT	DCNn CMIE INTF5	DCNn CMIECTL	INTWUP	DIAG_CH CAN machine wake-up interrupt status bit from SLEEP mode by CAN bus
10	CINTS5	DCNn CRISCLT	DCNn CRIE INTF5	DCNn CRIECTL		RXONLY_CH CAN machine wake-up interrupt status bit from SLEEP mode by CAN bus

Table 26-10 List of all module interrupt sources (2/2)

#	Interrupt status bit		Interrupt enable bit		Interrupt request signal	Interrupt source description
	Name	Register	Name	Register		
11	CINTS6	DCNn CMISCLT	DCNn CMIE INTF6	DCNn CMIECTL	INTERR	CAN module transmit abort interrupt status from DIAG_CH
12	CINTS6	DCNn CRISCLT	DCNn CRIE INTF6	DCNn CRIECTL	INTERR	The buffer overflow interrupt status bit, which is set 1 when the RXONLY_CH CAN machine fails to store a message because the upper 16 message buffer are all occupied (i.e. DN <DCNnCRLISTR +1> = 1).
13	CINTS7	DCNn CMISCLT	DCNn CMIE INTF7	DCNn CMIECTL	INTTRX	Signals a completed transmission for upper 16 message buffer during Mirror Mode. The status bit and DCNnCMISCLT7 are only meaningful when RXONLY_CH is operated in Mirror mode. DCNnCMISCLT7 has no function when DCNnCMIEINT0 in DCNnCMIECTL is cleared.

a) The DCNnMmIENF bit (message buffer interrupt enable bit) in the DCNnMmCTL register must set (1) for the message buffers, which shall participate in the interrupt generation process.

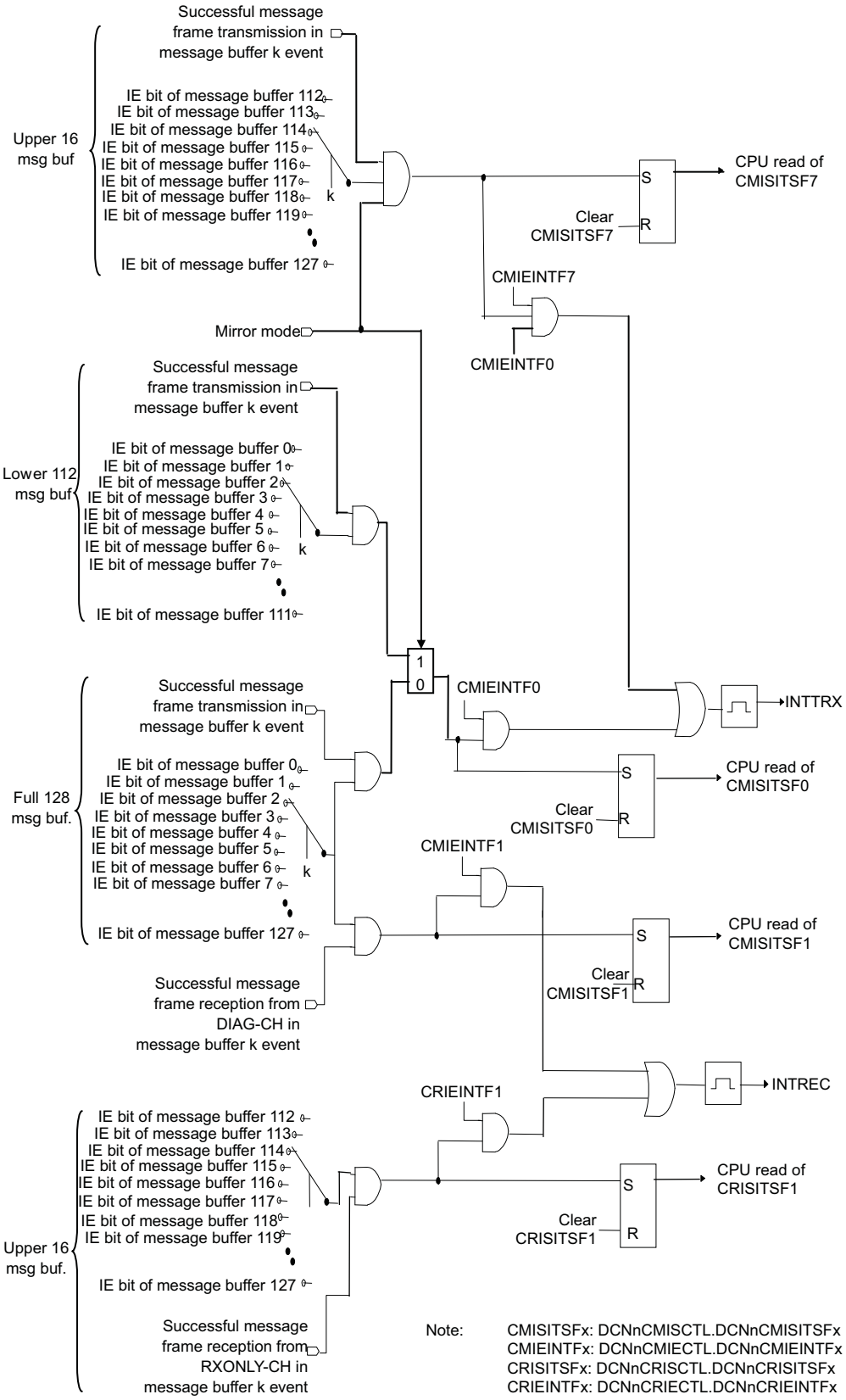


Figure 26-19 Schematic of the DCN module interrupt generation (1/2)

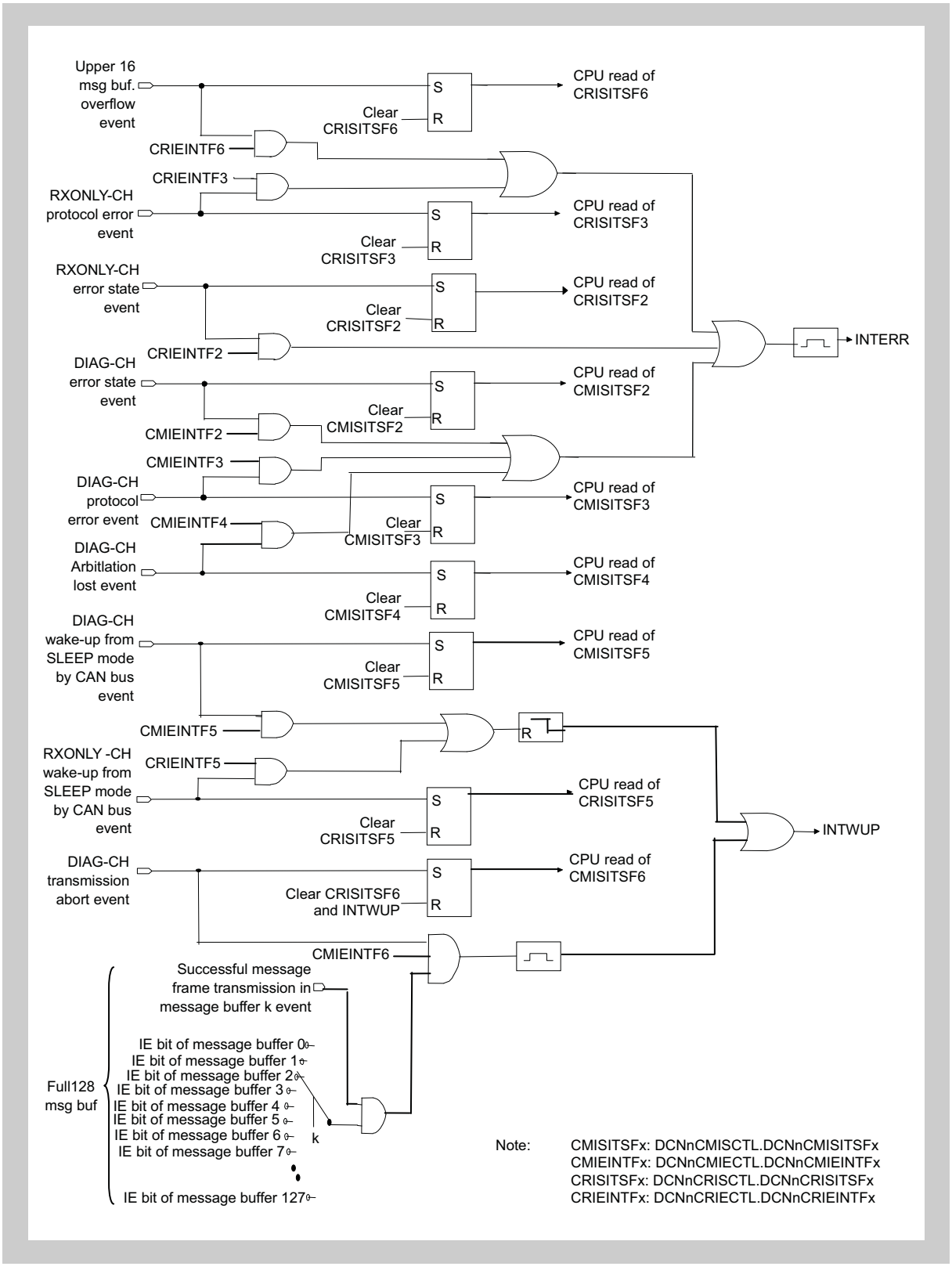


Figure 26-20 Schematic of the DCN module interrupt generation (2/2)

26.7 Message Reception of RXONLY Channel

26.7.1 Principal reception process

In general all operations linked to acceptance filtering are only available on the DIAG_CH. For the RXONLY_CH only simple receive operations apply. These are mentioned more detailed in chapter *26.9 “Operational Modes of RXONLY_CH” on page 2056* . The features of DIAG_CH regarding reception are identical with those of the CAN Controller module, so these are not mentioned within this addendum.

For the RXONLY_CH the acceptance filtering is obsolete. Any valid message will be stored in the upper 16 message buffer of the DCN module when Mirror Mode, mirror mode with TIF or Receive Only Mode is selected. In both modes the RXONLY_CH will store the message in the next higher buffer number where it finds a buffer with its DCNnMmDTNF bit cleared. At buffer #127 is wraps around to buffer #112.

In Receive Only Mode the application needs to read the messages and clear the DCNnMmDTNF bits in time to avoid the generation of the interrupt CINTS6 in DCNnCRISCLT.

In Mirror Mode the reception process will also set the DCNnMmDTNF bit but no receive interrupt is generated. Rather another state machine is triggered to evaluate if the DCNnMmTRQF of this message buffer can be set, and thus launch the transmission of the message on the DIAG_CH. The state machine handling the setting of DCNnMmTRQF´s for the upper 16 message buffers is activated every time the DIAG_CH or the host CPU clears a DCNnMmTRQF bit. Even DCNnMmTRQF bits for the lower 32 message buffers need to be considered. When the state machine detects that the DCNnMmTRQF for the message buffer it flagged for transmission the last time was cleared, the DCNnMmTRQF for the next buffer is set unless the history list implies no pending entry.

As the DCNnMmDTNF bit is used by the RXONLY_CH to detect an empty (already transmitted) buffer, the DCNnMmDTNF bit is cleared automatically after the message was transmitted successfully on the DIAG_CH.

26.7.2 Reception History

The RXONLY Channel does not support the Reception History List (RHL), as it is known for the standard CAN Controller channels and DIAG_CH. However, the functions of the Last-In-Message Pointer (LIPT) are implemented within the DCNnCRLISTR register.

Note The DCNnCRLISTR value is maintained throughout the power save mode STOP for RXONLY_CH. This register is only cleared when entering operation mode INIT.

Caution The RGPT of the DCN channel is not updated by the reception from the RXONLY_CH.

26.7.3 Reception of remote frames

The reception of remote frames applies to both, the DIAG_CH and the RXONLY_CH. However, for RXONLY_CH message storing follows different rules and acceptance filtering is not available.

A remote frame reception on the RXONLY_CH is handled as a normal data frame reception into the upper 16 message buffers without any acceptance filtering. The message will be stored according the ring buffer method.

26.8 Message Transmission

As the RXONLY_CH does only receive messages, only specific transmit operations by the DIAG_CH that are linked to the mirror mode of the RXONLY_CH are to be explained in *26.9.2 “Mirror mode” on page 2061* .

Regarding the transmission functionality of the DIAG_CH, please refer to the FCN CAN Controller chapter.

26.9 Operational Modes of RXONLY_CH

26.9.1 Receive–only mode

In Receive-only mode of the RXONLY_CH CAN module every data and every remote frame, which is received from the CAN bus in the RXONLY_CH CAN module without an error (i.e. valid reception), is stored in the upper 16 message buffers without any acceptance filtering.

Storing received messages from RXONLY_CH requires the following conditions:

- The message buffer has to be assigned to the CAN I/F channel, which received the data frame (DCNnMmSSMA bit-string in DCNnMmSTRB register has to hold the values 01H).
- The message buffer has to be marked ready for CAN protocol processing (DCNnMmRDYF bit set (1) in DCNnMmCTL register)
- Additional condition necessary for Mirror Mode operation:
  The upper 16 message buffer need to be configured as transmit message buffer (DCNnMmSSMT bit-string in DCNnMmSTRB has to hold the value 00H).

Caution In difference to a regular CAN Controller channel, any message is accepted by the buffer although it is configured as a transmit message buffer that normally would not except to receive data frames.
All transmit request bits of the upper 16 message buffer have to be cleared (DCNnMmTRQF bit reset (0) in DCNnMmCTL register) especially before Mirror Mode shall be invoked. This is necessary because the setting of DCNnMmTRQF for the upper 16 message buffer is handled by the mirror mode machine.
The upper 16 message buffer behave as ring buffer as shown in the figure below.

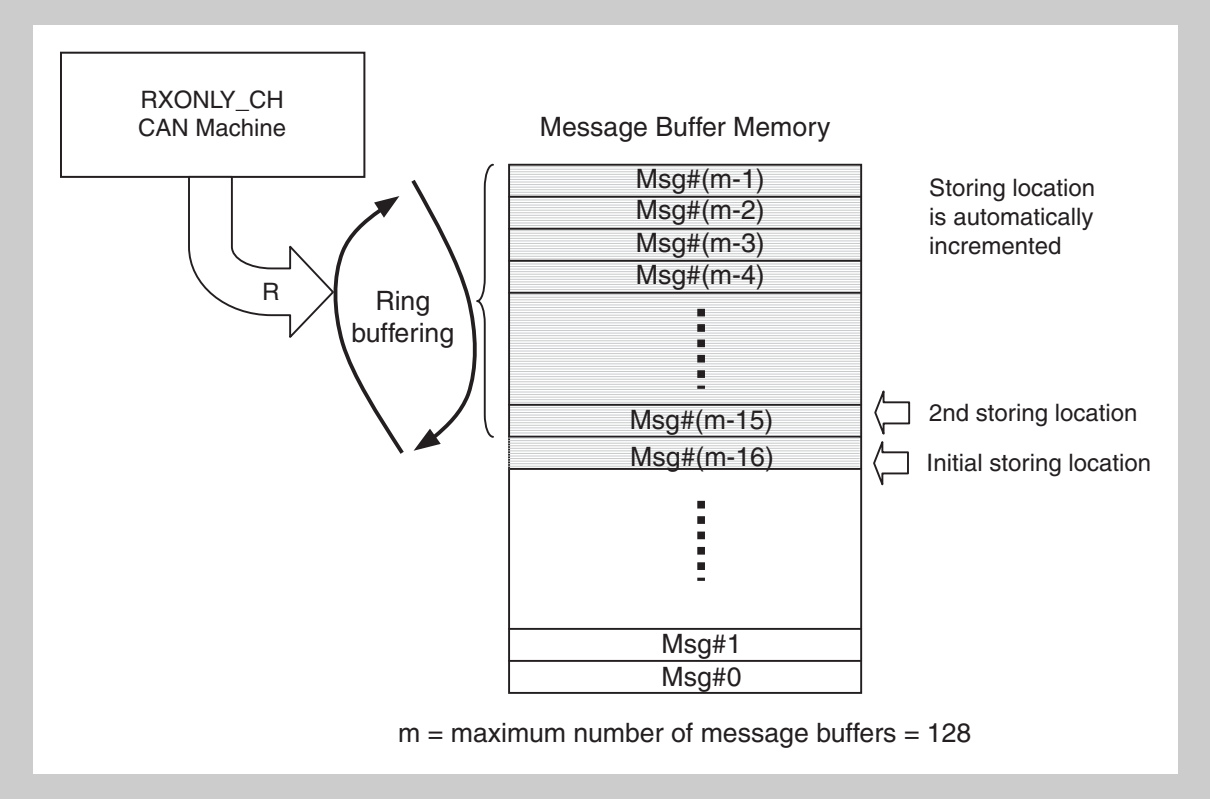


Figure 26-21 Reception Process of RXONLY

The storing pointer is initialised to (m-16) at the time when switching the assignment of upper 16 message buffer.

The reception process of the RXONLY_CH differs from the reception process of the regular CAN Controller module.

- There is no acceptance filtering; any data or remote frame will be received.
- No specific setting is required to receive standard and extended format frames in the upper 16 buffers during RECONLY and Mirror Mode. DCNnMmSSIE bit will be written by the RX-store machine with respect to the received frame type.
- For received remote frames the DCNnMmSSRT bit will be set by the RX-Store machine.

The table below compares the differences for each data type of the buffer.

Table 26-11 Differences in reception process between regular CAN Controller and RXONLY_CH

Register	Bit String	DCN /FCN CAN Controller	RXONLY_CH
DCNnMmDATx	-	Written by RX-Store machine for remote frames value is kept	Written by RX-Store machine
DCNnMmDTLGB	-	Written by RX-Store machine for remote frames value is kept	Written by RX-Store machine
DCNnMmSTRB	DCNnMmSSOW, DCNnMmMT[3:0], DCNnMmSSMA	Set up by CPU during configuration	Set up by CPU during configuration
	DCNnMmSSRT	Set up by CPU when preparing TX message	Written by RX-Store machine according received frame type
DCNnMmMIDx	DCNnMmSSID[28:0]	Written by RX-Store machine according rules of acceptance filtering	Written by RX-Store machine as received in message
	DCNnMmSSIE	Set up by CPU during configuration	Set up at reception by RX-Store machine
DCNnMmCTL	DCNnMmMOWF DCNnMmIENF	Set up by CPU during configuration	Set up by CPU during configuration
	DCNnMmDTNF	Set by RX-Store machine and cleared by CPU	Set by RX-Store machine and cleared by CPU in Receive-only mode. Autonomously operated by RXONLY_CH in Mirror mode
	DCNnMmTRQF	Set by CPU (in ABT mode TRQ is set by ABT-engine autonomously)	Set by RXONLY_CH when operated in Mirror Mode
	DCNnMmRDYF	Set/cleared by CPU	Cleared by CPU before RXONLY_CH is put to Receive-only or Mirror mode. Then set again by CPU in order to enable storing messages.

The transition into RECONLY mode is described in *26.5.7 “Transition for operational modes of RXONLY_CH” on page 2048* .

(1) Processing received frames in Receive-only mode of RXONLY_CH

When the RXONLY_CH is operated in Receive-only mode, all valid messages are stored in the upper 16 message buffer. At initial start of that operating mode (i.e. when leaving INIT), the first message is stored in buffer 112. When resuming Receive-only mode leaving SLEEP mode, the storage will resume from the last position (message buffer) reached previously. If the user has set the option to get an interrupt, a receive interrupt is signalled to the host CPU.

The CPU can identify the most recently stored message with the help of the DCNnCRLISTR register. In case the CPU chooses to randomly poll the upper 16 message buffer for new messages and more than 1 message has its DCNnMmDTNF flag set, the sequence of their reception can be retrieved by a special algorithm.

In that case the CPU reads the DCNnCRLISTR register and decrements the message buffer number by one. Then CPU checks if the DCNnMmDTNF flag of that buffer is set. If not set, the buffer with the oldest message was one position ahead (in reverse direction of the CPU search).

If the DCNnMmDTNF flag is set the CPU repeats decrementing the buffer number with wrap around from buffer number 112 to 127 until it encounters a DN flag that is not set or until it reaches the buffer number that equals DCNnCRLISTR + 1. In that case the CPU reads DCNnCRLISTR again in order to find out if newly received messages have been stored while the CPU was searching for the oldest message.

Then, this secondly read value incremented by one points to the oldest message.

The CPU needs to clear the DCNnMmDTNF flag after processing the received messages in order to avoid an overflow situation, which generates an error interrupt. In case this interrupt occurs, the host CPU can directly start processing messages at buffer DCNnCRLISTR + 1, because the RXONLY_CH does not overwrite any buffer with DCNnMmDTNF = 1. Once the CPU has cleared the DCNnMmDTNF flag of buffer DCNnCRLISTR + 1, the RXONLY_CH will resume storing received messages. Clearing DCNnMmDTNF flags at other locations beforehand, will not let the reception process resume because the RXONLY_CH always stores messages in ascending order starting at DCNnCRLISTR + 1.

The flow chart below illustrates the necessary algorithms for the software.

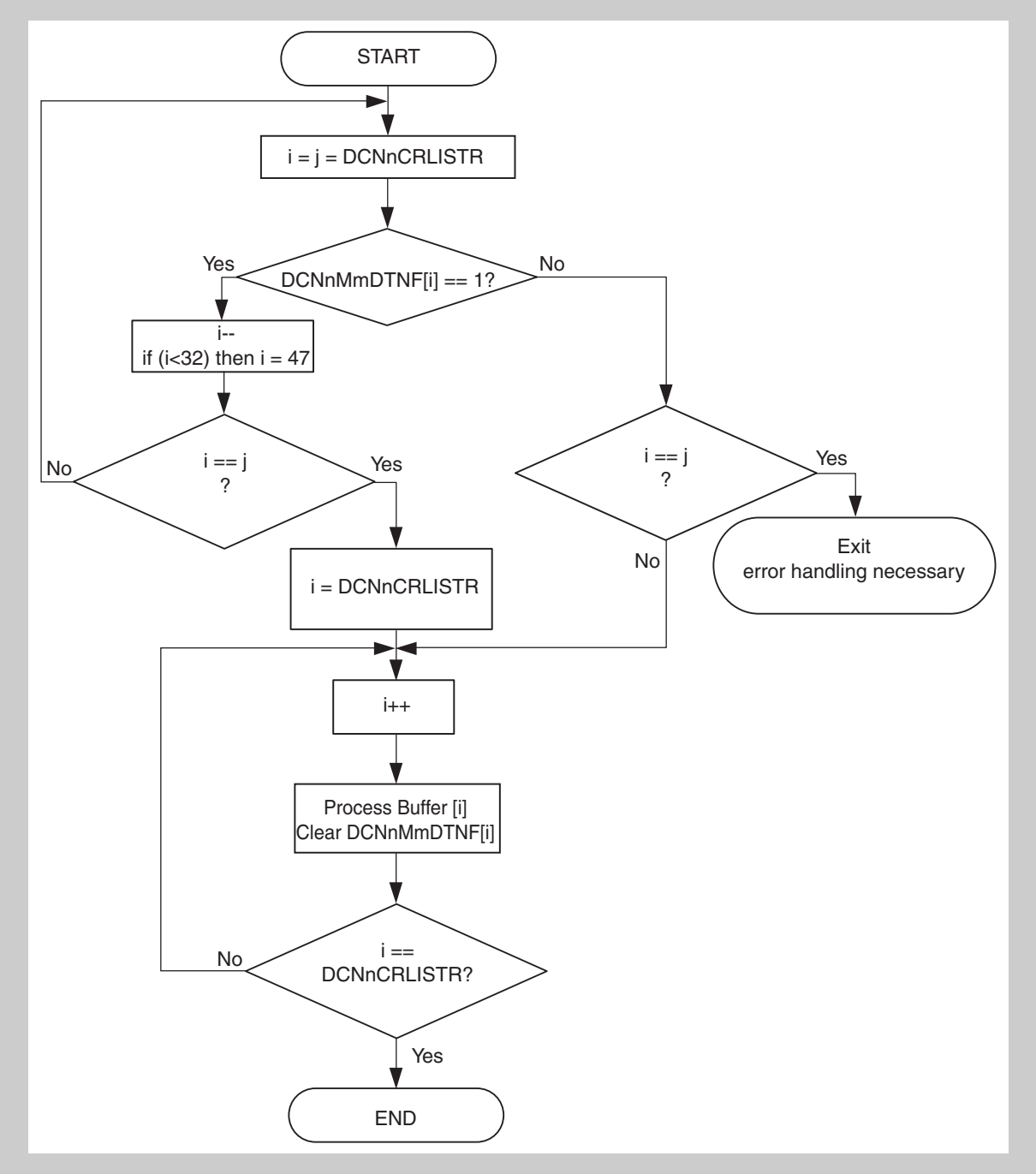


Figure 26-22 Receive operation for RXONLY_CH in receive-only mode

The exit that requires error handling is normally not encountered. In that case the application would process a receive interrupt but did not find any newly received message. Typically, the application has failed to clear the interrupt pending bit in a previous interrupt routine.

26.9.2 Mirror mode

The DCN module features a message mirroring function. It automatically copies messages from the RXONLY_CH to the bus served by the DIAG_CH. When this function, the mirror mode, is enabled, any valid data frame and any valid remote frame received by the RXONLY_CH will be stored in the upper 16 message buffers in the same manner as in Receive-only mode. From that location these messages are automatically transmitted by the DIAG_CH in FIFO manner. No acceptance filtering is applied to the frame receptions.

- The message mirroring is activated when the DCN module is configured as the follows:
- The operation mode of RXONLY_CH CAN machine is configured as “Mirror mode”.
- The operation mode of DIAG_CH CAN machine is configured as either “Normal operation mode” or “Normal operation mode with ABT”
- Neither of the CAN machines are in power save mode.

Other settings for the operation mode for the DIAG_CH I.e., SSHT are not forbidden, but they my lead to unexpected behaviour i.e. in case of transient bus errors on the diagnosis bus the mirrored message will not be repeated and thus be not visible although it was successfully communicated on the bus that was monitored. The application is strongly recommended to use only the configuration as mentioned in the listing above.

The mirror mode engine sets one DCNnMmTRQF at a time sequentially in FIFO manner. So there is only one DCNnMmTRQF bit, which is set (1) in the upper 16 buffers when mirroring is enabled. The application must not set any DCNnMmTRQF bit in this upper 16 buffers when mirroring or RECONLY mode is enabled. Neither the application shall leave a DCNnMmTRQF bit set (1) when the mirror mode, mirror mode with TIF or Receive-only mode shall be entered next.

Even when the mirroring function is enabled, the TX-search of the DIAG_CH will be applied to the upper 16 message buffers although they are assigned to the RXONLY_CH. The search engine will conduct the TX-search on the lower buffers and the candidate from the upper 16 buffers. Thus, the mirrored messages can suffer a delay before they can be seen on the diagnosis CAN bus depending on the message priority loaded in the lower buffers.

(1) Operations of the mirror mode engine

The following paragraphs describe the operation of the mirror mode engine (MME).

After RESET the DCNnCRLIST points to the buffer number #112 where the first frame from the RXONLY_CH is stored. As soon as the DCNnMmDTNF flag is set (1), the MME starts to operate using current DCNnCRLIST value. Note that DCNnCRLISTR value can be different as #112 when the user resumes Mirror mode from SLEEP mode of RXONYLY_CH.

If there is no other copy process from a previously handled message from the upper 16 buffers pending, the DCNnMmTRQF flag of the buffer with DCNnMmDTNF = 1 is set (1). The MME simultaneously memorises that a mirror mode operation is in progress (internal MMP flag is set) in order to queue additional requests (newly received messages from RXONLY_CH) for later execution. Up to 16 requests are stored by the MME. The MMP flag is cleared when all pending requests have been dealt with.

When the transmission by the DIAG_CH is finished, the MME clears the respective DCNnMmDTNF flag and it clears the current request for the MME

(i.e. decrement the up down counter). The buffer number can be obtained from THL of DIAG_CH.

If no further requests are pending (MMP = 0) the MME turns idle until the RXONLY_CH receives a new message. If there are other requests pending, the MME walks to the next buffer (in RX-Store direction of RXONLY_CH), writes MMP = 1, and sets the DCNnMmTRQF for that message buffer.

The 'transmission complete' state can be easily detected by the TX-pending signal for the TX-complete interrupt. Thus, in Mirror mode the interrupt enable bit has to be set for all upper 16 message buffers. This configuration will also generate RX-interrupts from RXONLY_CH. However the application can choose to suppress any interrupts generated by new receptions on the RXONLY_CH by masking the respective interrupt vector individually in the system interrupt controller or in the DCNnCMIECTL register of the CAN Controller. The TX-complete interrupt can not be masked completely because the same interrupt vector is also used for the transmissions for the lower buffer numbers (#0 - #111). However the interrupt can be masked partially by the host CPU. For the transmissions from the upper message buffers the CPU can choose to mask the interrupt generation separately. This partial mask is only effective for transmit interrupt of the DIAG_CH in Mirror mode, i.e. when the upper 16 message buffer are assigned to the RXONLY_CH. Thus the CPU can configure both interrupts (RX for REXONLY_CH and sub-mask (#112 - #127) for TX-complete on DIAG_CH) such that it is able to follow up the Mirror mode operations (intrusive operation) or not (i.e. non-intrusive operation of Mirror mode). Any TX-complete signal generated by the upper 16 buffers, activates the MME. Then the MME clears the TX-interrupt pending signal before it clears the DCNnMmDTNF flag, and it decrements the MMP-counter as described above.

Note that the MMP flag is cleared when Mirror Mode is cancelled by a transition to INIT mode of RXONLY_CH. Any changes of the operational or power save modes of the DIAG_CH and switching the RXONLY_MODE to SLEEP and further to STOP mode have no influence on MMP flag. Thus the mirror mode can be resumed later on at the position where it suspended.

The following diagrams illustrates the phases of operation in Mirror mode.

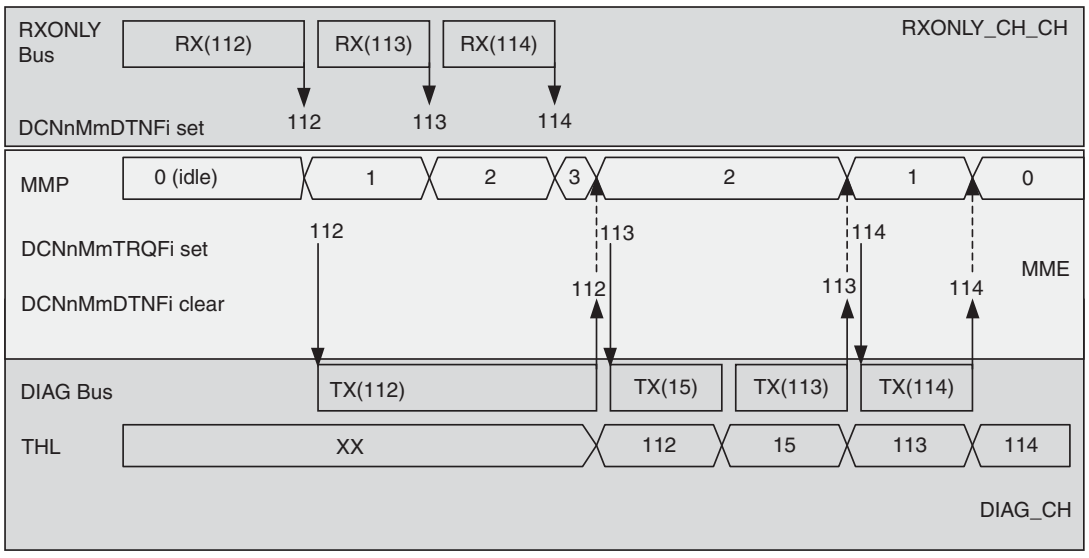


Figure 26-23 Mirror mode without application communication on DIAG_CH

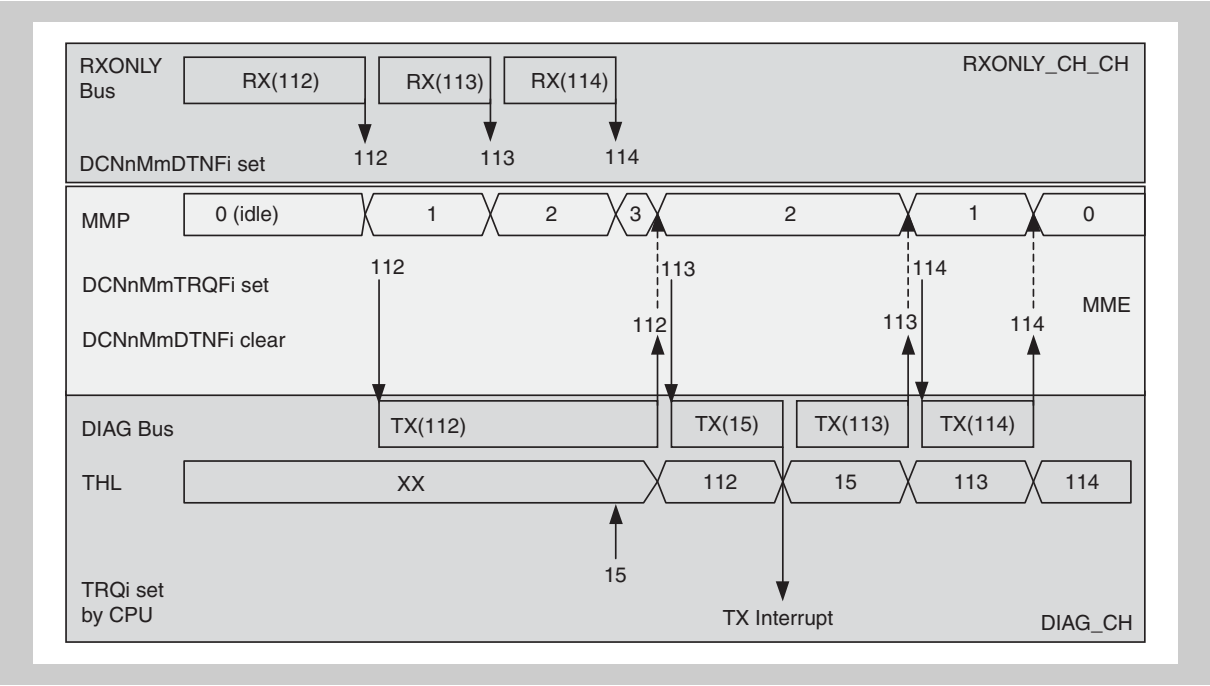


Figure 26-24 Mirror mode with interleaved application communication on DIAG_CH

The RXONLY_CH can monitor up to eight other CAN channels. These sources can be selected via the DCNnCRBSSSEL[2:0] bits in the DCNnCRBSSTR register during initialisation mode. Changing the selection during other operational modes is inhibited.

26.9.3 Mirror mode with TIF

The mirror mode with TIF (Transfer Identifier filtering) applies acceptance filtering to the messages received on RXONLY_CH. All other operations are identical to Mirror Mode (operation mode = 110_B).

Up to 8 FullCAN identifiers can be specified in the transfer ID reference registers. Additionally one mask (Transfer ID Mask) register can be linked individually to each of the reference registers.

Any message matching one of the filter criteria will be stored in the upper 16 message buffer and thus participate in the mirror mode function. All messages not matching any filter criteria are not stored. This feature provides the application with a tool that minimizes the data throughput to a relevant subset of messages.

26.10 Transitions for Buffer Assignment

The DCN module provides for each channel several operational modes and several power save modes. When considering all possible states with respect to the independent settings for the DIAG_CH and the RXONLY_CH, a huge number of transitions are basically available. However, the majority of mode transitions are not possible because i.e. the direct transition from NORMAL to NORMAL with ABT is not allowed, as well as STOP mode can not be directly followed by NORMAL mode.

This chapter describes the operations the host CPU has to perform in order to correctly switch between power save and operational modes of any of the two CAN channels of the DCN module. The flow chart below sketches the algorithm the host CPU needs to issue any time the buffer assignment changes.

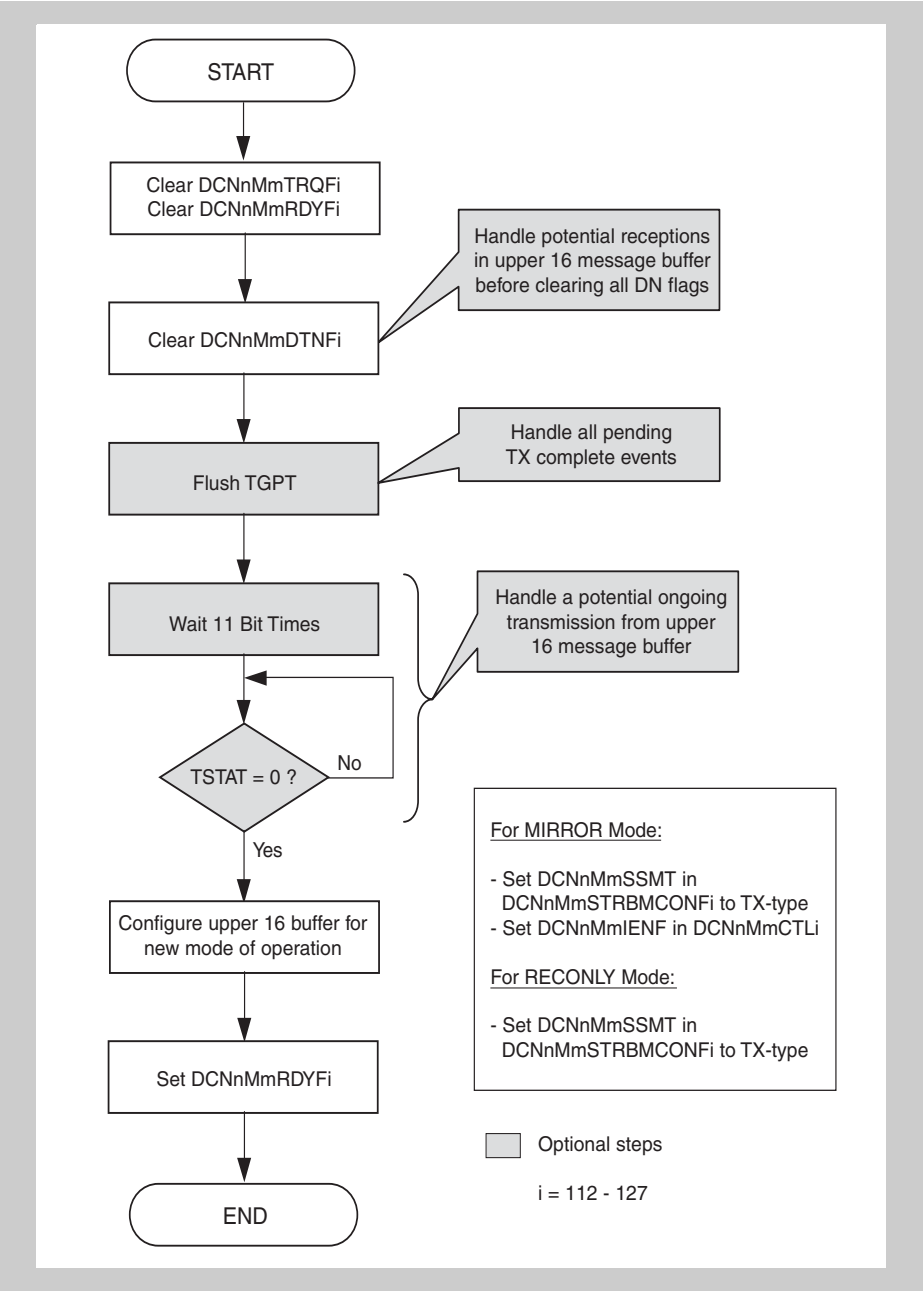


Figure 26-25 Changing the assignment of the upper 16 buffer

Whenever the assignment of the upper 16 buffers shall be changed, the host CPU needs to clear their DCNnMmRDYF, DCNnMmTRQF, and DCNnMmDTNF bits at first. Before the DN flags are cleared, the application can optionally serve newly received messages.

In a second step the TGPT needs to be flushed, which forces the host CPU to handle all pending TX-complete events. If the application does not care on these events, this step can be skipped.

The third step is again optional and only applicable if the application used at least one of the upper
16 buffers for transmitting application messages. If the application likes to follow the sequence of these messages, it needs to wait 11 bit times and then check for bus idle. After this wait-algorithm any ongoing TX-messages from the DIAG_CH is completed and the transition flow can step ahead.

Then, the new buffer configuration is set up. When switching to RXONLY_CH assignment, all upper
16 buffers need to be configured as TX-type. If additionally Mirror mode shall be run, the DCNnMmIENF bit of these buffers need to be set.

When switching the assignment to the DIAG_CH the application can setup its individual configuration.

In the second but last step, the new operational mode and the new power save mode for any of the channels can be invoked.

Finally, the DCNnMmRDYF bits of all upper 16 message buffers need to be set again when the assignment is switched to the RXONLY_CH. In opposite direction (DIAG_CH) the DCNnMmRDYF bits can be configured according the specific needs of the application.

26.11 Register Description

26.11.1 Register bit configuration

Not all registers will be discussed in detail within this chapter. Although all registers will be listed in the overview tables, the descriptions of registers, which have the same functionality as in the standard CAN Controller channels can be found in the CAN Controller chapter of this manual.

Table 26-12 DCN global 16-bit registers bit configuration (1/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
$0\ 8000_H$	DCNnGMCLCTL (W)	0	0	DCNnGMCLCLMB		0	0	0	DCNnGMCLCLOM
		0	0	0	DCNnGMCLSESR	0	0	DCNnGMCLSESD	DCNnGMCLSEOM
	DCNnGMCLCTL (R)	0	0	DCNnGMCLECCF	DCNnGMCLSORF	0	0	DCNnGMCLESDE	DCNnGMCLPWOM
		DCN0GMCLSSMO	0	0	0	0	0	0	0
$0\ 0008_H$	DCNnGMCSPRE	0	0	0		DCNnGMCSPRSC[3:0]			
$0\ 8018_H$	DCNnGMABCTL (W)	0	0	0	0	0	0	0	DCNnGMABCLAT
		0	0	0	0	0	0	DCNnGMABSEAC	DCNnGMABSEAT
	DCNnGMABCTL (R)	0	0	0	0	0	0	DCNnGMABCLRF	DCNnGMABABTT
		0	0	0	0	0	0	0	0
$0\ 0020_H$	DCNnGMADCTL	0	0	0	0	DCNnGMADSSAD[3:0]			
$0\ 8030_H$	DCNnTIDRTX00H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
$0\ 8038_H$	DCNnTIDRTX01H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$0\ 8040_H$	DCNnTIDRTX02H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
$0\ 8048_H$	DCNnTIDRTX03H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$0\ 8050_H$	DCNnTIDRTX04H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
$0\ 8058_H$	DCNnTIDRTX05H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$0\ 8060_H$	DCNnTIDRTX06H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							

Table 26-12 DCN global 16-bit registers bit configuration (2/2)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 8068$_H$	DCNnTIDRTX07H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
0 8070$_H$	DCNnTIDRTX08H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
0 8078$_H$	DCNnTIDRTX09H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
0 8080$_H$	DCNnTIDRTX10H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
0 8088$_H$	DCNnTIDRTX11H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
0 8090$_H$	DCNnTIDRTX12H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
0 8098$_H$	DCNnTIDRTX13H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
0 80A0$_H$	DCNnTIDRTX14H	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
0 80A8$_H$	DCNnTIDRTX15H	DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
0 80B0$_H$	DCNnTIDMTX0H	DCNnTIDMSSMA[7:0]							
		DCNnTIDMSSMA[15:8]							
0 80B8$_H$	DCNnTIDMTX01H	DCNnTIDMSSMA[23:16]							
		0	0	0	DCNnTIDMSSMA[28:24]				

Table 26-13 DCN global 32-bit registers bit configuration (1/3)

Address offset	Symbol	Bit 7/15/23/31	Bit 6/14/22/30	Bit 5/13/21/29/21	Bit 4/12/20/28	Bit 3/11/19/27	Bit 2/10/18/26	Bit 1/9/17/25	Bit 0/8/16/24
10 0030$_H$	DCNnTIDRTX00W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
10 0040$_H$	DCNnTIDRTX02W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				

Table 26-13 DCN global 32-bit registers bit configuration (2/3)

Address offset	Symbol	Bit 7/ 15/23/ 31	Bit 6/ 14/22/ 30	Bit 5/ 13/21/ 29/21	Bit 4/ 12/20/ 28	Bit 3/ 11/19/ 27	Bit 2/ 10/18/ 26	Bit 1/9/ 17/25	Bit 0/8/ 16/24
$10\ 0050_H$	DCNnTIDRTX04W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 0060_H$	DCNnTIDRTX06W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 0070_H$	DCNnTIDRTX08W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 0080_H$	DCNnTIDRTX10W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 0090_H$	DCNnTIDRTX12W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 00A0_H$	DCNnTIDRTX14W	DCNnTIDRSSID[7:0]							
		DCNnTIDRSSID[15:8]							
		DCNnTIDRSSID[23:16]							
		DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				
$10\ 00B0_H$	DCNnTIDMTX0W	DCNnTIDMSSMA[7:0]							
		DCNnTIDMSSMA[15:8]							
		DCNnTIDMSSMA[23:16]							
		0	0	0	DCNnTIDMSSMA[28:24]				
$10\ 00C0_H$	DCNnDNBMRX0W	DCNnDNBMSSDN[7:0]							
		DCNnDNBMSSDN[15:8]							
		DCNnDNBMSSDN[23:16]							
		DCNnDNBMSSDN[31:24]							
$10\ 00D0_H$	DCNnDNBMRX1W	DCNnDNBMSSDN[7:0]							
		DCNnDNBMSSDN[15:8]							
		DCNnDNBMSSDN[23:16]							
		DCNnDNBMSSDN[31:24]							

Table 26-13 DCN global 32-bit registers bit configuration (3/3)

Address offset	Symbol	Bit 7/ 15/23/ 31	Bit 6/ 14/22/ 30	Bit 5/ 13/21/ 29/21	Bit 4/ 12/20/ 28	Bit 3/ 11/19/ 27	Bit 2/ 10/18/ 26	Bit 1/9/ 17/25	Bit 0/8/ 16/24
$10\ 00E0_H$	DCNnDNBMRX2W	DCNnDNBMSSDN[7:0]							
		DCNnDNBMSSDN[15:8]							
		DCNnDNBMSSDN[23:16]							
		DCNnDNBMSSDN[31:24]							
$10\ 00F0_H$	DCNnDNBMRX3W	DCNnDNBMSSDN[7:0]							
		DCNnDNBMSSDN[15:8]							
		DCNnDNBMSSDN[23:16]							
		DCNnDNBMSSDN[31:24]							

Table 26-14 DCN module register bit configuration (1/3)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
$0\ 00248_H$	DCNnCM LCSTR (W)	0	0	0	0	0	0	0	0
	DCNnCM LCSTR (R)	0	0	0	0	0	FCN0CMLCSSLC[2:0]		
0 024CH	DCNnCM INSTR	0	0	0	DCNnCMINBOFF	DCNnCMINSSTE[1:0]		DCNnCMINSSRE[1:0]	
$0\ 8250_H$	DCNnCM ERCNT	DCNnCMERTECF[7:0]							
		DCNnCM ERRPSF	DCNnCMERRECF[6:0]						
$0\ 0268_H$	DCNnCM BRPRS	DCNnCMBRPRS[7:0]							
$0\ 0278_H$	DCNnCM LISTR	DCNnCMLISSLR[7:0]							
$0\ 0288_H$	DCNnCM LOSTR	DCNnCMLOSSLT[7:0]							
$0\ 002A8_H$	DCNnCR LCSTR (W)	0	0	0	0	0	0	0	0
	DCNnCR LCSTR (R)	0	0	0	0	0	DCNnCRLCSSLC[2:0]		
$0\ 02C8_H$	DCNnCR BRPRS	DCNnCRBRPRS[7:0]							
$0\ 02D8_H$	DCNnCR LISTR	DCNnCRLISSLR[7:0]							
$0\ 02E8_H$	DCNnCR BSSTR						DCNnCRBSSSEL[2:0]		

Table 26-14 DCN module register bit configuration (2/3)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 8240$_H$	DCNnCM CLCTL (W)	0	DCNnCM CLCLAL	DCNnCM CLCLVL	DCNnCMCLCLPS[1:0]		DCNnCMCLCLOP[2:0]		
		DCNnCM CLSERC	DCNnCM CLSEAL	0	DCNnCMSESEPS[1:0]		DCNnCMCSELOP[2:0]		
	DCNnCM CLCTL (R)	DCNnCM CLERCF	DCNnCM CLALBF	DCNnCM CLVALF	DCNnCMCLMDPF[1:0]		DCNnCMCLMDOF[2:0]		
		0	0	0	0	0	0	DCNnCMC LSSRS	DCNnCMC LSSTS
0 8258$_H$	DCNnCM IECTL (W)	DCNnCMI ECLIE7	DCNnCMIECLIE[6:0]						
		0	DCNnCMIESEIE[6:0]						
	DCNnCM IECTL (R)	DCNnCMI EINTF7	DCNnCMIEINTF[6:0]						
		0	0	0	0	0	0	0	0
0 8260$_H$	DCNnCM ISCTL (W)	DCNnCMI SCLTS7	DCNnCMISCLTS[6:0]						
		0	0	0	0	0	0	0	0
	DCNnCM ISCTL (R)	DCNnCMI SITSF7	DCNnCMISITSF[6:0]						
		0	0	0	0	0	0	0	0
0 8270$_H$	DCNnCM BTCTL	0	0	0	0	DCNnCMBTS1LG[3:0]			
		0	0	DCNnCMBTJWLG[1:0]		0	DCNnCMBTS2LG[2:0]		
0 8280$_H$	DCNnCM RGRX (W)	0	0	0	0	0	0	0	DCNnCMR GCLRV
		0	0	0	0	0	0	0	0
	DCNnCM RGRX (R)	0	0	0	0	0	0	DCNnCMR GSSPM	DCNnCMR GRVFF
		DCNnCMRGSSPT[7:0]							
0 8290$_H$	DCNnCM TGTX (W)	0	0	0	0	0	0	0	DCNnCMT GCLTV
		0	0	0	0	0	0	0	0
	DCNnCM TGTX (R)	0	0	0	0	0	0	DCNnCMT GSSPM	DCNnCMT GTVFF
		DCNnCMTGSSPT[7:0]							
0 8298$_H$	DCNnCM TSCTL (W)	0	0	0	0	0	DCNnCMT SCLLK	DCNnCMT SCLSL	DCNnCMT SCLTS
		0	0	0	0	0	DCNnCMT SSELK	DCNnCMT SSESL	DCNnCMT SSETS
	DCNnCM TSCTL (R)	0	0	0	0	0	DCNnCMT SLOKE	DCNnCMT SSELE	DCNnCMT STSGE
		0	0	0	0	0	0	0	0

Table 26-14 DCN module register bit configuration (3/3)

Address offset	Symbol	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8
0 82A0$_H$	DCNnCR CLCTL (W)	0	0	DCNnCR CLCLVL	DCNnCRCLCLPS[1:0]		DCNnCRCLCLOP[2:0]		
		DCNnCR CLSERC	0	0	DCNnCRSESEPS[1:0]		DCNnCRCSELOP[2:0]		
	DCNnCR CLCTL (R)	DCNnCR CLERCF	0	DCNnCR CLVALF	DCNnCRCLMDPF[1:0]		DCNnCRCLMDOF[2:0]		
		0	0	0	0	0	0	DCNnCRC LSSRS	0
0 82B0$_H$	DCNnCR ERCNT	0	0	0	0	0	0	0	0
		DCNnCR ERRPSF	DCNnCRERRECF[6:0]						
0 82B8$_H$	DCNnCR IECTL (W)	0	DCNnCRIECLIE[6:5]		0	DCNnCRIECLIE[3:1]			0
		0	DCNnCRIESEIE[6:5]		0	DCNnCRIESEIE[3:1]			0
	DCNnCR IECTL (R)	0	DCNnCRIEINTF[6:5]		0	DCNnCRIEINTF[3:1]			0
		0	0	0	0	0	0	0	0
0 82C0$_H$	DCNnCR ISCTL (W)	0	DCNnCRISCLTS[6:5]		0	DCNnCRISCLTS[3:1]			0
		0	0	0	0	0	0	0	0
	DCNnCR ISCTL (R)	0	DCNnCRISITSF[6:5]		0	DCNnCRISITSF[3:1]			0
		0	0	0	0	0	0	0	0
0 82D0$_H$	DCNnCR BTCTL	0	0	0	0	DCNnCRBTS1LG[3:0]			
		0	0	DCNnCRBTJWLG[1:0]		0	DCNnCRBTS2LG[2:0]		
0 82E0$_H$	DCNnCR TSCTL (W)	0	0	0	0	0	DCNnCRT SCLLK	DCNnCRT SCLSL	DCNnCRT SCLTS
		0	0	0	0	0	DCNnCRT SSELK	DCNnCRT SSESL	DCNnCRT SSETS
	DCNnCR TSCTL (R)	0	0	0	0	0	DCNnCRT SLOKE	DCNnCRT SSELE	DCNnCRT STSGE
		0	0	0	0	0	0	0	0

Table 26-15 DCN module mask control 16-bit registers bit configuration

Address offset	Symbol	Bit 15	Bit 14	Bit 13	Bit 12 to 0
0 8300$_H$	DCNnCMMKCTL01H	DCNnCMMKSSID[15:0]			
0 8308$_H$	DCNnCMMKCTL02H	0	0	0	DCNnCMMKSSID[28:16]
0 8310$_H$	DCNnCMMKCTL03H	DCNnCMMKSSID[15:0]			
0 8318$_H$	DCNnCMMKCTL04H	0	0	0	DCNnCMMKSSID[28:16]
0 8320$_H$	DCNnCMMKCTL05H	DCNnCMMKSSID[15:0]			
0 8328$_H$	DCNnCMMKCTL06H	0	0	0	DCNnCMMKSSID[28:16]
0 8330$_H$	DCNnCMMKCTL07H	DCNnCMMKSSID[15:0]			
0 8338$_H$	DCNnCMMKCTL08H	0	0	0	DCNnCMMKSSID[28:16]
0 8340$_H$	DCNnCMMKCTL09H	DCNnCMMKSSID[15:0]			
0 8348$_H$	DCNnCMMKCTL10H	0	0	0	DCNnCMMKSSID[28:16]
0 8350$_H$	DCNnCMMKCTL11H	DCNnCMMKSSID[15:0]			
0 8358$_H$	DCNnCMMKCTL12H	0	0	0	DCNnCMMKSSID[28:16]
0 8360$_H$	DCNnCMMKCTL13H	DCNnCMMKSSID[15:0]			
0 8368$_H$	DCNnCMMKCTL14H	0	0	0	DCNnCMMKSSID[28:16]
0 8370$_H$	DCNnCMMKCTL15H	DCNnCMMKSSID[15:0]			
0 8378$_H$	DCNnCMMKCTL16H	0	0	0	DCNnCMMKSSID[28:16]

Table 26-16 DCN module mask control 32-bit registers bit configuration (1/2)

Address offset	Symbol	Bit 31	Bit 30	Bit 29	Bit 28 to 0
1 0300$_H$	DCNnCMMKCTL01W	0	0	0	DCNnCMMKSSID[28:0]
1 0310$_H$	DCNnCMMKCTL03W	0	0	0	DCNnCMMKSSID[28:0]
1 0320$_H$	DCNnCMMKCTL05W	0	0	0	DCNnCMMKSSID[28:0]
1 0330$_H$	DCNnCMMKCTL07W	0	0	0	DCNnCMMKSSID[28:0]
1 0340$_H$	DCNnCMMKCTL09W	0	0	0	DCNnCMMKSSID[28:0]

Table 26-16 DCN module mask control 32-bit registers bit configuration (2/2)

Address offset	Symbol	Bit 31	Bit 30	Bit 29	Bit 28 to 0
$1\ 0350_H$	DCNnCMMKCTL11W	0	0	0	DCNnCMMKSSID[28:0]
$1\ 0360_H$	DCNnCMMKCTL13W	0	0	0	DCNnCMMKSSID[28:0]
$1\ 0370_H$	DCNnCMMKCTL15W	0	0	0	DCNnCMMKSSID[28:0]

Table 26-17 Message buffer register bit configuration (1/2)

Address offset	Symbol	Bit 7/15/23/31	Bit 6/14/22/30	Bit 5/13/21/29	Bit 4/12/20/28	Bit 3/11/19/27	Bit 2/10/18/26	Bit 1/9/17/25	Bit 0/8/16/24
$1\ 0000_H + m \times 40_H$	DCNnMmDAT0W	DCNnMmSSD[07:00]							
		DCNnMmSSD[17:10]							
		DCNnMmSSD[27:00]							
		DCNnMmSSD[37:30]							
$0\ 1000_H + m \times 40_H$	DCNnMmDAT0H	DCNnMmSSD[07:00]							
		DCNnMmSSD[17:10]							
$0\ 0000_H + m \times 40_H$	DCNnMmDAT0B	DCNnMmSSD[07:00]							
$0\ 0004_H + m \times 40_H$	DCNnMmDAT1B	DCNnMmSSD[17:10]							
$0\ 1008_H + m \times 40_H$	DCNnMmDAT2H	DCNnMmSSD[27:20]							
		DCNnMmSSD[37:30]							
$0\ 0008_H + m \times 40_H$	DCNnMmDAT2B	DCNnMmSSD[27:20]							
$0\ 000C_H + m \times 40_H$	DCNnMmDAT3B	DCNnMmSSD[37:30]							
$1\ 0010_H + m \times 40_H$	DCNnMmDAT4W	DCNnMmSSD[47:40]							
		DCNnMmSSD[57:50]							
		DCNnMmSSD[67:60]							
		DCNnMmSSD[77:70]							
$0\ 1010_H + m \times 40_H$	DCNnMmDAT4H	DCNnMmSSD[47:40]							
		DCNnMmSSD[57:50]							
$0\ 0010_H + m \times 40_H$	DCNnMmDAT4B	DCNnMmSSD[47:40]							
$0\ 0014_H + m \times 40_H$	DCNnMmDAT5B	DCNnMmSSD[57:50]							
$0\ 1018_H + m \times 40_H$	DCNnMmDAT6H	DCNnMmSSD[67:60]							
		DCNnMmSSD[77:70]							
$0\ 0018_H + m \times 40_H$	DCNnMmDAT6B	DCNnMmSSD[67:60]							
$0\ 001C_H + m \times 40_H$	DCNnMmDAT7B	DCNnMmSSD[77:70]							
$0\ 0020_H + m \times 40_H$	DCNnMmDTLGB	0				DCNnMmDTLG[3:0]			

Table 26-17 Message buffer register bit configuration (2/2)

Address offset	Symbol	Bit 7/15/23/31	Bit 6/14/22/30	Bit 5/13/21/29	Bit 4/12/20/28	Bit 3/11/19/27	Bit 2/10/18/26	Bit 1/9/17/25	Bit 0/8/16/24
$0\ 0024_H$ + m x 40_H	DCNnMm STRB	DCNnMm SSOW	DCNnMmSSMT[3:0]				DCNnMm SSRT	0	DCNnMm SSAM
$0\ 1028_H$ + m x 40_H	DCNnMm MID0H	DCNnMmSSID[7:0]							
		DCNnMmSSID[15:8]							
$0\ 1030_H$ + m x 40_H	DCNnMm MID1H	DCNnMmSSID[23:16]							
		DCNnMm SSIE	0	0	DCNnMmSSID[28:24]				
$1\ 0028_H$ + m x 40_H	DCNnMm MID0W	DCNnMmSSID[7:0]							
		DCNnMmSSID[15:8]							
		DCNnMmSSID[23:16]							
		DCNnMm SSIE	0	0	DCNnMmSSID[28:24]				
$0\ 1038_H$ + m x 40_H	DCNnMmCTL (W)	0	DCNnMm CLNH	0	DCNnMm CLMW	DCNnMm CLIE	DCNnMm CLDN	DCNnMm CLTR	DCNnMm CLRY
		0	DCNnMm SENH	0	0	DCNnMm SEIE	DCNnMm SEDN	DCNnMm CSETR	DCNnMm SERY
	DCNnMmCTL (R)	0	DCNnMm NHMF	0	DCNnMm MOWF	DCNnMm IENF	DCNnMm DTNF	DCNnMm TRQF	DCNnMm RDYF
		0	0	DCNnMm MUCF	0	0	0	DCNnMm TCPF	0

26.11.2 DCN global registers

(1) DCNnGMCLCTL - DCNn global control register

Refer to the description *"FCNnGMCLCTL - FCNn global control register"* in the chapter *"CAN Controller (FCN)"*.

(2) DCNnGMCSPRE - DCNn global clock selection register

Refer to the description *"FCNnGMCSPRE - FCNn global clock selection register"* in the chapter *"CAN Controller (FCN)"*.

(3) DCNnGMADCTL - DCNn global automatic block transmission delay register

Refer to the description *"FCNnGMCSPRE - FCNn global clock selection register"* in the chapter *"CAN Controller (FCN)"*.

(4) DCNnGMADCTL - DCNn global automatic block transmission delay register

Refer to the description *"FCNnGMADCTL - FCNn global automatic block transmission delay register"* in the chapter *"CAN Controller (FCN)"*.

(5) DCNnTIDRTXxH/yW - DCN transfer ID reference registers

The DCNnTIDRTXxH and DCNnTIDRTXyW registers are used to set the filter criteria for mirror mode with TIF of RXONLY_CH. These registers become only effective for Mirror mode with TIF. The application needs to initialise all these registers before entering mirror mode withTIF.

Note x = 00 to 15
y = 00, 02, 04, 06, 08, 10, 12, 14

Access The DCNnTIDRTXxH registers can be read/written in 16-bit units.
The DCNnTIDRTXyW registers can be read/written in 32-bit units.

Address DCNnTIDRTX00H: <DCNn_base> + 0 8030_H
DCNnTIDRTX01H: <DCNn_base> + 0 8038_H
DCNnTIDRTX02H: <DCNn_base> + 0 8040_H
DCNnTIDRTX03H: <DCNn_base> + 0 8048_H
DCNnTIDRTX04H: <DCNn_base> + 0 8050_H
DCNnTIDRTX05H: <DCNn_base> + 0 8058_H
DCNnTIDRTX06H: <DCNn_base> + 0 8060_H
DCNnTIDRTX07H: <DCNn_base> + 0 8068_H
DCNnTIDRTX08H: <DCNn_base> + 0 8070_H
DCNnTIDRTX09H: <DCNn_base> + 0 8078_H
DCNnTIDRTX10H: <DCNn_base> + 0 8080_H
DCNnTIDRTX11H: <DCNn_base> + 0 8088_H
DCNnTIDRTX12H: <DCNn_base> + 0 8090_H
DCNnTIDRTX13H: <DCNn_base> + 0 8098_H
DCNnTIDRTX14H: <DCNn_base> + 0 $80A0_H$
DCNnTIDRTX15H: <DCNn_base> + 0 $80A8_H$

DCNnTIDRTX00W: <DCNn_base> + 1 0030_H
DCNnTIDRTX02W: <DCNn_base> + 1 0040_H
DCNnTIDRTX04W: <DCNn_base> + 1 0050_H
DCNnTIDRTX06W: <DCNn_base> + 1 0060_H
DCNnTIDRTX08W: <DCNn_base> + 1 0070_H
DCNnTIDRTX10W: <DCNn_base> + 1 0080_H
DCNnTIDRTX12W: <DCNn_base> + 1 0090_H
DCNnTIDRTX14W: <DCNn_base> + 1 $00A0_H$

Initial Value Undefined

(a) DCNnTIDRTXxH (x = 00 to 15)

x = 00, 02, 04, 06, 08, 10, 12, 14:

15	14	13	12	11	10	9	8
DCNnTIDRSSID[15:8]							

7	6	5	4	3	2	1	0
DCNnTIDRSSID[7:0]							

x = 01, 03, 05, 07, 09, 11, 13, 15:

15	14	13	12	11	10	9	8
DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				

7	6	5	4	3	2	1	0
DCNnTIDRSSID[23:16]							

(b) DCNnTIDRTXyW (y = 00, 02, 04, 06, 08, 10, 12, 14)

31	30	29	28	27	26	25	24
DCNnTIDRSSIM	DCNnTIDRSSIE	0	DCNnTIDRSSID[28:24]				

23	22	21	20	19	18	17	16
DCNnTIDRSSID[23:16]							

15	14	13	12	11	10	9	8
DCNnTIDRSSID[15:8]							

7	6	5	4	3	2	1	0
DCNnTIDRSSID[7:0]							

DCNnTIDRSSID[28:0]	Message ID reference for transfer message
DCNnTIDRSSID[28:18]	Range for the 11-bit Standard identifier values
DCNnTIDRSSID[28:0]	Range for the 29-bit Extended identifier values

DCNnTIDRSSIM	Identifier extension bit for transfer message
0	11-bit Standard identifier[a]
1	29-bit Extended identifier

a) The bits DCNnTIDRSSID[17:0] are not used and may contain undefined values.

DCNnTIDRSSIE	Mask enable bit for transfer message
0	CnTIDRL/H register without a link to the CnTIDML/H
1	CnTIDRL/H register with a link to the CnTIDML/H

Caution Be sure to write “0” to bit 13 of the DCNnTIDRTXxH (x = 01, 03, 05, 07, 09, 11, 13, 15) registers or bit 29 of the DCNnTIDRTXyW register.

Note When CPU attempts to write DCNnTIDRTXxH in a mode other than INIT, it is simply ignored.

(6) DCNnTIDMTX0H/1H - DCN transfer ID mask registers

These registers are used to extend the number of receivable messages into the upper 16 message buffers by masking part of the ID of a message and invalidating the ID of the masked part.

Access These registers can be read/written in 16-bit units.

Address DCNnTIDMTX0H:<DCNn_base> + 0 $80B0_H$
DCNnTIDMTX1H:<DCNn_base> + 0 $80B8_H$

Initial Value Undefined

(a) DCNnTIDMTX0H

15	14	13	12	11	10	9	8
DCNnTIDMSSMA[15:8]							

7	6	5	4	3	2	1	0
DCNnTIDMSSMA[7:0]							

(b) DCNnTIDMTX1H

15	14	13	12	11	10	9	8
0	0	0	DCNnTIDMSSMA[28:4]				

7	6	5	4	3	2	1	0
DCNnTIDMSSMA[23:16]							

DCNnTIDMSSMA[28:0]	Mask identifier pattern for transfer message
0	Corresponding identifier bit in the received message frame and in the message buffer must match.
1	Corresponding identifier bit in the received message frame and in the message do not care.

Note When CPU attempts to write DCNnTIDMTX0H/1H in a mode other than INIT, it is simply ignored.

(7) DCNnDNBMRXk - DCNn data new bit monitor registers (k = 0 to 3)

Refer to the description *"DCNnDNBMRXk - DCNn data new bit monitor registers"* in the chapter *"CAN Controller (FCN)"*.

26.11.3 DCN module registers

(1) DCNnCMMKCTLaH - DCNn module mask control register

Refer to the description *"FCNnCMMKCTLaH - FCNn module mask control register"* in the chapter *"CAN Controller (FCN)"*.

(2) DCNnCMCLCTL - DCNn module control register

Refer to the description *"FCNnCMCLCTL - FCNn module control register"* in the chapter *"CAN Controller (FCN)"*.

(3) DCNnCMLCSTR - DCNn module last error information register

Refer to the description *"FCNnCMLCSTR - FCNn module last error information register"* in the chapter *"CAN Controller (FCN)"*.

(4) DCNnCMINSTR - DCNn module information register

Refer to the description *"FCNnCMINSTR - FCNn module information register"* in the chapter *"CAN Controller (FCN)"*.

(5) DCNnCMERCNT - DCNn module error counter register

Refer to the description *"FCNnCMERCNT - FCNn module error counter register"* in the chapter *"CAN Controller (FCN)"*.

(6) DCNnCMIECTL - DCNn module interrupt enable register

Refer to the description *"FCNnCMIECTL - FCNn module interrupt enable register"* in the chapter *"CAN Controller (FCN)"*.

(7) DCNnCMISCTL - DCNn module interrupt status register

Refer to the description *"FCNnCMISCTL - FCNn module interrupt status register"* in the chapter *"CAN Controller (FCN)"*.

(8) DCNnCMBRPRS - DCNn module bit rate prescaler register

Refer to the description *"FCNnCMBRPRS - FCNn module bit rate prescaler register"* in the chapter *"CAN Controller (FCN)"*.

(9) DCNnCMBTCTL - DCNn module bit rate register

Refer to the description *"FCNnCMBTCTL - FCNn module bit rate register"* in the chapter *"CAN Controller (FCN)"*.

(10) DCNnCMLISTR - DCNn module last in-pointer register

Refer to the description *"FCNnCMLISTR - FCNn module last in-pointer register"* in the chapter *"CAN Controller (FCN)"*.

(11) DCNnCMRGRX - DCNn module receive history list register

Refer to the description *"FCNnCMRGRX - FCNn module receive history list register"* in the chapter *"CAN Controller (FCN)"*.

(12) DCNnCMLOSTR - DCNn module last out-pointer register

Refer to the description *"FCNnCMLOSTR - FCNn module last out-pointer register"* in the chapter *"CAN Controller (FCN)"*.

(13) DCNnCMTGTX - DCNn module transmit history list register

Refer to the description *"FCNnCMTGTX - FCNn module transmit history list register"* in the chapter *"CAN Controller (FCN)"*.

(14) DCNnCMTSCTL - DCNn module time stamp register

Refer to the description *"FCNnCMTSCTL - FCNn module time stamp register"* in the chapter *"CAN Controller (FCN)"*.

(15) DCNnCRCLCTL - DCNn module RXONLY_CH control register

This register is used to control the operation mode of the RXONLY_CH channel.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 $82A0_H$

Initial Value 0000_H

(a) DCNnCRCLCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	DCNnCR CLSSRS	0

7	6	5	4	3	2	1	0
DCNnCR CLERCF	0	DCNnCR CLVALF	DCNnCM CLRDPF[1:0]		DCNnCR CLMDOF[2:0]		

DCNnCRCLSSRS	RXONLY_CH reception status bit
0	No receive activity on the RXONLY CAN bus.
1	Receive activity on the RXONLY CAN bus.

Notes

1. DCNnCRCLSSRS is set to 1 under the following conditions (timing)
   - The SOF bit of a receive frame is detected
   - On occurrence of arbitration loss during a transmit frame
2. DCNnCRCLSSRS is cleared to 0 under the following conditions (timing)
   - When a recessive level is detected at the second bit of the interframe space
   - On transition to the initialization mode at the first bit of the interframe space

DCNnCRCLERCF	RXONLY_CH error counter clear bit
0	The DCNnCRERCNT and DCNnCRLCSTR registers are not cleared in the initialization mode.
1	The DCNnCRERCNT and DCNnCRLCSTR registers are cleared in the initialization mode.

Caution The error counter register DCNnCRERCNT and the information register DCNnCRINSTR are cleared by DCNnCRCLERCF bit only following conditions:

- During Bus-Off state, while in initialization mode.
- After starting the DCN module (DCNnGMCLPWOM is set at DCNnGMCLPWOM = 0), while in initialization mode.

Notes

1. When the DCNnCRERCNT and DCNnCRLCSTR registers have been cleared, DCNnCRCLERCF is also cleared to 0 automatically.

2. DCNnCRCLERCF can be set to 1 at the same time as a request to change the initialization mode to an operation mode is made.
3. DCNnCRCLERCF is read-only in the DCN sleep mode or DCN stop mode.
4. The error counters can also be cleared by shut down or forced shut down of the CAN controller.

DCNnCRCLVALF	RXONLY_CH valid receive message frame detection bit
0	A valid message frame has not been received since DCNnCRCLVALF was last cleared to 0.
1	A valid message frame has been received since DCNnCRCLVALF was last cleared to 0.

Notes

1. Detection of a valid receive message frame is not dependent upon storage in the receive message buffer (data frame) or transmit message buffer (remote frame).
2. Clear DCNnCRCLVALF (0) before changing the initialization mode to an operation mode.
3. If only two CAN nodes are connected to the CAN bus with one transmitting a message frame in the normal mode and the other in the receive-only mode, DCNnCRCLVALF is not set to 1 before the transmitting node enters the error passive state, because in receive-only mode no acknowledge is generated.
4. To clear DCNnCRCLVALF, set DCNnCRCLCLVL to 1 first and confirm that DCNnCRCLVALF is cleared. If it is not cleared, perform clearing processing again.

Cautions

1. User must not set the DCNnCRCLVALF bit (1) by using set DCNnCRCLVALF bit.
2. A valid reception does not require an acceptance of the message frame into a receive message buffer (data or remote frame).
3. In case only two CAN nodes are connected to a CAN bus and one of the CAN nodes is in “Normal Operating mode” transmitting message frames while the other CAN node is in “Receive-only mode”, the DCNnCRCLVALF bit will be set (1) not before the transmitting node becomes error passive.

DCNnCRCLMDPF[1:0]	RXONLY_CH power save mode
00_B	No power save mode is selected.
01_B	DCN sleep mode
10_B	Setting prohibited
11_B	DCN stop mode

Cautions

1. Transition to and from the DCN stop mode must be made via DCN sleep mode. A request for direct transition to and from theDCN stop mode is ignored.
2. The DCNnGMCLSSMO flag ofDCNnGMCLCTL must be checked after releasing a power save mode, prior to access the message buffers again.

3. DCN sleep mode requests are kept pending, until cancelled by software or entered on appropriate bus condition (bus idle). Software can check the actual status by reading DCNnCDCLMDPF[1:0].
4. In order to achieve sleep mode or stop mode for the whole DCN module, the power save mode of the DIAG_CH must be set, too. Only if DIAG_CH and RXONLY_CH are in power save mode, power consumption is reduced and indicated to the user by clearing the DCNnGMCLSSMO flag.

Note In case that the CAN bus is blocked on dominant level, so that the DCN module could not synchronize since initialization mode was left, the sleep mode can be reached nevertheless. In this case however, the wake up from sleep mode will happen after the first recessive to dominant edge, after a synchronization was successful. Waking up by software is possible in any case.

DCNnCRCLMDOF[2:0]	RXONLY_CH operation mode
000_B	No operation mode is selected (RXONLY_CH FCN module is in the initialization mode).
001_B	Setting prohibited
010_B	
011_B	Receive-only mode
100_B	Setting prohibited
101_B	
110_B	Mirror mode
111_B	Mirror mode with Transfer ID Filter function (with TIF)

Caution Transit to initialization mode or power saving modes may take some time. Be sure to verify the success of mode change by reading the values, before proceeding.

Note DCNnCRCLMDOF[2:0] are read-only in the DCN sleep mode or DCN stop mode.

(b) DCNnCRCLCTL write

15	14	13	12	11	10	9	8
DCNnCR CLSERC	0	0	DCNnCR SESEPS[1:0]		DCNnCR CLSEOP[2:0]		
7	6	5	4	3	2	1	0
0	0	DCNnCR CLCLVL	DCNnCR SECLPS[1:0]		DCNnCR CLCLOP[2:0]		

DCNnCRCLSERC	Setting of DCNnCRCLERCF bit
1	DCNnCRCLERCF is set to 1.
Other than above	DCNnCRCLERCF is not changed.

DCNnCRCLCLVL	Setting of DCNnCRCLVALF bit
0	DCNnCRCLVALF is not changed.
1	DCNnCRCLVALF is cleared to 0.

DCNnCRSESEPS0	DCNnCRSECLPS0	Setting of DCNnCRCLMDPF0 bit
0	1	DCNnCRCLMDPF0 is cleared to 0.
1	0	DCNnCRCLMDPF0 is set to 1.
Other than above		DCNnCRCLMDPF0 is not changed.

DCNnCRSESEPS1	DCNnCRSECLPS1	Setting of DCNnCRCLMDPF1 bit
0	1	DCNnCRCLMDPF1 is cleared to 0.
1	0	DCNnCRCLMDPF1 is set to 1.
Other than above		DCNnCRCLMDPF1 is not changed.

DCNnCRCLSEOP0	DCNnCRCLCLOP0	Setting of DCNnCRCLMDOF0 bit
0	1	DCNnCRCLMDOF0 is cleared to 0.
1	0	DCNnCRCLMDOF0 is set to 1.
Other than above		DCNnCRCLMDOF0 is not changed.

DCNnCRCLSEOP1	DCNnCRCLCLOP1	Setting of DCNnCRCLMDOF1 bit
0	1	DCNnCRCLMDOF1 is cleared to 0.
1	0	DCNnCRCLMDOF1 is set to 1.
Other than above		DCNnCRCLMDOF1 is not changed.

DCNnCRCLSEOP2	DCNnCRCLCLOP2	Setting of DCNnCRCLMDOF2 bit
0	1	DCNnCRCLMDOF2 is cleared to 0.
1	0	DCNnCRCLMDOF2 is set to 1.
Other than above		DCNnCRCLMDOF2 is not changed.

(16) DCNnCRLCSTR - DCNn module RXONLY_CH last error information register

This register provides the error information of the CAN protocol.

Access This register can be read/written in 8-bit units.

Address <DCNn_base> + 0 $02A8_H$

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	DCN0CRLCSSLC[2:0]		

Notes

1. The contents of the DCNnCRLCSTR register are not cleared when the DCN module changes from an operation mode to the initialization mode.
2. If an attempt is made to write a value other than 00_H to the DCNnCRLCSTR register by software, the access is ignored.

DCN0CRLCSSLC[2:0]	Last DCN protocol error information
000_B	No error
001_B	Stuff error
010_B	Form error
110_B	CRC error
all other	unused

(17) DCNnCEERCNT - DCNn module RXONLY_CH error counter register

This register indicates the count value of the transmission/reception error counter.

Access This register is read-only in 16-bit units.

Address <DCNn_base> + $0\ 82B0_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
DCNnCR ERRPSF	DCNnCR ERRECF[6:0]						
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0

DCNnCRERRPSF	Reception error passive status bit
0	The reception error counter is not in the error passive range (< 128)
1	The reception error counter is in the error passive range (≥ 128)

DCNnCRERRECF[6:0]	Reception error counter bit
0 to 127	Number of reception errors. These bits reflect the status of the reception error counter. The number of errors is defined by the CAN protocol.

Note DCNnCRERRECF[6:0] are invalid in the reception error passive state (DCNnCRINSTR.DCNnCRINSSRE[1:0] = 11_B).

(18) DCNnCRIECTL - DCNn module RXONLY_CH interrupt enable register

This register is used to enable or disable the interrupts of the DCN module.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 $82B8_H$

Initial Value 0000_H

(a) DCNnCRIECTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	DCNnCRIEINTF[6:5]		0	DCNnCRIEINTF[3:1]			0

DCNnCRIEINTF[6:5] DCNnCRIEINTF[3:1]	DCN module interrupt enable bit
0	Output of the interrupt corresponding to interrupt status registerDCNnCRISCTL is disabled.
1	Output of the interrupt corresponding to interrupt status register DCNnCRISCTL is enabled.

(b) DCNnCRIECTL write

15	14	13	12	11	10	9	8
0	DCNnCRIESEIE[6:5]		0	DCNnCRIESEIE[3:1]			0
7	6	5	4	3	2	1	0
0	DCNnCRIECLIE[6:5]		0	DCNnCRIECLIE[3:1]			0

DCNnCRIESEIE[6:5] DCNnCRIESEIE[3:1]	DCNnCRIECLIE[6:0] DCNnCRIECLIE[3:1]	Setting of DCNnCRIEINTF[6:5]/ DCNnCRIEINTF[3:1] bit
0	1	Corresponding bit is cleared to 0.
1	0	Corresponding bit is set to 1.
Other than above		Corresponding bit is not changed.

(19) DCNnCRISCTL - DCNn module RXONLY_CH interrupt status register

This register indicates the interrupt status of the DCN module.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 $82C0_H$

Initial Value 0000_H

(a) DCNnCRISCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	DCNnCRISITSF[6:5]		0	DCNnCRISITSF[3:1]			0

DCNnCRISITSF[6:5] DCNnCRISITSF[3:1]	RXONLY_CH interrupt status bit
0	No related interrupt source event is pending
1	A related interrupt source event is pending

Interrupt status bit	RXONLY_CH related interrupt source event
FCNnCMISITSF6	RXONLY_CH transmission abort interrupt status bit
FCNnCMISITSF5	RXONLY_CH wake-up interrupt from sleep mode [a]
FCNnCMISITSF3	RXONLY_CH protocol error interrupt
FCNnCMISITSF2	RXONLY_CH error status interrupt
FCNnCMISITSF1	Interrupt on completion of reception of valid message frame to message buffer m

a) DCNnCRISITSF5 is set only when the DCN module is woken up from the DCN sleep mode by a CAN bus operation. It is not set when the DCN sleep mode has been released by software.

(b) DCNnCRISCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	DCNnCRISITSF[6:5]		0	DCNnCRISITSF[3:1]			0

DCNnCRISITSF[6:5] DCNnCRISITSF[3:1]	Clearing of DCNnCRISITSF[6:5]/ DCNnCRISITSF[3:1] bits
0	Corresponding bits are not changed
1	Corresponding bits are cleared to 0

Caution Clear the status bit of this register by software, when the confirmation of each status is necessary in the interrupt processing, because these bits are not cleared automatically.

(20) DCNnCMBRPRS - DCNn module RXONLY_CH bit rate prescaler register

This register is used to select the CAN protocol layer basic system clock (f_{TQ}). The communication baud rate is set to the DCNnCRBTCTL register.

Access This register can be read/written in 8-bit units.

Address <DCNn_base> + $0\ 02C8_H$

Initial Value FF_H

7	6	5	4	3	2	1	0
DCNnCRBRPRS[7:0]							

DCNnCRBRPRS[7:0]	Time quanta clock prescaler
0	$f_{CANMOD}/1$
1	$f_{CANMOD}/2$
n	$f_{CANMOD}/(n+1)$
:	:
255	$f_{CANMOD}/256$ (default value)

(21) DCNnCRBTCTL - DCNn module RXONLY_CH bit rate register

This register is used to control the data bit time of the communication baud rate.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 $82D0_H$

Initial Value $0370F_H$

15	14	13	12	11	10	9	8
0	0	DCNnCR BTJWLG[1:0]		0	DCNnCR BTS2LG[2:0]		
7	6	5	4	3	2	1	0
0	0	0	0	DCNnCRBTS1LG[3:0]			

DCNnCRBTJWLG[1:0]	Length of synchronization jump width
00_B	$1T_Q$
01_B	$2T_Q$
10_B	$3T_Q$
11_B	$4T_Q$ (default value)

DCNnCRBTS2LG[2:0]	Length of time segment 2
000_B	$1T_Q$
001_B	$2T_Q$
010_B	$3T_Q$
011_B	$4T_Q$
100_B	$5T_Q$
101_B	$6T_Q$
110_B	$7T_Q$
111_B	$8T_Q$ (default value)

DCNnCRBTS1LG[3:0]	Length of time segment 1
0000_B	Setting prohibited
0001_B	$2T_Q$[a]
0010_B	$3T_Q$[a]
0011_B	$4T_Q$
0100_B	$5T_Q$
0101_B	$6T_Q$
0110_B	$7T_Q$
0111_B	$8T_Q$
1000_B	$9T_Q$
1001_B	$10T_Q$
1010_B	$11T_Q$
1011_B	$12T_Q$
1100_B	$13T_Q$
1101_B	$14T_Q$
110_B	$15T_Q$
1111_B	$16T_Q$ (default value)

a) This setting must not be made when the DCNnCRBRPRS register = 00_H

(22) DCNnCRLISTR - DCNn module RXONLY_CH last in-pointer register

This register indicates the number of the message buffer in which a data frame or a remote frame was last stored.

Access This register is read-only in 8-bit units.

Address <DCNn_base> + 0 $02D8_H$

Initial Value Undefined.

7	6	5	4	3	2	1	0
DCNnCRLISSLR[7:0]							

DCNnCRLISSLR[7:0]	Last in-pointer of the receive history list for the RXONLY_CH
0 to 127	Reading the DCNnCRLISTR register delivers the message buffer number in which the last data frame has been saved or the last remote frame has been stored into the message buffer area of RXONLY_CH. The user can only evaluate DCNnCRLISTR when both of the following conditions are met: • The CAN channel is in receive-only or mirror mode. • A newly received message on RXONLY_CH was signalled by interrupt CINTS1 in DCNnCRISCTL. As a pre-condition RXONLY_CH needs to have been put in INIT or STOP mode beforehand at least once. The usage of DCNnCRLISTR requires proper interrupt handling.

Note As long no data frame has been stored into a message buffer or no received remote frame has been assigned, an undefined value is read from DCNnCRLISTR register. The user must not read the DCNnCRLISTR register until first message acceptance to the buffer from RXONLY_CH after switching the DCNnGMCLCTL.DCNnGMCLPWOM bit.

This register shows the reception history of the messages from RXONLY_CH only.

(23) DCNnCRTSCTL - DCNn module RXONLY_CH time stamp register

This register is used to control the time stamp function.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 $82E8_H$

Initial Value 0000_H

(a) DCNnCRTSCTL read

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
7	6	5	4	3	2	1	0
0	0	0	0	0	DCNnCRTSLOKE	DCNnCRTSSELE	DCNnCRTSTSGE

Note The Basic Time Stamp function must not be used when the DIAG_CH CAN module operates in 'Normal Operating mode with Automatic Block Transmission.

DCNnCRTSLOKE	Time stamp lock function enable bit
0	The time stamp lock function is disabled. Upon every selected time stamp capture event the TSOUT_R signal is toggled.
1	The time stamp lock function is enabled. The TSOUT_R signal generation is locked after a data frame was successfully received in message buffer 0.

Caution The TSLOCK function on RXONLY_CH is only invoked when mirror mode, mirror mode with TIF or receive-only mode are active.

DCNnCRTSSELE	Time stamp capture event selection bit
0	The time stamp capture event is the SOF event (Start-of-Frame on the RXONLY_CH CAN-bus)
1	The time stamp capture event is the EOF event (the last bit of the End-of-Frame field. TSOUT_R signal is generated at the sample point of the last bit in the EOF field.

DCNnCRTSTSGE	TSOUT_R operation setting bit
0	TSOUT_R toggle operation is disabled.
1	TSOUT_R toggle operation is enabled.

(b) DCNnCRTSCTL write

15	14	13	12	11	10	9	8
0	0	0	0	0	DCNnCRTSSELK	DCNnCRTSSES	DCNnCRTSSETS
7	6	5	4	3	2	1	0
0	0	0	0	0	DCNnCRTSCLLK	DCNnCRTSCLSL	DCNnCRTSCLTS

DCNnCRTSSELK	DCNnCRTSCLLK	Setting of DCNnCRTSLOKE bit
0	1	DCNnCRTSLOKE is cleared to 0.
1	0	DCNnCRTSLOKE is set to 1.
Other than above		DCNnCRTSLOKE is not changed.

DCNnCRTSSES	DCNnCRTSCLSL	Setting of DCNnCRTSSELE bit
0	1	DCNnCRTSSELE is cleared to 0.
1	0	DCNnCRTSSELE is set to 1.
Other than above		DCNnCRTSSELE is not changed.

DCNnCRTSSETS	DCNnCRTSCLTS	Setting of DCNnCRTSTSGE bit
0	1	DCNnCRTSTSGE is cleared to 0.
1	0	DCNnCRTSTSGE is set to 1.
Other than above		DCNnCRTSTSGE is not changed.

(24) DCNnCRBSSTR - DCNn module RXONLY_CH bus selector register

This register selects the CAN bus input to be selected as input for the RXONLY_CH.

Access This register is read-only in 8-bit units.

Address <DCNn_base> + 0 $02E8_H$

Initial Value 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	DCNnCRBSSSEL[2:0]		

DCNnCRBSSSEL[2:0][a]	Bus selector for selecting the CAN bus to be monitored
000_B	DIAG_CH bus input
001_B	CAN-bus input 1
010_B	CAN-bus input 2
011_B	CAN-bus input 3
100_B	CAN-bus input 4
101_B	CAN-bus input 5
110_B	CAN-bus input 6
111_B	CAN-bus input 7

a) Refer to the key word “Internal signals” in the first section of this chapter for information which CAN busses are connected and how to select them.

Note When CPU attempts to write DCNnCRBSSTR in the mode other than INIT, it is simply ignored.

Caution When selecting the DIAG_CH bus input (for self-testing purposes), the RXONLY_CH may only be operated in RXONLY mode. Otherwise, endless circulation of CAN frames with high bus load on the DIAG_CH will result.

26.11.4 DCN message buffers registers

(1) DCNnMmSTRB - DCNn message configuration register m

Refer to the description *"FCNnMmSTRB - FCNn message configuration register m"* in the chapter *"CAN Controller (FCN)"*.

(2) DCNnMmMID0H, DCNnMmMID1H, DCNnMmMID0W - DCNn message ID register m

These registers are used to set an identifier (ID).

Access DCNnMmMID0H, DCNnMmMID1H can be read/written in 16-bit units.
DCNnMmMID0W can be read/written in 32-bit units.

Address DCNnMmMID0H: <DCNn_base> + 0 1028_H + m x 40_H
DCNnMmMID1H: <DCNn_base> + 0 1030_H + m x 40_H

DCNnMmMID0W: <DCNn_base> + 1 0028_H + m x 40_H

Initial Value Undefined

(a) DCNnMmMID0H

15	14	13	12	11	10	9	8
DCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
DCNnMmSSID[7:0]							

(b) DCNnMmMID1H

15	14	13	12	11	10	9	8
DCNnMmSSIE	0	0	DCNnMmSSID[28:24]				

7	6	5	4	3	2	1	0
DCNnMmSSID[23:16]							

(c) DCNnCMmMID0W

31	30	29	28	27	26	25	24
DCNnMmSSIE	0	0	DCNnMmSSID[28:24]				

23	22	21	20	19	18	17	16
DCNnMmSSID[23:16]							

15	14	13	12	11	10	9	8
DCNnMmSSID[15:8]							

7	6	5	4	3	2	1	0
DCNnMmSSID[7:0]							

DCNnMmSSIE	Format mode specification bit
0	Standard format mode (DCNnMmSSID[28:18]: 11 bits, DCNnMmSSID[17:0] are not used)
1	Extended format mode (DCNnMmSSID[28:0]: 29 bits)

DCNnMmSSID[28:0]	Message ID
DCNnMmSSID[28:18]	Standard ID value of 11 bits (when DCNnMmSSIE = 0)
DCNnMmSSID[28:0]	Extended ID value of 29 bits (when DCNnMmSSIE = 1)

Note When RXONLY_CH is operated in Mirror or Receive-only mode, the IDE bit will be overwritten with the actually received frame format (i.e. standard or extended).

Cautions

1. Be sure to write 0 to bits 14 and 13 of DCNnMmMID1H, respectively bits 30 and 29 of DCNnMmMID0W register.
2. Be sure to align the ID value according to the given bit positions into this registers. Note that for standard ID, the ID value must be shifted to fit into DCNnMmSSID[28:18] bit positions.

(3) DCNnMmCTL - DCNn message control register m

This register is used to control the operation of the message buffer.

Access This register can be read/written in 16-bit units.

Address <DCNn_base> + 0 1038_H + m x 40_H

Initial Value 0000_B

(a) DCNnMmCTL read

15	14	13	12	11	10	9	8
0	0	DCNnMmMUCF	0	0	0	DCNnMmTCPF	0
7	6	5	4	3	2	1	0
0	DCNnMmNHMF	0	DCNnMmMOWF	DCNnMmIENF	DCNnMmDTNF	DCNnMmTRQF	DCNnMmRDYF

DCNnMmNHMF	History mask flag [a]
0	Updates of receive/transmit history list registers DCNnCMRGRX/DCNnCMRGTX are not masked.
1	Updates of receive/transmit history list registers DCNnCMRGRX/DCNnCMRGTX are masked.

a) When masked, the history lists are not updated upon reception or transmission completion activity on this message buffer.

DCNnMmMUCF[a]	Bit indicating that message buffer data is being updated
0	The DCN module is not updating the message buffer (reception and storage).
1	The DCN module is updating the message buffer (reception and storage).

a) DCNnMmMUCF is undefined until the first reception and storage is performed.

DCNnMmTCPF[a]	Transmission completely finished flag
0	Transmission failed.[b] [c]
1	Transmission (including ABT) completely finished.

a) DCNnMmTCPF is cleared if DCNnMmRDYF is changed or DCNnMmTRQF is set.
b) This indicates a successful transmission abort, if this was requested by the application by clearing the DCNnMmTRQF flag.
c) DCNnMmTCPF is not cleared, if DCNnMmTRQF is set by the ABT operation.

DCNnMmMOWF	Message buffer overwrite status bit
0	The message buffer is not overwritten by a newly received data or remote frame.
1	The message buffer is overwritten by a newly received data or remote frame.

DCNnMmIENF	Message buffer interrupt request enable bit
0	Receive message buffer: Valid message reception completion interrupt disabled. Transmit message buffer: Normal message transmission completion interrupt disabled.
1	Receive message buffer: Valid message reception completion interrupt enabled. Transmit message buffer: Normal message transmission completion interrupt enabled.

DCNnMmDTNF	Message buffer data update bit
0	A new data frame or remote frame has been stored in the message buffer.
1	No new data frame or remote frame has been stored in the message buffer.

Note The user must not set the DCNnMmDTNF (1) by using the set-DN instruction. Special care applies to DCNnMmDTNF, DCNnMmTRQF, and DCNnMmRDYF bit when in mirror mode, mirror mode with TIF or receive-only mode is activated. Please refer to *26.9 “Operational Modes of RXONLY_CH” on page 2056* .

DCNnMmTRQF	Message buffer transmission request bit
0	No message frame transmitting request that is pending or being transmitted is in the message buffer.
1	The message buffer is holding transmission of a message frame pending or is transmitting a message frame.

DCNnMmRDYF	Message buffer ready bit
0	The message buffer can be written by software. The DCN module cannot write to the message buffer.
1	Writing the message buffer by software is ignored (except a write access to the DCNnMmRDYF, DCNnMmTRQF, DCNnMmDTNF, and DCNnMmMOWF). The DCN module can write to the message buffer.

(b) DCNnMmCTL write

15	14	13	12	11	10	9	8
0	DCNnMmSENH	0	0	DCNnMmSEIE	DCNnMmSEDN	DCNnMmSETR	DCNnMmSERY
7	6	5	4	3	2	1	0
0	DCNnMmCLNH	0	DCNnMmCLMW	DCNnMmCLIE	DCNnMmCLDN	DCNnMmCLTR	DCNnMmCLRY

DCNnMmSENH	DCNnMmCLNH	Setting of DCNnMmNHMF bit
0	1	DCNnMmNHMF is cleared to 0.
1	0	DCNnMmNHMF is set to 1.
Other than above		DCNnMmNHMF is not changed.

DCNnMmCLMW	Setting of DCNnMmMOWF bit
0	DCNnMmMOWF is not changed.
1	DCNnMmMOWF is cleared to 0.

DCNnMmSEIE	DCNnMmCLIE	Setting of DCNnMmIENF bit
0	1	DCNnMmIENF is cleared to 0.
1	0	DCNnMmIENF is set to 1.
Other than above		DCNnMmIENF is not changed.

DCNnMmSEDN	DCNnMmCLDN	Setting of DCNnMmDTNF bit
0	1	DCNnMmDTNF is cleared to 0.
1	0	Prohibited
Other than above		DCNnMmDTNF is not changed.

Note If DCNnMmDTNF is cleared by the finish of ID field receiving, the message buffer participates in the search to store the receiving frame.

DCNnMmSETR	DCNnMmCLTR	Setting of DCNnMmTRQF bit
0	1	DCNnMmTRQF is cleared to 0.
1	0	DCNnMmTRQF is set to 1.
Other than above		DCNnMmTRQF is not changed.

DCNnMmSERY	DCNnMmCLRY	Setting of DCNnMmRDYF bit
0	1	DCNnMmRDYF is cleared to 0.
1	0	DCNnMmRDYF is set to 1.
Other than above		DCNnMmRDYF is not changed.

Cautions

1. Set DCNnMmIENF and DCNnMmRDYF always separately.
2. Do not set DCNnMmDTNF to 1 by software. Be sure to write 0 to bit 10.
3. Do not set DCNnMmTRQF and DCNnMmRDYF to 1 at the same time. Set DCNnMmRDYF = 1 before setting DCNnMmTRQF = 1.
4. Do not clear DCNnMmRDYF to “0” during message transmission. Follow the transmission abort process about clearing DCNnMmRDYF for redefinition of the message buffer.
5. Clearing of DCNnMmRDYF may take some time, depending on activity of the CAN Controller. Repeat the clearing access, until reading of DCNnMmRDYF confirms that the bit is cleared.
6. Be sure that DCNnMmRDYF is cleared before writing to the other message buffer registers, by checking the status of DCNnMmRDYF.

Chapter 27 Clocked Serial Interface G (CSIG)

This chapter contains a generic description of the Clocked Serial Interface G (CSIG).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

27.1 V850E2/Fx4-H CSIG Features

Instances

This microcontroller has following number of instances of the Clocked Serial Interface G.

Table 27-1 Instances of CSIG

Clocked Serial Interface G	V850E2/FK4-H	V850E2/FL4-H
Instance	2	3
Name	CSIG0, CSIG4	CSIG0, CSIG2, CSIG4

Instances index n

Throughout this chapter, the individual instances of a Clocked Serial Interface G is identified by the index "n" (n = 0, 2, 3, 4), for example, CSIGnCTL0 for the CSIGn control register 0.

Register addresses

All CSIGn register addresses are given as address offsets to the individual base address <CSIGn_base>.
The base address <CSIGn_base> of each CSIGn is listed in the following table:

Table 27-2 Register base addresses <CSIGn_base>

CSIGn instance	<CSIGn_base> address
CSIG0	FF70 0000_H
CSIG2	FF72 0000_H
CSIG4	FF74 0000_H

Clock supply All Clocked Serial Interface G provide one clock input:

Table 27-3 CSIGn clock supply

CSIGn instance	CSIGn clock	Connected to
CSIG0	PCLK	Clock Controller CKSCLK_108
CSIG2	PCLK	Clock Controller CKSCLK_107
CSIG4	PCLK	Clock Controller CKSCLK_011

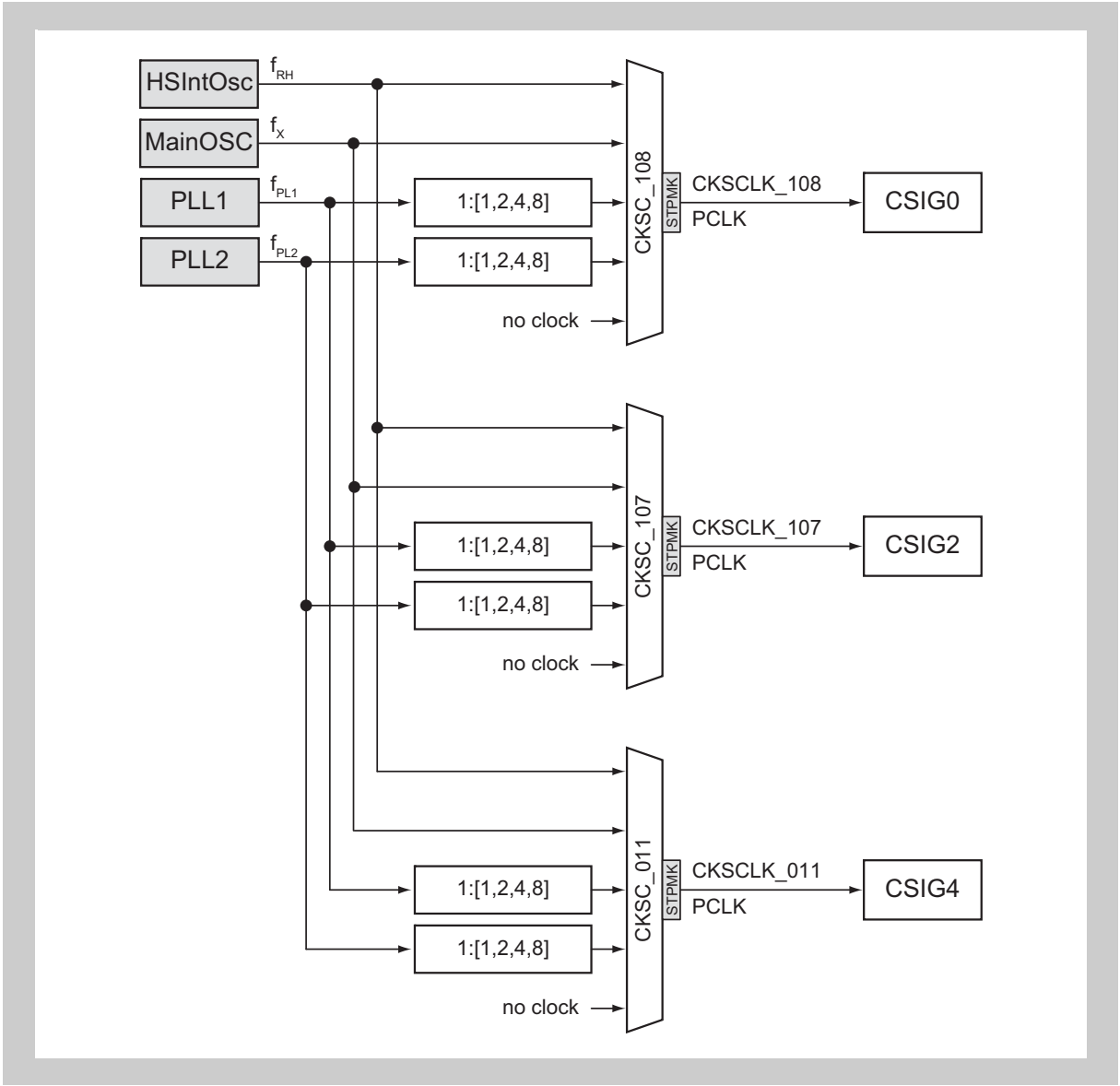


Figure 27-1 CSIG clock supply

Interrupts and DMA

The Clocked Serial Interface G can generate the following interrupt and DMA requests:

Table 27-4 CSIGn interrupt and DMA requests

CSIGn signals	Function	Connected to
CSIG0:		
CSIGTIC	Communication status interrupt	Interrupt Controller INTCSIG0IC DMA Controller trigger 43 DTS Controller trigger 80
CSIGTIR	Reception status interrupt	Interrupt Controller INTCSIG0IR DMA Controller trigger 42 DTS Controller trigger 79
CSIGTIRE	Reception error interrupt	Interrupt Controller INTCSIG0IRE
CSIG2:		
CSIGTIC	Communication status interrupt	Interrupt Controller INTCSIG2IC DMA Controller trigger 73 DTS Controller trigger 70
CSIGTIR	Reception status interrupt	Interrupt Controller INTCSIG2R DMA Controller trigger 72 DTS Controller trigger 69
CSIGTIRE	Reception error interrupt	Interrupt Controller INTCSIG2IRE
CSIG4:		
CSIGTIC	Communication status interrupt	Interrupt Controller INTCSIG4IC[a] DMA Controller trigger 83 DTS Controller trigger 101
CSIGTIR	Reception status interrupt	Interrupt Controller INTCSIG4IR[a] DMA Controller trigger 82 DTS Controller trigger 100
CSIGTIRE	Reception error interrupt	Interrupt Controller INTCSIG4IRE[a]

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

CSIG H/W reset

The Clocked Serial Interfaces G and their registers are initialized by the following reset signal:

Table 27-5 CSIGn reset signal

CSIGn	Reset signal
CSIG0, CSIG2	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)
CSIG4	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)

Internal signals

The internal signal connections of the Clocked Serial Interface G are listed in the following table.

Table 27-6 CSIGn internal signal connections

CSIGn signal	Function	Connected to
CSIG0:		
CSIGTSSO	CSIGTSO output buffer control	Port CSIG0SO output buffer control
CSIG2:		
CSIGTSSO	CSIGTSO output buffer control	Port CSIG2SO output buffer control
CSIG4:		
CSIGTSSO	CSIGTSO output buffer control	Port CSIG4SO output buffer control

I/O signals

The I/O signals of the Clocked Serial Interface G are listed in the following table.

Table 27-7 CSIGn I/O signals

CSIGn signal	Function	Connected to
CSIG0:		
CSIGTSCK	Serial clock signal	Port CSIG0SC[a]
CSIGTSI	Serial data input signal	Port CSIG0SI[a]
CSIGTSO	Serial data output signal	Port CSIG0SO
$\overline{\text{CSIGTSSI}}$	Slave select input signal	Port $\overline{\text{CSIG0SSI}}$[a]
CSIGTSHSG	Handshake signal	Port CSIG0RY[a]
CSIG2:		
CSIGTSCK	Serial clock signal	Port CSIG2SC[a]
CSIGTSI	Serial data input signal	Port CSIG2SI[a]
CSIGTSO	Serial data output signal	Port CSIG2SO
$\overline{\text{CSIGTSSI}}$	Slave select input signal	not connected
CSIGTSHSG	Handshake signal	Port CSIG2RY[a]
CSIG4:		
CSIGTSCK	Serial clock signal	Port CSIG4SC[a]
CSIGTSI	Serial data input signal	Port CSIG4SI[a]
CSIGTSO	Serial data output signal	Port CSIG4SO
$\overline{\text{CSIGTSSI}}$	Slave select input signal	Port $\overline{\text{CSIG4SSI}}$[a]
CSIGTSHSG	Handshake signal	Port CSIG4RY[a]

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

Caution The initial state of the input signals noise filters block the input signals. Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

Activation of the input signals noise filters has an influence on the timely relation of the signals.

Handshake function

The CSIGn handshake function uses the CSIGn module's handshake signal CSITSHSG, which is connected to the port signals CSIGnRY.
If the CSIGn module operates in master mode, CSIGnRY is an input and in slave mode an output signal.
The same CSIGnRY can not be used as alternative input and output port function. The following table summarizes the usage options of the different CSIGnRY signals.

Table 27-8 CSIGn handshake signal ports

CSIGn I/O port signal	Port	Alternative function	CSIGnRY usage
CSIG0:			
CSIG0RY	P3_4	ALT_IN4 ALT_OUT4	in CSIG0 master or slave mode
	P4_9	ALT_OUT2	only in CSIG0 slave mode
CSIG2:			
CSIG2RY	P3_3	ALT_IN4 ALT_OUT4	in CSIG4 master or slave mode
CSIG4:			
CSIG4RY	P4_10	ALT_IN2	only in CSIG4 master mode

27.1.1 Data consistency check

The following table shows the CSIGnSO ports and their capability to use them for data consistency checks. Refer to *"Error detection"* in the section *"Functional Description"* below for details about data consistency checks.

Table 27-9 CSIGn handshake signal ports

CSIGn I/O port signal	Port	Alternative function	CSIGnRY usage
CSIG0:			
CSIG0RY	P3_4	ALT_IN4 ALT_OUT4	in CSIG0 master or slave mode
	P4_9	ALT_OUT2	only in CSIG0 slave mode
CSIG2:			
CSIG2RY	P3_3	ALT_IN4 ALT_OUT4	in CSIG4 master or slave mode
CSIG4:			
CSIG4RY	P4_10	ALT_IN2	only in CSIG4 master mode

Port configuration

Input/output control of the CSIGnSO port is done by the CSIGn module, thus set PIPCn.PIPCn_m = 1.
All other necessary configuration is done automatically, when the data consistency check is enabled.

27.2 Functional Overview

Features summary

- Three-wire serial synchronous data transfer
- Master mode and slave mode
- Slave select input signal ($\overline{\text{CSIGTSSI}}$)
- Built-in baud rate generator
- Adjustable baud rate; in slave mode it is determined by the input clock
- Maximum transmission speed:
  - in master mode: PCLK/4
  - in slave mode: PCLK/6
- Adjustable clock phase and data phase
- Data transfer with MSB or LSB first
- Transfer data length selectable from 7 to 16 bits in 1-bit units
- EDL (Extended Data Length) function for transferring data with more than 16 bits
- Three selectable transfer modes:
  - transmit-only mode
  - receive-only mode
  - transmit/receive mode
- Built-in handshake function
- Separate transmit and receive buffers (two 16-bit registers)
- Error detection (data consistency check, parity, overrun)
- Three different interrupt request signals (CSIGTIC, CSIGTIR, CSIGTIRE)
- Various conditions for interrupt generation
- LBM (Loop Back Mode) function for self test

The block diagram shows the main components of the CSIG.

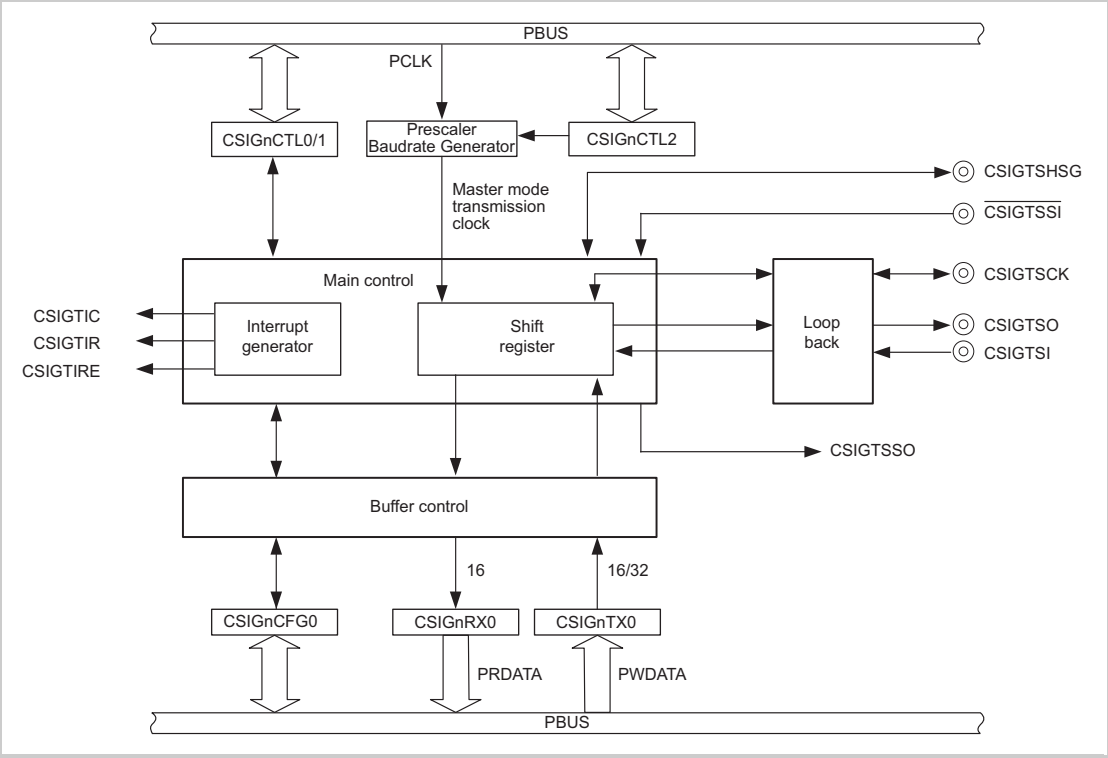


Figure 27-2 CSIG block diagram

In master mode, the transmission clock CSIGTSCK is generated by the built-in baud rate generator (BRG). In slave mode, the transmission clock is received from an external source.

27.3 Functional Description

The Clocked Serial Interface G uses three signals for communication:

- Transmission clock CSIGTSCK (output in master mode, in slave mode input)
- Data output signal CSIGTSO
- Data input signal CSIGTSI

If the CSIGn is operated in slave mode, the additional input signal $\overline{\text{CSIGTSSI}}$ acts as a chip select signal, that selects the CSIG to take part in data transfers.

Data transmission is bit-wise and serial and synchronous to the transmission clock.

The most important registers for setting up the CSIG are:

Register	Function
CSIGnCTL0	Enables/disables transmission clock, data transmission, and data reception
CSIGnCTL1	Controls options like interrupt timing, extended data length, data consistency check, loop-back mode, handshake, etc.
CSIGnCTL2	Selects master/slave mode and – effective in master mode – the baud rate of the internal generator
CSIGnCFG0	Configures the communication protocol

27.3.1 Master/slave mode

The master or slave mode primarily determines the source of the transmission clock.

(1) Master mode

In master mode, the serial communication clock is generated by the internal baud rate generator (BRG) and provided by signal CSIGTSCK.

Master mode is enabled by setting CSIGnCTL2.CSIGnPRS[2:0] to anything but 111_B. In master mode, the frequency setting of the BRG becomes effective.

The initial level of CSIGTSCK depends on the clock phase selection bit: it is high when CSIGnCTL1.CSIGnCKR = 0, and is low when CSIGnCTL1.CSIGnCKR = 1.

The example below shows the communication in master mode for 8 data bits, CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0, and MSB first:

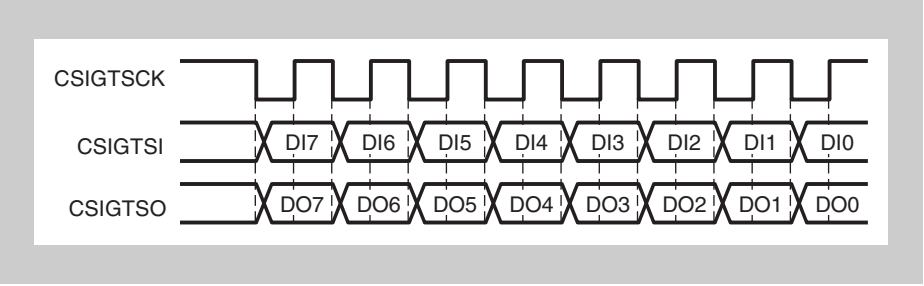


Figure 27-3 Transmit/receive in master mode

(2) Slave mode

In slave mode, another device is the communication master. The external clock is received by signal CSIGTSCK. Send/receive operation starts as soon as a clock signal is detected.

Slave mode is selected by setting CSIGnCTL2.CSIGnPRS[2:0] to 111_B.

Note When using slave mode, disable the baud rate generator (BRG) by clearing bits CSIGnCTL2.CSIGnBRS[11:0]. When the BRG is disabled, CSIGTSCK stays on the level specified by CSIGnCTL1.CSIGnCKR.

The example below shows the communication in slave mode for 8 data bits, CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0, and MSB first:

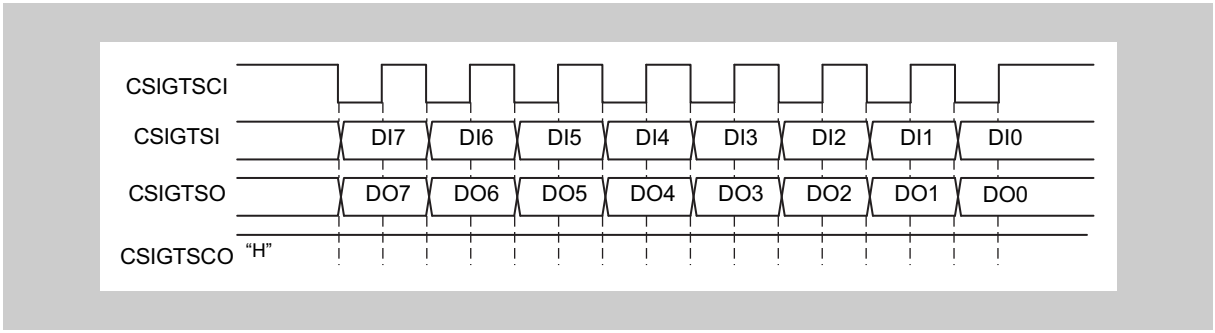


Figure 27-4 Transmit/receive in slave mode

27.3.2 Master/slave connections

(1) One master and one slave

The following figure illustrates the connections between one master and one slave.

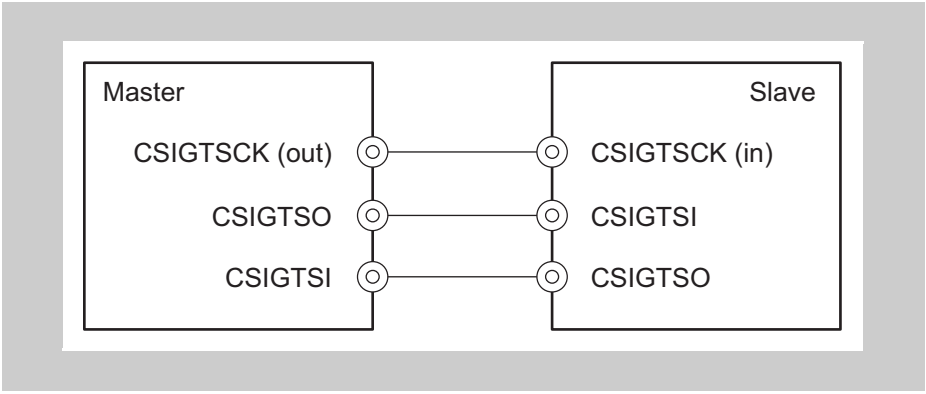


Figure 27-5 Simple master/slave connection

(2) One master and multiple slaves

The following figure illustrates the connections between one master and multiple slaves. In this case, the master must provide one slave select (SS) signal to each of the slaves. This signal is connected to the slave select input $\overline{\text{CSIGTSSI}}$ of the slave.

The recognition of the $\overline{\text{CSIGTSSI}}$ signal can be enabled/disabled by bit CSIGnCTL1.CSIGnSSE.

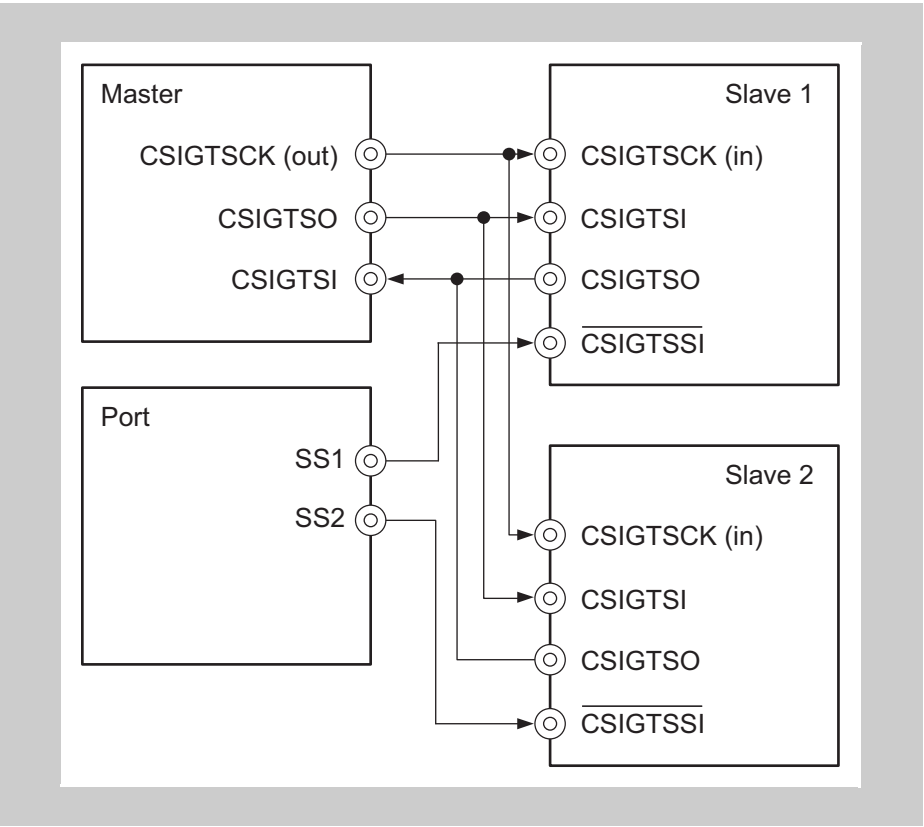


Figure 27-6 Master to multiple slaves connection

A slave is selected (enabled) when its $\overline{\text{CSIGTSSI}}$ signal is low.

If it is not selected, the slave will neither receive nor transmit data. In addition, the CSIGTSO output buffer is disabled order to avoid interference with the output of another slave which was selected.

CSIGTSO buffer control

The CSIG provides the signal CSIGTSSO to control the port output buffer of the serial output signal CSIGTSO. By use of this signal the CSIGTSO output buffer can be disabled under following conditions:

- The CSIG is enabled (CSIGnCTL0.CSIGnPWR = 1)
- The CSIG is operated in transmit-only or transmit/receive mode (CSIGnCTL0.CSIGnTXE = 1)
- The CSIG is operated with slave select enabled (CSIGnCTL1.CSIGnSSE = 1).
- The slave mode selection signal $\overline{\text{CSIGTSSI}}$ is inactive, i.e. on high level.

By this signal congestions on the external CSIGTSO signal line are avoided.

27.3.3 Transmission clock selection

In master mode, the transmission baud rate is selectable using the CSIGnPRS[2:0] and CSIGnBRS[11:0] bits in the CSIGnCTL2 register. The baud rate generator (BRG) counts up at every rising edge of PCLK.

The following figure shows a block diagram of the BRG.

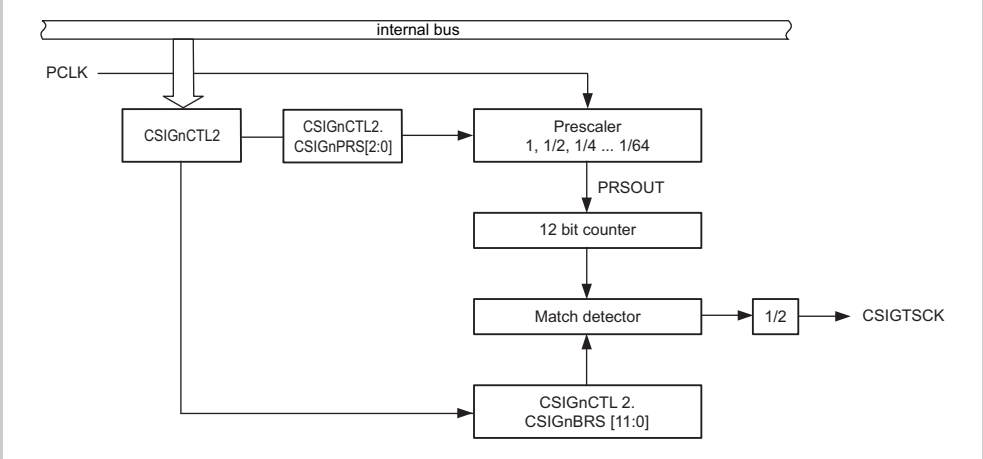


Figure 27-7 BRG block diagram

Clearing CSIGnCTL2.CSIGnBRS[11:0] disables the BRG.

Baud rate calculation

The baud rate is calculated as: PCLK / ($2^m \times k \times 2$), where

m = CSIGnCTL2.CSIGnPRS[2:0] = 0 to 6

k = CSIGnCTL2.CSIGnBRS[11:0] = 1 to 4095

Baud rate limits

When setting the baud rate, please note:

- Maximum acceptable baud rate in master mode is PCLK / 4.
- Maximum acceptable baud rate in slave mode is PCLK / 6 and must be ensured by the external master.
- Minimum baud rate in both modes is PCLK / 524160.

27.3.4 Data transfer modes

(1) Transmit-only mode

Setting CSIGnCTL0.CSIGnTXE = 1 and CSIGnCTL0.CSIGnRXE = 0 puts the CSIG in transmit-only mode. Transmission starts when transmit data is written in the CSIGnTX0W or CSIGnTX0H register.

Caution In case transmit-only mode has been entered after any reception mode, the data in the CSIGnRX0 buffer becomes undefined after completion of the first transmission.
Consequently the reception register CSIDnRX0 has to be read before changing to transmit-only mode.

(2) Receive-only mode

Setting CSIGnCTL0.CSIGnTXE = 0 and CSIGnCTL0.CSIGnRXE = 1 puts the CSIG in receive-only mode.

In master mode, reception starts when dummy data is written to the CSIGnTX0W or CSIGnTX0H register.
All following receptions are triggered by a read from the receive data register CSIGnRX0, as long as CSIGnBCTL0.CSIGnSCE = 1.

In slave mode, reception starts when the communication clock CSIGTSCK from the master is received. In this case, it is not necessary to write data to the CSIGnTX0W or CSIGnTX0H register of the slave.

Note In receive-only mode, any previously received data must be read from the reception register CSIGnRX0 in order to avoid any overwrite situation.

Moreover the communication start bit CSIGnBCTL0.CSIGnSCE has to be set to 1 and has to set back to 0 before reading the last received data from CSIGnRX0.

The recommended procedure is:

1. Set CSIGnBCTL0.CSIGnSCE = 1.
2. Before starting the first receive operation, read CSIGnRX0 (dummy data).
3. Wait for the reception interrupt CSIGTIR.
4. Read CSIGnRX0 (received data).
   In case of further data receptions continue at 4. until all data has been received.
   Before reading the last received data from CSIGnRX0, set CSIGnBCTL0.CSIGnSCE = 0.

(3) Transmit / receive mode

Setting CSIGnCTL0.CSIGnTXE = 1 and CSIGnCTL0.CSIGnRXE = 1 puts the CSIG in transmit/receive mode.

Data transfer (transmission and reception) starts when transmit data is written to the CSIGnTX0W or CSIGnTX0H register.

27.3.5 Data length selection

(1) Data length selection without extended length

Transmission data length is selectable from 7 to 16 bits using the CSIGnDLS[3:0] bits in the CSIGnCFG0 register. The examples below show the communication with MSB first (CSIGnCFG0.CSIGnDIR = 0):

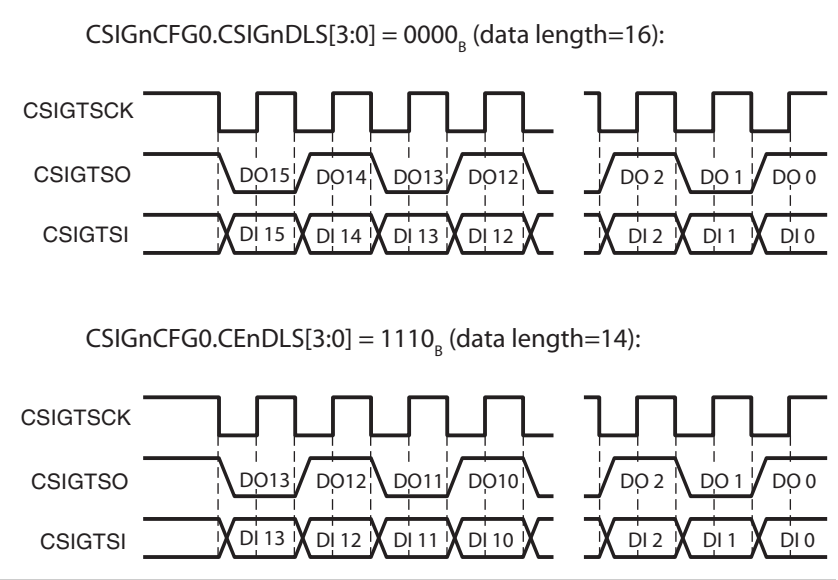


Figure 27-8 Data length select function

(2) Data length selection with extended data length

If the data to be sent/received exceeds 16 bits, the extended data length (EDL) feature can be used.

The EDL function is enabled by setting bit CSIGnCTL1.CSIGnEDLE to 1.

The EDL function works as follows:

- The data has to be broken into 16-bit blocks plus remainder. For example, a string of 42 bits would be broken into two 16-bit blocks plus 10 bits.
- The remainder defines the “data length” that has to be specified in CSIGnCFG0.CSIGnDLS[3:0].
- CSIGnTX0W.CSIGNEDL must be set to 1. In this case, the data writen to CSIGnTX0W is sent as a 16-bit data length regardless of the CSIGnCFG0.CSIGnDLS[3:0] bits.
- The transfer is complete after the data with the specified data length (the remainder with CSIGnTX0W.CSIGNEDL = 0) has been sent.

Example Example for sending 40-bit data, for example the string $123456789A_H$:

40 bits are split into 2 x 16 bits plus 8 bits.

- Initialize CSIGnCFG0.CSIGnDLS[3:0] = 8.
- To send the string $123456789A_H$ with MSB first, write the following sequence to CSIGnTX0W:
  - $2000\ 1234_H$ (CSIGnTX0W.CSIGnEDL = 1)
  - $2000\ 5678_H$ (CSIGnTX0W.CSIGnEDL = 1)
  - $0000\ 009A_H$ (CSIGnTX0W.CSIGnEDL = 0)

The following figure illustrates the timing.

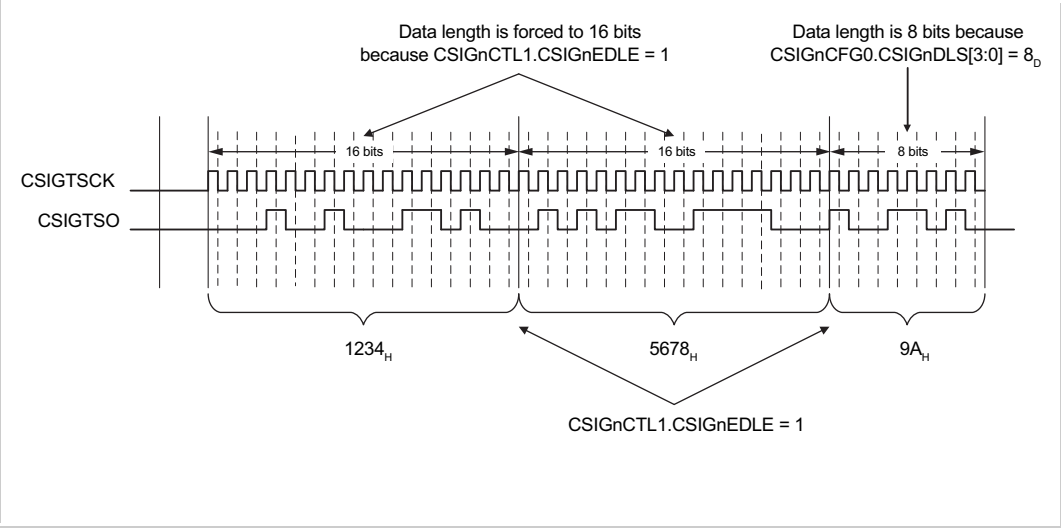


Figure 27-9 EDL timing diagram

Notes

1. It is not possible to send two consecutive data with a data length of less than 7 bits.
2. If parity is enabled, the parity bit is added after the last bit.
3. To consider the data direction, pay attention to the following example: (example data to be sent 123456_H)
   - CSIGnCFG0.CSIGnDIR = 1: LSB first
     – CSIGnTX0 = $2000\ 3456_H$ (EDL bit = 1)
     – CSIGnTX0 = $0000\ 0012_H$ (EDL bit = 0)
   - CSIGnCFG0.CSIGnDIR = 0: MSB first
     – CSIGnTX0 = $2000\ 1234_H$ (EDL bit = 1)
     – CSIGnTX0 = $0000\ 0056_H$ (EDL bit = 0)

27.3.6 Serial data direction select function

The serial data direction is selectable using the CSIGnDIR bit in the CSIGnCFG0 register. The examples below show the communication for 8-bit data (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B):

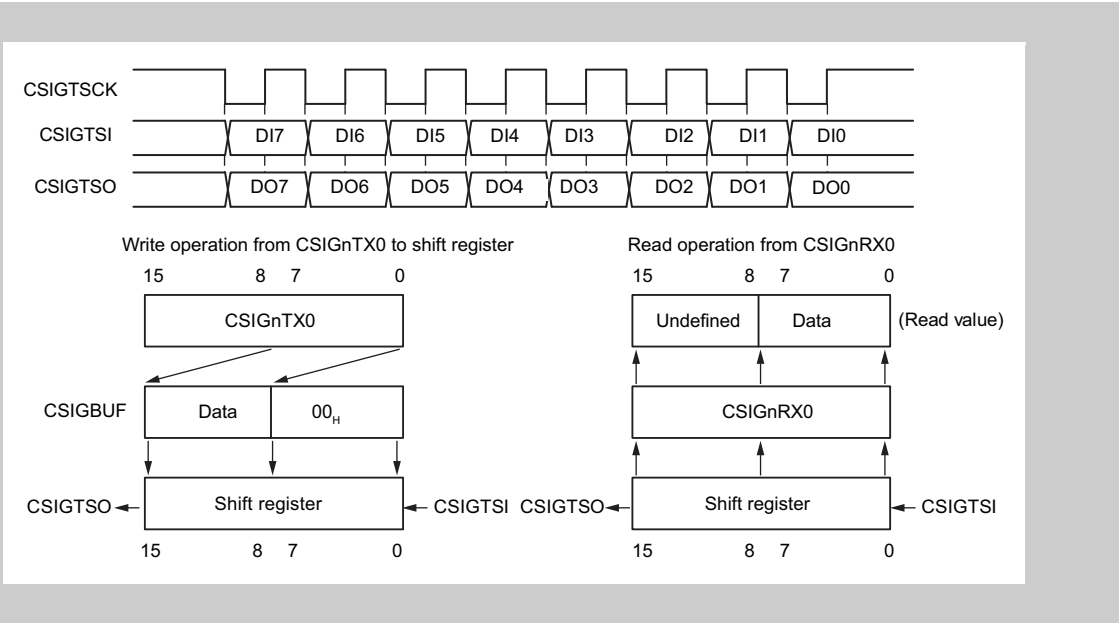


Figure 27-10 Serial data direction select function - MSB first (CSIGnDIR = 0)

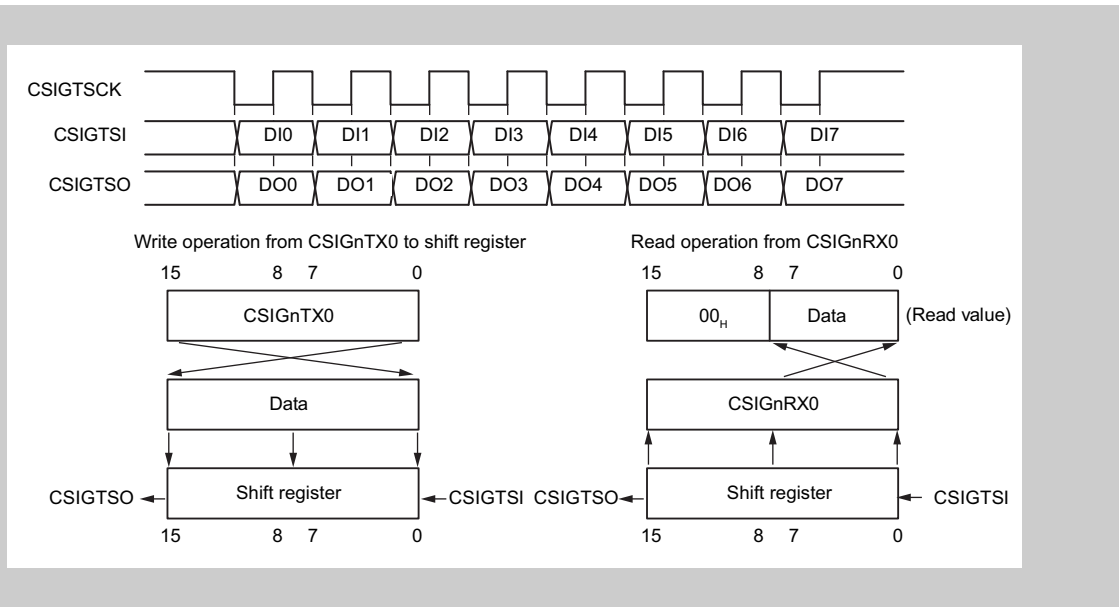


Figure 27-11 Serial data direction select function - LSB first (CSIGnDIR = 1)

27.3.7 Communication in slave mode

The following figure illustrates the communication signals and timings in slave mode.

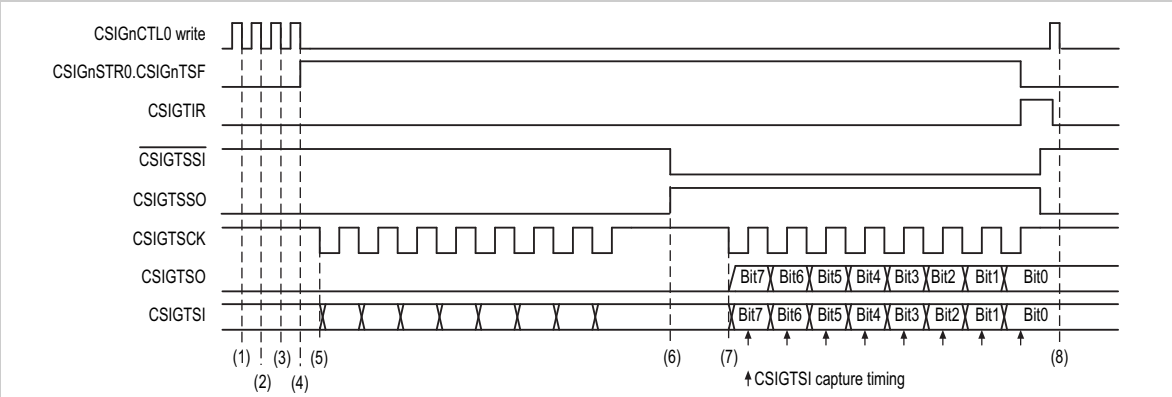


Figure 27-12 Rx/Tx communication timing in slave mode

1. CSIG is put into slave mode by setting CSIGnCTL2.CSIGnPRS[2:0] = 111_B. CSIGnCTL1.CSIGnCKR and CSIGnCFG0.CSIGnDAP are 0.
2. Data length is 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B). Data direction is MSB first (CSIGnCFG0.CSIGnDIR = 0).
3. CSIG is set to transmit/receive operation (CSIGnCTL0.CSIGnTXE = 1, CSIGnCTL0.CSIGnRXE = 1). Communication start is enabled.
4. The “transmission in progress” flag CSIGnSTR0.CSIGnTSF is automatically set when transfer data is written to the transmission register CSIGnTX0H.
5. As long as signal $\overline{\text{CSIGTSSI}}$ is high, transmission/reception is not started, even if an external transmission clock is applied to clock input signal CSIGTSCI. Input at CSIGTSI is ignored.
6. As soon as $\overline{\text{CSIGTSSI}}$ falls to low level, signal CSIGTSSO goes high, indicating that CSIGTSO is enabled and ready for transmission.
7. Now, as soon as the external clock signal appears at CSIGTSCK, the slave transmits data to CSIGTSO and simultaneously captures data from CSIGTSI.
8. Interrupt CSIGTIR indicates when the reception is complete. The register CSIGnRX0 can be read.

27.3.8 CSIG interrupts

CSIG can generate the following interrupts:

- CSIGTIC
- CSIGTIR
- CSIGTIRE

Interrupt delay

In master mode, all interrupts generated by the master can be delayed by one half period of the transmission clock CSIGTSCK. This is not possible in slave mode.

The delay is specified by setting bit CSIGnCTL1.CSIGnSIT = 1.

The following example illustrates the interrupt delay function, assuming a setting of CSIGnCTL1.CSIGnSIT = 1 (interrupt delay enabled), CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock and data phase), and CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (data length 8 bits).

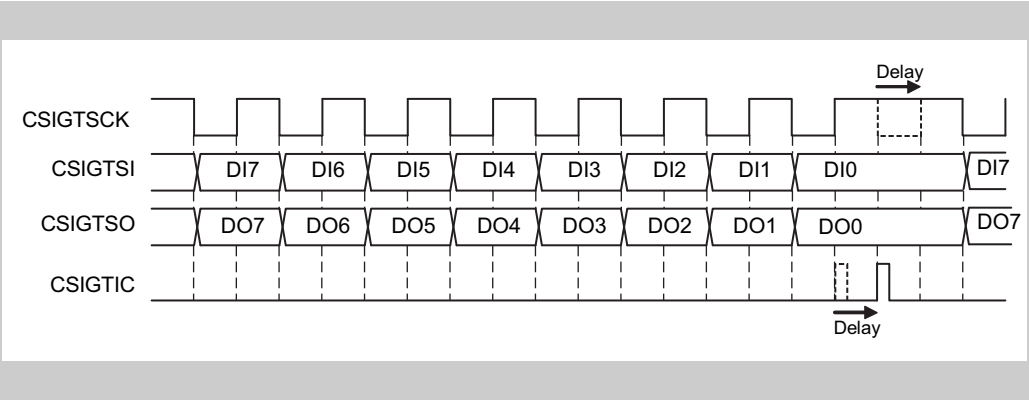


Figure 27-13 Interrupt delay function (CSIGnCTL1.CSIGnSIT = 1)

(1) CSIGTIC communication interrupt

This interrupt is normally generated after every data transfer. It can be used to trigger a DMA for writing new transmission data to register CSIGnTX0W or CSIGnTX0H.

The following example assumes master mode and a setting of CSIGnCTL1.CSIGnSIT = 0 (no interrupt delay), CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock and data phase), CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (data length 8 bits), and CSIGnCTL1.CSIGnSLIT = 0 (normal interrupt timing).

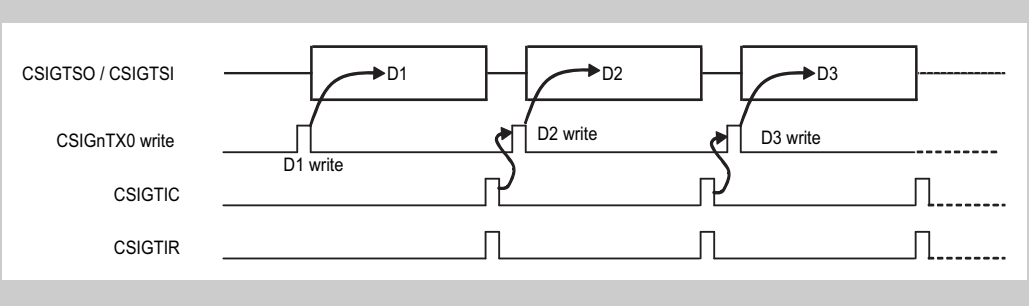


Figure 27-14 Generation of CSIGTIC after communication (CSIGnCTL1.CSIGnSLIT = 0)

However, CSIGTIC can also be set up to occur when the CSIGnTX0 register is free for the next data. This is specified by setting CSIGnCTL1.CSIGnSLIT = 1.

This mode allows more efficient data transfers.

The effect is illustrated in the figure below.

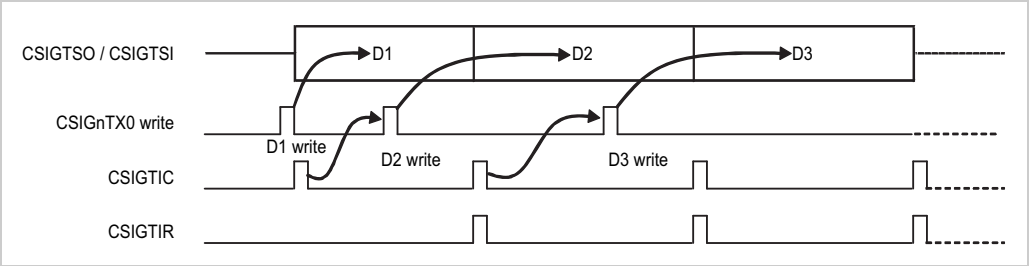


Figure 27-15 Generation of CSIGTIC at the beginning of communication

(2) CSIGTIR reception interrupt

This interrupt is generated in receive-only and transmit/receive mode after data has been received and is available in the reception register. It can be used to trigger a DMA for reading the received data from register CSIGnRX0.

The following example assumes master mode and a setting of CSIGnCTL1.CSIGnSIT = 0 (no interrupt delay), CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0 (normal clock and data phase), and CSIGnCFG0.CSIGnDLS[3:0] = 1000_B (data length 8 bits).

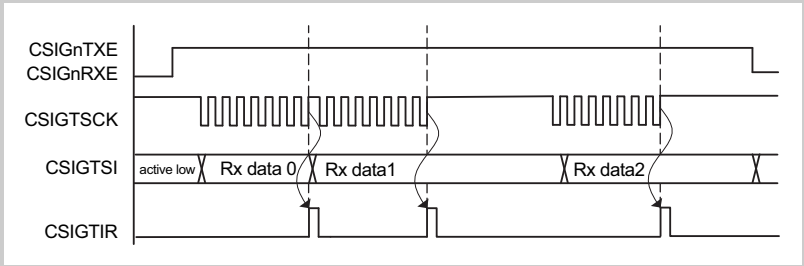


Figure 27-16 Generation of CSIGTIR

(3) CSIGTIRE reception error interrupt

This interrupt is generated whenever an error is detected.

Table 27-10 Data error types

Error type	Communication status after error interrupt
Parity error	Interrupt is generated and communication continues
Data consistency error	Interrupt is generated and communication continues
Overrun error	Interrupt is generated and communication is stopped

The type of error that caused the generation of CSIGTIRE is indicated in register CSIGnSTR0.

For details about the various error types refer to *27.3.11 “Error detection” on page 2127* .

27.3.9 Handshake function

CSIG features a handshake function to synchronize the master and the slave devices. This function can be enabled/disabled by bit CSIGnCTL1.CSIGnHSE. For handshake, the signal CSIGTSHSG is used.

The timing depends on the data phase selection bit CSIGnCFG0.CSIGnDAP.

(1) Slave mode

When CSIGnCTL1.CSIGnHSE = 1, the slave outputs CSITSHSG = 0 when it is busy. This happens when previous receive data is still in the CSIGnRX0 register, and new data cannot be copied from the shift register to CSIGnRX0 (CSIGnRX0 full condition).

The following examples assume a data length of 8 bits.

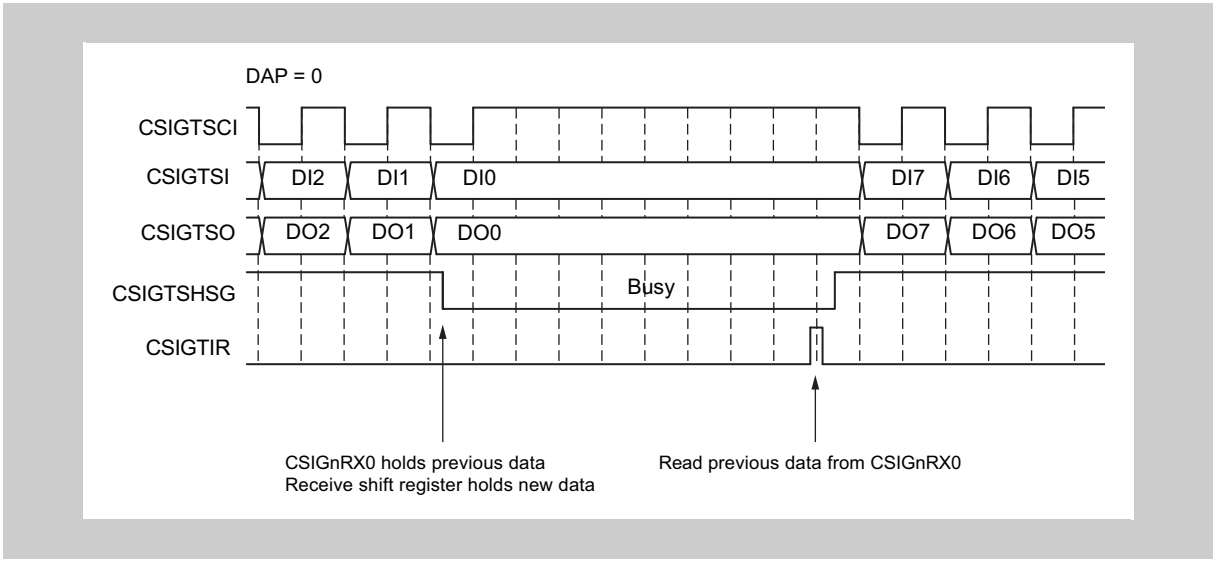


Figure 27-17 Ready/busy signal from slave (CSIGnCFG0.CSIGnDAP = 0)

As long as the slave is busy, the master has to wait (i.e. suspend the transmission clock). The slave sets CSIGTSHSG to high (“ready”) as soon as the reception register CSIGnRX0 has been read.

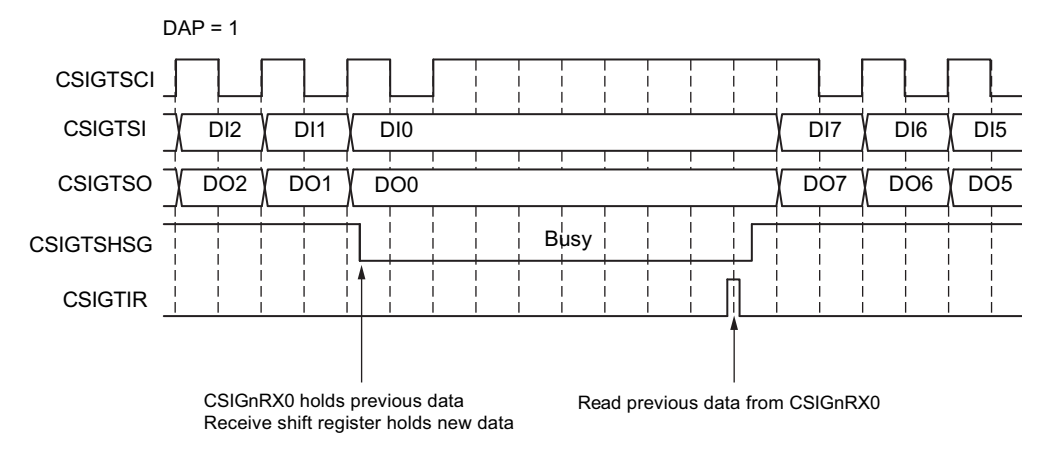


Figure 27-18 Ready/busy signal from slave (CSIGnCFG0.CSIGnDAP = 1)

(2) Master mode

When the master detects CSIGTSHSG = 0, the following transfer is put on hold, and the master goes into wait status. It suspends the clock at CSIGTSCK.

The CSIGTSHSG level is checked at each half clock cycle of CSIGTSCK.

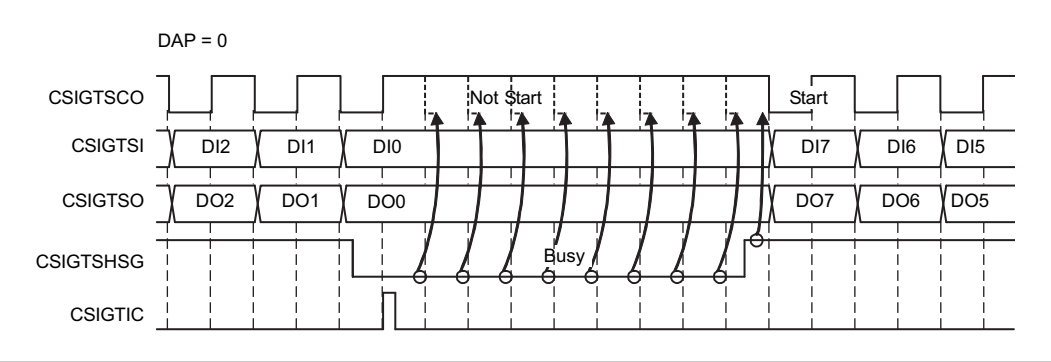


Figure 27-19 Master's reaction to CSIGTSHSG (CSIGnCFG0.CSIGnDAP = 0)

If the CSIGTSHSG low signal comes from the slave while data transfer is in progress, the serial clock is suspended after the transfer is complete.

The master resumes the communication as soon as CSIGTSHSG becomes high (slave is "ready").

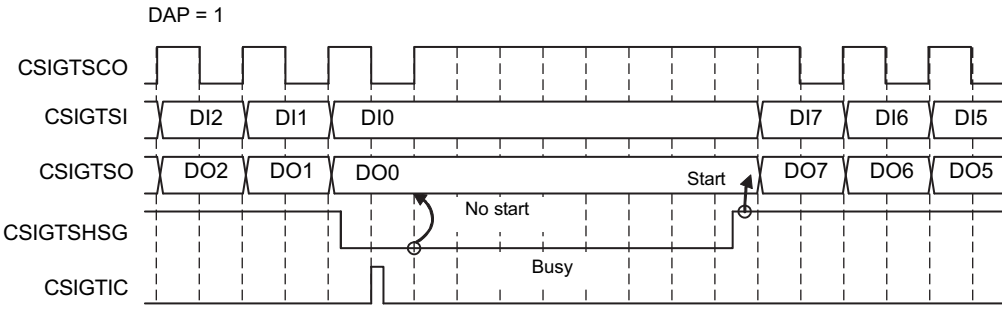


Figure 27-20 Master's reaction to CSIGTSHSG (CSIGnCFG0.CSIGnDAP = 1)

Caution If multiple slaves are connected, the master must only detect the CSIGTSHSG signal of the slave it has selected for communication.

CSIGTSHSG must be pulled down by the external slave before the next transfer starts. Even if the signal is pulled down by the slave during the transfer, the transfer will be finished.

27.3.10 Loop-back mode

Loop-back mode is a special mode for self-test. This feature is only available in master mode.

When this mode is active, the transmit and receive signals are internally connected, as shown in the figures below. The signals CSIGTSCK, CSIGTSO, and CSIGTSI are disconnected from the ports. In addition, the CSIGTSO output level is fixed to low, and CSIGTSCK is set to the inactive state, as defined by CSIGnCTL1.CSIGnCKR. The rest of CSIG works as in normal operation.

In order to test the CSIG, enable loop-back mode and carry out normal transfer operations. Then check that the received data is the same as the transmitted data.

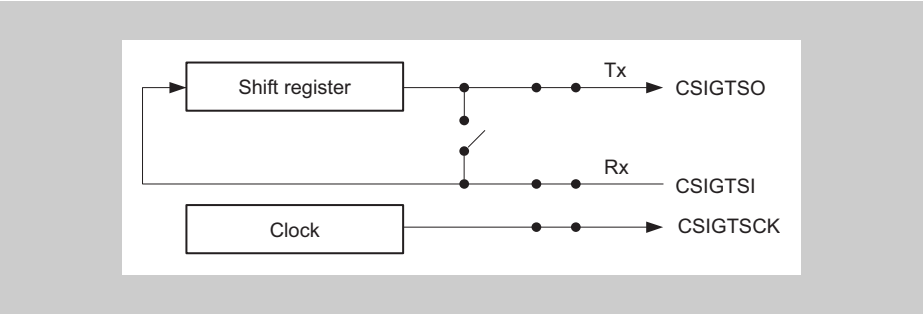


Figure 27-21 Normal operation

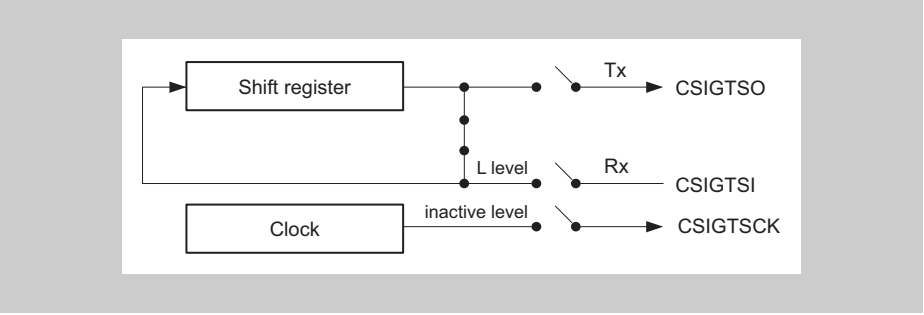


Figure 27-22 Operation in loop-back mode

27.3.11 Error detection

CSIG can detect three error types:

- Data consistency error (transmission data)
- Parity error (received data)
- Overrun error

Error check can be individually enabled/disabled for each type.

If one of these errors is detected, the interrupt CSIGTIRE is generated.

(1) Data consistency check

The purpose of the data consistency check is to ensure that the data physically sent to the output pin is identical to the original data that was copied to the shift register.

The data consistency check can be enabled/disabled by bit CSIGnCTL1.CSIGnDCS. It is not active if data transmission is disabled (CSIGnCTL0.CSIGnTXE = 0).

When the data consistency check is active, the data transferred from CSIGnTX0 to the shift register is copied to a separate register. In addition, the physical levels at CSIGTSO is read back via the CSIGTDCS signal into an own shift register.

After completion of the transmission, the data sent is compared with the original transmission data.

Mismatch is considered as a data consistency error:

- Interrupt CSIGTIRE is generated.
- Bit CSIGnSTR0.CSIGnDCE is set.

The function is illustrated in the following block diagram.

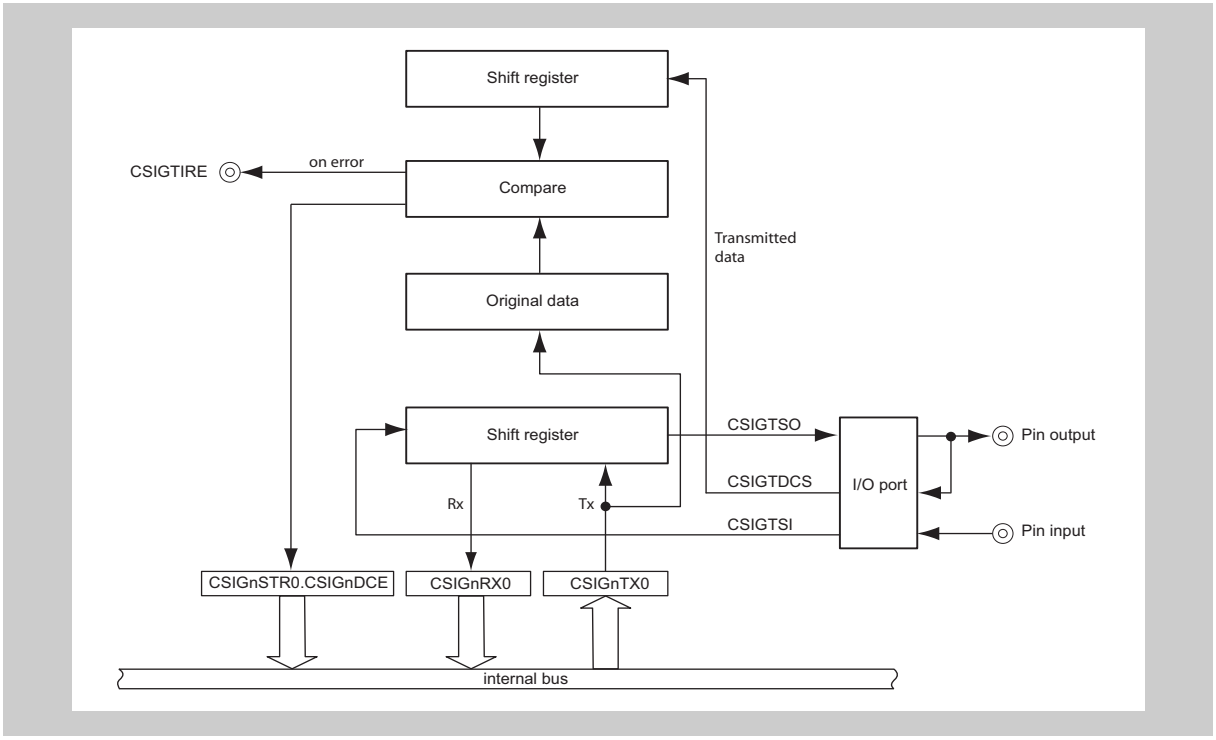


Figure 27-23 Functional block diagram of the data consistency check

(2) Parity check

Parity is a common mean to detect a single bit failure during data transmission. CSIG can append a parity bit to the last data bit (even if extended data length is used).

The use and type of parity is specified in CSIGnCFG0.CSIGnPS[1:0].

Parity check is enabled if CSIGnCFG0.CSIGnPS[1] = 1.

The parity bit is checked after reception is complete. If a parity error occurs:

- Interrupt CSIGTIRE is generated
- Bit CSIGnSTR0.CSIGnPE is set

The following figure shows an example.

Data length is 8 bits. The data transmitted is 05_H and 35_H. Parity type is odd.

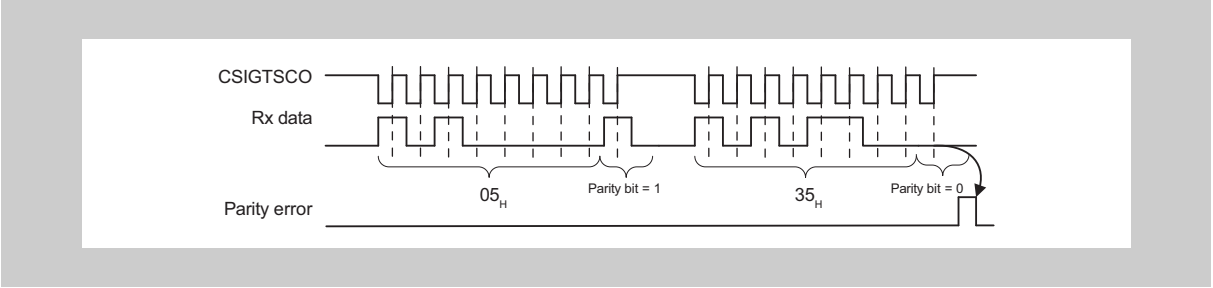


Figure 27-24 Parity check example

For the first 8 bits, the parity bit is 1. There is no parity error, because the total number of ones (including the parity bit) is odd.

For the second 8 bits, the parity bit is 0. This is detected as a parity error, because the total number of ones (including the parity bit) is even.

(3) Overrun error

This error occurs when previously received data still resides in the reception register CSIGnRX0, because it wasn't read, and new data is received.

The overrun error is not generated if data reception is disabled (CSIGnCTL0.CSIGnRXE = 0).

If overrun occurs:

- Interrupt CSIGTIRE is generated
- Bit CSIGnSTR0.CSIGnOVE is set
- Communication is stopped

The following figure illustrates the function.

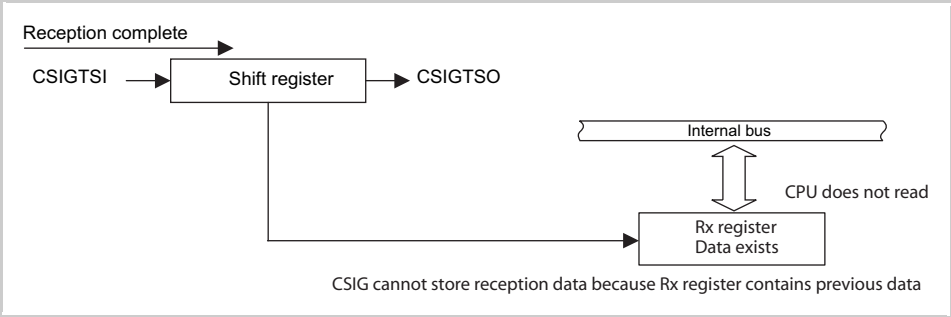


Figure 27-25 Overrun error detection

The following figure illustrates an example where:

- Rx data 3 was not read
- Rx data 4 was received, but cannot be stored

Thus an overrun error occurs.

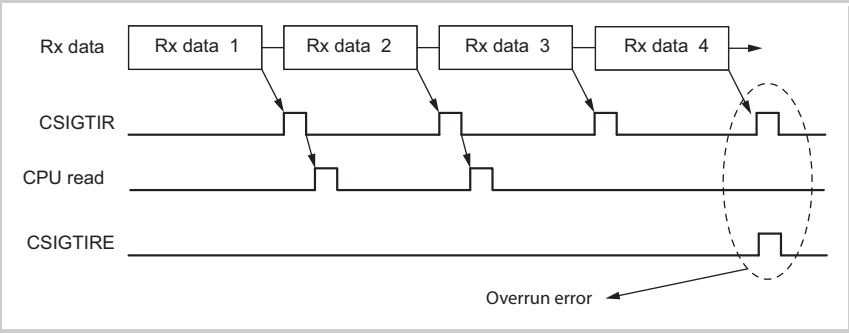


Figure 27-26 Overrun error detection - example

Notes

1. An Rx data overrun error can be avoided in slave mode by using the handshake.
   When handshake is used in slave mode, the receiver (slave) signals to the transmitter (master) that it is busy. The transmitter then waits until the receiver has read its reception register and is ready again.
   For details see *27.3.9 "Handshake function" on page 2123* .
2. In master mode no overrun error can occur, since the data clock CSIGTSCK is stopped, until the data in the CSIGnRX0 register is read.

27.4 CSIG Control Registers

The CSIGn is controlled and operated by means of the following registers:

Table 27-11 CSIGn register overview

Register name	Shortcut	Address
Control register 0	CSIGnCTL0	<CSIGn_base>
Control register 1	CSIGnCTL1	<CSIGn_base> + 10_H
Control register 2	CSIGnCTL2	<CSIGn_base> + 14_H
Status register 0	CSIGnSTR0	<CSIGn_base> + 04_H
Status clear register 0	CSIGnSTCR0	<CSIGn_base> + 08_H
Rx-only mode control register 0	CSIGnBCTL0	<CSIGn_base> + 1000_H
Configuration register 0	CSIGnCFG0	<CSIGn_base> + 1010_H
Transmit data register 0 for word access	CSIGnTX0W	<CSIGn_base> + 1004_H
Transmit data register 0 for half word access	CSIGnTX0H	<CSIGn_base> + 1008_H
Receive data register 0	CSIGnRX0	<CSIGn_base> + $100C_H$
Emulation register	CSIGnEMU	<CSIGn_base> + 0018_H

<CSIGn_base> The base addresses <CSIGn_base> of the CSIGn is defined in the first section of this chapter under the key word “Register addresses”.

(1) CSIGnCTL0 - CSIG control register 0

This register controls the operation clock and enables/disables transmission/ reception.

Access This register can be read/written in 8-bit units.

Address <CSIGn_base> + 00_H

Initial Value 00_H

7	6	5	4	3	2	1	0
CSIGn PWR	CSIGn TXE	CSIGn RXE	0	0	0	0	0[a]
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

a) The default value “0” of bit 0 of CSIGnCTL0 must be changed to “1” before the CSIGn is used.

Caution The default value “0” of bit 0 of CSIGnCTL0 must be changed to “1” before the CSIGn is used.

Table 27-12 CSIGnCTL0 register contents

Bit position	Bit name	Function
7	CSIGnPWR	Controls operation clock: 0: Stop operation clock 1: Provide operation clock Clearing CSIGnPWR to 0 resets the internal circuits, stops operation, and sets the CSIG to standby state. No clock is provided to internal circuits. If CSIGnPWR is cleared during communication, ongoing communication is aborted. A restart of the communication is then required.
6	CSIGnTXE	Enables/disables transmission: 0: Transmission disabled 1: Transmission enabled
5	CSIGnRXE	Enables/disables reception: 0: Receive disabled 1: Receive enabled
0	Bit 0	**Caution** The default value “0” of this must be changed to “1” before the CSIGn is used.

Cautions

1. Do not modify CSIGnRXE or CSIGnTXE while CSIGnPWR = 0. However both bits can be modified in the same write operation when setting CSIGnPWR = 1.
2. Do not modify CSIGnRXE or CSIGnTXE while a data transmission is pending or ongoing, i.e. if CSIGnSTR0.CSIGnTSF = 1.

(2) CSIGnCTL1 - CSIG control register 1

This register specifies the interrupt timing and the interrupt delay mode. It enables/disables extended data length control, data consistency check, loop-back mode, handshake function, and slave select function.

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 10_H

Initial Value 0000 0000H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	CSIGn CKR	CSIGn SLIT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	CSIGn EDLE	0	CSIGn DCS	0	CSIGn LBM	CSIGn SIT	CSIGn HSE	CSIGn SSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIGnCTL0.CSIGnPWR = 0.

Table 27-13 CSIGnCTL1 register contents (1/2)

Bit position	Bit name	Function
17	CSIGnCKR	Selects the initial level of the data clock CSIGTSCK 0: Initial level of CSIGTSCK is high 1: Initial level of CSIGTSCK is low For the relation between CSIGTSCK, CSIGTSO and the sample point, refer to the description of the CSIGnCFG0.CSIGnDAP bit.
16	CSIGnSLIT	Selects the timing of interrupt CSIGTIC: 0: Normal interrupt timing (interrupt is generated after the transfer) 1: Interrupt generation when CSIGnTX0 is free for next data. For details refer to *1 “CSIGTIC communication interrupt” on page 2121* .
7	CSIGnEDLE	Enables/disables extended data length (EDL) mode: 0: Extended data length mode disabled 1: Extended data length mode enabled For details refer to *2 “Data length selection with extended data length” on page 2117* .
5	CSIGnDCS	Enables/disables data consistency check: 0: Data consistency check disabled 1: Data consistency check enabled For details refer to *1 “Data consistency check” on page 2127* .
3	CSIGnLBM	Controls loop-back mode (LBM): 0: Loop-back mode deactivated 1: Loop-back mode activated For details refer to *27.3.10 “Loop-back mode” on page 2126* .

Table 27-13 CSIGnCTL1 register contents (2/2)

Bit position	Bit name	Function
2	CSIGnSIT	Selects interrupt delay mode: 0: No delay 1: Half clock delay for all interrupts This bit is only valid in master mode. In slave mode, no delay is generated. For details refer to *27.3.8 "CSIG interrupts" on page 2121* .
1	CSIGnHSE	Enables/disables handshake mode: 0: Handshake function disabled 1: Handshake function enabled For details refer to *27.3.9 "Handshake function" on page 2123* .
0	CSIGnSSE	Enables/disables slave select function: 0: Input signal $\overline{\text{CSIGTSSI}}$ is ignored 1: Input signal $\overline{\text{CSIGTSSI}}$ is enabled If the slave select function is not used, this bit must be set to 0 (see also *27.3.2 "Master/slave connections" on page 2113*).

Details about CSIGnCTL1.CSIGnSSE:

Table 27-14 Operation of the slave select function during reception

CSIGnCTL0. CSIGnRXE	CSIGnCTL1. CSIGnSSE	$\overline{\text{CSIGTSSI}}$	Receive operation
0	-	-	Reception disabled
1	0	-	Possible
1	1	0	Possible
1	1	1	Impossible

Table 27-15 Operation of the slave select function during transmission

CSIGnCTL0. CSIGnTXE	CSIGnCTL1. CSIGnSSE	$\overline{\text{CSIGTSSI}}$	Transmit operation
0	-	-	Transmission disabled
1	0	-	Possible
1	1	0	Possible
1	1	1	Impossible

(3) CSIGnCTL2 - CSIG control register 2

This register selects the communication clock.

Access This register can be read/written in 16-bit units.

Address <CSIGn_base> + 14_H

Initial Value $E000_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGn PRS[2:0]			0	CSIGn BRS[11:0]											
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIGnCTL0.CSIGnPWR = 0.

Table 27-16 CSIGnCTL2 register contents

Bit position	Bit name	Function
15 to 13	CSIGnPRS [2:0]	Selects the value m of the prescaler:
11 to 0	CSIGnBRS [11:0]	Selects the baud rate (m = CSIGnPRS[2:0]):

CSIGn PRS2	CSIGn PRS1	CSIGn PRS0	Prescaler output (PRSOUT)
0	0	0	PCLK (master mode)
0	0	1	PCLK / 2 (master mode)
0	1	0	PCLK / 4 (master mode)
0	1	1	PCLK / 8 (master mode)
1	0	0	PCLK / 16 (master mode)
1	0	1	PCLK / 32 (master mode)
1	1	0	PCLK / 64 (master mode)
1	1	1	External clock via CSIGTSCI (slave mode)

CSIGnBRS[11:0]	Baud rate at CSIGTSCK
0	BRG is stopped
1	PCLK / ($2^m \times 1 \times 2$)
2	PCLK / ($2^m \times 2 \times 2$)
3	PCLK / ($2^m \times 3 \times 2$)
4	PCLK / ($2^m \times 4 \times 2$)
...	...
4095	PCLK / ($2^m \times 4095 \times 2$)

(4) CSIGnSTR0 - CSIG status register 0

This register indicates the status of the CSIG.

Access This register can be read in 32-bit units.

Address <CSIGn_base> + 04_H

Initial Value $0000\ 0010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	CSIGn TSF	0	0	1	CSIGn DCE	0	CSIGn PE	CSIGn OVE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 27-17 CSIGnSTR0 register contents (1/2)

Bit position	Bit name	Function
7	CSIGnTSF	Transfer status flag: 0: Idle state 1: Transmission is in progress or being prepared Setting and clearing of this bit is as follows:

Master mode	Set by	Cleared by
Tx only mode	Writing to transmit register	Rising edge of CSIGTIC
Tx / Rx mode		Rising edge of CSIGTIR
Rx only mode		

Slave mode	Set by	Cleared by
Tx only mode	Writing to transmit register	Rising edge of CSIGTIC
Tx / Rx mode		Rising edge of CSIGTIR
Rx only mode	CSIGTSCK	

Table 27-17 CSIGnSTR0 register contents (2/2)

Bit position	Bit name	Function
3	CSIGnDCE	Data consistency check error flag: 0: No data consistency error detected 1: Data consistency error detected This bit is cleared by writing 1 to CSIGnSTCR0.CSIGnDCEC. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0.
1	CSIGnPE	Parity error flag: 0: No parity error detected 1: Parity error detected This bit is cleared by writing 1 to CSIGnSTCR0.CSIGnPEC. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0.
0	CSIGnOVE	Overrun error flag: 0: No overrun error detected 1: Overrun error detected This bit is cleared by writing 1 to CSIGnSTCR0.CSIGnOVEC. This bit is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0.

(5) CSIGnSTCR0 - CSIG status clear register 0

This register clears the status flags of the CSIGnSTR0 status register.

Access This register can be read/written in 16-bit units.

Address <CSIGn_base> + 08_H

Initial Value Reading this register returns always 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	CSIGn DCEC	0	CSIGn PEC	CSIGn OVEC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-18 CSIGnSTCR0 register contents

Bit position	Bit name	Function
3	CSIGnDCEC	0: No operation. Read value is always 0. 1: Clear data consistency error flag (CSIGnSTR0.CSIGnDCE)
1	CSIGnPEC	0: No operation. Read value is always 0. 1: Clear parity error flag (CSIGnSTR0.CSIGnPE)
0	CSIGnOVEC	0: No operation. Read value is always 0. 1: Clear overrun error flag (CSIGnSTR0.CSIGnOVE)

(6) CSIGnBCTL0 - CSIG Rx-only mode control register 0

This register enables/disables the data transfer in Rx-only mode.

Access This register can be read/written in 8-bit units.

Address <CSIGn_base> + 1000_H

Initial Value 01_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	CSIGnSCE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-19 CSIGnBCTL0 register contents

Bit position	Bit name	Function
0	CSIGnSCE	Disables/enables next data reception start by reading CSIGnRX0: 0: Next reception disabled 1: Next reception enabled For details refer to *2 "Receive-only mode" on page 2116* and *27.3.7 "Communication in slave mode" on page 2120* .

Caution CSIGnSCE must be set to 0 if transmit mode is enabled (CSIGnCTL0.CSIGnTXE = 1).

(7) CSIGnCFG0 - CSIG configuration register 0

This register configures the communication protocol – data length, parity, transfer direction, clock phase, and data phase.

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 1010_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	CSIGn PS[1:0]		CSIGn DLS[3:0]				0	0	0	0	0	CSIGn DIR	0	CSIGn DAP
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

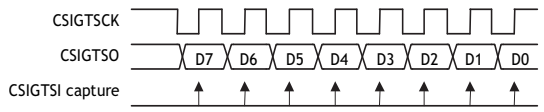
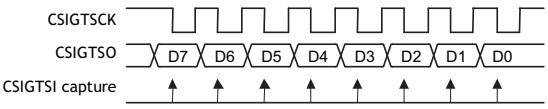
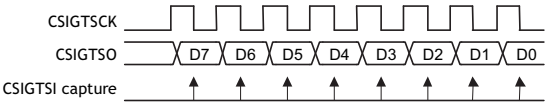
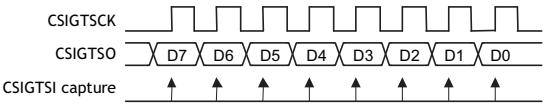
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIGnCTL0.CSIGnPWR = 0.

Table 27-20 CSIGnCFG0 register contents (1/2)

Bit position	Bit name	Function
29 to 28	CSIGnPS [1:0]	Specifies parity: CSIGn PS1 / CSIGn PS0 / Transmission / Reception 0 / 0 / No parity transmitted / No parity expected 0 / 1 / Add parity bit fixed at 0 / Parity bit is expected but not judged 1 / 0 / Add odd parity / Odd parity bit is expected 1 / 1 / Add even parity / Even parity bit is expected
27 to 24	CSIGnDLS [3:0]	Specifies data length: 0: Data length is 16 bits 1: Data length is 1 bit 2: Data length is 2 bits ... 15: Data length is 15 bits **Caution** Data lengths of less than 7 bits is only allowed in combination with the extended data length function. It is forbidden to transmit two consecutive data with a data length of less than 7 bits.

Table 27-20 CSIGnCFG0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>18</td><td>CSIGnDIR</td><td>Selects the serial data direction:
0: Data is sent/received with MSB first
1: Data is sent/received with LSB first</td></tr>
<tr><td>16</td><td>CSIGnDAP</td><td>Data phase selection bit
The control bits CSIGnDAP and CSIGnCTL1.CSIGnCKR determine the CSIGTSCK clock edge
• to shift serial data out via CSIGTSO
• to sample serial input data at CSIGTSI.
The following diagram shows the relation between CSIGTSCK, CSIGTSO and the sample point of CSIGTSI, depending on CSIGnCTL1.CSIGnCKR and CSIGnDAP:
<table>
<tr><th>CSIGn CKR</th><th>CSIGn DAP</th><th>Clock and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td> 
</td></tr>
<tr><td>0</td><td>1</td><td> 
</td></tr>
<tr><td>1</td><td>0</td><td> 
</td></tr>
<tr><td>1</td><td>1</td><td> 
</td></tr>
</table>
</td></tr>
</table>

(8) CSIGnTX0W - CSIG transmission register 0 for word access

This register stores the transmission data. It has to be used when the extended data length function is enabled (CSIGnCTL1.SCIGnEDLE = 1).

Access This register can be read/written in 32-bit units.

Address <CSIGn_base> + 1004_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	CSIGn EDL	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-21 CSIGnTX0W register contents

Bit position	Bit name	Function
29	CSIGnEDL	Specifies the extended data length configuration: 0: Normal operation 1: Extended data length activated The associated data is transmitted as 16-bit data. This bit can only be set if CSIGnCTL1.CSIGnEDLE = 1. It is automatically cleared if CSIGnCTL1.CSIGnEDLE is cleared.
15 to 0	CSIGn TX[15:0]	Data to be transmitted

(9) CSIGnTX0H - CSIG transmission register 0 for half word access

This register stores the transmission data. It can be used when the Extended Data Length function is disabled (CSIGnCTL1.CSIGnEDLE = 0).

Access This register can be read/written in 16-bit units.

Address <CSIGn_base> + 1008_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-22 CSIGnTX0H register contents

Bit position	Bit name	Function
15 to 0	CSIGn TX[15:0]	Data to be transmitted

(10) CSIGnRX0 - CSIG reception register 0

This register stores the received data.

Access This register can be read in 16-bit units.

Address <CSIGn_base> + $100C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIGn RX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 27-23 CSIGnRX0 register contents

Bit position	Bit name	Function
15 to 0	CSIGn RX[15:0]	Received data This register is initialized when CSIGnCTL0.CSIGnPWR changes from 0 to 1 or from 1 to 0.

(11) CSIGnEMU - CSIG emulation register

This register controls whether the CSIGn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <CSIGn_base> + 18_H

Initial Value 00_H

7	6	5	4	3	2	1	0
CSIGn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 27-24 CSIGnEMU register contents

Bit position	Bit name	Function
7	CSIGn SVSDIS	Emulation control 0: CSIGn can be stopped during emulation 1: CSIGn continuous operating during emulation

27.5 Operating Procedure Example

In the following a transmit/receive example in master mode in combination with a DMA is described.

The following instructions are based on the assumption that:

- Transmission data length is 8 bits (CSIGnCFG0.CSIGnDLS[3:0] = 1000_B)
- MSB is transmitted first (CSIGnCFG0.CSIGnDIR = 0)
- CSIGTIC interrupt at the end of the transfer (CSIGnCTL1.CSIGnSLIT = 0)
- Normal clock and data phase (CSIGnCTL1.CSIGnCKR = 0, CSIGnCFG0.CSIGnDAP = 0)
- The number of data packets is 10 (0 to 9)

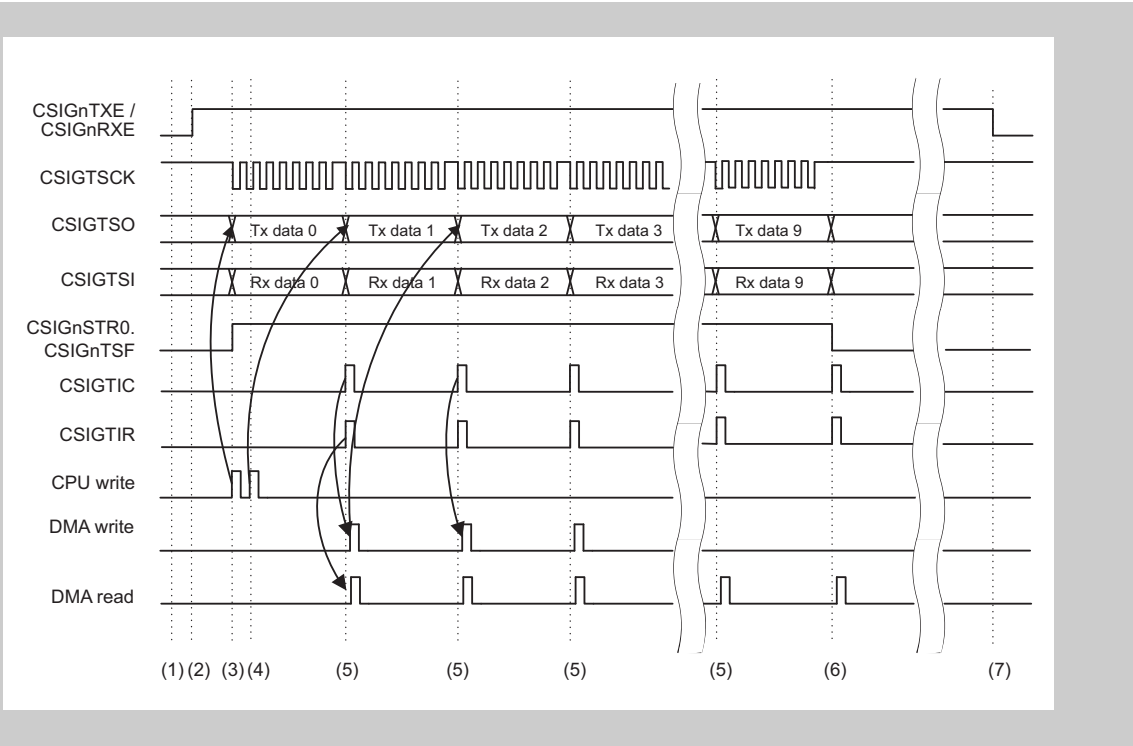


Figure 27-27 Communication in master mode

Procedure:

1. Configure the communication protocol in register CSIGnCFG0.
2. In the CSIGnCTL0 register, set bits CSIGnPWR = 1 (enable the clock), CSIGnTXE = 1 (enable transmission), CSIGnRXE = 1 (enable reception). The data output CSIGTSO is enabled.
3. Write the first data packet to be sent to the transmission register CSIGnTX0H. Transmission starts automatically when the first data is available.
4. Write the second data to CSIGnTX0H. Writing the second packet immediately after the first one avoids unnecessary delays between the packets.
5. After every packet that has been transmitted, the interrupts CSIGTIC and CSIGTIR are generated. CSIGTIC indicates that the next packet can be

written to CSIGnTX0H. CSIGTIR indicates that the reception register CSIGnRX0 must be read.
In this example, CPU write and DMA write are equivalent.

6. No more write action is required after completion of packet 8. Packet 9 (the last packet) has been written in advance.
However, the reception register CSIGnRX0 must be read after completion of packets 8 and 9.

7. To finally disable the transmit/receive operation, clear CSIGnCTL0.CSIGnTXE and CSIGnCTL0.CSIGnRXE.

Chapter 28 Clocked Serial Interface H (CSIH)

This chapter contains a generic description of the Clocked Serial Interface H (CSIH).

The first section describes all V850E2/Fx4-H specific properties, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

28.1 V850E2/Fx4-H CSIH Features

Instances

This microcontroller has following number of instances of the Clocked Serial Interface H.

Table 28-1 Instances of CSIH

Clocked Serial Interface H	
Instance	3
Name	CSIH0 to CSIH2

Instances index n

Throughout this chapter, the individual instances of a Clocked Serial Interface H is identified by the index "n" (n = 0 to 2), for example, CSIHnCTL0 for the CSIHn control register 0.

Chip select index x

The Clocked Serial Interface H has up to 8 chip select signals. Throughout this chapter, the individual chip select signals are identified by the index “x” with x = 0 to 7, thus a certain chip select signal is denoted as CSx.

Register addresses

All CSIHn register addresses are given as address offsets to the individual base address <CSIHn_base>.
The base address <CSIHn_base> of each CSIHn is listed in the following table:

Table 28-2 Register base addresses <CSIHn_base>

CSIHn instance	<CSIHn_base> address
CSIH0	FF6C 0000_H
CSIH1	FF6D 0000_H
CSIH2	FF6E 0000_H

Clock supply All Clocked Serial Interfaces H provide one clock input:

Table 28-3 CSIHn clock supply

CSIHn instance	CSIHn clock	Connected to
CSIH0	PCLK	Clock Controller CKSCLK_109
CSIH1	PCLK	Clock Controller CKSCLK_109
CSIH2	PCLK	Clock Controller CKSCLK_109

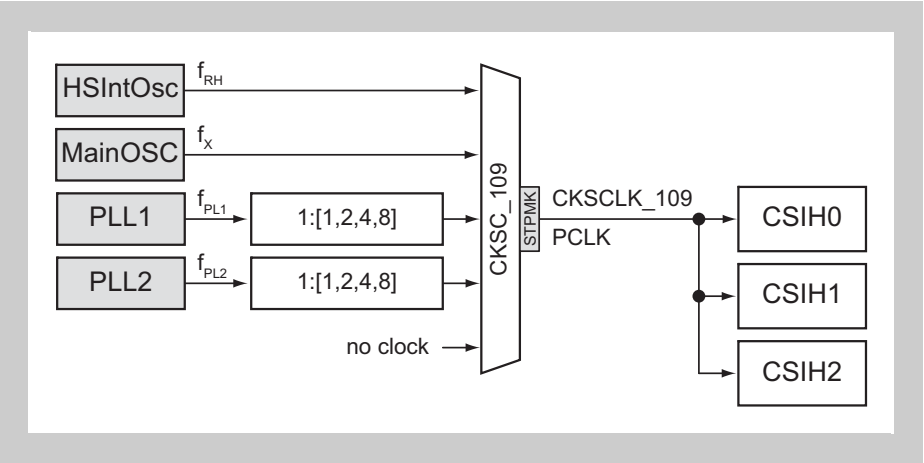


Figure 28-1 CSIH clock supply

Interrupts and DMA/DTS The Clocked Serial Interfaces H can generate the following interrupt and DMA/DTS requests:

Table 28-4 CSIHn interrupt and DMA/DTS requests (1/2)

CSIHn signals	Function	Connected to
CSIH0:		
CSIHTIC	Communication status interrupt	Interrupt Controller INTCSIH0IC DMA Controller trigger 77 DTS Controller trigger 12
CSIHTIR	Reception status interrupt	Interrupt Controller INTCSIH0IR DMA Controller trigger 81 DTS Controller trigger 14
CSIHTIRE	Communication error interrupt	Interrupt Controller INTCSIH0IRE
CSIHTIJC	Job completion interrupt	Interrupt Controller INTCSIH0IJC DMA Controller trigger 78 DTS Controller trigger 13
CSIH1:		
CSIHTIC	Communication status interrupt	Interrupt Controller INTCSIH1IC DMA Controller trigger 88 DTS Controller trigger 19
CSIHTIR	Reception status interrupt	Interrupt Controller INTCSIH1IR DMA Controller trigger 87 DTS Controller trigger 18
CSIHTIRE	Communication error interrupt	Interrupt Controller INTCSIH1IRE

Table 28-4 CSIHn interrupt and DMA/DTS requests (2/2)

CSIHn signals	Function	Connected to
CSIHTIJC	Job completion interrupt	Interrupt Controller INTCSIH1IJC DMA Controller trigger 89 DTS Controller trigger 20
CSIH2:		
CSIHTIC	Communication status interrupt	Interrupt Controller INTCSIH2IC DMA Controller trigger 97
CSIHTIR	Reception status interrupt	Interrupt Controller INTCSIH2IR DMA Controller trigger 96
CSIHTIRE	Communication error interrupt	Interrupt Controller INTCSIH2RE
CSIHTIJC	Job completion interrupt	Interrupt Controller INTCSIH2IJC DMA Controller trigger 98

CSIH H/W reset

The Clocked Serial Interfaces H and their registers are initialized by the following reset signal:

Table 28-5 CSIGHn reset signal

CSIHn	Reset signal
CSIHn	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

Internal signals

The internal signal connections of the Clocked Serial Interface H are listed in the following table.

Table 28-6 CSIHn internal signal connections

CSIHn signal	Function	Connected to
CSIH0:		
CSIHTSSO	CSIHTSO output buffer control	Port CSIH0SO output buffer control
CSIH1:		
CSIHTSSO	CSIHTSO output buffer control	Port CSIH1SO output buffer control
CSIH2:		
CSIHTSSO	CSIHTSO output buffer control	Port CSIH2SO output buffer control

I/O signals The I/O signals of the Clocked Serial Interface H are listed in the following table.

Table 28-7 CSIHn I/O signals

CSIHn signal	Function	Connected to
CSIH0:		
CSIHTSCK	Serial clock signal	Port CSIH0SC[a]
CSIHTSI	Serial data input signal	Port CSIH0SI[a]
CSIHTSO	Serial data output signal	Port CSIH0SO
$\overline{\text{CSIHTSSI}}$	Slave select input signal	Port $\overline{\text{CSIH0SSI}}$[a]
CSIHTSHSG	Handshake signal	Port CSIH0RY[a]
CSIHTCSS[7:0]	Chip select signals	Port CSIH0CSS[7:0]
CSIH1:		
CSIHTSCK	Serial clock signal	Port CSIH1SC[a]
CSIHTSI	Serial data input signal	Port CSIH1SI[a]
CSIHTSO	Serial data output signal	Port CSIH1SO
$\overline{\text{CSIHTSSI}}$	Slave select input signal	Port $\overline{\text{CSIH1SSI}}$[a]
CSIHTSHSG	Handshake signal	Port CSIH1RY[a]
CSIHTCSS[7:0]	Chip select signals	Port CSIH1CSS[7:0]
CSIH2:		
CSIHTSCK	Serial clock signal	Port CSIH2SC[a]
CSIHTSI	Serial data input signal	Port CSIH2SI[a]
CSIHTSO	Serial data output signal	Port CSIH2SO
$\overline{\text{CSIHTSSI}}$	Slave select input signal	Port $\overline{\text{CSIH2SSI}}$[a]
CSIHTSHSG	Handshake signal	Port CSIH2RY[a]
CSIHTCSS[7:0]	Chip select signals	Port CSIH2CSS[7:0]

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

Caution The initial state of the input signals noise filters block the input signals. Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

28.1.1 Data consistency check

The following table shows the CSIHnSO ports and their capability to use them for data consistency checks. Refer to *"Error detection"* in the section *"Functional Description"* below for details about data consistency checks.

Table 28-8 CSIHn data consistency check ports

CSIGn I/O port signal	Port	Alternative function	Data consistency check
CSIH0:			
CSIH0SO	P4_1	ALT_OUT3	possible
CSIH1:			
CSIH4SO	P1_8	ALT_OUT3	possible
CSIH2:			
CSIH2SO	P1_3	ALT_OUT4	possible
	P21_3	ALT_OUT2	possible

Port configuration

Input/output control of the CSIHnSO port is done by the CSIHn module, thus set PIPCn.PIPCn_m = 1.
All other necessary configuration is done automatically, when the data consistency check is enabled.

28.2 Functional Overview

Features summary

- Three-wire serial synchronous data transfer
- Master mode and slave mode selectable
- Multiple slaves configuration plus RCB (Recessive Configuration for Broadcasting) thanks to eight configurable chip select output signals
- Slave select input signal ($\overline{\text{CSIHTSSI}}$)
- Built-in baud rate generator
- Baud rate adjustable; in slave mode determined by input clock
- Maximum transmission speed:
  - in master mode: PCLK/4
  - in slave mode: PCLK/6
- Phase of clock and data selectable
- Data transfer with MSB or LSB first selectable
- Transfer data length selectable from 7 to 16 bits in 1-bit units
- EDL (Extended Data Length) function for transferring data with more than 16 bits
- Three selectable transfer modes:
  - transmit-only mode
  - receive-only mode
  - transmit/receive mode
- Built-in handshake function
- Error detection (data consistency check, parity, time-out, overrun)
- Full support of job concept
- 128 words I/O buffer memory
- Memory mode selectable (FIFO, dual buffer, Tx-only buffer, direct access)
- Four different interrupt request signals (CSIHTIC, CSIHTIR, CSIHTIRE, CSIHTIJC)
- LBM (Loop Back Mode) function for self test

The block diagram shows the main components of the CSIH.

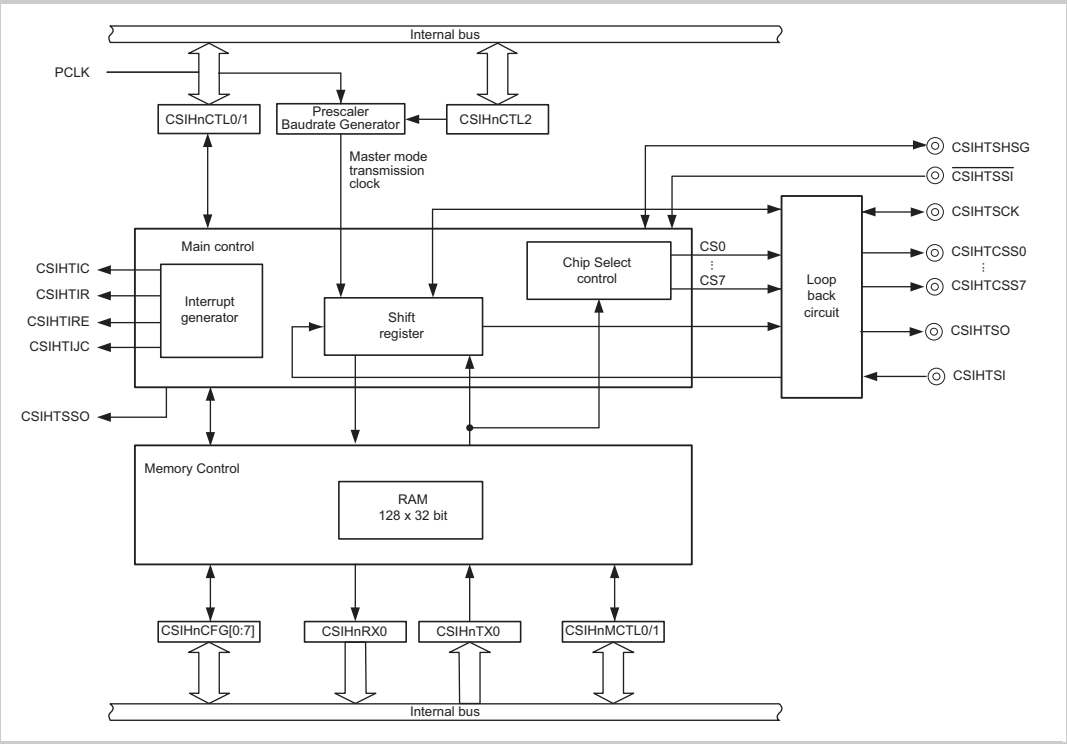


Figure 28-2 CSIH block diagram

In master mode, the transmission clock CSIHTSCK is generated by the built-in baud rate generator (BRG). In slave mode, the transmission clock is received from an external source.

The built-in memory can be configured as FIFO, dual buffer (separate transmit and receive buffers), or transmit-only buffer. It can also be bypassed for data transmission and reception without buffering.

The loop back circuit disconnects the CSIH completely from the ports and supports internal self test.

Note This chapter describes the following modes:

- The “operating mode” separates between master and slave mode. In this context, only a master can control and communicate with several slaves (for details see *28.3.1 “Operating modes (master/slave)” on page 2152*).
- The “job mode” is related to the Autosar job concept (for details see *28.3.4 “The job concept” on page 2157*).
- The “memory mode” takes the various configurations of the associated buffer memory into account (for details see *28.3.7 “CSIH buffer memory” on page 2160*).
- The “data transfer mode” specifies the kind of the communication – transmit-only, receive only, or both (for details see *28.3.8 “Data transfer modes” on page 2162*).

28.3 Functional Description

The Clocked Serial Interface uses three signals for communication:

- Transmission clock CSIHTSCK (output in master mode, input in slave mode)
- Data output signal CSIHTSO
- Data input signal CSIHTSI

Additional signals are available for external control and monitoring.

Data transmission is bit-wise and serial and synchronous to the transmission clock.

The most important registers for setting up the CSIH are:

Register	Function
CSIHnCTL0	Enables/disables transmission clock, data transmission, and data reception. Defines end-of-job behavior and enables/ disables buffering (bypass of the buffer).
CSIHnCTL1	Controls options like interrupt timing, extended data length, job feature, data consistency check, loop-back mode, handshake, etc.
CSIHnCTL2	Selects master/slave mode and – effective in master mode – the baud rate of the internal generator
CSIHnMCTL0	Selects memory mode and specifies time-out
CSIHnMCTL1	Controls the memory in FIFO mode
CSIHnMCTL2	Controls the memory in buffer mode
CSIHnCFGx	Registers to configure the communication protocol for each chip select signal

28.3.1 Operating modes (master/slave)

For a particular CSIH module, the master or slave mode determines the source of the transmission clock.

(1) Master mode

In master mode, the serial transmission clock is generated by the internal baud rate generator (BRG) and provided to the slave(s) by the signal CSIHTSCK.

Master mode is enabled by setting CSIHnCTL2.CSIHnPRS[2:0] to anything but 111_B. In master mode, the frequency setting of the BRG becomes effective.

Chip select signals

In master mode, one or several chip select signals can be used. If several slaves are connected to the master, the chip select signals can be used to address one or several of the slaves. Only a selected slave is then enabled for communication.

The communication protocol as well as additional parameters are stored separately for each chip select signal. This makes it possible to adapt the data transfer individually to the requirements of each slave. For details see *28.3.3 “Chip selection (CS) features” on page 2155* .

Clock defaults

The default level of CSIHTSCK depends on the clock phase selection bit: It is high when CSIHnCFGx.CSIHnCKPx = 0, and is low when CSIHnCFGx.CSIHnCKPx = 1.

The example below shows the communication in master mode for 8 data bits, CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0, and MSB first:

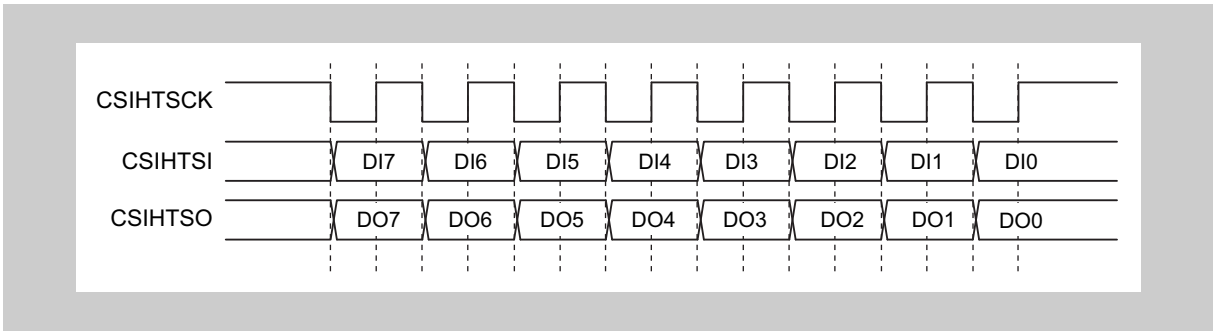


Figure 28-3 Transmit/receive in master mode

(2) Slave mode

In slave mode, another device is the communication master and provides the transmission clock. Send/receive operation normally starts as soon as a clock signal is detected.

Slave mode is selected by setting CSIHnCTL2.CSIHnPRS[2:0] to 111_B.

In slave mode the transmission protocol setting of the CSIHnCFG0 register are relevant:

- CSIHnPS0: parity usage
- CSIHnDLS0: data length selection
- CSIHnDIR0: data direction
- CSIHnCKP0,CSIHnDAP0: clock and data phase

Note When using slave mode, disable the baud rate generator (BRG) by clearing bits CSIHnCTL2.CSIHnBRS[11:0].

28.3.2 Master/slave connections

(1) One master and one slave

The following figure illustrates the connections between one master and one slave.

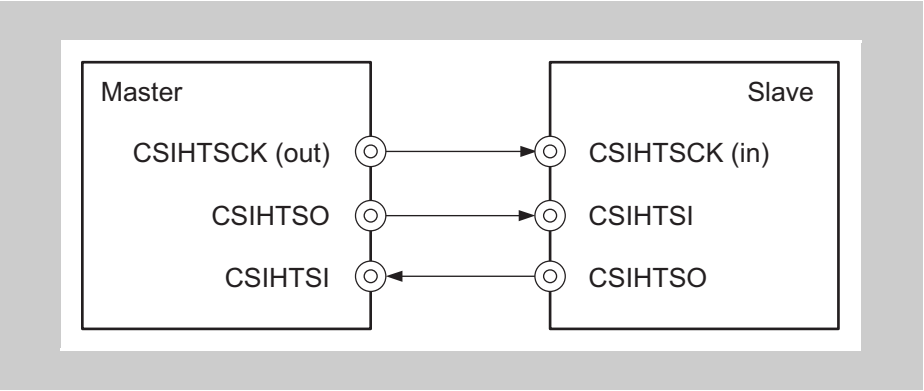


Figure 28-4 Direct master/slave connection

(2) One master and multiple slaves

The following figure illustrates the connections between one master and multiple slaves. In this example, the master provides one chip select (CS) signal to each of the slaves. This signal is connected to the slave select input $\overline{\text{CSIHTSSI}}$ of the slave.

The recognition of the $\overline{\text{CSIHTSSI}}$ signal can be enabled/disabled by bit CSIHnCTL1.CSIHnSSE.

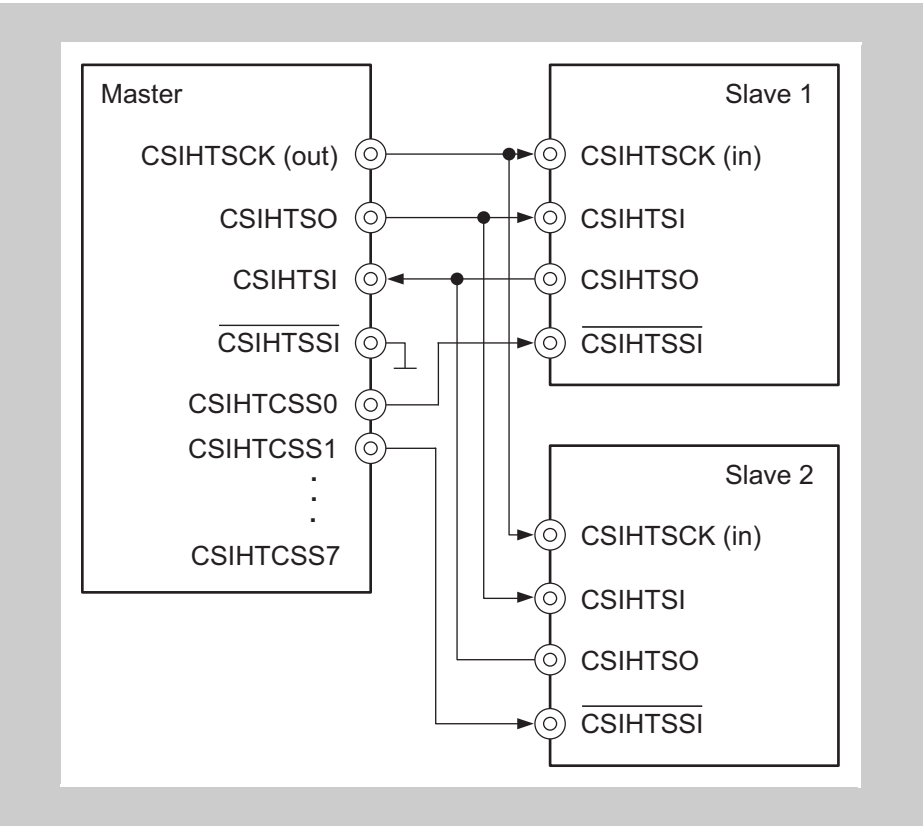


Figure 28-5 Master to multiple slaves connection

By default, the chip select level is active low. That means, a CSIH slave is selected (enabled) when its $\overline{\text{CSIHTSSI}}$ signal has low level. However, to adapt the CS to other devices, the output level of each chip select signal can also be programmed to be active high.

If a slave is not selected, it will neither receive nor transmit data. In addition, its output CSIHTSO is set to input mode in order to avoid interference with the output of another slave that was selected.

(3) CSIHTSO output control

The CSIH provides the signal CSIHTSSO to control the buffer of the serial data output signal CSIHTSO. By use of this signal the CSIHTSO port output buffer is disabled when all of the following conditions are met:

- The CSIH is enabled (CSIHnCTL0.CSIHnPWR = 1)
- The CSIH is operated in transmit-only or transmit/receive mode (CSIHnCTL0.CSIHnTXE = 1)
- The CSIH is operated with slave select enabled (CSIHnCTL1.CSIHnSSE = 1).
- The slave mode selection signal $\overline{\text{CSIHTSSI}}$ is inactive, i.e. on high level.

By this signal congestions on the external CSIHTSO signal line are avoided.

28.3.3 Chip selection (CS) features

The chip select signals CSIHTCSSx can be used by the master to select one or several slaves for communication.

(1) Configuration registers

The parameters for each chip select signal CSIHTCSSx are defined in the corresponding configuration register CSIHnCFGx. The parameters include the communication protocol and additional CS parameters.

The communication protocol specifies:

- Data length: The number of bits to be sent or received.
- Transfer direction: MSB or LSB first.
- Parity usage: Odd, even, or none.
- Clock phase and data phase.

Additional parameters for each chip select and only available in master mode are:

- Prescaler selection of the baudrate generator separately for each chip select
- Chip select priority: Separates between “dominant” and “recessive” chip select signals. The priority applies if two or more chip selects with different configurations are simultaneously activated for message broadcasting. In this case, the configuration that is set as dominant is used.

  The principle is also called “Recessive Configuration for Broadcasting” (RCB).

Caution It is forbidden to specify several chip select signals as dominant with different configurations unless all dominant chip selects have the same configuration.

- Chip select timing:
  - Setup time T_{setup}: The time from setting the CS signal active to starting data output.
  - Inter-data time T_{inter}: The time between data while the same CS signal is active.
  - Hold time T_{hold}: Hold time of CS active level before changing the CS.
  - Idle time T_{idle}: Inactive time after terminating a CS signal or after every data transfer to the same CSx.

The various CS timings are illustrated in the following figure. The example shows the default – the CS1 and CS2 signals are active low. But the active level can be specified individually for each CS.

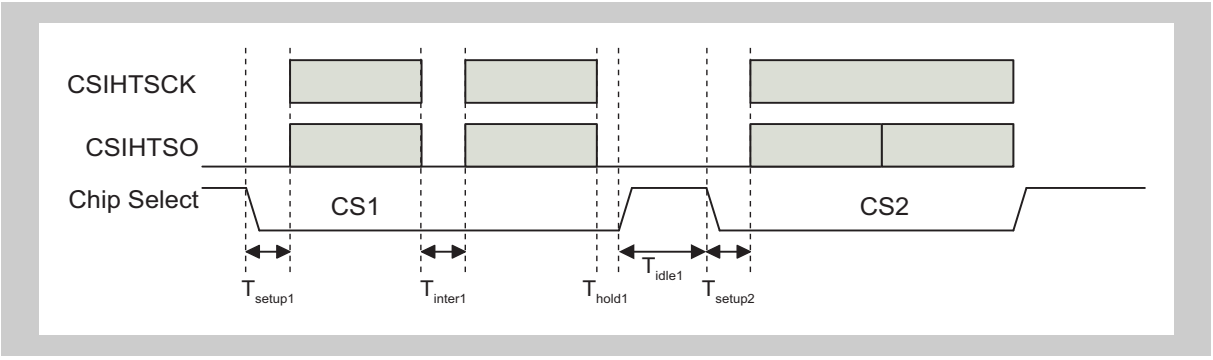


Figure 28-6 Chip select timings

Note that each CS can have a different value for setup, inter-data gap, hold and idle periods.

A particular chip select signal is activated by setting the appropriate bit in the transmission register CSIHnTX0W.CSIHnCS[7:0].

The reception register indicates in CSIHnRX0W.CSIHnCS[7:0] the chip select signal associated with the received data.

(2) CS example

The following figure shows an example of two consecutive transmissions.

The first communication uses CS0 to address one single slave. The second enables CS0 and CS1 to broadcast a message to two slaves. The priority of CS0 is set to “recessive”, the priority of CS1 to “dominant”.

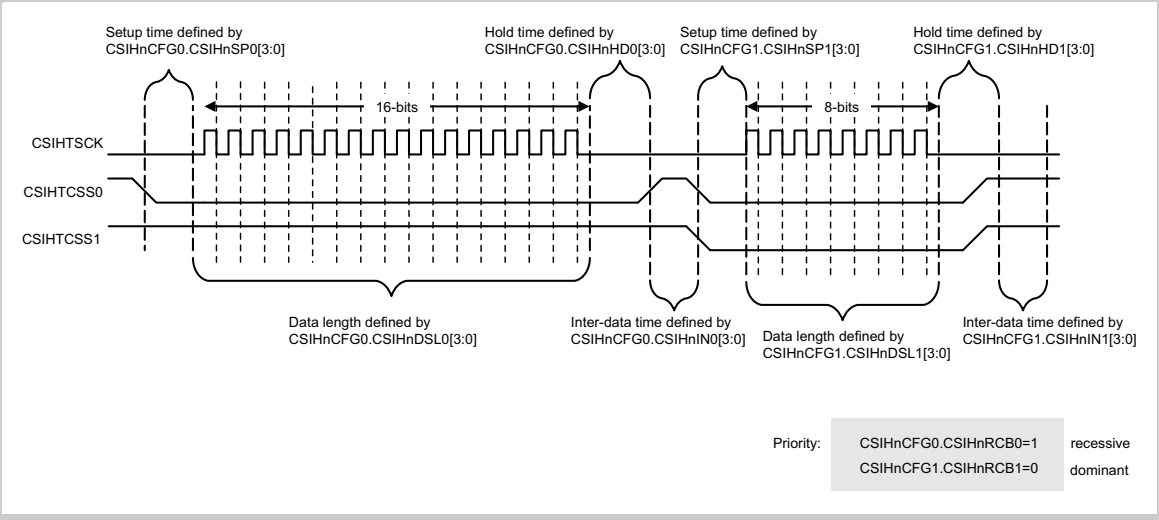


Figure 28-7 Chip select and RCB example

28.3.4 The job concept

In terms of CSIH, a job consists of a number of data that are transferred.

Note The job mode can only be used in master mode (CSIHnCTL2.CSIHnPRS[2:0] ≠ 111_B).

Job mode enable The job mode is enabled and disabled by CSIHnCTL1.CSIHnJE, while the CSIH is disabled by CSIHnCTL0.CSIHnPWR = 0.

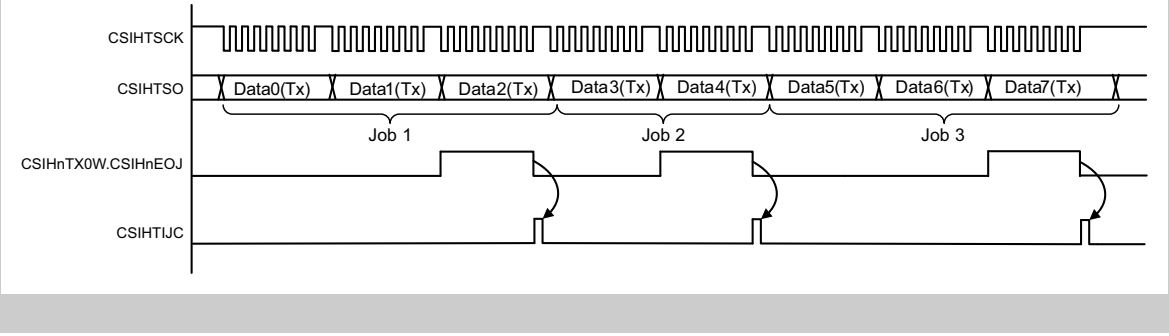


Figure 28-8 Job examples

A job ends when a data with the end-of-job bit set, i.e. with CSIHnTX0W.CSIHnEOJ = 1.

A communication stop can be specified to occur after a job has finished. This is done by setting CSIHnCTL0.CSIHnJOBE. When CSIHnJOBE is set, the communication continues until a data is sent, for which the CSIHnEOJ bit was set. After this data is sent, the communication is stopped and the end-of job-interrupt CSIHTIJC is generated.

28.3.5 Chip select timing details

(1) Changing the clock phase

The serial clock level specified by CSIHnCFGx.CSIHnCKPx may be changed during an idle time. The minimum value of an idle time is one period of transmission clock (CSIHTSCK).

If the idle time is set to 0.5 transmission clock periods (in CSIHnCFGx.CSIHnIDx[2:0]) and two consecutive data are sent with different CSIHnCFGx.CSIHnCKPx configuration, the idle time is automatically extended to one period of CSIHTSCK.

(2) Changing the data phase

The bit CSIHnCFGx.CSIHnDAPx defines the phase of the data bits relative to the clock.

If CSIHnCFGx.CSIHnDAPx = 0, the transmission clock CSIHTSCK holds its level after the last bit of a data is transferred.

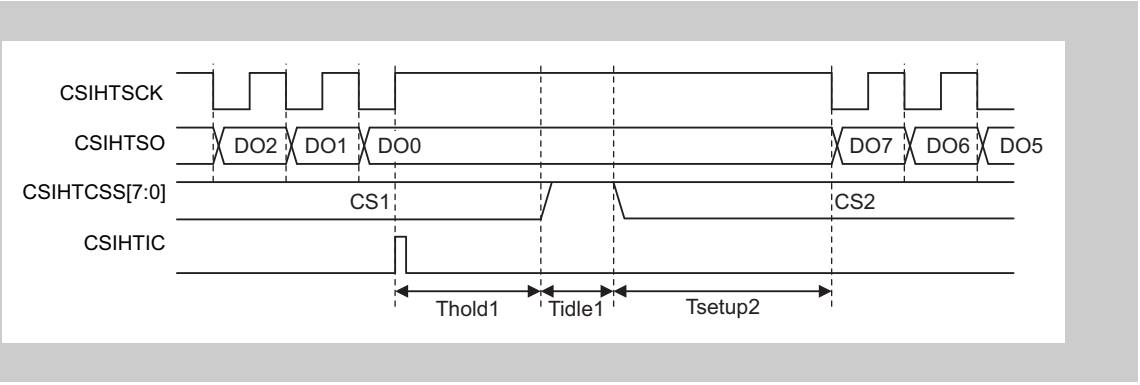


Figure 28-9 Data phase timing with CSIHnCFG1.CSIHnCKP1 = 0, CSIHnCFG1.CSIHnDAP1 = 0 and CSIHnCFG2.CSIHnCKP2 = 0, CSIHnCFG2.CSIHnDAP2 = 0

If the default clock phase changes between two consecutive chip selects, the transmission clock CSIHTSCK changes its level after the last bit of the first data is transferred:

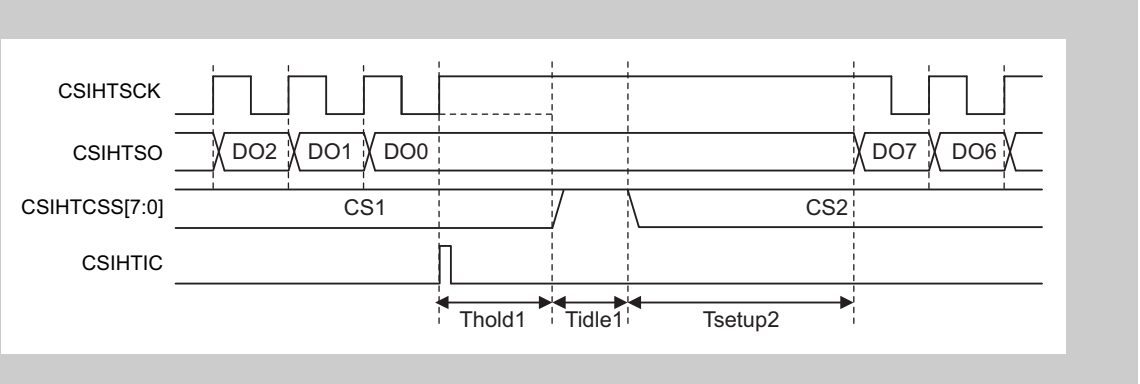


Figure 28-10 Data phase timing with CSIHnCFG1.CSIHnCKP1 = 1, CSIHnCFG1.CSIHnDAP1 = 0 and CSIHnCFG2.CSIHnCKP2 = 0, CSIHnCFG2.CSIHnDAP2 = 1

Note that the minimum idle time of one CSIHTSCK period is automatically inserted, if CSIHnCFGx.CSIHnIDx[2:0] = 0 (T_{idle1} = 0.5 CSIHTSCK periods).

28.3.6 Transmission clock selection

In master mode, the transmission baudrate is selectable using

- CSIHnCTL2.CSIHnPRS[2:0]
- CSIHnCTL2.CSIHnBRS[11:0]
- CSIHnCFGx.CSIHn.CSIHnPSCLx

While the settings in the CSIHnCTL2 register determine the transmission base clock CSIHBCLK, a chip select dedicated prescaler, controlled by CSIHnCFGx.CSIHn.CSIHnPSCLx, allows to generated different baudrates for different chip selects.

The following figure shows a block diagram of the baudrate generator.

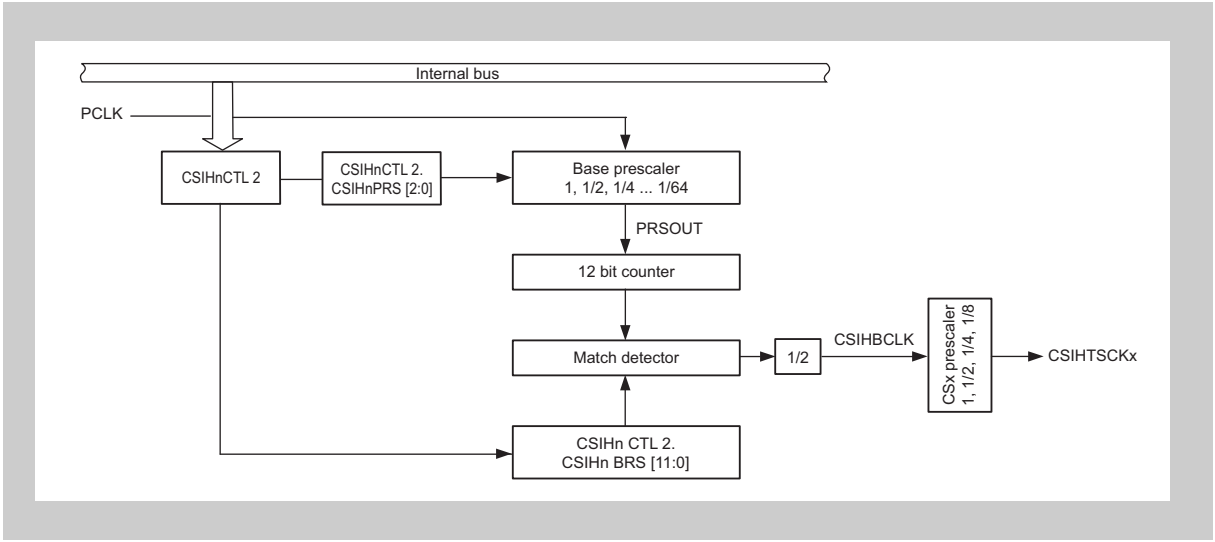


Figure 28-11 Baud rate generator block diagram

Clearing CSIHnCTL2.CSIHnBRS[11:0] disables the baudrate generator, and thus all CSIHTSCKx are stopped.

Baud rate calculation

The baudrate is calculated as:

$$CSIHTSCKx = PCLK / (2^m \times k \times 2 \times 2^j)$$

where

m = CSIHnCTL2.CSIHnPRS[2:0] = 0 to 6

k = CSIHnCTL2.CSIHnBRS[11:0] = 1 to 4095

j = CSIHnCFGx.CSIHnPSCLx[1:0]

Baud rate limits When setting the baud rate, please note:

- Maximum acceptable baud rate in master mode is PCLK / 4.
- Maximum acceptable baud rate in slave mode is PCLK / 6 (must be ensured by the external master).
- Minimum baud rate in both modes is PCLK / 524160.

Example If PCLK = 48 MHz, the maximum baud rate is

- 16 Mbps (PCLK / 4) in master mode
- 8 Mbps (PCLK / 6) in slave mode

The slowest baud rate is 91.6 bps (PCLK / 524160).

28.3.7 CSIH buffer memory

The CSIH has a configurable RAM that can be used for buffered I/O. The size is 128 words. One word is comprised of 32 bits data plus 7 bits ECC.

The following configurations are available:

Mode	CSIHnCTL0. CSIHnMBS	CSIHnMCTL0. CSIHnMMS[1:0]
FIFO mode	0	00_B
Dual buffer mode		01_B
Transmit-only buffer mode		10_B
Direct access mode	1	X

(1) FIFO mode

In FIFO mode, data can be written to the CSIHnTX0W register without waiting for completion of the transmission, and data can be received without reading the CSIHnRX0W register immediately, provided the FIFO is not full.

Data to be transmitted is stored to the FIFO memory. Transmission and reception occur simultaneously – one bit is sent, one bit is received. That means, received data overwrites the transmitted data in the FIFO.

The CSIH automatically updates the respective FIFO memory pointers when a data package is committed, sent or received:

Pointer description	Control bits	Range
Number of unsent data	CSIHnSTR0.CSIHnSPF[7:0]	0 to 128
Number of receivable words	CSIHnSTR0.CSIHnSRP[7:0]	0 to 128
Address of data to be sent	CSIHnMRWP0.CSIHnTRWA[6:0]	0 to 127
Address of received data	CSIHnMRWP0.CSIHnRRA[6:0]	0 to 127

The CSIH status register contains also two FIFO status flags:

- CSIHnSTR0.CSIHnFLF: FIFO full
- CSIHnSTR0.CSIHnEMF: FIFO empty

When this mode is started, bit CSIHnSTCR0.CSIHnPCT must be set. This resets all FIFO pointers and flags.

(2) Dual buffer mode

In this mode, the memory is divided into two parts of equal size – this means 64 words for transmit data and 64 words for received data. In dual buffer mode, the respective buffer pointers indicate:

Pointer description	Pointers[a]	Range
Destination address for data written to or read from CSIHnTX0W/H	CSIHnMRWP0.CSIHnTRWA[6:0]	0 to 63
Address of data read from CSIHnRX0W/H	CSIHnMRWP0.CSIHnRRA[6:0]	0 to 63

a) Both pointers are automatically incremented after each read/write.

(3) Transmit-only buffer mode

In this mode the entire memory is used to save transmission data.

Received data must be read directly from CSIHnRX0W/H.

In transmit-only buffer mode, the respective buffer pointer is:

Pointer description	Pointer	Range
Destination address for data written to or read from CSIHnTX0W/H	CSIHnMRWP0.CSIHnTRWA[6:0]	0 to 127

(4) Direct access mode

In direct access mode, the CSIH memory is completely bypassed:

- Transmission data provided by the CPU to the transmission register CSIHnTX0W or CSIHnTX0H is directly copied to the shift register.
- Reception data is directly copied from the shift register to the reception register CSIHnRX0W or CSIHnRX0H.

28.3.8 Data transfer modes

(1) Transmit-only mode

Setting CSIHnCTL0.CSIHnTXE = 1 and CSIHnCTL0.CSIHnRXE = 0 puts the CSIH in transmit-only mode. Start of transmission depends on the memory mode:

- In case of FIFO or direct access mode, transmission starts when transmit data is written to the CSIHnTX0W or CSIHnTX0H register.
- In case of dual buffer or transmit-only buffer mode, transmission starts when bit CSIHnMCTL2.CSIHnBTST is set.

(2) Receive-only mode

Setting CSIHnCTL0.CSIHnTXE = 0 and CSIHnCTL0.CSIHnRXE = 1 puts the CSIH in receive-only mode.

In master mode, the start of reception depends on the memory mode:

- In case of FIFO, transmit-only buffer or direct access mode, reception starts when dummy data is written in the CSIHnTX0W or CSIHnTX0H register.
- In case of dual buffer mode, reception starts when bit CSIHnMCTL2.CSIHnBTST is set.

In slave mode, reception starts as soon as the transmission clock CSIHTSCK from the master is received. It is not necessary to write data to the CSIHnTX0W or CSIHnTX0H register of the slave.

(3) Transmit & receive mode

Setting CSIHnCTL0.CSIHnTXE = 1 and CSIHnCTL0.CSIHnRXE = 1 puts the CSIH in transmit / receive mode.

The start of the communication (transmission and reception) depends on the memory mode:

- In case of FIFO or direct access mode, communication starts when transmit data is written to the CSIHnTX0W or CSIHnTX0H register.
- In case of dual buffer or transmit-only buffer mode, communication starts when bit CSIHnMCTL2.CSIHnBTST is set.

(4) Summary

The following table provides a summary. It shows how the data transfer is started in the various memory, operating, and transfer modes.

Table 28-9 Start of data transfer

Memory and operating mode		Transfer mode	
		Transmit-only Transmit / receive	Receive-only
FIFO, direct access	Master	Writing to CSIHnTX0 register	Writing to CSIHnTX0 register
	Slave	Writing to CSIHnTX0 register and start of master clock	Incoming clock from master
Transmit-only buffer	Master	CSIHnMCTL2.CSIHnBTST = 1	CSIHnMCTL2.CSIHnBTST = 1
	Slave	CSIHnMCTL2.CSIHnBTST = 1 and start of master clock	Incoming clock from master
Dual buffer	Master	CSIHnMCTL2.CSIHnBTST = 1	CSIHnMCTL2.CSIHnBTST = 1
	Slave	CSIHnMCTL2.CSIHnBTST = 1 and start of master clock	Incoming clock from master

28.3.9 Data length selection

(1) Data length between 7 and 16 bits

The length of a data is selectable for each chip select signal from 7 to 16 bits using CSIHnCFGx.CSIHnDLSx[3:0]. The examples below show the communication with MSB first (CSIHnCFGx.CSIHnDIRx = 0).

Data length = 16 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 0000_B):

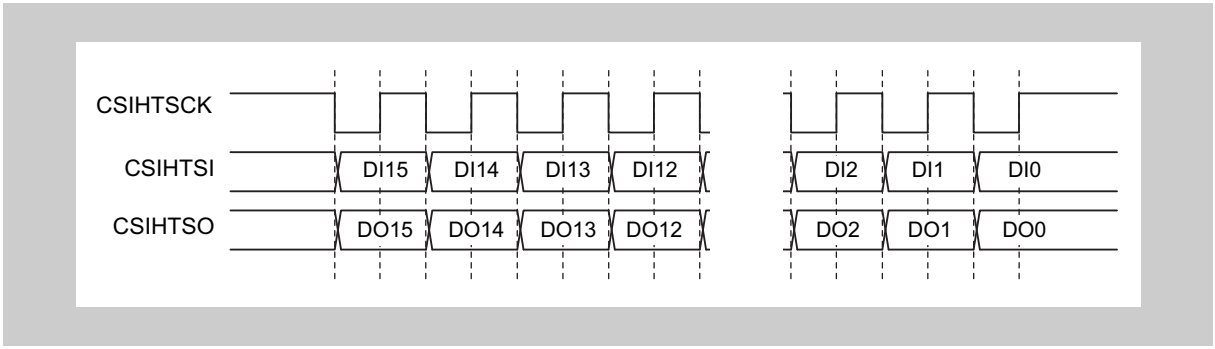


Figure 28-12 16 bit data length, MSB first

Data length = 14 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1110_B):

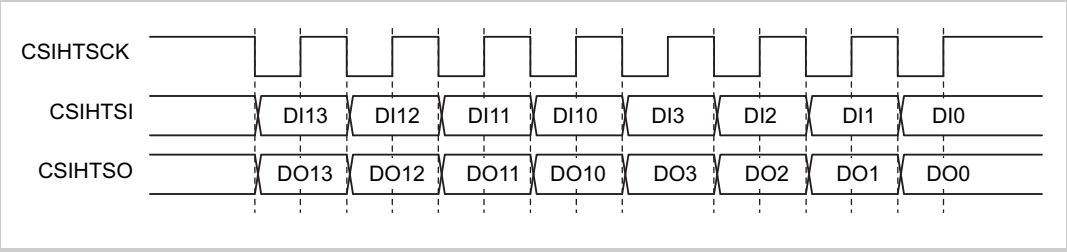


Figure 28-13 14 bit data length, MSB first

(2) Data length greater than 16 bits

If the length of the data to be sent/received exceeds 16 bits, the extended data length (EDL) feature can be used.

EDL is enabled by setting bit CSIHnCTL1.CSIHnEDLE to 1.

EDL works as follows:

- The data has to be broken into 16-bit blocks plus remainder. For example, a string of 42 bits would be broken into two 16-bit blocks plus 10 bits.
- The remainder defines the “data length” that has to be specified in CSIHnCFGx.CSIHnDLSx[3:0].
- For transmitting the 16-bit blocks, CSIHnTX0W.CSIHnEDL must be set to 1. In this case, the data written to CSIHnTX0W is sent as a 16-bit data length regardless of the CSIHnCFG0.CSIHnDLS[3:0] bits.
- The transfer is complete after a block with the specified data length (the remainder – with CSIHnTX0W.CSIHnEDL = 0) has been sent.

Example Example for sending 40-bit data, for example the string $123456789A_H$:

40 bits are split into 2 x 16 bits plus 8 bits.

- Initialize CSIHnCFGx.CSIHnDLSx[3:0] = 8.
- To send the string $123456789A_H$ with MSB first, write the following sequence to CSIHnTX0W:
  - $2000\ 1234_H$ (CSIHnTX0W.CSIHnEDL = 1)
  - $2000\ 5678_H$ (CSIHnTX0W.CSIHnEDL = 1)
  - $0000\ 009A_H$ (CSIHnTX0W.CSIHnEDL = 0)

The following figure illustrates the timing.

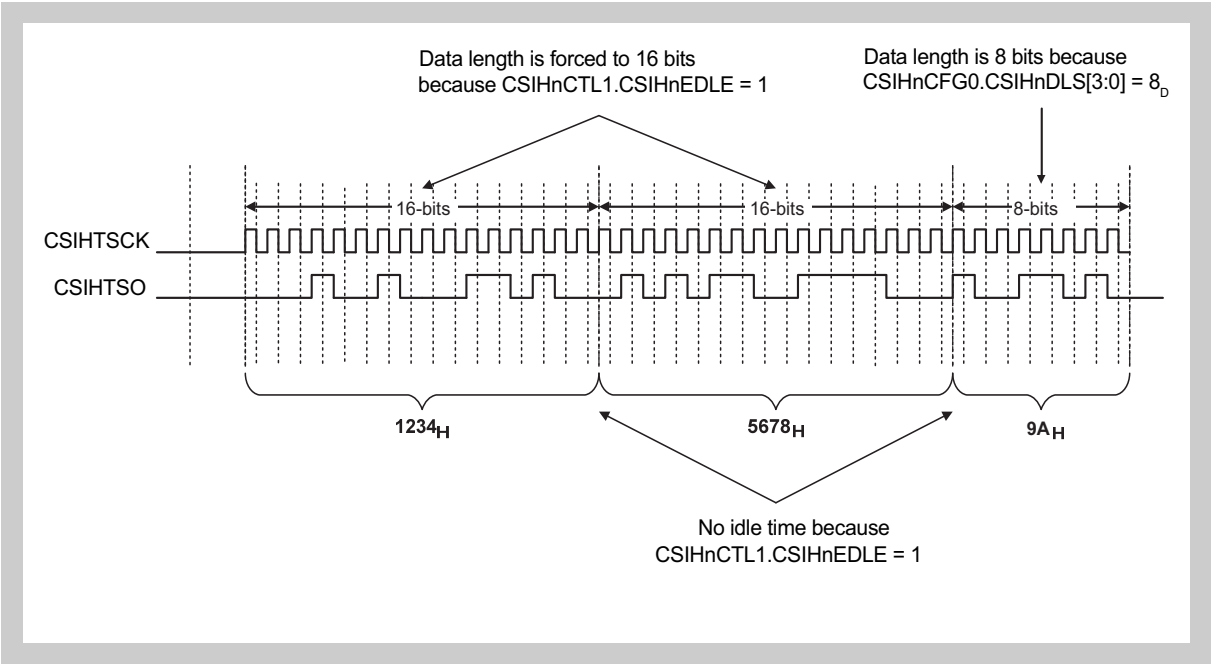


Figure 28-14 EDL timing diagram

Notes

1. Data lengths settings lower than 7 are only permitted in combination with EDL mode.
2. It is not possible to send two consecutive data with a data length of less than 7 bits.
3. If parity is enabled, the parity bit is added after the last bit.
4. To consider the data direction, pay attention to the following example:

   Data to be sent:123456_H

   MSB first:
   Set CSIHnCFGx.CSIHnDIR = 0
   Write CSIHnTX0W = 2000 1234_H (EDL bit = 1)
   Write CSIHnTX0W = 0000 0056_H (EDL bit = 0)

   LSB first:
   Set CSIHnCFGx.CSIHnDIR = 1
   Write CSIHnTX0W = 2000 3456_H (EDL bit = 1)
   Write CSIHnTX0W = 0000 0012_H (EDL bit = 0)

28.3.10 Serial data direction selection

The serial data direction is selectable for each chip select signal using the CSIHnDIRx bit in the CSIHnCFGx register.

The examples below show the communication for a data length of 8 bit (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B):

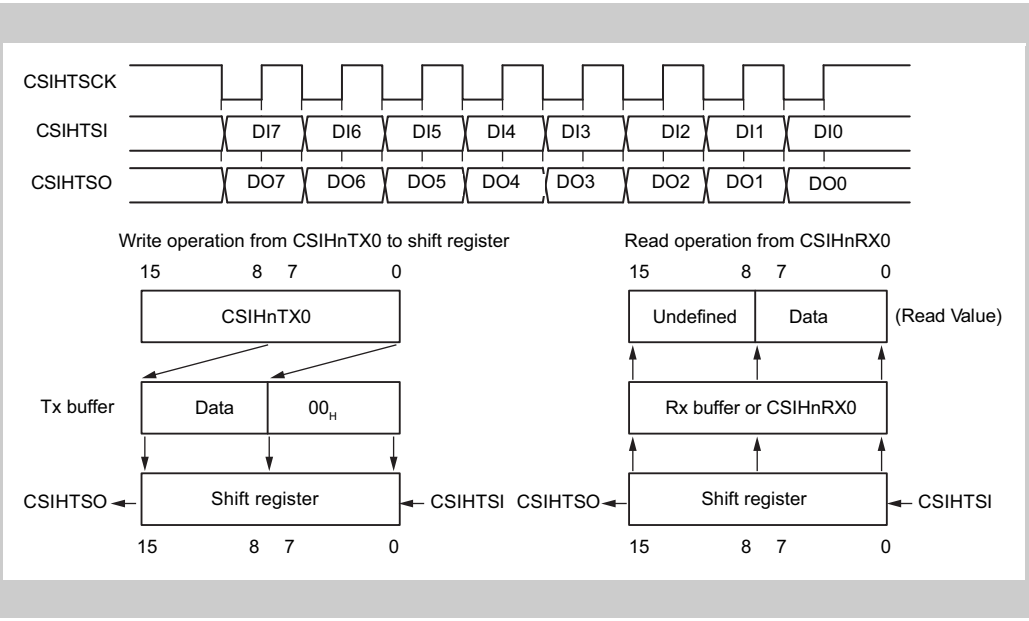


Figure 28-15 Serial data direction select function - MSB first (CSIHnDIR = 0)

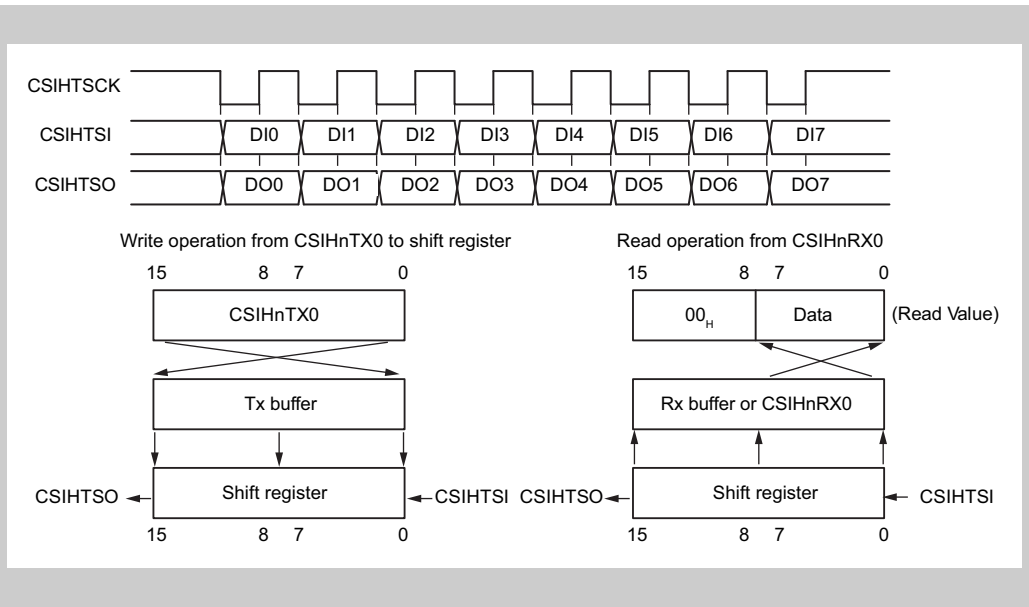


Figure 28-16 Serial data direction select function - LSB first (CSIHnDIR = 1)

28.3.11 Communication in slave mode

The following figure illustrates the communication signals and timings in slave mode.

In slave mode, the data transfer configuration is determined by the CSIHnCFG0 register.

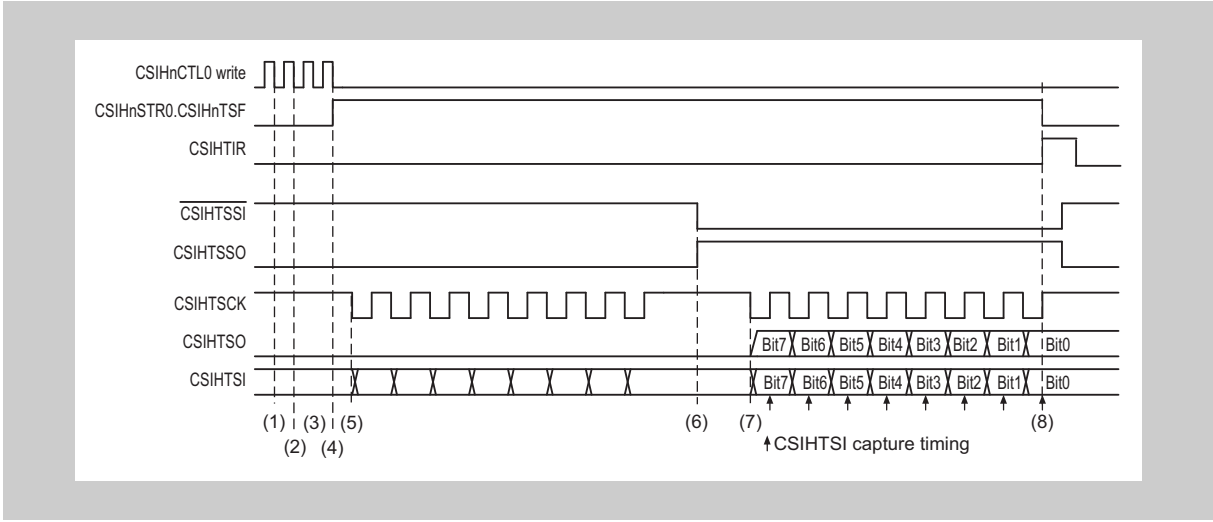


Figure 28-17 Transmit / receive communication timing in slave mode

1. CSIH is put into slave mode by setting CSIHnCTL2.CSIHnPRS[2:0] = 111_B. CSIHnCFGx.CSIHnCKPx and CSIHnCFGx.CSIHnDAPx are 0.
2. Data length is 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B). Data direction is MSB first (CSIHnCFGx.CSIHnDIRx = 0).
3. CSIH is set to transmit/receive operation (CSIHnCTL0.CSIHnTXE = 1, CSIHnCTL0.CSIHnRXE = 1). Communication start is enabled.
4. The “transmission in progress” flag CSIHnSTR0.CSIHnTSF is automatically set when transfer data is written to the transmission register CSIHnTX0W or CSIHnTX0H.
5. As long as signal CSIHTSSI is high, transmission/reception is not started, even if an external transmission clock CSIHTSCK is applied. Input at CSIHTSI is ignored.
6. As soon as CSIHTSSI falls to low level, signal CSIHTSSO goes high, indicating that CSIHTSO is enabled and ready for transmission.
7. Now, as soon as the external clock signal CSIHTSCK appears, the slave transmits data to CSIHTSO and simultaneously captures data from CSIHTSI.
8. Interrupt CSIHTIR indicates when the reception is complete. The register CSIHnRX0W/H can be read.

28.3.12 CSIH interrupt requests

CSIH can generate the following interrupt requests:

- CSIHTIC (communication interrupt)
- CSIHTIR (communication interrupt)
- CSIHTIRE (error interrupt)
- CSIHTIJC (job completion interrupt)

(1) Overview

The error interrupt CSIHTIRE is generated when an error is detected. The generation of the other interrupts depends on the memory mode, the job mode, and – in case of the job completion interrupt CSIHTIJC –also the operating mode.

The job completion interrupt CSIHTIJC is only generated when job mode is enabled (CSIHnCTL1.CSIHnJE = 1). It is not available in slave mode.

The following table gives an overview.

Table 28-10 Interrupt generation

Memory mode	Interrupt	Master and slave	
		Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO	CSIHTIC	Tx data empty[a]	Tx data empty[a]
	CSIHTIR	Rx data full[b] *and* CSIHnCTL0.CSIHnRXE = 1	Rx data full[b] *and* CSIHnCTL0.CSIHnRXE = 1
	CSIHTIRE	Error detected	Error detected
	CSIHTIJC[c]	not applicable	CSIHnTX0W.CSIHnCIRE = 1, or job abortion[d]
Transmit-only buffer, dual buffer	CSIHTIC	End of communication	End of communication, or CSIHnTX0W.CSIHnCIRE = 1
	CSIHTIR	Data received *and* CSIHnCTL0.CSIHnRXE = 1	Data received *and* CSIHnCTL0.CSIHnRXE = 1
	CSIHTIRE	Error detected	Error detected
	CSIHTIJC[a]	not applicable	Job abortion[d]
Direct access	CSIHTIC	Every data transferred	Every data transferred, if not aborted by job abortion[d]
	CSIHTIR	Data received *and* CSIHnCTL0.CSIHnRXE = 1	Data received *and* CSIHnCTL0.CSIHnRXE = 1
	CSIHTIRE	Error detected	Error detected
	CSIHTIJC[a]	not applicable	Job abortion[d]

a) "Tx data empty" refers to the FIFO fill level, defined by CSIHnMCTL1.CSIHnFES[6:0].
b) "Rx data full" refers to the FIFO fill level, defined by CSIHnMCTL1.CSIHnFFS[6:0].
c) CSIHTIJC is not available in slave mode.
d) Job abortion condition: CSIHnTX0W.CSIHnEOJ = 1 *and* CSIHnCTL0.CSIHnJOBE = 1

(2) General interrupt delay

In master mode, all interrupts generated by the master can be delayed by one half period of the transmission clock CSIHTSCK. This is not possible in slave mode.

The delay is specified by setting bit CSIHnCTL1.CSIHnSIT = 1.

The following example illustrates the interrupt delay function, assuming a setting of CSIHnCTL1.CSIHnSIT = 1 (interrupt delay enabled), CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0 (normal clock and data phase), and CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B (data length 8 bits).

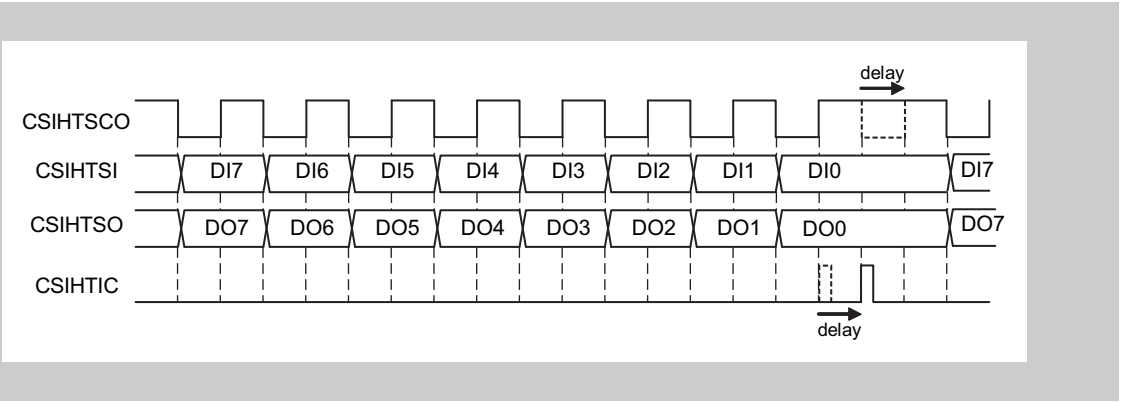


Figure 28-18 Interrupt delay function (CSIHnCTL1.CSIHnSIT = 1)

Setting CSIHnCTL1.CSIHnSIT = 1 adds half period delay to the transmission clock. This delays also the end of the present chip select signal.

(3) CSIHTIC communication interrupt

Depending on the memory mode and the job mode, this interrupt is generated according to the following conditions:

Table 28-11 CSIHTIC interrupt generation

Memory mode	Master and slave	
	Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO	This interrupt occurs when transmission data is about to be missing in the FIFO, indicating to the application that new data should be added. CSIHTIC is generated, if the number of data to be sent remaining in the FIFO CSIHnSTR0.CSIHnSPF[7:0] equals CSIHnMCTL1.CSIHnFES[6:0].	
Transmit-only buffer, dual buffer	End of communication (as specified in register CSIHnMCTL2).	End of communication, (as specified in register CSIHnMCTL2) or data with CSIHnTX0W.CSIHnCIRE = 1 was sent. However, if data with CSIHnTX0W.CSIHnCIRE = 1 and job abortion[a] is sent, interrupt CSIHTIJC is generated instead.
Direct access	Happens after every data transfer.	Happens after every data transfer, except the communication was aborted[a].

a) Job abortion condition: CSIHnTX0W.CSIHnEOJ = 1 and CSIHnCTL0.CSIHnJOBE = 1.

CSIHTIC in direct access mode

The following example shows the CSIHTIC behavior in direct access mode.

The following example assumes:

- Master mode
- Direct access memory mode
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Normal CSIHTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

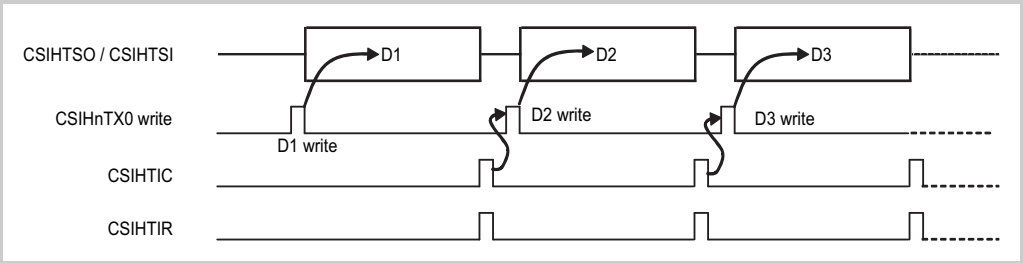


Figure 28-19 Generation of CSIHTIC after transfer (CSIHnCTL1.CSIHnSLIT = 0)

If job mode is enabled (CSIHnCTL1.CSIHnJE = 1) and a job ends because data is sent with CSIHnTX0W.CSIHnEOJ = 1 and communication stop is requested (CSIHnCTL0.CSIHnJOBE = 1), then CSIHTIC is replaced by the job completion interrupt CSIHTIJC.

CSIHTIC can also be set up to occur as soon as the CSIHnTX0 register is free for the next data. This is specified by setting CSIHnCTL1.CSIHnSLIT = 1.

Note This mode allows faster data transfer but is only available in direct access memory mode.

The effect is illustrated in the figure below.

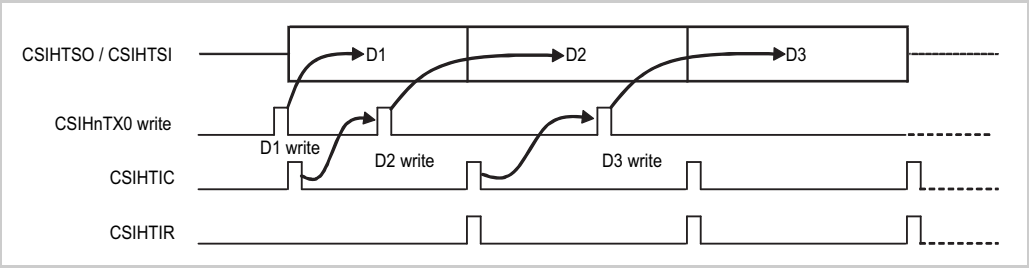


Figure 28-20 Immediate generation of CSIHTIC (CSIHnCTL1.CSIHnSLIT = 1)

Thus, the new data can be written in advance.

CSIHTIC in FIFO mode

The following example shows the CSIHTIC behavior in FIFO mode.

The following example assumes:

- Master mode
- FIFO memory mode
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)

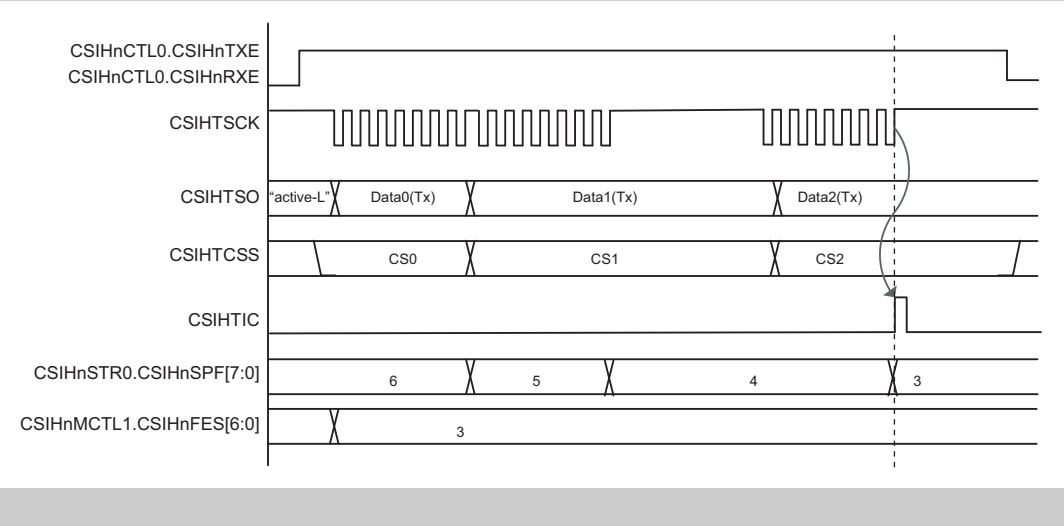


Figure 28-21 Generation of CSIHTIC in FIFO memory mode

The condition for “FIFO empty” is specified in CSIHnMCTL1.CSIHnFES[6:0]: In the example of the diagram above, minimum stock is 3 data. CSIHnSTR0.CSIHnSPF[7:0] indicates the number of unsent data. When both match, the interrupt CSIHTIC occurs.

CSIHTIC in job mode

The following example shows the CSIHTIC behavior in job mode.

The following example assumes:

- Master mode
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- Normal CSIHTIC interrupt timing (CSIHnCTL1.CSIHnSLIT = 0)

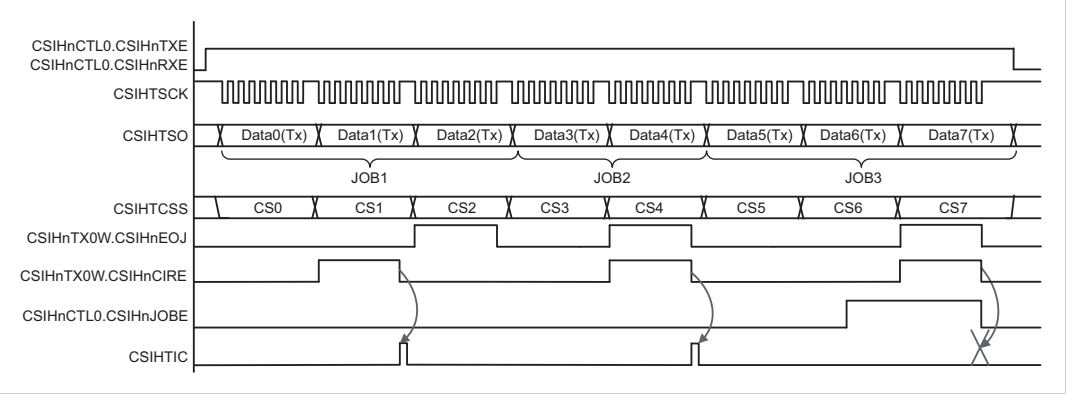


Figure 28-22 Generation of CSIHTIC in job mode

The rules for generating CSIHTIC in job mode are:

Table 28-12 Generation of CSIHTIC in job mode

CSIHnTX0W. CSIHnEOJ	CSIHnTX0W. CSIHnCIRE	CSIHTIC
0	0	not generated
0	1	generated
1	0	not generated
1	1	CSIHnCTL0.CSIHnJOBE = 0: generated
		CSIHnCTL0.CSIHnJOBE = 1: not generated, replaced by interrupt CSIHTIJC

(4) CSIHTIR reception interrupt

Depending on the memory mode and the job mode, this interrupt is generated according to the following conditions:

Table 28-13 CSIHTIR interrupt generation

Memory mode	Master and slave	
	Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO	This interrupt occurs when the FIFO buffer is almost full with received data, indicating to the application that the FIFO must be emptied. CSIHTIR is generated, if the number of received data in the FIFO CSIHnSTR0.CSIHnSRP[7:0] equals CSIHnMCTL1.CSIHnFFS[6:0].	
Transmit-only buffer	Occurs after every received data when CSIHnCTL0.CSIHnRXE = 1.	Occurs after every received data when CSIHnCTL0.CSIHnRXE = 1
Dual buffer	Occurs when the communication has finished (as specified by register CSIHnMCTL2) and CSIHnCTL0.CSIHnRXE = 1.	
Direct access	Happens after every data transfer.	

In transmit-only or dual buffer mode, this interrupt is generated in receive-only and transmit/receive mode after each data has been received.

CSIHTIR in direct access mode

The following example shows the CSIHTIR behavior in direct access mode.

The following example assumes:

- Master mode
- Direct access mode
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)

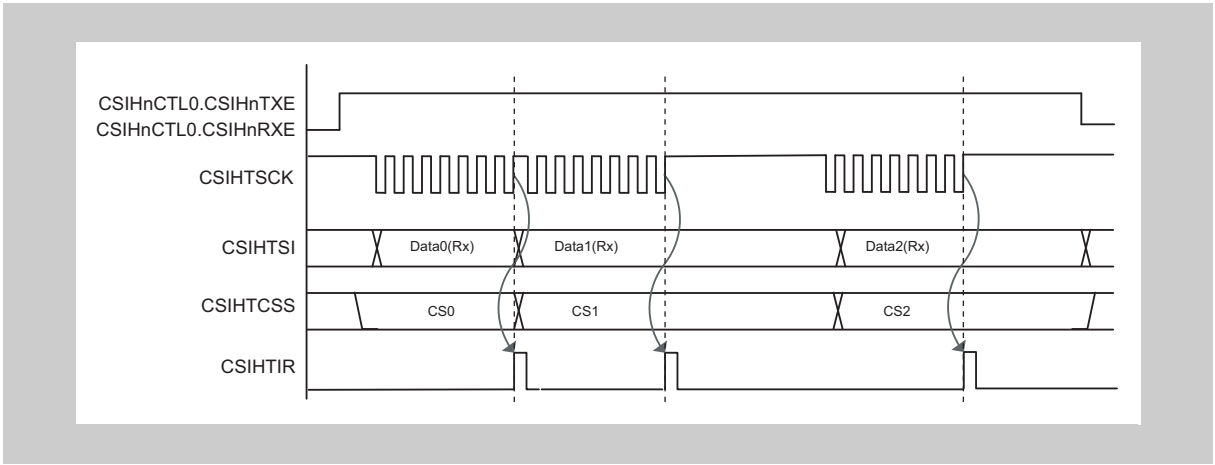


Figure 28-23 Generation of CSIHTIR in direct access memory mode

CSIHTIR in buffer mode

The following example shows the CSIHTIR behavior in buffer mode.

The following example assumes:

- Master mode
- Transmit-only or dual buffer mode
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Default clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- Data length 8 bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)

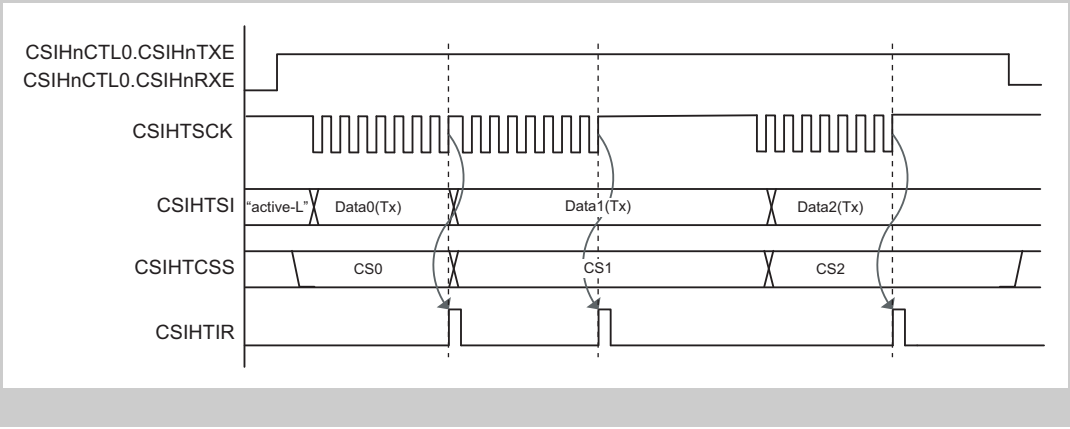


Figure 28-24 Generation of CSIHTIR in transmit-only or dual buffer memory mode

(5) CSIHTIRE reception error interrupt

This interrupt is generated whenever an error is detected.

Table 28-14 Data error types

Error type	Communication status after error interrupt	Comment
FIFO overflow error	Interrupt is generated and communication continues	The data written to the FIFO is lost, but previously started communications are continued.
Parity error	Interrupt is generated and communication continues	-
Data consistency error	Interrupt is generated and communication continues	-
Time-out error	Interrupt is generated and communication continues	-
Overrun error	Interrupt is generated and communication continues	This error is generated when the number of received data is 0 and the CPU attempts to read reception data.
	Interrupt is generated and communication is stopped	This error is generated when data was received, but FIFO is full. In Tx-only buffer mode or direct access mode: Reception data is left in the CSIHnRX0 register.

The type of error that caused the generation of CSIHTIRE is flagged in register CSIHnSTR0.

Additionally a parity and data consistency error flag is attached to the reception data in CSIHnRX0W.

For details about the various error types refer to *28.3.14 “Error detection” on page 2179* .

(6) CSIHTIJC job completion interrupt

This interrupt supports the handling of jobs – refer to *28.3.4 “The job concept” on page 2157* . This interrupt is only available in master mode.

Job mode is enabled by setting CSIHnCTL1.CSIHnJE = 1. When CSIHnCTL1.CSIHnJE = 0, CSIHTIJC is not generated.

Depending on the memory mode, this interrupt is generated according to the following conditions:

Table 28-15 CSIHTIJC interrupt generation

Memory mode	Job mode disabled CSIHnCTL1.CSIHnJE = 0	Job mode enabled CSIHnCTL1.CSIHnJE = 1
FIFO	Not applicable	Indicates that the communication has stopped at the end of a job after a job abortion[a] was triggered or data with CSIHnTX0W.CSIHnCIRE = 1 was sent
Transmit-only buffer		Indicates that the communication has stopped at the end of a job after a job abortion[a] was triggered
Dual buffer		
Direct access		

a) Job abortion condition: CSIHnTX0W.CSIHnEOJ = 1 and CSIHnCTL0.CSIHnJOBE = 1

28.3.13 Handshake function

CSIH features a handshake function to synchronize the master and the slave devices. This function can be enabled/disabled by bit CSIHnCTL1.CSIHnHSE. For handshake, the signal CSIHTSHSG is used.

The timing depends on the data phase selection bit CSIHnCFGx.CSIHnDAPx.

(1) Slave mode

When CSIHnCTL1.CSIHnHSE = 1, the slave outputs CSIHTSHSG = 0 when it is busy. This can happen in two cases:

1. Memory mode is FIFO mode.
   The slave is in transmit-only or transmit/receive mode but has no transmission data in its buffer. This status is indicated by the flag CSIHnSTR0.CSIHnEMF.

   The following examples assume a data length of 8 bits.

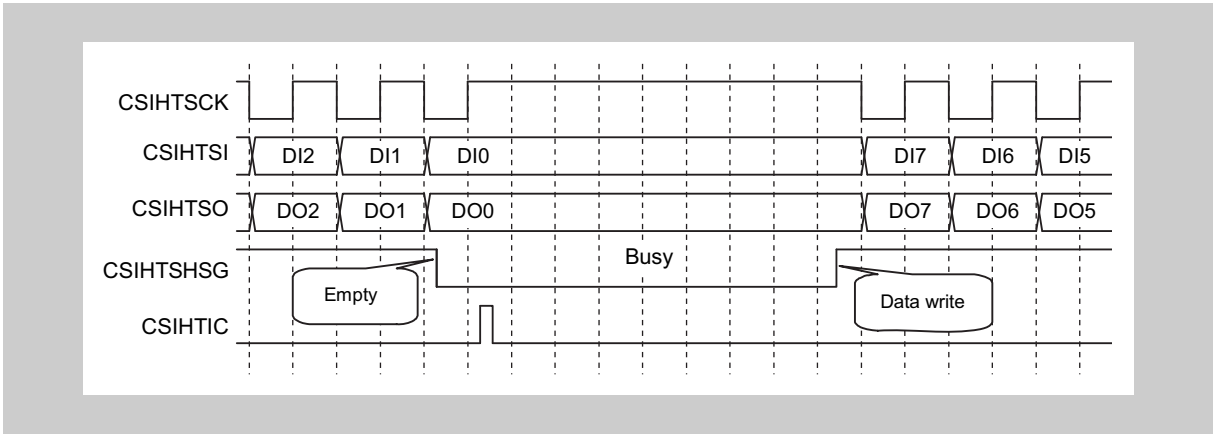


Figure 28-25 Busy signal from slave (FIFO mode; CSIHnCFGx.CSIHnDAPx = 0)

The slave sets CSIHTSHSG to high (“ready”) as soon as new transmission data is written to the FIFO.

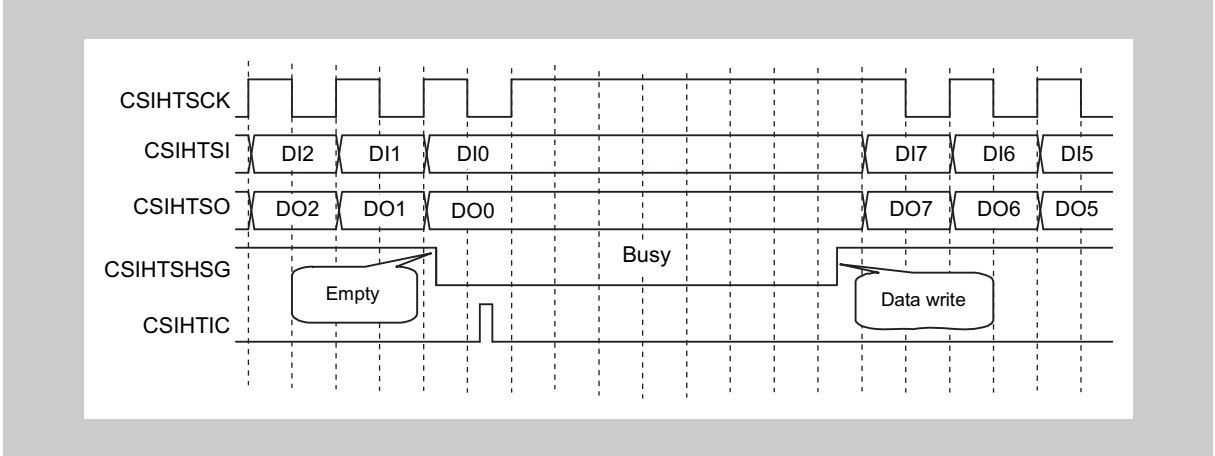


Figure 28-26 Busy signal from slave (FIFO mode; CSIHnCFGx.CSIHnDAPx = 1)

2. Memory mode is direct access mode.
   The slave is in receive-only or transmit/receive mode but previously received data is still in the CSIHnRX0 register, and new data cannot be copied from the shift register to CSIHnRX0 (CSIHnRX0 full condition).

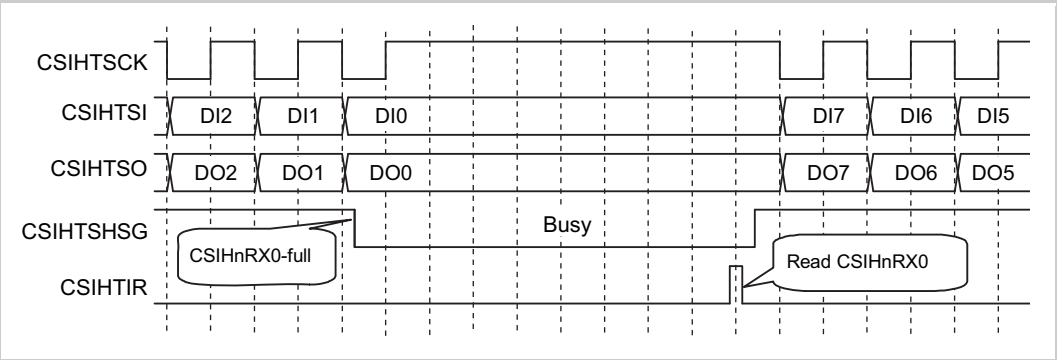


Figure 28-27 Busy signal from slave (Direct access; CSIHnCFGx.CSIHnDAPx = 0)

The slave sets CSIHTSHSG to high (“ready”) as soon as the reception register CSIHnRX0 has been read.

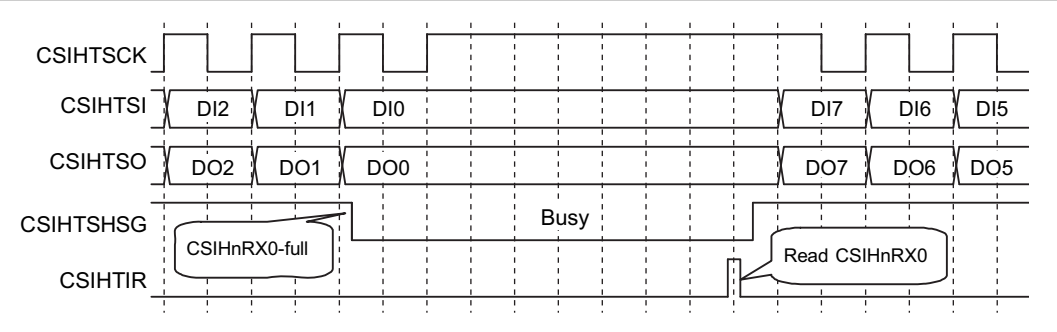


Figure 28-28 Busy signal from slave (CSIHnCFGx.CSIHnDAPx = 1)

(2) Master mode

When the master detects CSIHTSHSG = 0, the following transfer is put on hold, and the master goes into wait status. It suspends the clock CSIHTSCK.

The CSIHTSHSG level is checked at each half clock cycle of CSIHTSCK.

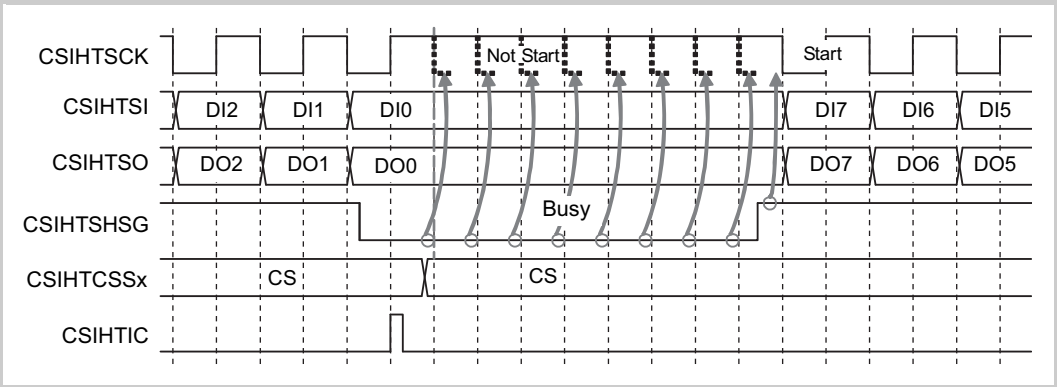


Figure 28-29 Master’s reaction on CSIHTSHSG (CSIHnCFGx.CSIHnDAPx = 0)

CSIHTSHSG must be pulled down by the slave before the next transfer starts. If the CSIHTSHSG signal is pulled down while a data transfer is in progress, the serial clock is suspended after the transfer is complete.

The master resumes the communication as soon as CSIHTSHSG becomes high (slave is “ready”).

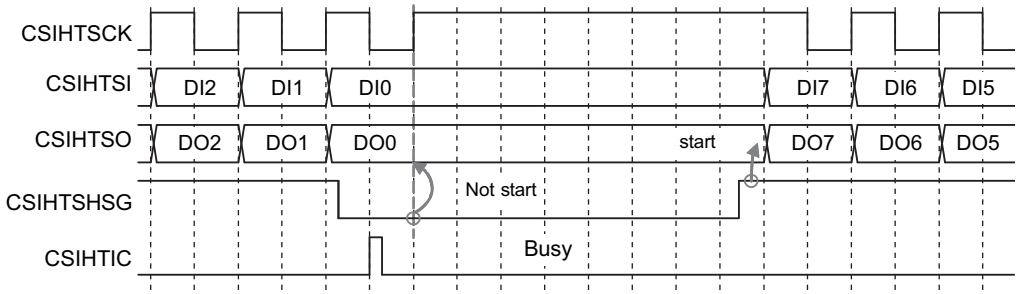


Figure 28-30 Master’s reaction on CSIHTSHSG (CSIHnCFGx.CSIHnDAPx = 1)

Caution If multiple slaves are connected, the master must only detect the CSIHTSHSG signal of the slave it has selected for communication.

28.3.14 Error detection

The CSIH can detect five error types:

- Data consistency error (transmission data)
- Parity error (received data)
- Overrun error (received data)
- Time-out error (in FIFO mode)
- Overflow error (in FIFO mode)

Check for parity, data consistency and time-out erros can be enabled/disabled individually.

If one of these errors is detected, the interrupt request CSIHTIRE is generated and the corresponding flags are set.

(1) Data consistency check

The purpose of the data consistency check is to ensure that the data physically sent as output pin is identical to the original data that was copied to the shift register.

The data consistency check can be enabled/disabled by bit CSIHnCTL1.CSIHnDCS. It is not active if data transmission is disabled (CSIHnCTL0.CSIHnTXE = 0).

When the data consistency check is active, the data transferred from CSIHnTX0 to the shift register is copied to a separate register. In addition, the physical levels at CSIHTSO are read back via the CSIHTDCS signal into an own shift register.

After completion of the transmission, the sent data is compared with the original transmission data.

Mismatch is considered as a data consistency error:

- Interrupt CSIHTIRE is generated.
- Bit CSIHnSTR0.CSIHnDCE is set.

Additionally, CISHnRX0W.CSIHnTDCE is set with the corresponding data.

The function is illustrated in the following block diagram.

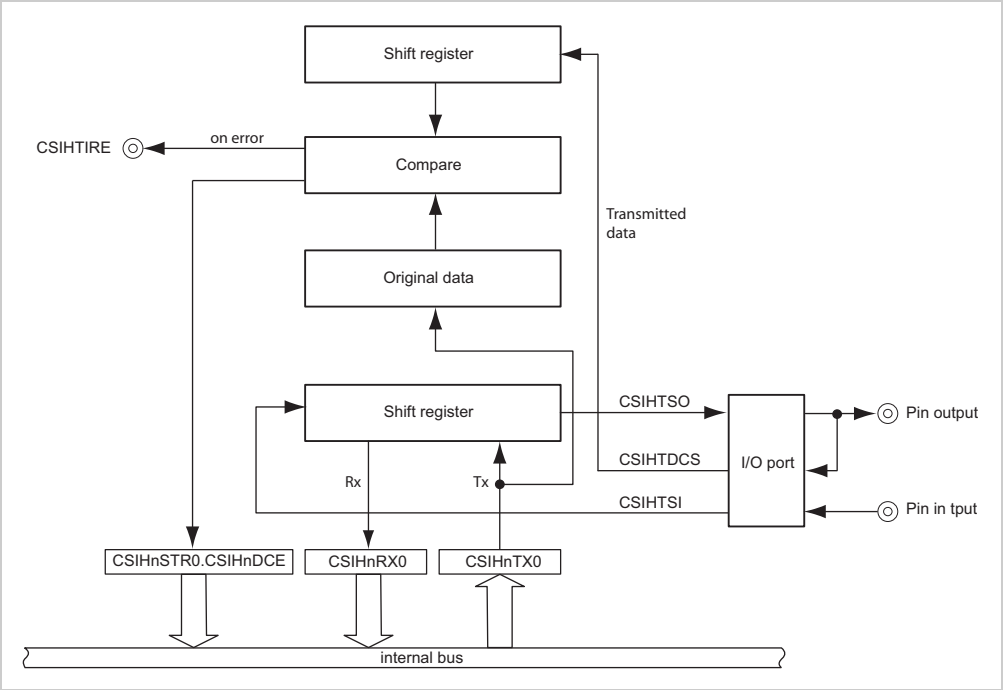


Figure 28-31 Data consistency check functional block diagram

(2) Parity check

CSIH can append a parity bit to the last data bit (even if extended data length is used).

The use and type of parity is specified in CSIHnCFGx.CSIHnPSx[1:0].

Parity check is enabled if CSIHnCFGx.CSIHnPSx[1] = 1.

The parity bit is checked after a reception is complete. In case of parity error:

- Interrupt CSIHTIRE is generated.
- Bit CSIHnSTR0.CSIHnPE is set.

Additionally, CISHnRX0W.CSIHnRPE is set with the corresponding data.

The following figure shows an example.

- Data length is 8 bits.
- The data transmitted is 05_H and 35_H.
- Data direction is LSB first.
- Parity type is odd.

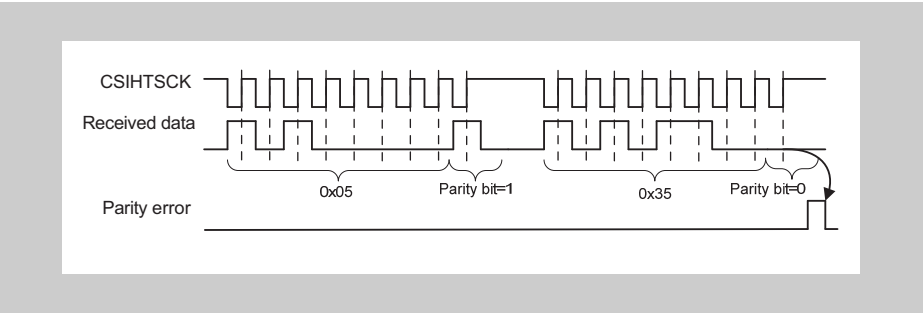


Figure 28-32 Parity check example

For the first 8 bits, the parity bit is 1. There is no parity error, because the total number of ones (including the parity bit) is odd.

For the second 8 bits, the parity bit is 0. This is detected as a parity error, because the total number of ones (including the parity bit) is even.

(3) Time-out error

Time-out erros checks are only possible in slave FIFO mode.

This error occurs when received data in the FIFO is not read or no data from CSIHTSI is received in the FIFO within a certain time.

The time is defined in CSIHnMCTL0.CSIHnTO[4:0] in multiples of 8 times the transmission clock CSIHSCK. Time-out error occurs when the specified time is exceeded.

A dedicated time-out counter measures the time between the last and the next read operation.

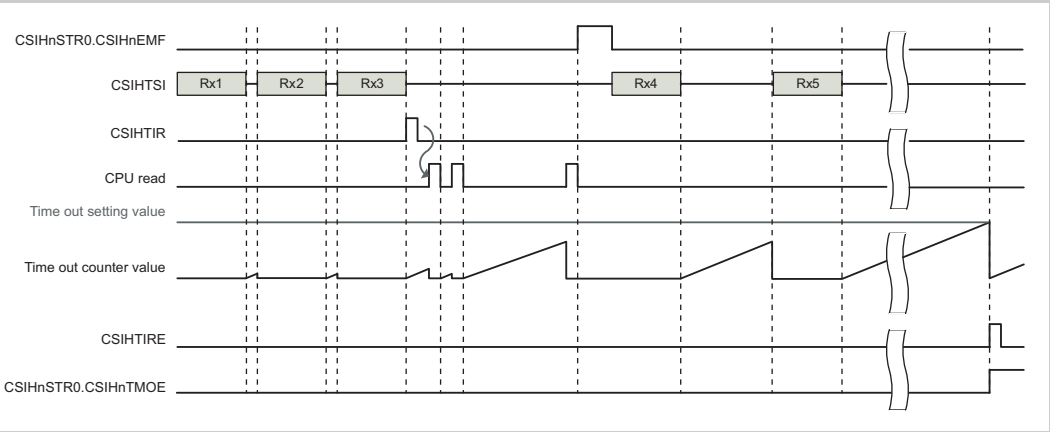


Figure 28-33 Time-out check functional timing diagram

The counter (re-)starts after every data reception and after every read operation, if data is still available in the FIFO. It does not start when the buffer is marked as empty (CSIHnSTR0.CSIHnEMF = 1).

The counter is reset after each read or new data has arrived.

In case of time-out error:

- Interrupt CSIHTIRE is generated.
- Bit CSIHnSTR0.CSIHnTMOE is set.

(4) Overflow error

Overflow error can happen in FIFO memory mode. It occurs when transmission data is written to the CSIHnTX0W register while the FIFO buffer is filled with reception data.

Example 100 data have been transmitted. That means, the FIFO contains 100 received data. The application starts to read the received data.

While the read operation is in progress, the application begins to write another set of 50 transmission data to the FIFO. However, only 10 received data have been read up to now, 90 are still in the FIFO.

In this case, only 38 cells are available for new transmission data. When the CPU tries to write the 39th data, an overflow error happens.

This is illustrated in the following figure:

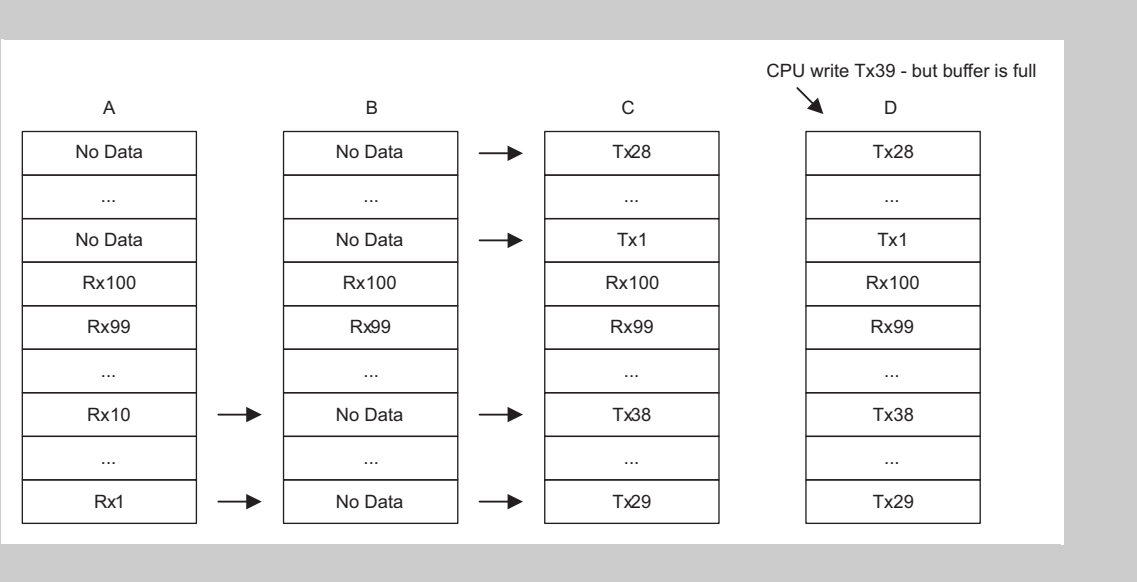


Figure 28-34 FIFO overflow

The data after 39 are discarded. The following figure shows the associated timing.

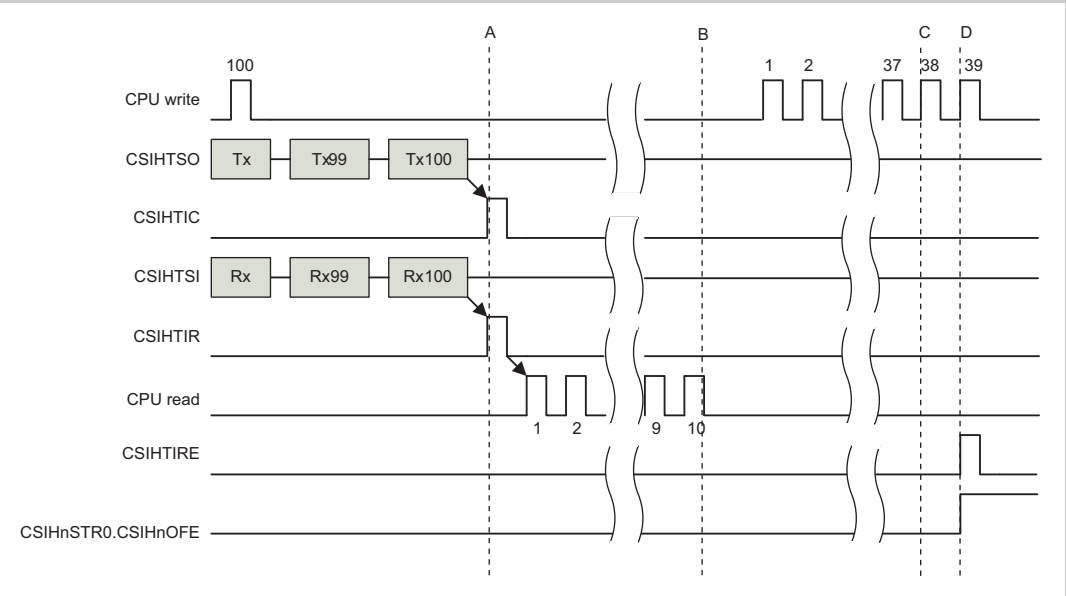


Figure 28-35 FIFO overflow timing

In case of overflow error:

- Interrupt CSIHTIRE is generated.
- Bit CSIHnSTR0.CSIHnOFE is set.

(5) Overrun error

An overrun error can happen in direct access, transmit-only buffer, and FIFO mode. It cannot happen in dual buffer mode. The overrun error is not generated if data reception is disabled (CSIHnCTL0.CSIHnRXE = 0).

Direct access / transmit-only buffer

In direct access and transmit-only buffer mode, this error occurs when newly received data cannot be transferred from the shift register to the reception register CSIHnRX0. This happens when CSIHnRX0 was not read and therefore contains previous reception data.

The following figure illustrates the function.

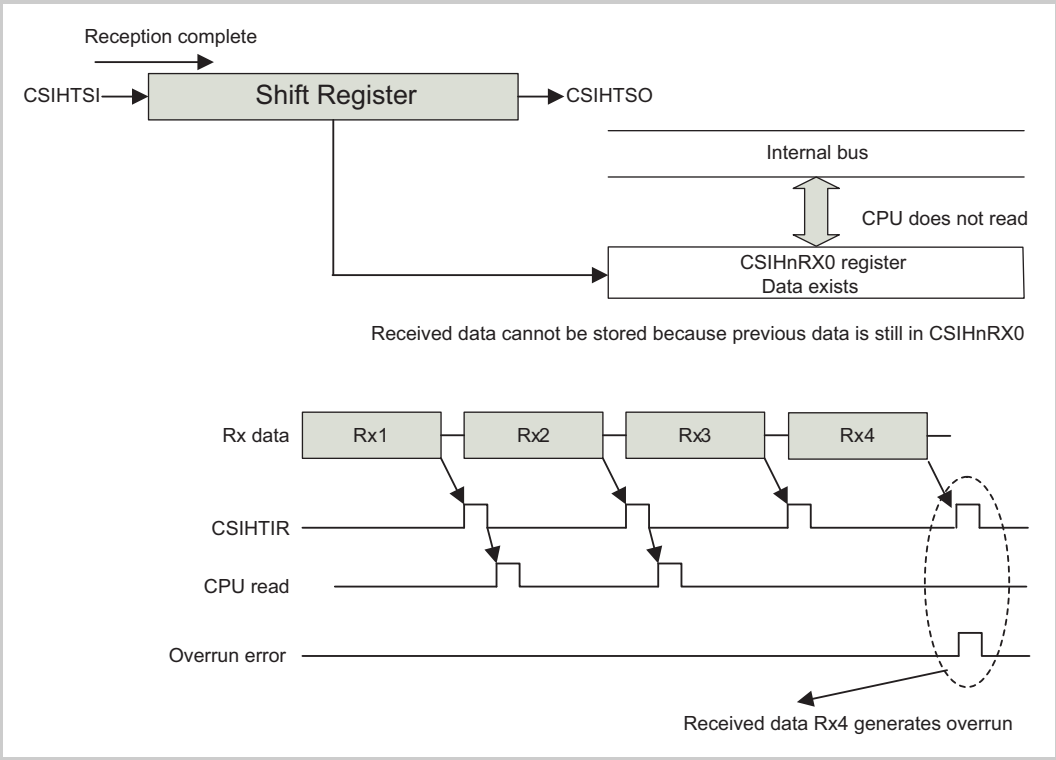


Figure 28-36 Overrun error detection in direct access and transmit-only buffer mode

FIFO mode In FIFO mode, this error occurs if:

1. Newly received data cannot be transferred from the shift register to the FIFO because the FIFO is full

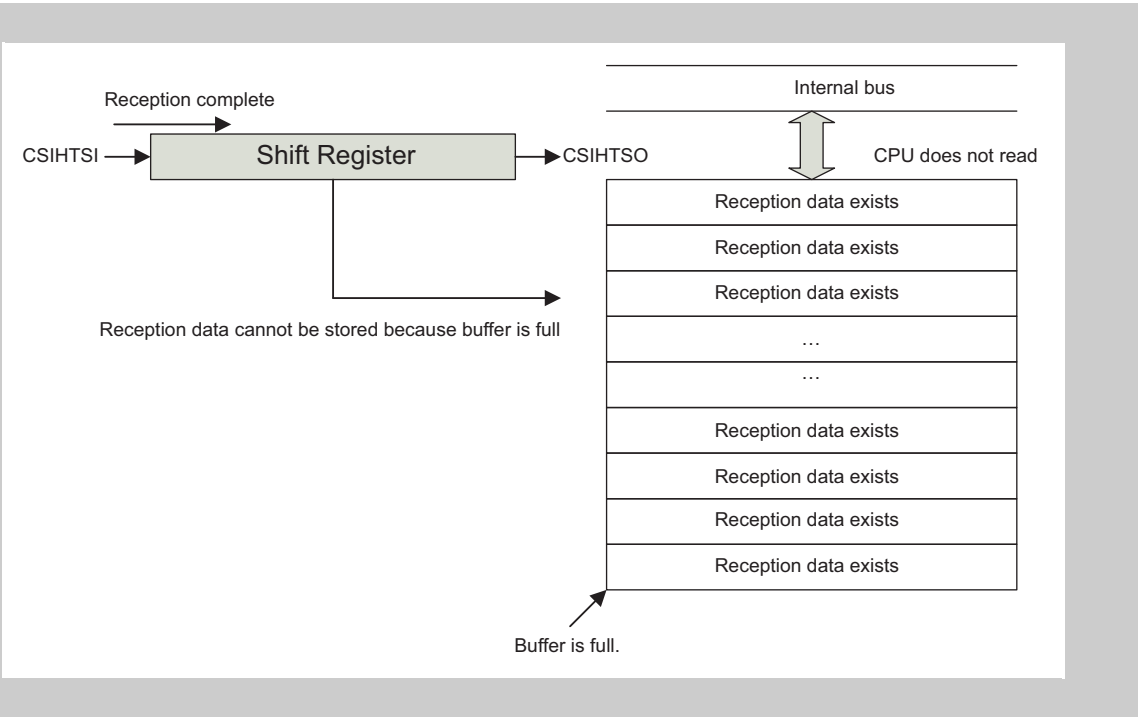


Figure 28-37 Overrun error detection in FIFO mode (FIFO full)

2. The CPU attempts to read non existing reception data

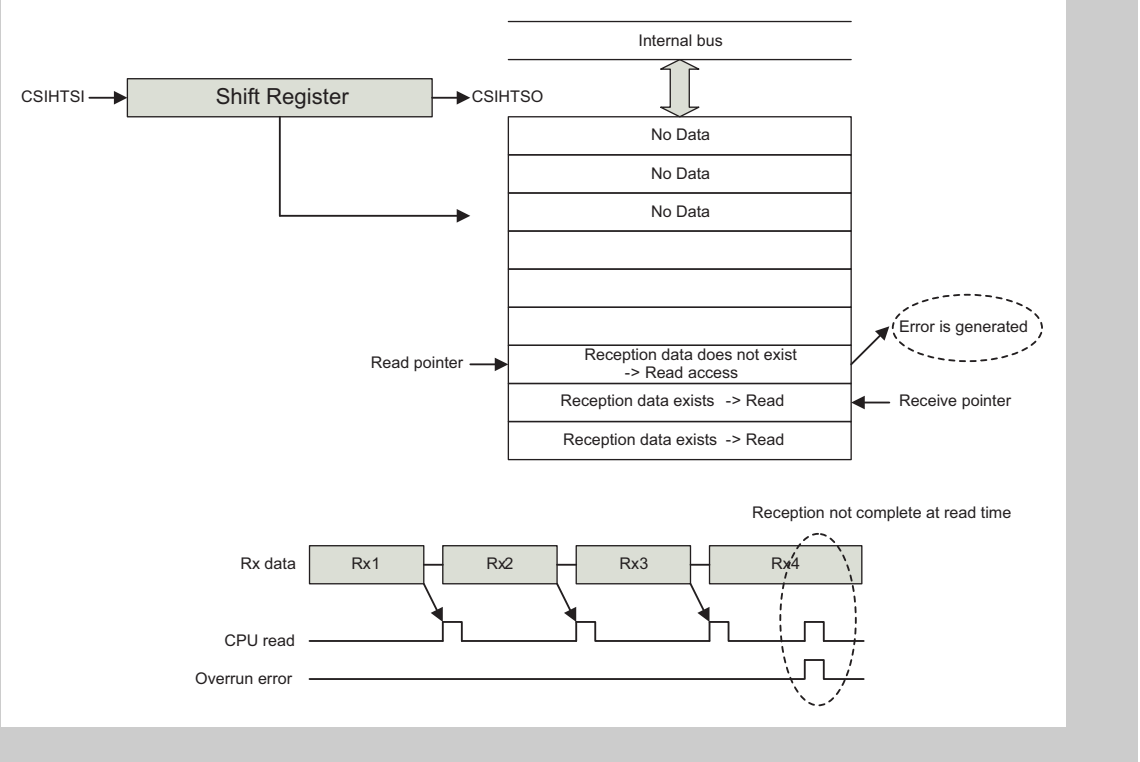


Figure 28-38 Overrun error detection in FIFO mode (no data)

In case of overrun error:

- Interrupt CSIHTIRE is generated.
- Bit CSIHnSTR0.CSIHnOVE is set.
- Communication is stopped (except if the CPU tried to read non existing data).

Note An overrun error can be avoided in slave mode by using the handshake. When handshake is used in slave mode, the receiver (slave) signals to the transmitter (master) that it is busy. The transmitter then waits until the receiver has read its reception register and is ready again.The overrun error can be avoided when the handshake function is used.

When handshake is used, the receiver (slave) signals to the transmitter (master) that it is busy. The transmitter then waits until the receiver has read its reception register and is ready again.

For details see *28.3.13 “Handshake function” on page 2176* .

28.3.15 Loop-back mode

Loop-back mode is a special mode for self-test. This feature is only available in master mode.

When this mode is active, the transmit and receive signals are internally connected, as shown in the figures below. The signals CSIHTSCK, CSIHTSO, and CSIHTSI are disconnected from the ports. In addition, the CSIHTSO output level is fixed to low, and CSIHTSCK is set to the inactive level, as defined by CSIHnCFGx.CSIHnCKPx. The rest of CSIH works as in normal operation.

In order to test the CSIH macro, put the macro in loop-back mode and carry out normal transfer operations. Then check that the received data is the same as the transmitted data.

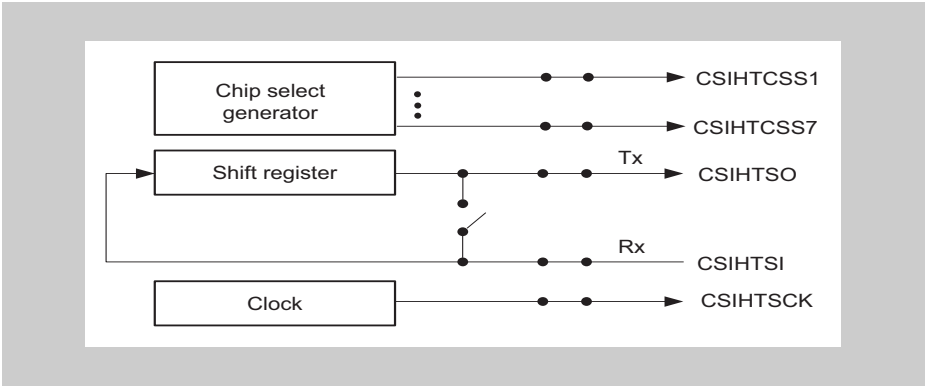


Figure 28-39 Normal operation

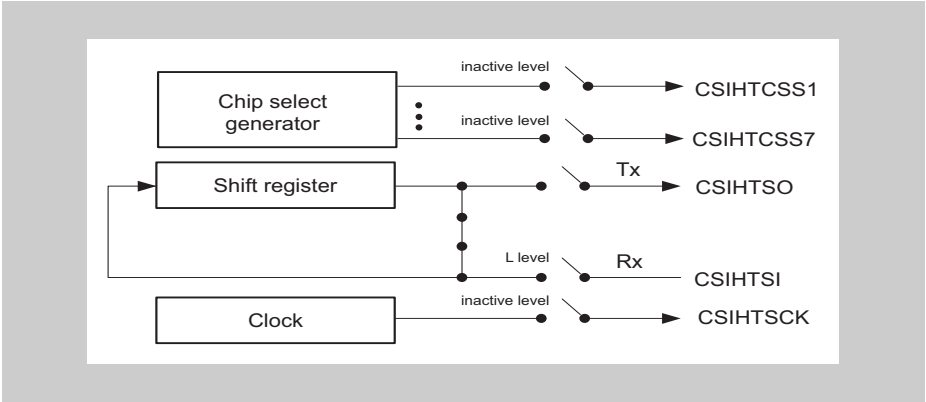


Figure 28-40 Loop-back operation

28.4 CSIH Control Registers

The CSIHn is controlled and operated by means of the following registers:

Table 28-16 CSIH register overview

Register name	Shortcut	Address
Control register 0	CSIHnCTL0	<CSIHn_base> + 0000_H
Control register 1	CSIHnCTL1	<CSIHn_base> + 0010_H
Control register 2	CSIHnCTL2	<CSIHn_base> + 0014_H
Status register 0	CSIHnSTR0	<CSIHn_base> + 0004_H
Status clear register 0	CSIHnSTCR0	<CSIHn_base> + 0008_H
Memory control register 0	CSIHnMCTL0	<CSIHn_base> + 1040_H
Memory control register 1	CSIHnMCTL1	<CSIHn_base> + 1000_H
Memory control register 2	CSIHnMCTL2	<CSIHn_base> + 1004_H
Transmit data register 0 for word access	CSIHnTX0W	<CSIHn_base> + 1008_H
Transmit data register 0 for half word access	CSIHnTX0H	<CSIHn_base> + $100C_H$
Receive data register 0 for word access	CSIHnRX0W	<CSIHn_base> + 1010_H
Receive data register 0 for half word access	CSIHnRX0H	<CSIHn_base> + 1014_H
Memory read/write pointer register 0	CSIHnMRWP0	<CSIHn_base> + 1018_H
Configuration register 0	CSIHnCFG0	<CSIHn_base> + 1044_H
Configuration register 1	CSIHnCFG1	<CSIHn_base> + 1048_H
Configuration register 2	CSIHnCFG2	<CSIHn_base> + $104C_H$
Configuration register 3	CSIHnCFG3	<CSIHn_base> + 1050_H
Configuration register 4	CSIHnCFG4	<CSIHn_base> + 1054_H
Configuration register 5	CSIHnCFG5	<CSIHn_base> + 1058_H
Configuration register 6	CSIHnCFG6	<CSIHn_base> + $105C_H$
Configuration register 7	CSIHnCFG7	<CSIHn_base> + 1060_H
Emulation register	CSIHnEMU	<CSIHn_base> + 0018_H

<CSIHn_base> The base addresses <CSIHn_base> of the CSIHn is defined in the first section of this chapter under the key word “Register addresses”.

28.4.1 CSIH registers details

(1) CSIHnCTL0 - CSIH control register 0

This register controls the operation clock and enables/disables transmission/ reception and the memory part for transmission and/or reception. It forces the stop of communication at the end of the current job.

Access This register can be read/written in 8-bit units.

Address <CSIHn_base> + 00_H

Initial Value 00_H

7	6	5	4	3	2	1	0
CSIHn PWR	CSIHn TXE	CSIHn RXE	0	0	0	CSIHn JOBE	CSIHn MBS
R/W	R/W	R/W	R	R	R	R/W	R/W

Table 28-17 CSIHnCTL0 register contents

Bit position	Bit name	Function
7	CSIHnPWR	Controls the operation clock: 0: Stops operation clock 1: Provides operation clock Clearing CSIHnPWR to 0 resets the internal circuits, stops operation, and sets the CSIH to standby state. No clock is provided to internal circuits, thus the power consumption of the CSIHn is minimized. If CSIHnPWR is cleared during communication, ongoing communication is immediately aborted.
6	CSIHnTXE	Enables/disables transmission: 0: Transmission disabled 1: Transmission enabled
5	CSIHnRXE	Enables/disables reception: 0: Receive disabled 1: Receive enabled
1	CSIHnJOBE	Stops the communication at the end of the current job (data written to the transmit buffer with CSIHnTX0W.CSIHnEOJ = 1): 0: Communication stop is not required 1: Communication stop This bit can be used to abort an ongoing job. It is automatically cleared when the communication has stopped. In FIFO mode, the next communication should then be started after clearing the pointers by setting CSIHnSTCR0.CSIHnPCT = 1. **Note** CSIHnJOBE is only valid when CSIHnCTL1.CSIHnJE = 1. This bit has no meaning in slave mode.
0	CSIHnMBS	Bypasses the memory for transmission and/or reception data 0: Memory mode CSIH memory is used for transmission and/or reception data 1: Direct access mode CSIH memory is bypassed **Note** This bit must not be modified in slave mode.

Cautions 1. Do not modify any of these bits while CSIHnPWR = 0. These bits can be modified with the same write operation when setting CSIHnPWR = 1.

2. Do not modify CSIHnTXE or CSIHnRXE or CSIHnMBS while a data transmission is pending or going on, i.e. if CSIHnSTR0.CSIHnTSF = 1.

(2) CSIHnCTL1 - CSIH control register 1

This register specifies the interrupt timing and the interrupt delay mode. It enables/disables extended data length control, data consistency check, loop-back mode, handshake function, and job mode. It selects the active output level of each chip select and the behavior of the chip select signals after the transfer of the final data.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 10_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	CSIHnCKR	CSIHnSLIT
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnCSL7	CSIHnCSL6	CSIHnCSL5	CSIHnCSL4	CSIHnCSL3	CSIHnCSL2	CSIHnCSL1	CSIHnCSL0	CSIHnEDLE	CSIHnJE	CSIHnDCS	CSIHnCSRI	CSIHnLBM	CSIHnSIT	CSIHnHSE	CSIHnSSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 28-18 CSIHnCTL1 register contents (1/2)

Bit position	Bit name	Function
17	CSIHnCKR	Selects the initial level of the data clock CSIHTSCK 0: Initial level of CSIHTSCK is high 1: Initial level of CSIHTSCK is low
16	CSIHnSLIT	Selects the timing of interrupt CSIHTIC: 0: Normal interrupt timing (interrupt is generated after the transfer) 1: Interrupt is generated as soon as new data can be written to register CSIHnTX0 (works only in direct access memory mode). For details refer to *3 “CSIHTIC communication interrupt” on page 2169* .
15 to 8	CSIHnCSLx	Selects the active output level of chip select signal x (CSIHTCSSx). 0: Chip select is active low 1: Chip select is active high For details refer to *28.3.3 “Chip selection (CS) features” on page 2155* .
7	CSIHnEDLE	Enables/disables extended data length (EDL) mode: 0: Extended data length mode disabled 1: Extended data length mode enabled For details refer to *2 “Data length greater than 16 bits” on page 2164* .
6	CSIHnJE	Enables/disables job mode. 0: Job mode disabled 1: Job mode enabled For details refer to *28.3.4 “The job concept” on page 2157* . **Caution** In slave mode CSIHnJE must be set to 0.

Table 28-18 CSIHnCTL1 register contents (2/2)

Bit position	Bit name	Function
5	CSIHnDCS	Enables/disables data consistency check: 0: Data consistency check disabled 1: Data consistency check enabled For details refer to *1 "Data consistency check" on page 2179* .
4	CSIHnCSRI	Defines chip select behavior after last data transfer: 0: Chip select holds active level 1: Chip select returns to inactive level
3	CSIHnLBM	Controls loop-back mode (LBM): 0: Loop-back mode deactivated 1: Loop-back mode activated For details refer to *28.3.15 "Loop-back mode" on page 2188* .
2	CSIHnSIT	Selects interrupt delay mode: 0: No delay 1: Half clock delay for all interrupts This bit is only valid in master mode. In slave mode, no delay is generated. For details refer to *2 "General interrupt delay" on page 2169* .
1	CSIHnHSE	Enables/disables handshake mode: 0: Handshake function disabled 1: Handshake function enabled For details refer to *28.3.13 "Handshake function" on page 2176* .
0	CSIHnSSE	Enables/disables slave select function: 0: Input signal $\overline{\text{CSIHTSSI}}$ is ignored 1: Input signal $\overline{\text{CSIHTSSI}}$ is recognized If the slave select function is not used, this bit must be set to 0 (see also *28.3.2 "Master/slave connections" on page 2153*).

Details about CSIHnCTL1.CSIHnSSE:

Table 28-19 Operation of the slave select function during reception

CSIHnCTL0. CSIHnRXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHTSSI}}$	Receive operation
0	-	-	Reception disabled
1	0	-	Possible
1	1	0	Possible
1	1	1	Disabled

Table 28-20 Operation of the slave select function during transmission

CSIHnCTL0. CSIHnTXE	CSIHnCTL1. CSIHnSSE	$\overline{\text{CSIHTSSI}}$	Transmit operation
0	-	-	Transmission disabled
1	0	-	Possible
1	1	0	Possible
1	1	1	Disabled

(3) CSIHnCTL2 - CSIH control register 2

This register selects the transmission clock and specifies the baud rate.

For details refer to *28.3.6 "Transmission clock selection" on page 2159* .

Access This register can be read/written in 16-bit units.

Address <CSIHn_base> + 14_H

Initial Value $E000_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnPRS[2:0]			0	CSIHnBRS[11:0]											
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Changing the contents of this register is only permitted when CSIHnCTL0.CSIHnPWR = 0.

Table 28-21 CSIHnCTL2 register contents

Bit position	Bit name	Function
15 to 13	CSIHnPRS [2:0]	Selects the operating mode and the value of the prescaler:
11 to 0	CSIHnBRS [11:0]	Selects the baud rate:

CSIHn PRS2	CSIHn PRS1	CSIHn PRS0	Prescaler output (PRSOUT)
0	0	0	PCLK (master mode)
0	0	1	PCLK / 2 (master mode)
0	1	0	PCLK / 4 (master mode)
0	1	1	PCLK / 8 (master mode)
1	0	0	PCLK / 16 (master mode)
1	0	1	PCLK / 32 (master mode)
1	1	0	PCLK / 64 (master mode)
1	1	1	External clock via CSIHTSCK (slave mode)

CSIHnBRS[11:0]	Baud rate at CSIHTSCK
0	BRG is stopped
1	PCLK / ($2^m \times 1 \times 2$)
2	PCLK / ($2^m \times 2 \times 2$)
3	PCLK / ($2^m \times 3 \times 2$)
4	PCLK / ($2^m \times 4 \times 2$)
...	...
4095	PCLK / ($2^m \times 4095 \times 2$)

(4) CSIHnSTR0 - CSIH status register 0

This register indicates the status of the CSIH.

Access This register can be read in 32-bit units.

Address <CSIHn_base> + 04_H

Initial Value $0000\ 0010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnSRP[7:0]								CSIHnSPF[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn TMOE	CSIHn OFE	0	0	0	0	0	0	CSIHn TSF	0	CSIHn FLF	CSIHn EMF	CSIHn DCE	0	CSIHn PE	CSIHn OVE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 28-22 CSIHnSTR0 register contents (1/3)

Bit position	Bit name	Function
31 to 24	CSIHnSRP[7:0]	Indicates the number of received words in FIFO mode. **CSIHnSRP[7:0]** / **Description** 00_H — Number of received words (0 to 128) … 80_H > 80_H — Undefined These bits are cleared by CSIHnSTCR0.CSIHnPCT. In dual or transmit-only buffer memory mode, this value is fixed to 00_H.[a]
23 to 16	CSIHnSPF[7:0]	Indicates the number of unsent data in FIFO mode. **CSIHnSPF[7:0]** / **Description** 00_H — Number of unsent data (0 to 128) … 80_H > 80_H — Undefined These bits are cleared by CSIHnSTCR0.CSIHnPCT. In dual or transmit-only buffer memory mode, this value is fixed to 00_H.[a]

Table 28-22 CSIHnSTR0 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>CSIHnTMOE</td><td>Time-out error flag in FIFO mode.
Indicates whether a time-out error was detected:
0: No time out error detected
1: Time out error detected
For details see 3 “Time-out error” on page 2182 .
This bit is cleared by CSIHnSTCR0.CSIHnTMOEC.
This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.</td></tr>
<tr><td>14</td><td>CSIHnOFE</td><td>Overflow error flag in FIFO mode.
Indicates whether an overflow error was detected:
0: No overflow error detected
1: Overflow error detected
For details see 4 “Overflow error” on page 2183 .
This bit is cleared by CSIHnSTCR0.CSIHnOFEC.
This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.</td></tr>
<tr><td>7</td><td>CSIHnTSF</td><td>Transfer status flag:
0: Idle state
1: Transmission is in progress or being prepared
Setting and clearing of this bit is as follows:
<table>
<tr><th>Master mode</th><th>Set by</th><th>Cleared by</th></tr>
<tr><td>Tx only mode</td><td rowspan="3">Writing to transmit register or by setting trigger bit CSIHnMCTL2.CSIHBTST</td><td>Rising edge of CSIHTIC</td></tr>
<tr><td>Tx / Rx mode</td><td rowspan="2">Rising edge of CSIHTIR</td></tr>
<tr><td>Rx only mode</td></tr>
</table>
<table>
<tr><th>Slave mode</th><th>Set by</th><th>Cleared by</th></tr>
<tr><td>Tx only mode</td><td rowspan="2">Writing to transmit register or by setting trigger bit CSIHnMCTL2.CSIHBTST</td><td>Rising edge of CSIHTIC</td></tr>
<tr><td>Tx / Rx mode</td><td rowspan="2">Rising edge of CSIHTIR</td></tr>
<tr><td>Rx only mode</td><td>CSIHSCK clock input</td></tr>
</table>
This bit is cleared by CSIHnSTCR0.CSIHnPCT.</td></tr>
<tr><td>5</td><td>CSIHnFLF</td><td>FIFO full flag: Indicates a completely filled FIFO buffer[b]:
0: FIFO buffer is not full
1: FIFO buffer is full
This bit is cleared by CSIHnSTCR0.CSIHnPCT.</td></tr>
<tr><td>4</td><td>CSIHnEMF</td><td>FIFO empty flag: Indicates an empty FIFO buffer[b]:
0: FIFO buffer is not empty
1: FIFO buffer is empty
This bit is cleared by CSIHnSTCR0.CSIHnPCT.</td></tr>
</table>

Table 28-22 CSIHnSTR0 register contents (3/3)

Bit position	Bit name	Function
3	CSIHnDCE	Data consistency check error flag: 0: No data consistency error detected 1: Data consistency error detected This bit is cleared by writing 1 to CSIHnSTCR0.CSIHnDCEC. This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.
1	CSIHnPE	Parity error flag: 0: No parity error detected 1: Parity error detected This bit is cleared by writing 1 to CSIHnSTCR0.CSIHnPEC. This bit is initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.
0	CSIHnOVE	Overrun error flag: 0: No overrun error detected 1: Overrun error detected This bit is cleared by writing 1 to CSIHnSTCR0.CSIHnOVEC. This bit is Initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.

a) This bit is fixed to 0 in buffer mode because the number of data is managed by CSIHnMCTL2.CSIHnND[6:0].
b) The FIFO buffer can be filled with unsent data and received data.

Following table summarizes the meaning of the CSIHnSTR0 register bits in the different modes:

Table 28-23 CSIHnSTR0 register bits in different modes

Bit name	Mode			
	Direct access	FIFO	Transmit-only	Dual buffer
CSIHnSRP[7:0]	fixed to 0	Number of received words	fixed to 0	fixed to 0
CSIHnSPF[7:0]	fixed to 0	Number of unsent data	fixed to 0	fixed to 0
CSIHnTMOE	fixed to 0	Time-out error flag	fixed to 0	fixed to 0
CSIHnOFE	fixed to 0	Overflow error flag	fixed to 0	fixed to 0
CSIHnTSF	Transfer status flag			
CSIHnFLF	fixed to 0	FIFO full flag	fixed to 0	fixed to 0
CSIHnEMF	fixed to 1	FIFO empty flag	fixed to 1	fixed to 1
CSIHnDCE	Data consistency check error flag			
CSIHnPE	Parity error flag			
CSIHnOVE	Overrun error flag			fixed to 0

(5) CSIHnSTCR0 - CSIH status clear register 0

This register clears the status flags of the CSIHnSTR0 status register.

Access This register can be read/written in 16-bit units.

When read, the value 0000_H is always returned.

Address <CSIHn_base> + 08_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn TMOEC	CSIHn OFEC	0	0	0	0	0	CSIHn PCT	0	0	0	0	CSIHn DCEC	0	CSIHn PEC	CSIHn OVEC
R/W	R/W	R	R	R	R	R	R/W	R	R	R	R	R/W	R	R/W	R/W

Table 28-24 CSIHnSTCR0 register contents

Bit position	Bit name	Function
15	CSIHnTMOEC	Time-out error flag clear command: 0: No operation. Read value is always 0. 1: Clear time out error flag (CSIHnSTR0.CSIHnTMOE)
14	CSIHnOFEC	Overflow error flag clear command: 0: No operation. Read value is always 0. 1: Clear overflow error flag (CSIHnSTR0.CSIHnOFE)
8	CSIHnPCT	FIFO pointer clear command: 0: No operation. Read value is always 0. 1: Clear all FIFO buffer pointers - CSIHnMRWPO.CSIHnTRWA[6:0], - CSIHnMRWPO.CSIHnRRA[6:0] and - CSIHnMCTL2.CSIHnSOP[6:0] and the status bits - CSIHnSTR0.CSIHnSPF[7:0], - CSIHnSTR0.CSIHnSRP[7:0], - CSIHnSTR0.CSIHnFLF and - CSIHnSTR0.CSIHnTSF). Additionally, bit CSIHnSTR0.CSIHnEMF is set to 1 (FIFO empty). **Caution** If this bit is cleared during communication, ongoing communication is aborted.
3	CSIHnDCEC	Data consistency error flag clear command: 0: No operation. Read value is always 0. 1: Clear data consistency error flag (CSIHnSTR0.CSIHnDCE)
1	CSIHnPEC	Parity error flag clear command: 0: No operation. Read value is always 0. 1: Clear parity error flag (CSIHnSTR0.CSIHnPE)
0	CSIHnOVEC	Overrun error flag clear command: 0: No operation. Read value is always 0. 1: Clear overrun error flag (CSIHnSTR0.CSIHnOVE)

(6) CSIHnMCTL0 - CSIH Memory control register 0

This register selects the memory mode and the time-out setting.

Access This register can be read/written in 16-bit units.

Address <CSIHn_base> + 1040_H

Initial Value $001F_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	CSIHn MMS[1:0]		0	0	0	CSIHnTO[4:0]				
R	R	R	R	R	R	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

Table 28-25 CSIHnMCTL0 register contents

Bit position	Bit name	Function
9 to 8	CSIHn MMS[1:0]	Selects the memory mode: **CSIHn MMS1** / **CSIHn MMS0** / **Description** 0 / 0 / FIFO mode 0 / 1 / Dual buffer mode 1 / 0 / Transmit-only buffer mode 1 / 1 / Prohibited After a change of the memory mode, the respective buffer pointers must be reset by software. In direct access mode, the setting of these bits is ignored. **Caution** Changing the memory mode is only permitted when CSIHnCTL0.CSIHnPWR = 0, CSIHnCTL0.CSIHnTXE = 0, and CSIHnCTL0.CSIHnRXE = 0.
4 to 0	CSIHn TO[4:0]	Selects the time-out setting: **CSIHnTO[4:0]** / **Description** 00000_B / No time-out detection 00001_B / Time-out is (1 x 8 x BRG output clocks) 00010_B / Time-out is (2 x 8 x BRG output clocks) ... / 11111_B / Time-out is (31 x 8 x BRG output clocks) **Caution** Changing the time-out is only permitted when CSIHnCTL0.CSIHnPWR = 0, CSIHnCTL0.CSIHnTXE = 0, and CSIHnCTL0.CSIHnRXE = 0. For details on time-out detection see also *3 "Time-out error" on page 2182*.

(7) CSIHnMCTL1 - CSIH Memory control register 1

This register selects the conditions to generate the interrupt requests CSIHTIC and CSIHTIR in FIFO mode.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 1000_H

Initial Value 0000 $007F_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	CSIHnFES[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnFFS[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note Writing during communication is possible.

Table 28-26 CSIHnMCTL1 register contents

Bit position	Bit name	Function
22 to 16	CSIHnFES[6:0]	Selects the conditions to generate the CSIHTIC interrupt request in FIFO mode: CSIHTIC interrupt request is generated when the number of unsent data in the FIFO to be transmitted equals CSIHnFES[6:0], i.e. when CSIHnSTR0.CSIHnSPF[7:0] is decreased and matches CSIHnFES[6:0].
6 to 0	CSIHnFFS[6:0]	Selects the conditions to generate the CSIHTIR interrupt request in FIFO mode: CSIHTIR interrupt request is generated when the number of received data in the FIFO equals CSIHnFFS[6:0], i.e. when CSIHnSTR0.CSIHnSRP[7:0] is increased and matches CSIHnFFS[6:0].

(8) CSIHnMCTL2 - CSIH Memory control register 2

This register controls the operation of the memory in dual or transmit-only buffer mode and initiates a communication start in buffer mode.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 1004_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnBTST	0	0	0	0	0	0	0	CSIHnND[7:0]							
R/W	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnSOP[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Cautions

1. Write access to this register is forbidden when CSIHnCSTRO.CSIHnTSF = 1 (transfer in progress).
2. Write access to this register in FIFO mode is forbidden.

Table 28-27 CSIHnMCTL2 register contents (1/2)

Bit position	Bit name	Function
31	CSIHnBTST	Provides a start trigger for buffer transfer: 0: No operation 1: Start transfer command The read value is always 0. **Caution** This bit can only be used in dual buffer mode and transmit-only buffer mode. In direct access mode and FIFO mode, this bit is disabled.

Table 28-27 CSIHnMCTL2 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>23 to 16</td><td>CSIHnND[7:0]</td><td>Specifies the number of data for the different modes:
<table>
<tr><th>CSIHn ND[7:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>Direct access mode</th><th>FIFO mode</th></tr>
<tr><td>00_H</td><td>Send 0 data</td><td>Send 0 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>01_H</td><td>Send 1 data</td><td>Send 1 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>No influence</td><td>No influence</td></tr>
<tr><td>$3F_H$</td><td>Send 63 data</td><td>Send 63 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>40_H</td><td>Send 64 data</td><td>Send 64 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>No influence</td><td>No influence</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>Send 127 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>80_H</td><td>Prohibited</td><td>Send 128 data</td><td>No influence</td><td>No influence</td></tr>
<tr><td>$> 80_H$</td><td colspan="4">Undefined</td></tr>
</table>
The values are automatically decremented after data transfer.</td></tr>
<tr><td>6 to 0</td><td>CSIHn SOP[6:0]</td><td>Selects the pointer of the data to be sent:
<table>
<tr><th>CSIHn SOP[6:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>Direct access mode</th><th>FIFO mode[a]</th></tr>
<tr><td>00_H</td><td>0000_H</td><td>0000_H</td><td>No influence</td><td>0000_H</td></tr>
<tr><td>01_H</td><td>0004_H</td><td>0004_H</td><td>No influence</td><td>0004_H</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$3F_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>No influence</td><td>$00FC_H$</td></tr>
<tr><td>40_H</td><td>Prohibited</td><td>0100_H</td><td>No influence</td><td>0100_H</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>$01FC_H$</td><td>No influence</td><td>$01FC_H$</td></tr>
</table>
These bits are cleared by hardware when communication is forced to stop.</td></tr>
</table>

a) In FIFO mode, these bits indicate the send address.

(9) CSIHnMRWP0 - CSIH memory read/write pointer register 0

This register sets the pointers for reading from and writing to the dual or transmit-only buffer.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 1018_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	CSIHnRRA[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	CSIHnTRWA[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note Writing during the communication is possible.

Caution Write access to this register in FIFO mode is forbidden.

Table 28-28 CSIHnMRWP0 register contents (1/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>22 to 16</td><td>CSIHn RRA[6:0]</td><td>Selects the read pointer of the Rx buffer:
<table>
<tr><th>CSIHn RRA[6:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>Direct access mode</th><th>FIFO mode[a]</th></tr>
<tr><td>00_H</td><td>0000_H</td><td>0000_H</td><td>No influence</td><td>0000_H</td></tr>
<tr><td>01_H</td><td>0004_H</td><td>0004_H</td><td>No influence</td><td>0004_H</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$3F_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>No influence</td><td>$00FC_H$</td></tr>
<tr><td>40_H</td><td>Prohibited</td><td>0100_H</td><td>No influence</td><td>0100_H</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>$01FC_H$</td><td>No influence</td><td>$01FC_H$</td></tr>
</table>
These bits are automatically incremented when the reception data is read.
If an overrun error is generated while reading the Rx register, the read pointer is not incremented.
These bits are cleared when CSIHnSTCR0.CSIHnPCT is set to 1.</td></tr>
</table>

Table 28-28 CSIHnMRWP0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>6 to 0</td><td>CSIHn TRWA[6:0]</td><td>Selects the read/write pointer of the Tx buffer:
<table>
<tr><th>CSIHn TRWA[6:0]</th><th>Dual buffer mode</th><th>Transmit-only buffer mode</th><th>Direct access mode</th><th>FIFO mode[b]</th></tr>
<tr><td>00_H</td><td>0000_H</td><td>0000_H</td><td>No influence</td><td>0000_H</td></tr>
<tr><td>01_H</td><td>0004_H</td><td>0004_H</td><td>No influence</td><td>0004_H</td></tr>
<tr><td>...</td><td>...</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$3F_H$</td><td>$00FC_H$</td><td>$00FC_H$</td><td>No influence</td><td>$00FC_H$</td></tr>
<tr><td>40_H</td><td>Prohibited</td><td>0100_H</td><td>No influence</td><td>0100_H</td></tr>
<tr><td>...</td><td>Prohibited</td><td>...</td><td>No influence</td><td>...</td></tr>
<tr><td>$7F_H$</td><td>Prohibited</td><td>$01FC_H$</td><td>No influence</td><td>$01FC_H$</td></tr>
</table>
These bits are automatically incremented when the transmission data is written or read.
These bits are cleared when CSIHnSTCR0.CSIHnPCT is set to 1.</td></tr>
</table>

a) In FIFO mode, these bits indicate the read address of the reception data.
b) In FIFO mode, these bits indicate the read/write address of the transmission data.

(10) CSIHnCFGx - CSIH Configuration register x

These eight registers specify for each chip select signal CSIHTCSSx prescaler, parity, data length, recessive configuration for broadcasting, serial data direction, clock and data phase, idle enforcement configuration, idle timing, hold timing, inter-data delay timing and setup timing.

Slave mode In slave mode the transmission protocol setting of the CSIHnCFG0 register are relevant:

- CSIHnPS0: parity usage
- CSIHnDLS0: data length selection
- CSIHnDIR0: data direction
- CSIHnCKP0,CSIHnDAP0: clock and data phase

All other register settings are disregarded.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 1044_H + x x 4

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHn PSCLx[1:0]		CSIHn PSx[1:0]		CSIHnDLSx[3:0]				0	0	0	0	CSIHn RCBx	CSIHn DIRx	CSIHn CKPx	CSIHn DAPx
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHn IDLx	CSIHnIDx[2:0]			CSIHnHDx[3:0]				CSIHnINx[3:0]				CSIHnSPx[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Write access is only possible while CSIHnCTL0.CSIHnPWR = 0.

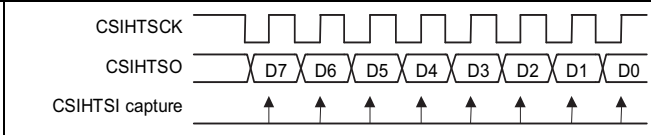
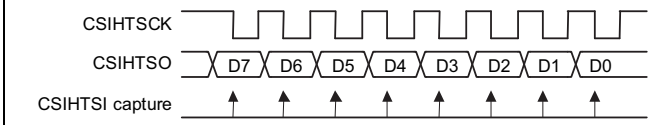
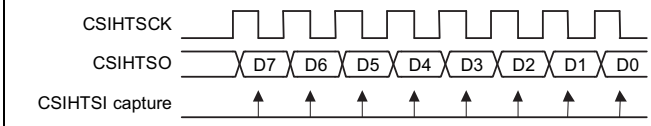
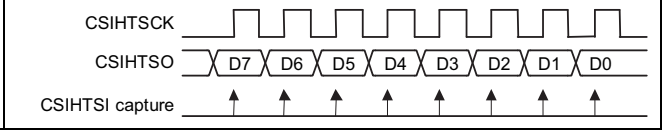
Table 28-29 CSIHnCFGx register contents (1/4)

Bit position	Bit name	Function
31 to 30	CSIHn PSCLx[1:0]	Selects the prescaler for chip select x CSIHn PSCLx1 / CSIHn PSCLx0 / Prescaler output 0 / 0 / CSIHBPCLK 0 / 1 / CSIHBPCLK / 2 1 / 0 / CSIHBPCLK / 4 1 / 1 / CSIHBPCLK / 8 These bits are only available in master mode.

Table 28-29 CSIHnCFGx register contents (2/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>29 to 28</td><td>CSIHn PSx[1:0]</td><td>Selects the parity for chip select x for transmission and reception:
<table>
<tr><th>CSIHn PSx1</th><th>CSIHn PSx0</th><th>Transmission</th><th>Reception</th></tr>
<tr><td>0</td><td>0</td><td>No parity transmitted</td><td>No parity expected</td></tr>
<tr><td>0</td><td>1</td><td>Add parity bit fixed at 0</td><td>Parity bit is expected but not judged</td></tr>
<tr><td>1</td><td>0</td><td>Add odd parity</td><td>Odd parity bit is expected</td></tr>
<tr><td>1</td><td>1</td><td>Add even parity</td><td>Even parity bit is expected</td></tr>
</table></td></tr>
<tr><td>27 to 24</td><td>CSIHn DLSx[3:0]</td><td>Selects the data length for chip select x:
<table>
<tr><th>CSIHn DLSx[3:0]</th><th>Data length</th></tr>
<tr><td>0000_B</td><td>16-bit</td></tr>
<tr><td>0001_B</td><td>1-bit</td></tr>
<tr><td>0010_B</td><td>2-bit</td></tr>
<tr><td>...</td><td>...</td></tr>
<tr><td>1111_B</td><td>15-bit</td></tr>
</table>
Note:
Data length between 1 bit and 7 bit requires that EDL function is used (see also 2 “Data length greater than 16 bits” on page 2164). In addition, it is forbidden to transmit two consecutive data with a length of less than 7 bits.</td></tr>
<tr><td>19</td><td>CSIHn RCBx</td><td>Selects the recessive configuration for broadcasting for chip select x:
0: Dominant (higher priority)
1: Recessive (lower priority)
For details see 1 “Configuration registers” on page 2155</td></tr>
<tr><td>18</td><td>CSIHn DIRx</td><td>Selects the serial data direction for chip select x:
0: Data is sent/received with MSB first
1: Data is sent/received with LSB first
For details see 28.3.10 “Serial data direction selection” on page 2166</td></tr>
</table>

Table 28-29 CSIHnCFGx register contents (3/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>17 to 16</td><td>CSIHn CKPx
CSIHn DAPx</td><td>Selects the clock and data phase for chip select x:
<table>
<tr><th>CSIHn CKPx</th><th>CSIHn DAPx</th><th>Clock and data phase selection</th></tr>
<tr><td>0</td><td>0</td><td> 
</td></tr>
<tr><td>0</td><td>1</td><td> 
</td></tr>
<tr><td>1</td><td>0</td><td> 
</td></tr>
<tr><td>1</td><td>1</td><td> 
</td></tr>
</table></td></tr>
<tr><td>15</td><td>CSIHn IDLx</td><td>Selects the idle enforcement configuration for chip select x:
0: If all bits of the chip select value CSIHnTX0W.CSIHnCS[7:0] of the next data to transmit remain the same as the previous one or the buffer becomes empty, the chip select signal stays active. If a different chip select value is defined, chip select x becomes idle.
1: An idle state is inserted after every transfer to chip select x.

This bit is only available in master mode.</td></tr>
<tr><td>14 to 12</td><td>CSIHn IDx[2:0]</td><td>Selects the idle time for chip select x:
<table>
<tr><th>CSIHn DLSx[3:0]</th><th>Idle timing</th></tr>
<tr><td>000_B</td><td>0.5 transmission clock periods</td></tr>
<tr><td>001_B</td><td>1 transmission clock period</td></tr>
<tr><td>010_B</td><td>1.5 transmission clock periods</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5)</td></tr>
<tr><td>111_B</td><td>8.5 transmission clock periods</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

Table 28-29 CSIHnCFGx register contents (4/4)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>11 to 8</td><td>CSIHn HDx[3:0]</td><td>Selects the hold time for chip select x in transmission clock periods:
<table>
<tr><th>CSIHn INx[3:0]</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 0</th><th>Hold timing with CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0.5</td><td>1</td></tr>
<tr><td>0001_B</td><td>1</td><td>1.5</td></tr>
<tr><td>0010_B</td><td>1.5</td><td>2</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td><td>... (2, 3, 4, 6, 5, 7, 9, 10, 11, 12, 13, 15, 17, 19)</td></tr>
<tr><td>1111_B</td><td>20.5</td><td>21</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>7 to 4</td><td>CSIHn INx[3:0]</td><td>Selects the inter-data delay time for chip select x in transmission clock periods:
<table>
<tr><th>CSIHn INx[3:0]</th><th>Inter-data delay with CSIHnCTL1.CSIHnSIT = 0</th><th>Inter-data delay with CSIHnCTL1.CSIHnSIT = 1</th></tr>
<tr><td>0000_B</td><td>0</td><td>0.5</td></tr>
<tr><td>0001_B</td><td>0.5</td><td>1</td></tr>
<tr><td>0010_B</td><td>1</td><td>1.5</td></tr>
<tr><td>0011_B</td><td>2</td><td>2.5</td></tr>
<tr><td>...</td><td>... (3, 4, 6, 8, 9, 10, 11, 12, 14, 16, 18)</td><td>... (3.5, 4.5, 6.5, 8.5, 9.5, 10.5, 11.5 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20</td><td>20.5</td></tr>
</table>
These bits are only available in master mode.</td></tr>
<tr><td>3 to 0</td><td>CSIHn SPx[3:0]</td><td>Selects the setup time for chip select x in transmission clock periods:
<table>
<tr><th>CSIHn SPx[3:0]</th><th>Setup delay</th></tr>
<tr><td>0000_B</td><td>0.5</td></tr>
<tr><td>0001_B</td><td>1</td></tr>
<tr><td>0010_B</td><td>1.5</td></tr>
<tr><td>...</td><td>... (2.5, 3.5, 4.5, 6.5, 8.5, 9.5 10.5, 11.5, 12.5, 14.5, 16.5, 18.5)</td></tr>
<tr><td>1111_B</td><td>20.5</td></tr>
</table>
These bits are only available in master mode.</td></tr>
</table>

(11) CSIHnTX0W - CSIH transmit data register 0 for word access

This register stores the transmission data. In addition, it specifies the communication interrupt request, the end-of-job, the extended data length, and the chip select activation.

Access This register can be read/written in 32-bit units.

Address <CSIHn_base> + 1008_H

Initial Value Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CSIHnCIRE	CSIHnEOJ	CSIHnEDL	0	0	0	0	0	CSIHnCS[7:0]							
R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Read access to this register in FIFO mode is prohibited.

Table 28-30 CSIHnTX0W register contents (1/2)

Bit position	Bit name	Function
31	CSIHnCIRE	Enables the communication interrupt request CSIHTIC in dual or transmit-only buffer mode or the job completion interrupt CSIHTJC request in FIFO mode: 0: No interrupt requested 1: Interrupt requested. Generates interrupt CSIHTIC or CSIHTJC after transmission. For details see *3 "CSIHTIC communication interrupt" on page 2169* and *6 "CSIHTIJC job completion interrupt" on page 2175* . **Caution** This bit is only valid when job mode is enabled (CSIHnCTL1.CSIHnJE = 1).
30	CSIHnEOJ	Specifies the end of a job: 0: No end-of-job data 1: End-of-job data **Caution** This bit is only valid when job mode is enabled (CSIHnCTL1.CSIHnJE = 1).

Table 28-30 CSIHnTX0W register contents (2/2)

Bit position	Bit name	Function
29	CSIHnEDL	Specifies whether the associated data requires the extended data length (EDL) option: 0: Normal operation 1: Extended data length activated The associated data is transmitted as of 16 bits. No inter-data or idle timing will be inserted after the data is transmitted. If CSIHnCTL1.CSIHnEDLE = 1 the following data must have the same CS selection. If CS changes while CSIHnCTL1.CSIHnEDLE = 1, the correct operation is not assured. **Caution** This bit is only available if CSIHnCTL1.CSIHnEDLE = 1. This bit is cleared when CSIHnCTL1.CSIHnEDLE is cleared.
23 to 16	CSIHnCSx	Activates one or several chip select signals: 0: Chip select x is activated for the associated transmission 1: Chip select x is deactivated for the associated transmission If CSIHnTX0W.CSIHnCS[7:0] = FF_H, the data is sent with the configuration of register CSIHnCFG0. In this case no chip select is activated. **Caution** If several chip select signals are enabled for broadcasting, the configuration of one with CSIHnCFGx.CSIHnRCBx = 0 (dominant) is used. In this case, all dominant chip selects must be set to precisely the same configuration. If the CSIHnCSx bits are written in slave mode, FE_H is written to these bits.
15 to 0	CSIHnTX[15:0]	Stores the transmission data

(12) CSIHnTX0H - CSIH transmit data register 0 for half word access

This register stores the transmission data. This register is the same as bits 15 to 0 of register CSIHnTX0W.

Access This register can be read/written in 16-bit units.

Address <CSIHn_base> + $100C_H$

Initial Value Undefined

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnTX[15:0]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Caution Read access to this register in FIFO mode is prohibited.

Table 28-31 CSIHnTX0H register contents

Bit position	Bit name	Function
15 to 0	CSIHnTX[15:0]	Stores the transmission data

(13) CSIHnRX0W - CSIH receive data register 0 for word access

This register stores the received data.

Access This register can be read in 32-bit units.

Address <CSIHn_base> + 1010_H

Initial Value Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	CSIHnRPE	CSIHnTDCE	CSIHnCS[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Cautions

1. This register can be read in direct access mode and Tx only buffer mode while CSIHnCTL0.CSIHnPWR = 1.
2. This register is Initialized when CSIHnCTL0.CSIHnPWR changes from 0 to 1 or from 1 to 0.

Table 28-32 CSIHnRX0W register contents

Bit position	Bit name	Function
25	CSIHnRPE	Indicates whether a reception data parity error was detected: 0: No parity error on the associated reception data detected 1: Parity error on the associated reception data detected
24	CSIHnTDCE	Indicates whether a transmission data consistency error was detected: 0: No consistency error on the associated transmission data detected 1: Consistency error on the associated transmission data detected
23 to 16	CSIHnCSx	Indicates which chip select signal was activated: 0: Chip select x was activated for the associated transmission 1: Chip select x was deactivated for the associated transmission
15 to 0	CSIHnRX[15:0]	Stores the reception data.

(14) CSIHnRX0H - CSIH receive data register 0 for half word access

This register stores the reception data. This register is the same as bits 15 to 0 of register CSIHnTX0W.

Access This register can be read/written in 16-bit units.

Address <CSIHn_base> + 1014_H

Initial Value Undefined

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSIHnRX[15:0]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 28-33 CSIHnRX0H register contents

Bit position	Bit name	Function
15 to 0	CSIHnRX[15:0]	Stores the reception data.

(15) CSIHnEMU - CSIH emulation register

This register controls whether the CSIHn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <CSIHn_base> + 18_H

Initial Value 00_H

7	6	5	4	3	2	1	0
CSIHn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 28-34 CSIHnEMU register contents

Bit position	Bit name	Function
7	CSIHn SVSDIS	Emulation control 0: CSIHn can be stopped during emulation 1: CSIHn continuous operating during emulation

28.5 Operating Procedures

The following examples and instructions are sorted according to the memory mode:

- Direct access
- Transmit-only buffer
- Dual buffer
- FIFO

28.5.1 Procedures in direct access mode

Two examples for a master are given, one with job mode disabled, one with job mode enabled.

(1) Transmit/receive in master mode, job mode disabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode is disabled (CSIHnCTL1.CSIHnJE = 0)
- The number of data is 10 (CSIHnMCTL2.CSIHnND[7:0] = $0A_H$)
- Normal CSIHTIC interrupt timing (CSHICTL1.CSIHnSLIT = 0)
- Direct access mode (CSHICTL0.CSIHnMBS = 1)

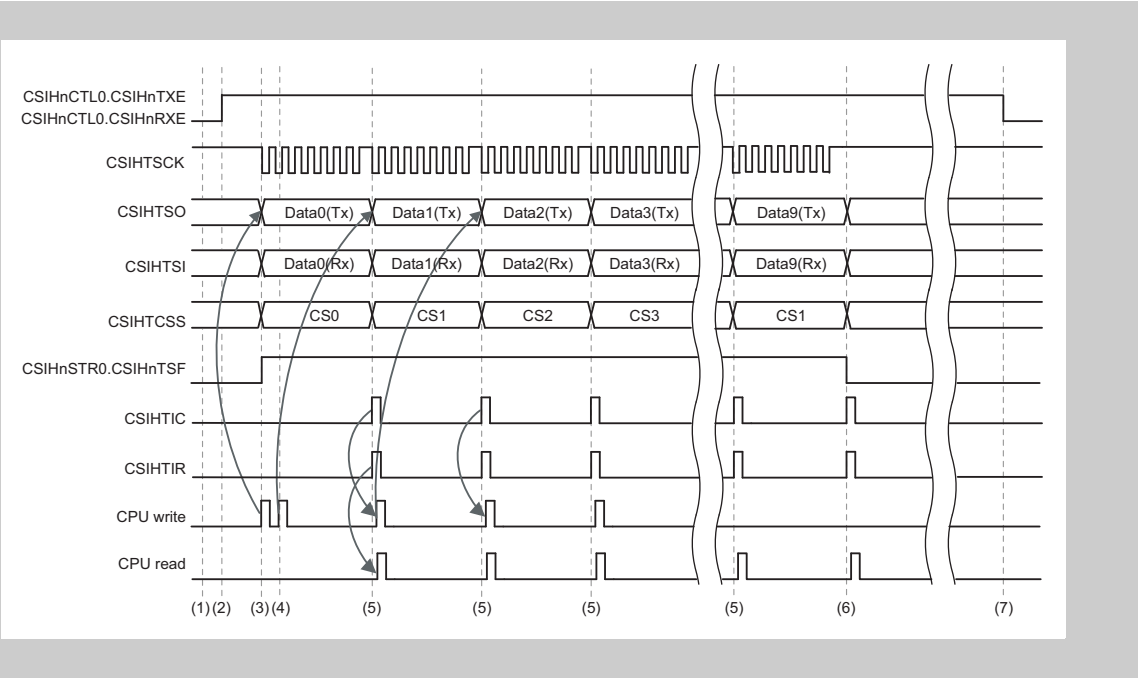


Figure 28-41 Master in direct access mode, CSIHnCTL1.CSIHnJE = 0

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. This example uses the chip select signals CS0 to CS3.
2. In the CSIHnCTL0 register, set bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission), CSIHnRXE = 1 (enable reception). This enables the output signal CSIHTSO.
3. Write the first data to be sent to the transmission register CSIHnTX0W. Within the same write operation activate CS0. Transmission starts automatically when the first data is available.
4. Write the second data to CSIHnTX0W. If required, you can change the CS to address a different device. Writing the second data immediately after the first one avoids unnecessary delays between the data.
5. After every data that has been transmitted, the interrupts CSIHTIC and CSIHTIR are generated:
   – CSIHTIC indicates that the next data can be written to CSIHnTX0W.
   – CSIHTIR indicates that the reception register CSIHnRX0 must be read.

6. No more write action is required after completion of Data8. Data9 (the last data) has been written in advance.
   However, the reception register CSIHnRX0 must be read after completion of data 8 and 9.
7. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

(2) Transmit/receive in master mode, job mode enabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- Normal CSIHTIC interrupt timing (CSHICTL1.CSIHnSLIT = 0)
- Direct access mode (CSHICTL0.CSIHnMBS = 1)
- Two jobs, each with three data are sent

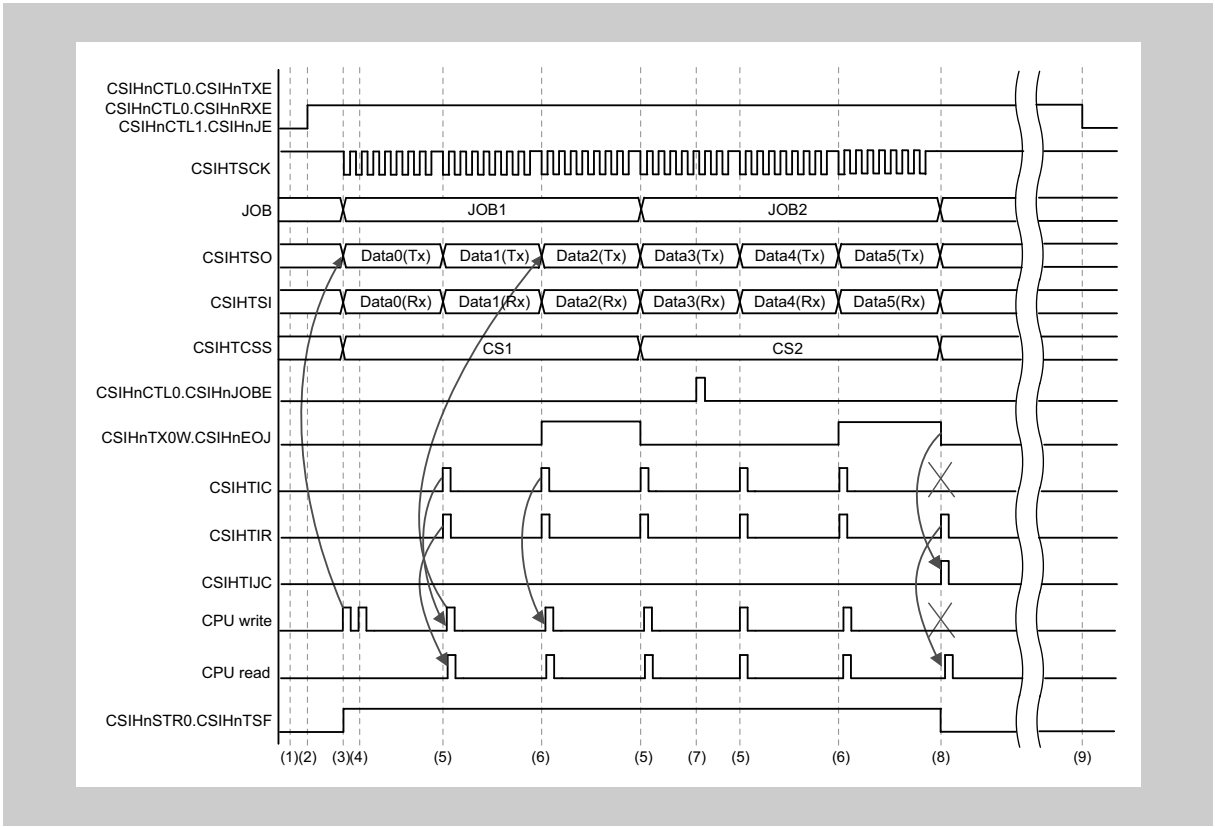


Figure 28-42 Master in direct access mode, CSIHnCTL1.CSIHnJE = 1

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. This example uses the chip select signals CS1 and CS2.
   Specify the transfer mode and job mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
2. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission), CSIHnRXE = 1 (enable the reception) and CSIHnMBS = 1 (select direct access mode).
3. Write the first data to be sent to the transmission register CSIHnTX0W. Transmission starts automatically when the first data is available.
   The flag CSIHnSTR0.CSIHnTSF signals that a communication is in progress.

4. Write the second data to CSIHnTX0H. Writing the second data immediately after the first one avoids unnecessary delays between the data.
5. After every data that has been transmitted, the interrupt requests CSIHTIC and CSIHTIR are generated:
   – CSIHTIC indicates that the next data can be written to CSIHnTX0.
   – CSIHTIR indicates that the reception register CSIHnRX0 must be read.
6. Setting CSIHnTX0W.CSIHnEOJ = 1 specifies that the last data of the current job is sent. After that, the next job may begin.
7. By setting CSIHnCTL0.CSIHnJOBE = 1, the communication is forced to stop at the end of the current job (JOB2).
8. After the forced stop of communication, the interrupt request CSIHTIC is replaced by CSIHTIJC. CSIHTIR is generated as usual.
   The interrupt request CSIHTIJC indicates a forced stop of communication at the end of the current job.
   The interrupt request CSIHTIC is not generated. Additionally, the transmission data available in register CSIHnTX0 is not sent.
9. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

28.5.2 Procedures in transmit-only buffer mode

Two examples for a master, one with job mode disabled, one with job mode enabled

(1) Transmit/receive in master mode, job mode disabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- The number of data is 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- The transfer start address is 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)

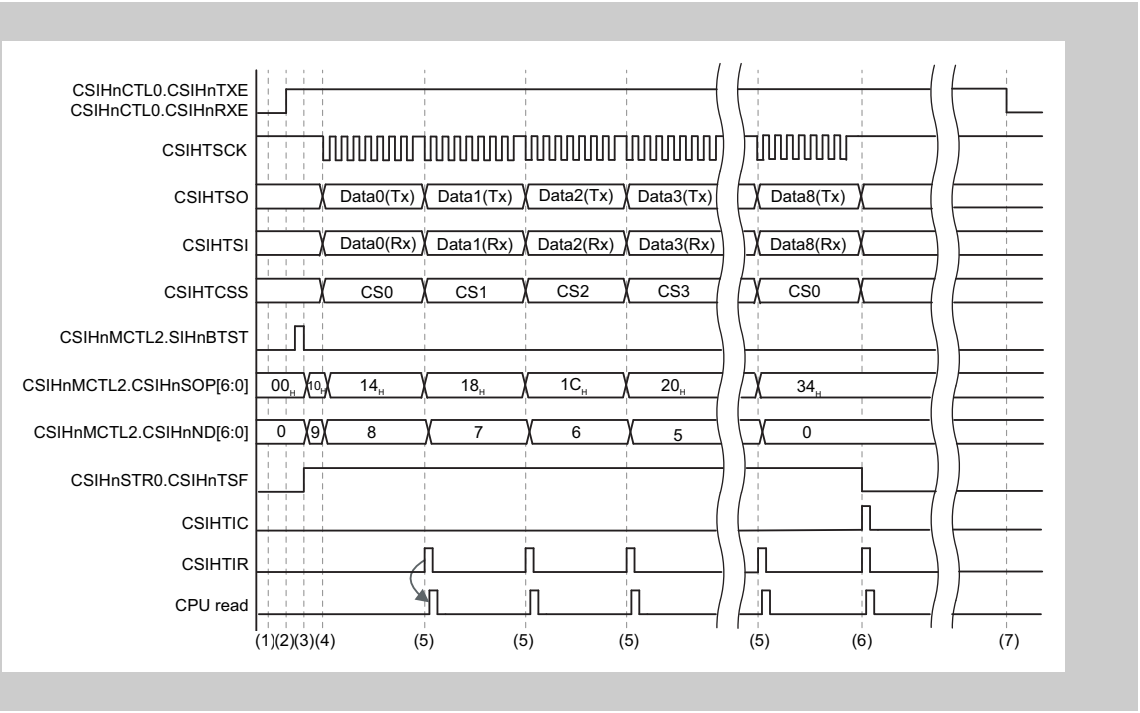


Figure 28-43 Master in transmit-only buffer mode, CSIHnCTL1.CSIHnJE = 0

Note The process of writing the data into the buffer is not described.

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. This example uses the chip select signals CS0 to CS3.
   Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
   Configure the memory mode in CSIHnMCTL0.CSIHnMMS[1:0]. Set CSIHnMCTL0.CSIHnMMS[1:0] = 10_B.
   Set the buffer write pointer CSIHnMRWP0.CSIHnTRWA[6:0] to the first data address 10_H and write the transmit data to CSIHnTX0. The pointer CSIHnMRWP0.CSIHnTRWA[6:0] is automatically incremented after each write.

2. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
3. Configure the send pointer and the number of data by setting bits CSIHnMCTL2.SCIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0]. Start the buffer transfer by setting CSIHnMCTL2.CSIHnBTST.
4. Bits CSIHnMCTL2.CSIHnSOP[6:0] are automatically incremented and bits CSIHnMCTL2.CSIHnND[7:0] decremented after each data. Transmission/reception is started.
5. After every data that has been transmitted, the interrupt request CSIHTIR is generated. CSIHTIR indicates that the reception register CSIHnRX0 must be read.
6. When all transmissions are complete, the interrupt request CSIHTIC is generated.
7. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

(2) Transmit/receive in master mode, job mode enabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode enabled (CSIHnCTL1.CSIHnJE = 1)
- The number of data is 8 (CSIHnMCTL2.CSIHnND[7:0] = 08_H)
- The transfer start address is 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)

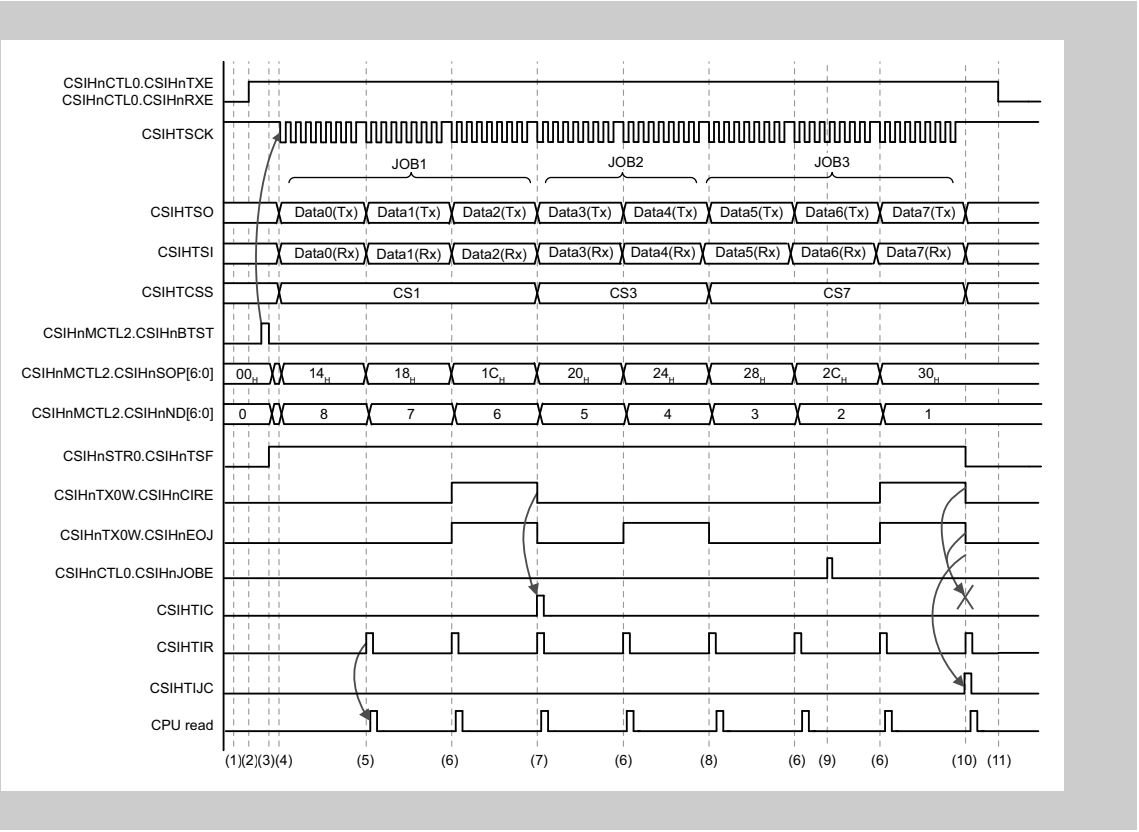


Figure 28-44 Master in transmit-only buffer mode, CSIHnCTL1.CSIHnJE = 1

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers.This example uses the chip select signals CS1, CS3, and CS7.
   Specify the transfer mode and job mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
2. Configure the memory mode in CSIHnMCTL0.CSIHnMMS[1:0]. Set CSIHnMCTL0.CSIHnMMS[1:0] = 10_B.
   Set the buffer write pointer CSIHnMRWP0.CSIHnTRWA[6:0] to the first data address 10_H and write the transmit data to CSIHnTX0W. The pointer CSIHnMRWP0.CSIHnTRWA[6:0] is automatically incremented after each write.
   Since data 2, 4 and 7 shall indicate an end-of-job, the CSIHnTX0W.CSIHnEOJ bit of these data must be set to 1.
   In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock),

CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.

3. Configure the send pointer and the number of data by setting bits CSIHnMCTL2.SCIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0]. Start the buffer transfer by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission is started. Bits CSIHnMCTL2.CSIHnSOP[6:0] are automatically incremented and bits CSIHnMCTL2.CSIHnND[7:0] decremented after each data.
5. After every data that has been transmitted, the interrupt request CSIHTIR is generated. CSIHTIR indicates that the reception register CSIHnRX0 must be read.
6. The CSIHnTX0W.CSIHnEOJ = 1 of the Tx data indicates that the last data of the current job is sent.
7. The interrupt request CSIHTIC is generated. CSIHTIC indicates that the last data of the current job (CSIHnTX0W.CSIHnEOJ = 1) was sent with CSIHnTX0W.CSIHnCIRE = 1.
8. The interrupt request CSIHTIC is not generated, because the last data of the current job (CSIHnTX0W.CSIHnEOJ = 1) was sent with CSIHnTX0W.CSIHnCIRE = 0.
9. By setting CSIHnCTL0.CSIHnJOBE = 1 the communication is forced to stop at the end of JOB3.
10. After the forced stop of communication, the interrupt requests CSIHTIJC and CSIHTIR are generated at the end of job3.
    The interrupt request CSIHTIJC indicates a forced stop of communication at the end of the current job.
    The interrupt request CSIHTIC is not generated because the interrupt request CSIHTIJC is generated instead.
11. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

28.5.3 Procedures in dual buffer mode

Four examples, two for a master and two for a slave with job mode disabled and enabled.

(1) Transmit/receive in master mode, job mode disabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Default clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- The number of data is 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- The transfer start address is 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)

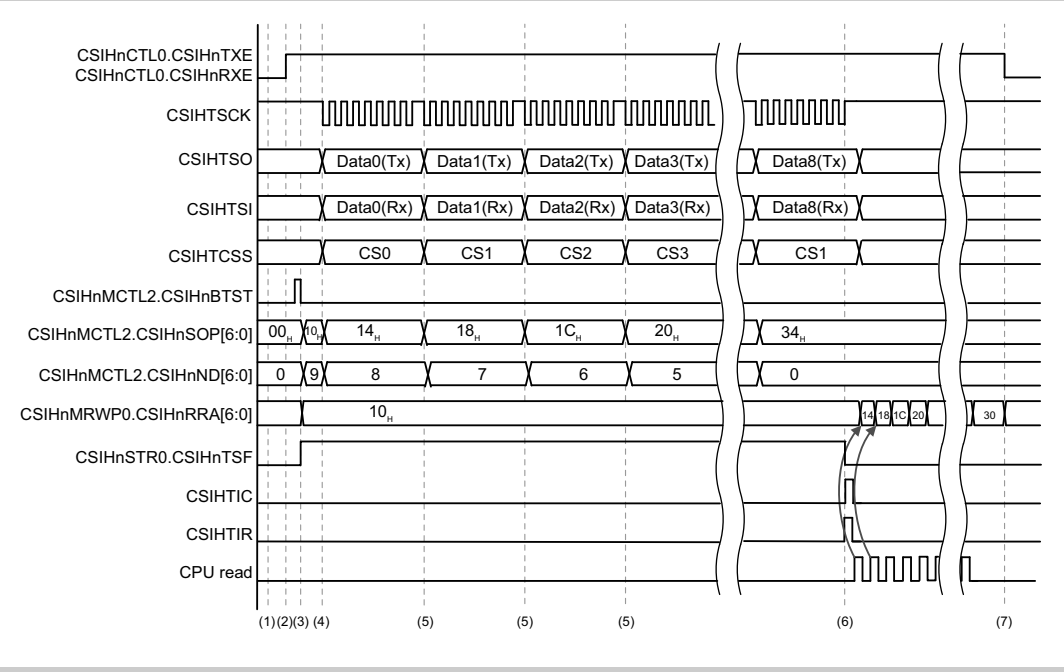


Figure 28-45 Master in dual buffer mode, CSIHnCTL1.CSIHnJE = 0

Note The process of writing the data into the buffer is not described.

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. The example uses the chip select signals CS0 to CS3.
   Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
   Set bit CSIHnSTCR0.CSIHnPCT = 1 to clear all buffer pointers.

2. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable the reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
3. Configure the communication by setting CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0]. Trigger the buffer transfer by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission is started.
5. Bits CSIHnMCTL2.CSIHnSOP[6:0] are automatically incremented and bits CSIHnMCTL2.CSIHnND[7:0] decremented after each data. This is repeated until the last data is transmitted/received.
   The interrupt requests CSIHTIC and CSIHTIR are not generated.
6. When the last data is transmitted/received, the interrupt requests CSIHTIC and CSIHTIR are generated.
   The CPU starts to read the received data from the Rx buffer. The start adress of the read access is specified in CSIHnMRWP0.CSIHnRRA[6:0]. These bits are incremented after the reading of every data.
7. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

(2) Transmit/receive in master mode, job mode enabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode is enabled (CSIHnCTL1.CSIHnJE = 1)
- The number of data is 8 (CSIHnMCTL2.CSIHnND[7:0] = 08_H)
- The transfer start address is 00_H (CSIHnMCTL2.CSIHnSOP[6:0] = 00_H)

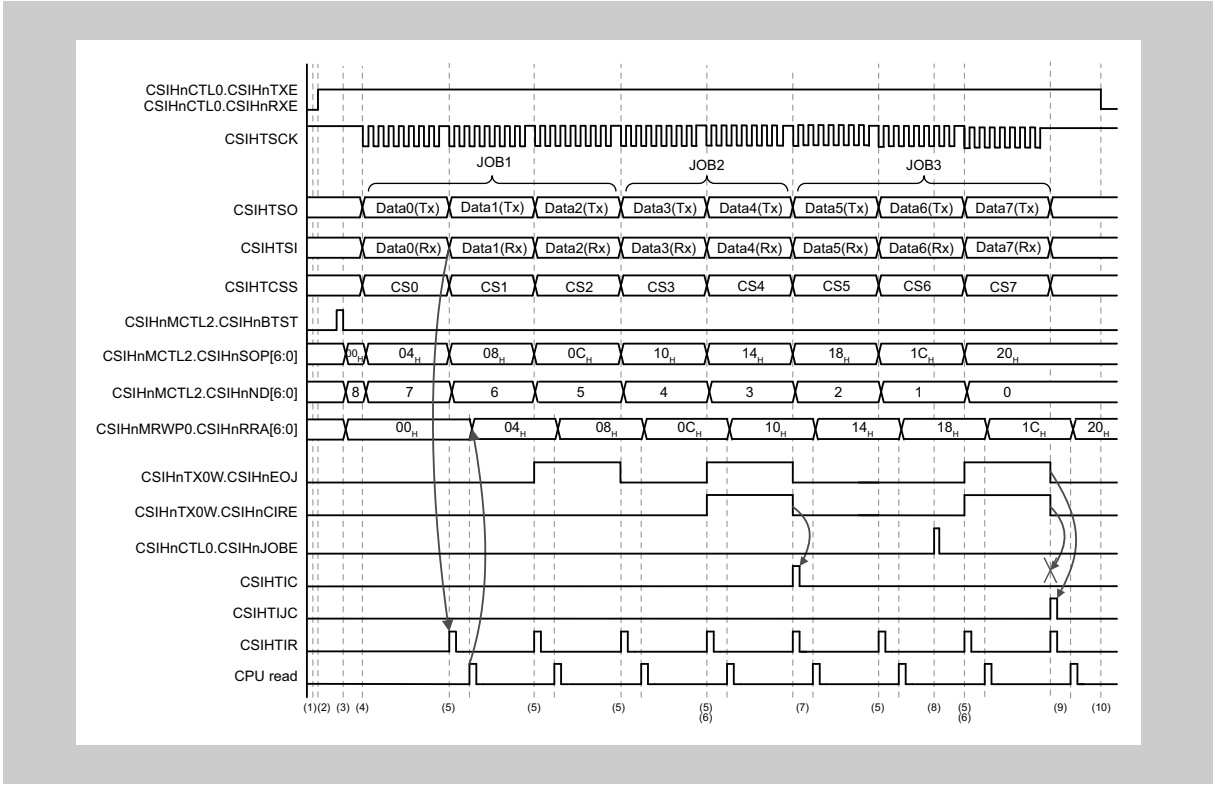


Figure 28-46 Master in dual buffer mode, CSIHnCTL1.CSIHnJE = 1

Note The process of writing the data into the buffer is not described.

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. The example uses the chip select signals CS0 to CS7.
   Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
   Set bit CSIHnSTCR0.CSIHnPCT = 1 to clear all buffer pointers.
   Set the buffer write pointer CSIHnMRWP0.CSIHnTRWA[6:0] to the first data address 00_H and write the transmit data to CSIHnTX0W. The pointer CSIHnMRWP0.CSIHnTRWA[6:0] is automatically incremented after each write.
   Since data 2, 4 and 7 shall indicate an end-of-job, the CSIHnTX0W.CSIHnEOJ bit of these data must be set to 1.

2. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
3. Configure the communication by setting bits CSIHnMCTL2.CSIHnSOP[6:0] and CSIHnMCTL2.CSIHnND[7:0]. Start the buffer transfer by setting CSIHnMCTL2.CSIHnBTST.
4. Transmission is started.
5. Bits CSIHnMCTL2.CSIHnSOP[6:0] are automatically incremented and bits CSIHnMCTL2.CSIHnND[7:0] decremented after each data. This is repeated until the last data is transmitted/received.
   The interrupt request CSIHTIR is generated after every data.
   The interrupt request CSIHTIC is not generated, because the last data of the current job (CSIHnTX0W.CSIHnEOJ = 1) was sent with CSIHnTX0W.CSIHnCIRE = 0.
6. CSIHnTX0W.CSIHnEOJ = 1 indicates that the corresponding data the end of a job.
7. The interrupt request CSIHTIC is generated. CSIHTIC indicates that the last data of the current job (CSIHnTX0W.CSIHnEOJ = 1) was sent with CSIHnTX0W.CSIHnCIRE = 1.
8. By setting CSIHnCTL0.CSIHnJOBE = 1 the communication is forced to stop at the end of job3.
9. After the forced stop of communication, the interrupt requests CSIHTIJC and CSIHTIR are generated at the end of JOB3.
   The interrupt request CSIHTIJC indicates a forced stop of communication at the end of the current job.
   The interrupt request CSIHTIC is not generated because the interrupt request CSIHTIJC is generated instead. Additionally, the transmission data available in register CSIHnTX0 is not sent.
10. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

(3) Transmit/receive in slave mode

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Default clock and data phase (CSIHnCFG0.CSIHnCKP0 = 0, CSIHnCFG0.CSIHnDAP0 = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)
- The number of data is 9 (CSIHnMCTL2.CSIHnND[7:0] = 09_H)
- The transfer start address is 10_H (CSIHnMCTL2.CSIHnSOP[6:0] = 10_H)

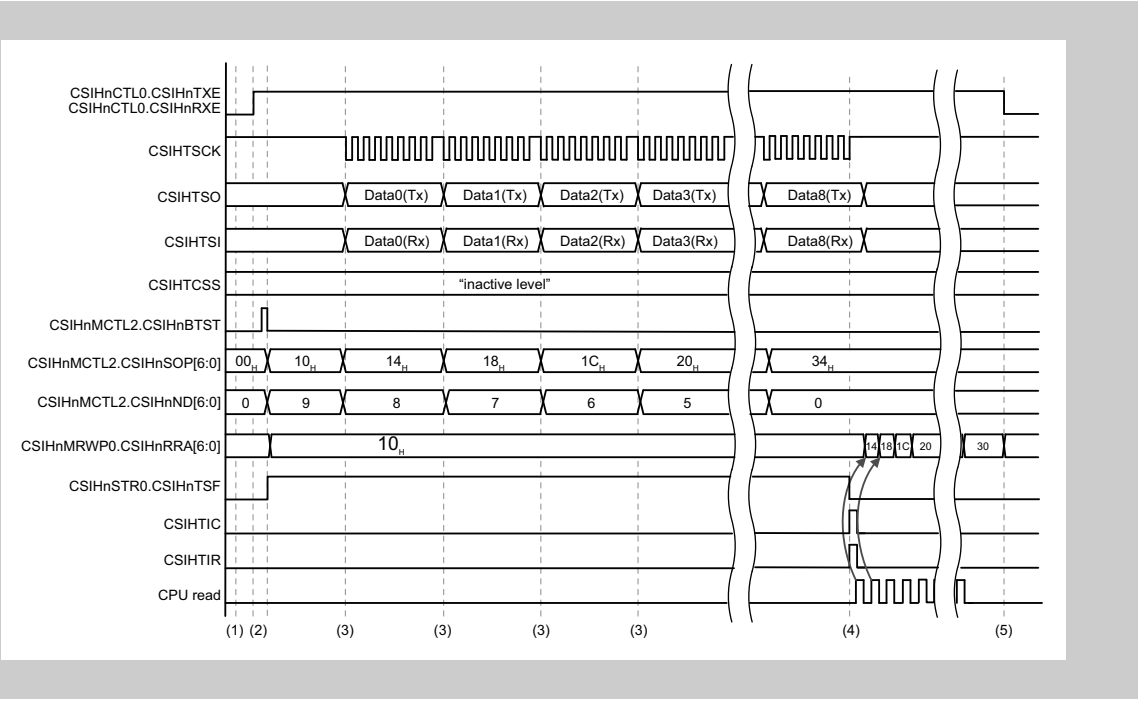


Figure 28-47 Slave in dual buffer mode with CSIHnCTL1.CSIHnJE = 0

Note The process of writing the data into the buffer is not described.

Procedure:

1. Configure the communication protocol in register CSIHnCFG0.
   Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
   Set bit CSIHnSTCR0.CSIHnPCT = 1 to clear all buffer pointers.
2. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
   Specify the transfer start address by setting CSIHnMCTL2.CSIHnSOP[6:0] and the number of data by setting CSIHnMCTL2.CSIHnND[7:0]. Enable the buffer transfer by setting CSIHnMCTL2.CSIHnBTST.

   Transmission is started when the input clock from the master is received.
3. Bits CSIHnMCTL2.CSIHnSOP[6:0] are automatically incremented and bits CSIHnMCTL2.CSIHnND[7:0] decremented after each data. This is repeated until the last data is transmitted/received.

The interrupt requests CSIHTIC and CSIHTIR are not generated, because transmission data comes from, received data is stored in the buffer.

4. When the last data is transmitted/received, the interrupt requests CSIHTIC and CSIHTIR are generated.
   The CPU starts to read the received data that is stored in the Rx buffer. The start adress of the read access is specified in bits CSIHnMRWP0.CSIHnRRA[6:0]. These bits are incremented after the reading of every data.
5. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

28.5.4 Procedures in FIFO mode

Two examples for a master, one with job mode disabled, one with job mode enabled.

(1) Transmit/receive in master mode, job mode disabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B)
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Default clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode disabled (CSIHnCTL1.CSIHnJE = 0)

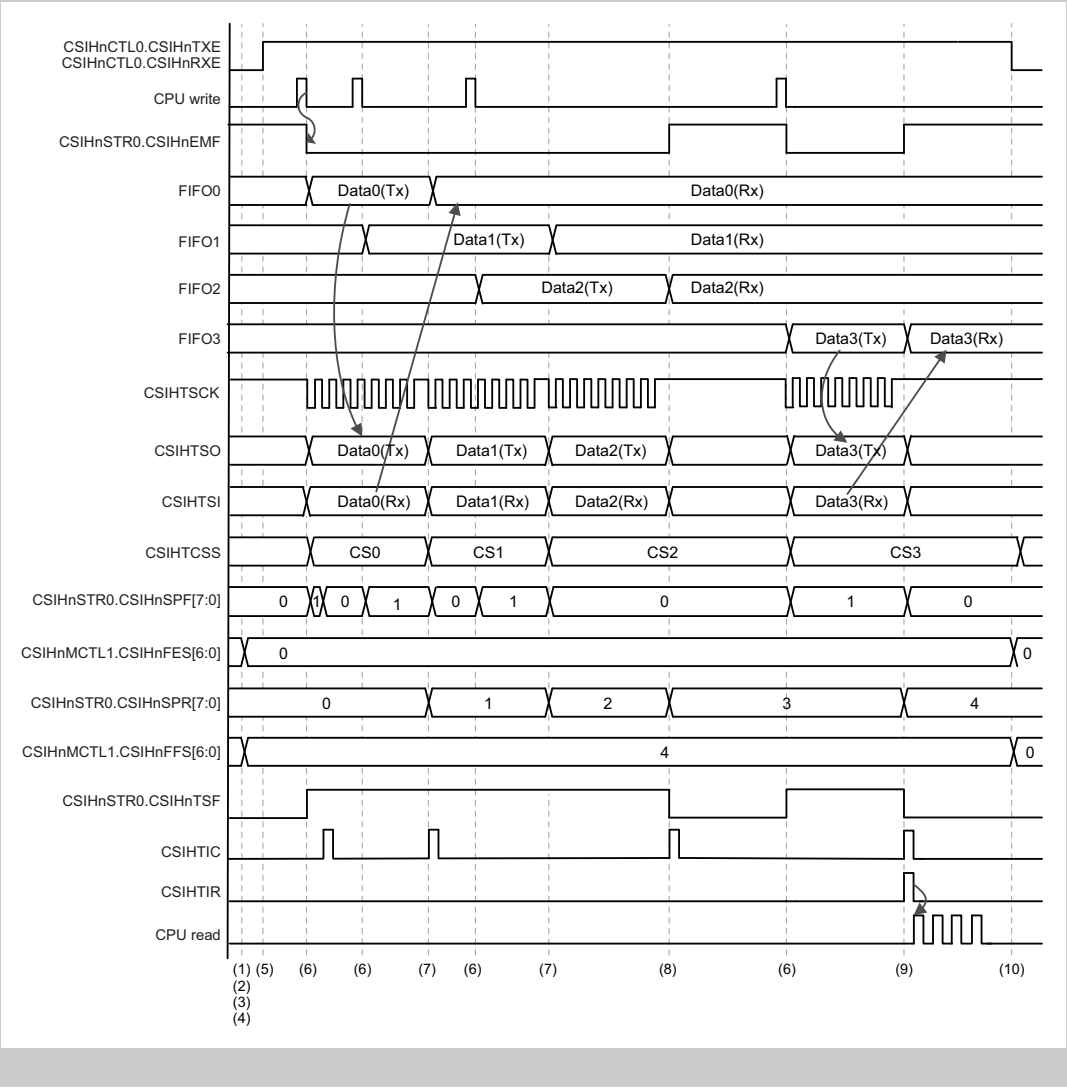


Figure 28-48 Master in FIFO mode, CSIHnCTL1.CSIHnJE = 0

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. This example uses the chip select signals CS0 to CS3.
2. Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
3. Set CSIHnMCTL0.CSIHnMMS[1:0] = 00_B (FIFO mode).
   Set bit CSIHnSTCR0.CSIHnPCT = 1 to clear all buffer pointers.
4. Specify in CSIHnMCTL1.CSIHnFES[6:0] the conditions for the CSIHTIC interrupt and the FIFO empty flag (i.e. CSIHnSTR0.CSIHnEMF[6:0]). Specify in the same register with CSIHnFFS[6:0] the conditions for the CSIHTIR interrupt for the FIFO full flag (i.e. CSIHnSTR0.CSIHnFLF).
5. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable the reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
6. Write the first data to be sent to the transmission register CSIHnTX0. Transmission starts automatically when the first data is available. Since Data0 is immediately sent and CSIHnSTR0.CSIHnSPF[7:0] is decreased to 0 and thus equals CSIHnMCTL1.CSIHnFES[6:0], the interrupt request CSIHTIC is asserted.
7. The first transmission is complete.
   Since CSIHnSTR0.CSIHnSPF[7:0] is decreased to 0 and thus equals CSIHnMCTL1.CSIHnFES[6:0], the interrupt request CSIHTIC is asserted.
8. Bit CSIHnSTCR0.CSIHnEMF is set to 1 when there is no further data to be transferred into the buffer.
   The interrupt request CSIHTIC is generated because CSIHnFES[6:0] = CSIHnSPF[7:0].
9. The interrupt request CSIHTIR is generated because CSIHnFFS[6:0] = CSIHnSRP[7:0].
   The interrupt request CSIHTIC is generated because CSIHnFES[6:0] = CSIHnSPF[7:0].
   The CPU starts to read the received data that are stored in the buffer.
10. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

(2) Master in transmit/receive mode, job mode enabled

The following instructions are based on the assumption that:

- Transmission data length is 8-bits (CSIHnCFGx.CSIHnDLSx[3:0] = 1000_B
- MSB is transmitted first (CSIHnCFGx.CSIHnDIRx = 0)
- Normal clock and data phase (CSIHnCFGx.CSIHnCKPx = 0, CSIHnCFGx.CSIHnDAPx = 0)
- No general interrupt delay (CSIHnCTL1.CSIHnSIT = 0)
- Job mode is enabled (CSIHnCTL1.CSIHnJE = 1)
- JOB1 = 4 data, JOB2 = 3 data, JOB3 = 5 data

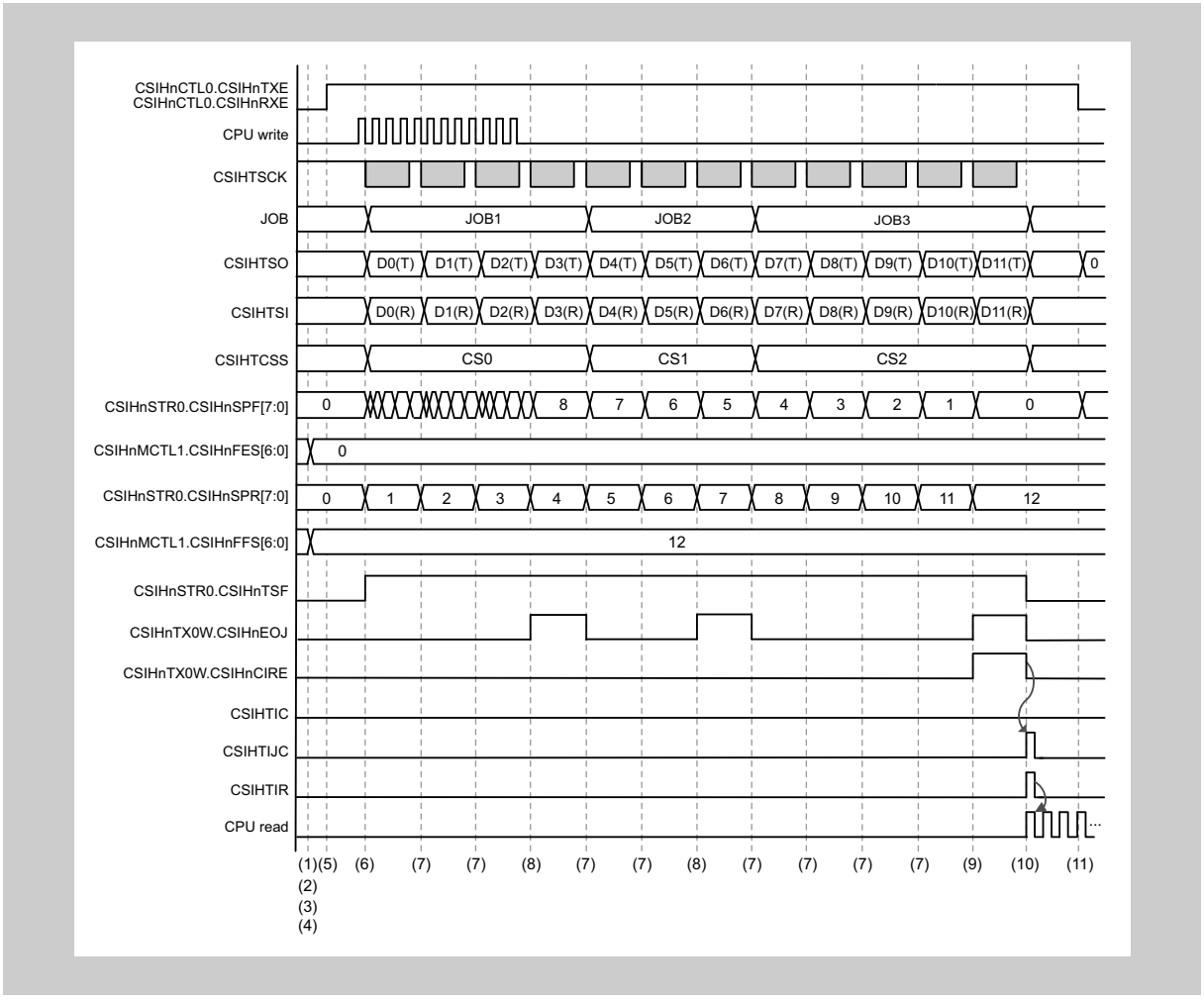


Figure 28-49 Master in FIFO mode, CSIHnCTL1.CSIHnJE = 1

Procedure:

1. Configure the communication protocol in the registers CSIHnCFGx and the baudrate via the CSIHnCTL2 and CSIHnCFGx registers. The example uses the chip select signals CS0 to CS2.
2. Select the memory mode by setting CSIHnMCTL0.CSIHnMMS[1:0] = 00_B. Set the FIFO write pointer CSIHnMRWP0.CSIHnTRWA[6:0] to the first data address 00_H and write the transmit data to CSIHnTX0W. The pointer CSIHnMRWP0.CSIHnTRWA[6:0] is automatically incremented after each write.
   Since data 3, 6 and 11 shall indicate an end-of-job, the CSIHnTX0W.CSIHnEOJ bit of these data must be set to 1.
   Specify the transfer and operating mode by setting the corresponding bits in registers CSIHnCTL1 and CSIHnCTL2.
3. Set bit CSIHnSTCR0.CSIHnPCT = 1 to clear all buffer pointers.
4. Specify the generation conditions of the interrupt request CSIHTIC in (CSIHnMCTL1.CSIHnFES[6:0]) and interrupt request CSIHTIR in (CSIHnMCTL1.CSIHnFFS[6:0]).
5. In the CSIHnCTL0 register set the bits CSIHnPWR = 1 (enable the clock), CSIHnTXE = 1 (enable transmission) and CSIHnRXE = 1 (enable reception). The bit CSIHnCTL0.CSIHnMBS must be cleared.
6. Write the first data to be sent to the transmission register CSIHnTX0. Transmission starts automatically when the first data is available.
7. The current transmission is completed.
8. Setting CSIHnTX0W.CSIHnEOJ = 1 specifies that the last data of the current job are sent.
   The interrupt request CSIHTIC is not generated, because the last data of the current job (CSIHnTX0W.CSIHnEOJ = 1) was sent with CSIHnTX0W.CSIHnCIRE = 0.
9. The final data D11(T) is started to be transferred with EOJ = CIRE = 1.
10. After the last data was transmitted, the interrupt requests CSIHTIJC and CSIHTIR are generated at the end of JOB3.
    The interrupt request CSIHTIC is not generated because the interrupt request CSIHTIJC is generated instead. Additionally, the transmission data available in register CSIHnTX0H is not sent.
11. To finally disable the transmit/receive operation, clear CSIHnCTL0.CSIHnTXE and CSIHnCTL0.CSIHnRXE.

Chapter 29 I^2C Interface (IICB)

This chapter contains a generic description of the I^2C Interface (IICB).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

29.1 V850E2/Fx4-H IICB Features

Instances

This microcontroller has following number of instances of the I^2C Interface.

Table 29-1 Instances of IICB

I^2C Interface	
Instances	1
Names	IICB0

Instances index n

Throughout this chapter, the individual instances of a I^2C Interface are identified by the index “n” (n = 0), for example, IICBnCTL0 for the IICBn control register 0.

Register addresses

All IICBn register addresses are given as address offsets from the individual base address <IICBn_base>.
The base address <IICBn_base> of each IICBn is listed in the following table:

Table 29-2 Register base addresses <IICBn_base>

IICBn instance	<IICBn_base> address
IICB0	FF82 0000_H

Clock supply All I^2C Interfaces provide one clock input:

Table 29-3 IICB clock supply

IICBn instance	Clock	Connected to
IICB0	PCLK	Clock Controller CKSCLK_108

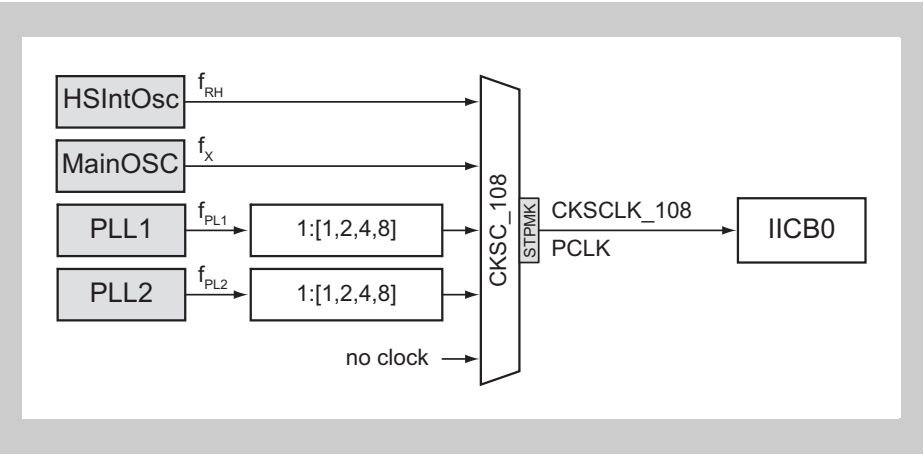


Figure 29-1 IICB clock supply

Interrupts and DMA The I^2C Interfaces can generate following interrupt and DMA requests:

Table 29-4 IICB interrupt and DMA requests

IICBn signals	Function	Connected to
IICB0:		
IICBTIA	Data transmission/reception interrupt	Interrupt Controller INTIICB0IA DMA Controller trigger 57
IICBTIS	Status interrupt	Interrupt Controller INTIICB0IS

IICB H/W reset The I^2C Interfaces and their registers are initialized by the following reset signal:

Table 29-5 IICBn reset signal

IICBn	Reset signal
IICBn	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the I^2C Interfaces are listed in the following table:

Table 29-6 IICB I/O signals

IICBn signals	Function	Connected to
IICB0:		
SCL	IICB0 clock signal	Port IICB0SCL
SDA	IICB0 data/address signal	Port IICB0SDA

29.2 I^2C Interface Port Settings

The I^2C interface function requires to define the ports for the IICBnSCL and IICBnSDA signals as input and open drain output pins simultaneously.

In the following table the port configuration register bits are listed to be set up properly for port m of the port group n, if it shall be used for I^2C interface signals IICBnSCL or IICBnSDA.

The bold printed values in grey cells differ from the initial register values after reset release and thus have to be changed.

Table 29-7 I^2C Interface port settings

Register		Value to set	Initial value	Function
Port function configuration registers:				
PMCn_m	=	**1**	0	alternative mode
PIPCn_m	=	0	0	S/W I/O control
PMn_m	=	**0**	1	output mode
PIBCn_m	=	x	0	overruled by PMCn_m = 1
PFCn_m, PFCEn_m	=	**1 or 0**	0	select correct alternative function
Port data input/output registers:				
PBDCn_m	=	**1**	0	bi-directional I/O
Configuration of electrical characteristics registers:				
PUn_m	=	x	0	no pull-up resistor, overruled by PMn_m = 0
PDn_m	=	x	0	no pull-down resistor, overruled by PMn_m = 0
PDSCn_m	=	**1 or 0**	0	output drive strength control can be selected based on capacitive load.
PODCn_m	=	**1**	0	open drain output
PISn_m, PISEn_m	=	**1 or 0**	0	input characteristic can be selected based on application signal quality

29.3 Functional Overview

Operating mode	Standard mode (SCL clock frequency: 100 kHz max.) Fast mode (SCL clock frequency: 400 kHz max.)
Transfer mode	Single transfer mode Continuous transfer mode
Pin configuration	SCLn: Serial clock pin SDAn: Serial transmit/receive data pin
Interrupt request signal	Data transmit/receive interrupt request signal (IICBTIAn) Status interrupt request signal (IICBTISn)
Communication data length	8 bits
Multimaster support	Multiple masters can control the bus simultaneously.
SCLn level width	The high-level width and low-level width of the serial clock signal (SCLn) can be changed.
Automatic detection	The start and stop conditions can be detected automatically.

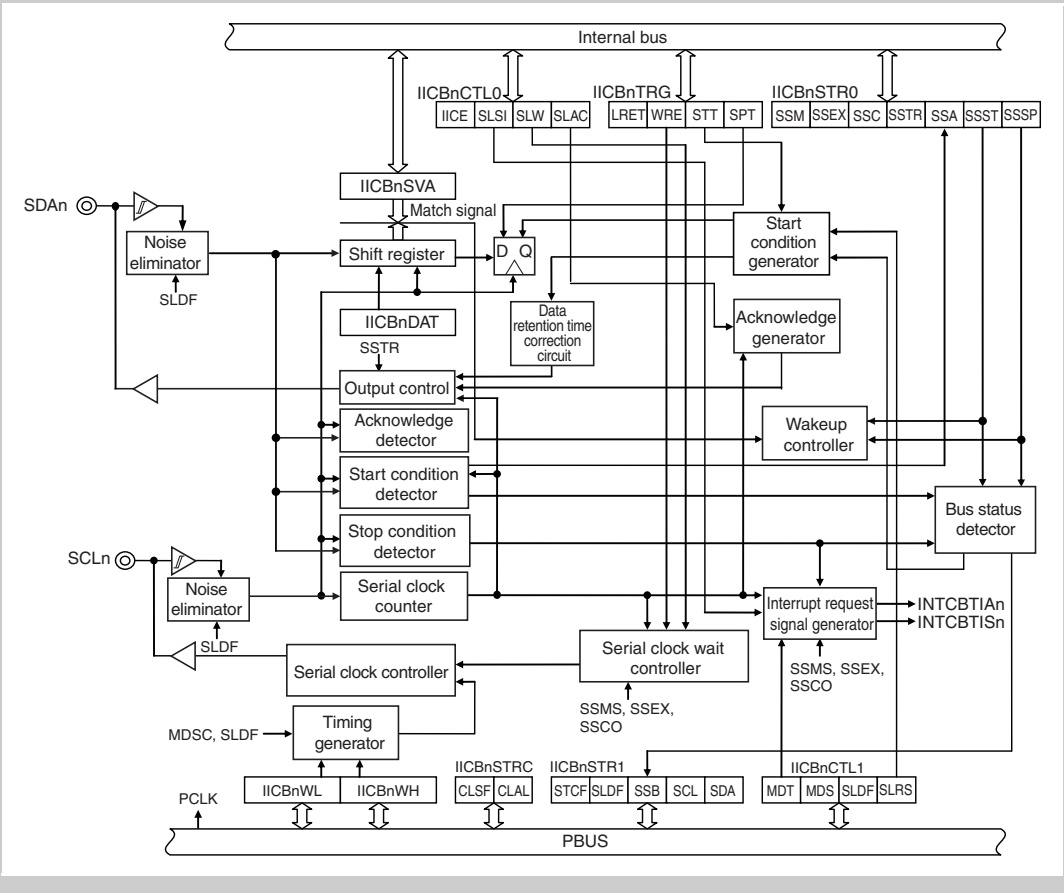


Figure 29-2 Block diagram of IICBn

29.4 I²C Bus Mode Functions

29.4.1 Pin configuration

The serial clock pin (SCLn) and serial data bus pin (SDAn) are configured as follows.

SCLn... This pin is used for serial clock input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

SDAn... This pin is used for serial data input and output.
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Because the outputs of the serial clock line and serial data bus line are N-ch open-drain outputs, an external pull-up resistor must be connected to these lines.

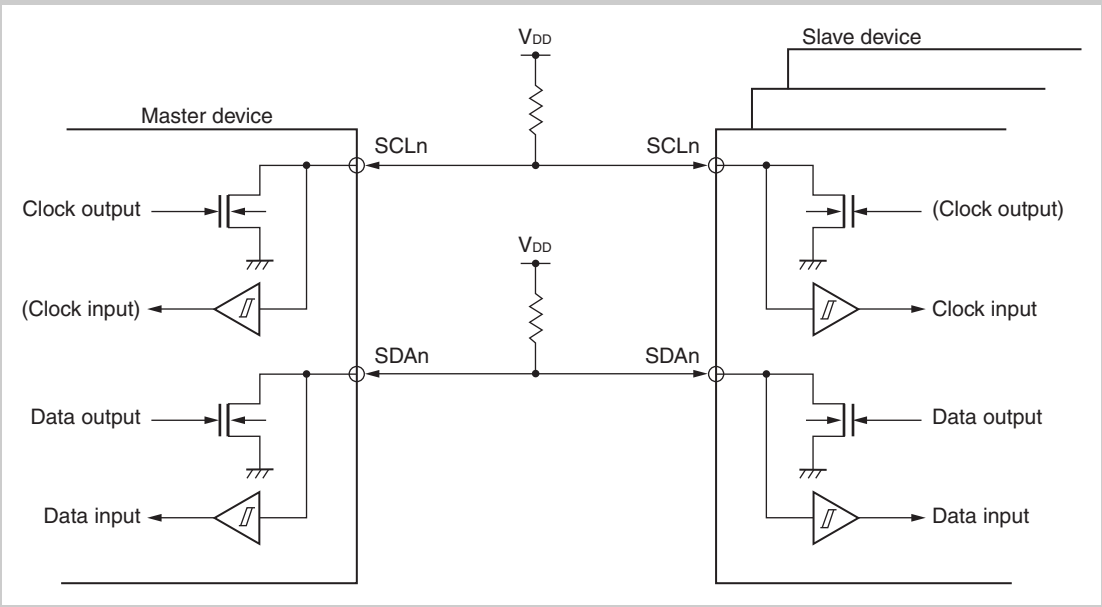


Figure 29-3 Pin configuration diagram

29.5 I²C Bus Definition

This section describes the I²C bus's serial data communication format and the signals used by the I²C bus.

Figure 29-4 "I²C bus serial data transfer timing" shows the transfer timing for the "start condition", "address", "transfer direction specification", "data", and "stop condition", which are output onto the I²C bus's serial data bus.

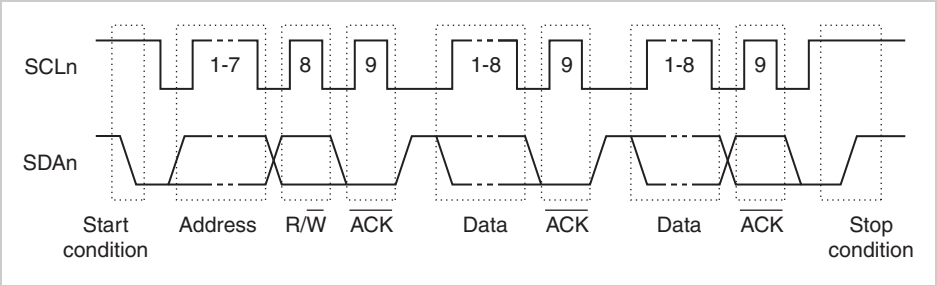


Figure 29-4 I²C bus serial data transfer timing

The start condition, slave address, and stop condition are output by the master device.

$\overline{ACK}$ can be output by either the master or slave device. (Normally, it is output by the device that receives 8-bit data.)

The serial clock signal (SCLn) is continuously output by the master device. In the slave device, the low-level period of the SCLn signal can be extended to insert a wait.

29.5.1 Start Condition

The start condition is met if the SDAn signal level changes from high to low while the SCLn signal is high. The start condition is output when the master device starts serial data transfer to a slave device. When the IICBn is in the slave mode, it detects the start condition.

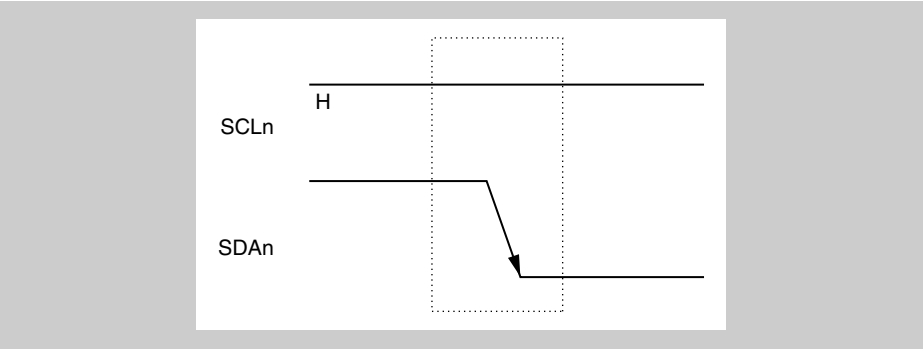


Figure 29-5 Start condition

29.5.2 Addresses

The 7 bits of data following the start condition are defined as an address.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave device checks whether the 7-bit data matches its own address. If they match, that slave device is selected as the communication destination and communicates with the master device until the master device outputs another start condition or a stop condition.

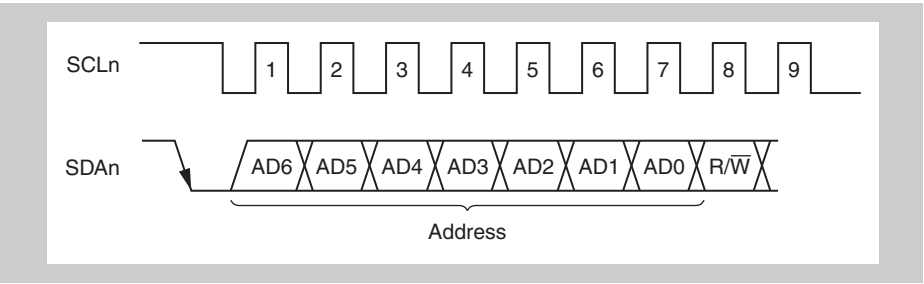


Figure 29-6 Addresses

29.5.3 Extension code

When the higher 4 bits of the address are 0000 or 1111, these bits are called extension code. *Table 29-8 "Extension code bit definitions"* lists the bit definitions of extension code.

Table 29-8 Extension code bit definitions

Slave address	R/$\overline{W}$ bit	Description
0000 000	0	General call address
0000 000	1	Start byte
0000 001	×	CBUS address
0000 010	×	Address reserved for different bus format
0000 011	×	Reserved for future use
0000 1xx	×	HS mode master code[a]
1111 0xx	×	10-bit slave address specification
1111 1xx	×	Reserved for future use

a) The HS mode cannot be used with this IICB module.

29.5.4 Transfer direction specification

After the 7-bit address data, the master device transmits 1 bit that specifies the transfer direction.

If this transfer direction specification bit is 0, it indicates that the master device transmits data to a slave device. If this bit is 1, it indicates that the master device receives data from a slave device.

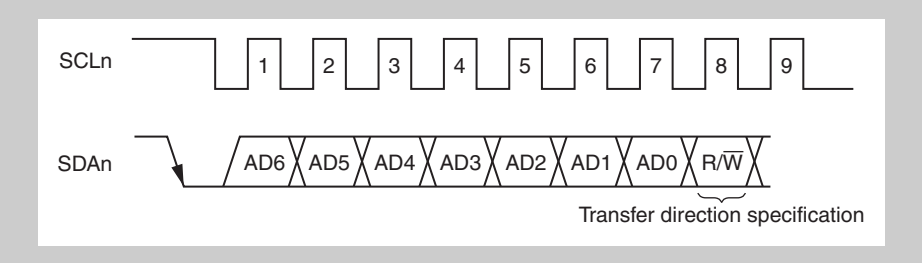


Figure 29-7 Transfer direction specification

29.5.5 Acknowledge ($\overline{ACK}$)

The 1-bit data after the transfer direction bit (R/$\overline{W}$) and the 1-bit data after the 8-bit data during address transfer are defined as an acknowledge signal ($\overline{ACK}$). $\overline{ACK}$ is used to check the serial data status of the transmitting and receiving devices.

The receiving device returns $\overline{ACK}$ after receiving 8-bit data.

The transmitting device normally receives $\overline{ACK}$ after transmitting 8-bit data. If the transmitting device receives $\overline{ACK}$ from the receiving device, it continues processing assuming that the transmitted data is normally received.

If the master device is the receiving device and receives the final data, it does not return $\overline{ACK}$ and outputs a stop condition. If the slave device is the receiving device and does not return $\overline{ACK}$, the master device outputs either a stop condition or a restart condition and then stops the current transmission. Failure to return $\overline{ACK}$ may be caused by the following factors:

- The transmitted data has not been received normally.
- The final data has been received.
- The receiving device (slave) does not exist for the specified address.

$\overline{ACK}$ is output when the SDAn line of the receiving device changes to low level at the 9th clock (normal reception).

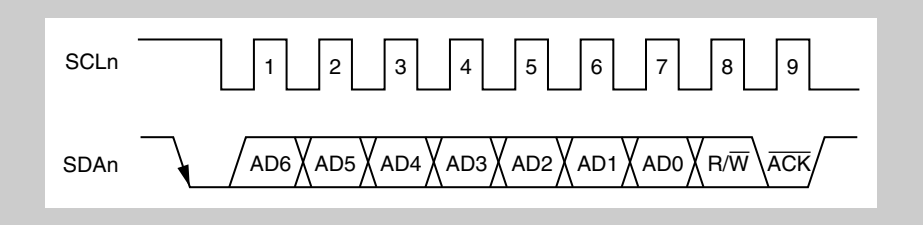


Figure 29-8 Acknowledge ($\overline{ACK}$)

29.5.6 Data

Except for a 9-bit data string (consisting of a 7-bit address, 1-bit R/$\overline{W}$, and 1-bit $\overline{ACK}$) transferred after the start condition, the bits other than $\overline{ACK}$ are defined as data.

If a 10-bit address is specified using an extension code, the 8-bit data that is transferred after the address is used as the second address.

29.5.7 Stop condition

A stop condition is met if the SDAn signal level changes from low to high while the SCLn signal is high.
The stop condition is output when serial data transfer from the master device to the slave device has been completed.

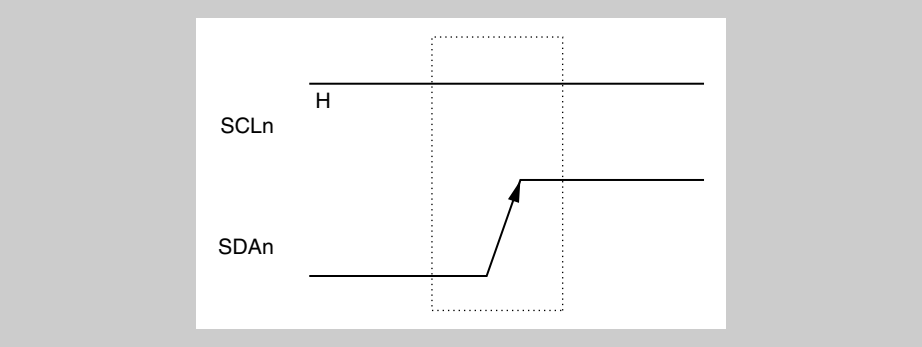


Figure 29-9 Stop condition

29.5.8 Wait state

A wait state is used to report to the communication destination that the IICBn (master or slave) is preparing to transmit or receive data.

The wait state is reported to the communication destination by setting the SCLn signal to low. The next data transfer cannot start until both the master and slave devices exit the wait state.

(a) When a wait at the 9th clock is specified for the master and a wait at the 8th clock for the slave (master: transmission, slave: reception)

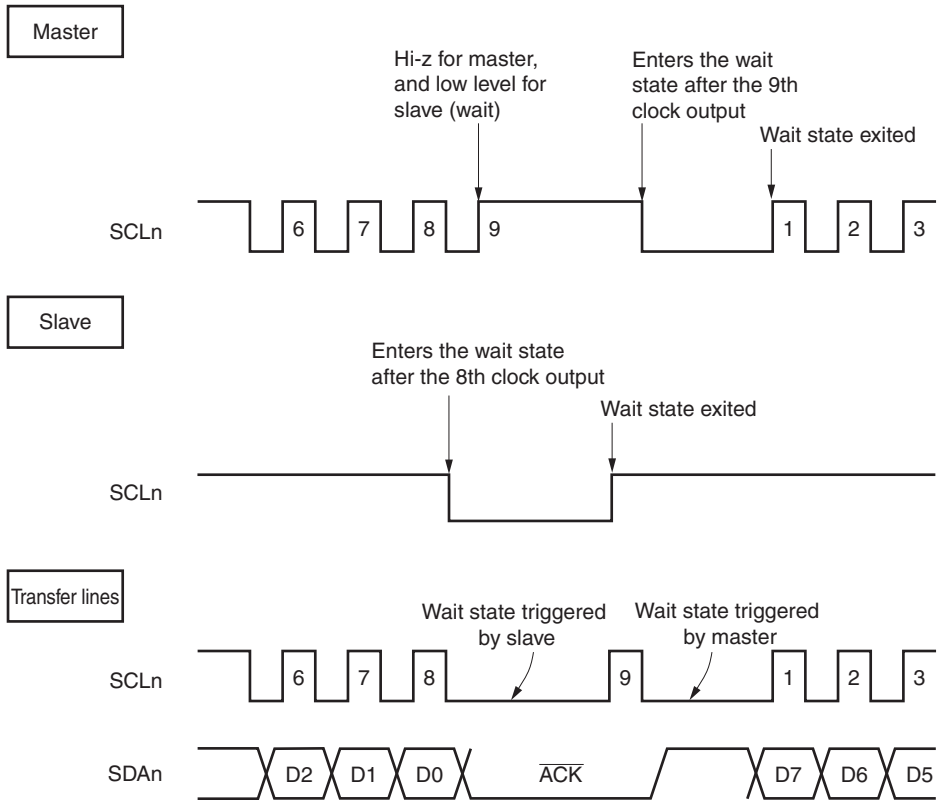


Figure 29-10 Wait state (1/2)

(b) When a wait at the 9th clock is specified for both the master and slave
(master: transmission, slave: reception)

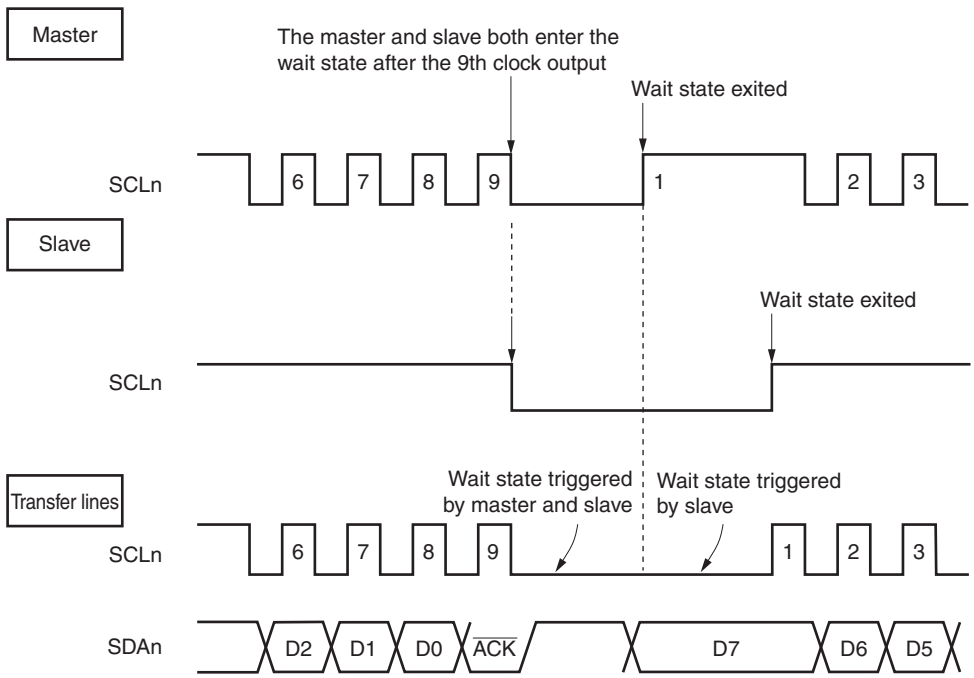


Figure 29-10 Wait state (2/2)

29.5.9 Arbitration

When several master devices simultaneously output a start condition, communication with the master devices continues until the data differs, while adjusting the clocks. An example where two masters simultaneously output a start condition and arbitration is conducted is described below.

This example assumes that one master outputs the SDAn line high (master 1) and the other master outputs the SDAn line low (master 2) while the SCLn line is low.
In this case, the communication with master 2 is prioritized, and communication is not authorized for master 1.

This kind of operation is called arbitration, and the state in which communication is not authorized is called arbitration loss. The master that lost arbitration releases the bus by setting both the SCLn and SDAn line to high impedance.

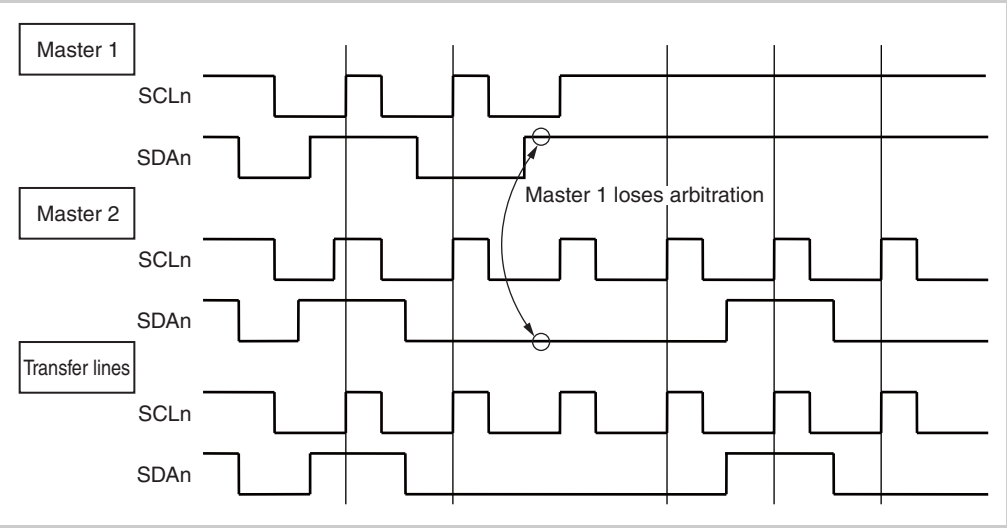


Figure 29-11 Arbitration timing example

29.6 Registers

Caution In this section, the operation when an extension code is received is omitted. For details about the extension code, refer to *29.7.5 "Extension code"*.

(1) IICBnDAT – IICBn data register

This register is used to transmit and receive transfer data.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 0000_H

Initial Value 0000_H. This register is also initialized by changing the value of the IICBnCTL0.IICBnIICE bit from 1 to 0 or from 0 to 1.

Cautions

1. When the IICBn becomes a master in the single transfer mode or continuous transfer mode, after the IICBnTRG.IICBnSTT bit has been set to 1, writing to the IICBnDAT register is allowed only once to transfer the address and communication direction.
2. When transferring data in the single transfer mode, writing to the IICBnDAT register in communication state other than the wait state is prohibited.
3. When transferring data in the continuous transfer mode, writing to the IICBnDAT register in response to an INTIICBTIAn interrupt request signal is only allowed once.
4. When executing transmission operations in the continuous transfer mode, do not read the IICBnDAT register.
   Similarly, when performing reception operations in the continuous transfer mode, do not write to the IICBnDAT register.

7	6	5	4	3	2	1	0
IICBnDAT7	IICBnDAT6	IICBnDAT5	IICBnDAT4	IICBnDAT3	IICBnDAT2	IICBnDAT1	IICBnDAT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 29-9 IICBnDAT register contents

Bit position	Bit name	Function
7 to 0	IICBnDAT[7:0]	During reception, these bits hold the received data. During transmission, these bits write the transmit data. The prescribed procedure must be followed during access (read, write) to the IICBnDAT register. For the setting sequence, refer to *29.10 "Setting Sequence"*. The IICBn exits the wait state by performing access to the IICBnDAT register. • In single transfer mode - When write access to the IICBnDAT register is performed • In continuous transfer mode - When write access to the IICBnDAT register is performed - When read access to the IICBnDAT register is performed during a wait state for data transfer that is not triggered by NACK signal reception

(2) IICBnSVA – IICBn slave address register

This register stores the slave address of the I^2C bus.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 0004_H

Initial Value 0000_H

Caution Write access to the IICBnSVA register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

7	6	5	4	3	2	1	0
IICBnSVA7	IICBnSVA6	IICBnSVA5	IICBnSVA4	IICBnSVA3	IICBnSVA2	IICBnSVA1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R

Table 29-10 IICBnSVA register contents

Bit position	Bit name	Function
7 to 1	IICBnSVA[7:1]	Store the slave address of the I^2C bus. Address match/address mismatch is judged by comparing the received address and the IICBnSVA register. If the received address matches the IICBnSVA register, the IICBnSTR0.IICBnSSCO bit is set to 1.

(3) IICBnCTL0 – IICBn control register 0

This register is used to control the operations of the IICBn.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 0008_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
IICBn IICE	0	0	IICBn MDTX1	IICBn MDTX0	IICBn SLSI	IICBn SLWT	IICBn SLAC
R/W	R	R	R/W	R/W	R/W	R/W	R/W

Table 29-11 IICBnCTL0 register contents (1/3)

Bit position	Bit name	Function
7	IICBnIICE	Enables/disables operation of the IICBn. 0: Disables operation of IICBn. 1: Enables operation of IICBn. Synchronous reset of the following registers is executed when the value of the IICBnIICE bit changes from 1 to 0, or the value of the IICBnIICE bit changes from 0 to 1. • IICBnDAT and IICBnSTR0 registers When IICBnIICE is 0, the SCLn and SDAn pins go into the high impedance state.
4	IICBnMDTX1	Specifies the transfer mode upon detection of expansion code in the slave. 0: Single transfer mode 1: Continuous transfer mode • Single transfer mode The IICBn enters a wait state after each transfer according to the setting of the IICBnSLWT bit. • Continuous transfer mode The IICBn performs continuous communication without entering a wait state when the IICBnDAT register is read or written upon the output of the data transmit/receive interrupt request signal (INTIICBTIAn). For the operation in each mode, refer to *29.7 "Operation"*. **Caution** Rewrite is allowed only when IICBnIICE is 0.
3	IICBnMDTX0	Specifies the transfer mode when the address matches between the master and slave. 0: Single transfer mode 1: Continuous transfer mode • Single transfer mode The IICBn enters a wait state after each transfer according to the setting of the IICBnSLWT bit. • Continuous transfer mode The IICBn performs continuous communication without entering a wait state when the IICBnDAT register is read or written upon the output of the data transmit/receive interrupt request signal (INTIICBTIAn). For the operation in each mode, refer to *29.7 "Operation"*. **Caution** Rewrite is allowed only when IICBnIICE is 0.

Table 29-11 IICBnCTL0 register contents (2/3)

Bit position	Bit name	Function
2	IICBnSLSI	Enables/disables status interrupt request signal (INTIICBTISn) output when a stop condition is detected. 0: Disables INTIICBTISn signal output when stop condition is detected. 1: Enables INTIICBTISn signal output when stop condition is detected. Set this bit to 1 when performing the following types of communication. - When the IICBn performs communication as a master while the communication reserve function is enabled - When the IICBn participates in communications as a slave - When the IICBn may lose in arbitration (when making the IICBn operate as a master in a multi-master environment))
1	IICBnSLWT	Controls a wait and interrupt request output timing. 0: The IICBn enters the wait state and an interrupt request is output at the falling edge of the 8th clock during single transfer. 1: The IICBn enters the wait state and an interrupt request is output at the falling edge of the 9th clock during single transfer. The IICBnSLWT bit controls wait state transition and interrupt request output at the following timing. - 8th and 9th clocks during data transfer For the conditions for transition to the wait state, refer to *29.7.4 "Entering and exiting wait state"*. During address transfer, the conditions for transiting to the wait state and for interrupt request output are as follows, regardless of the setting of the IICBnSLWT bit. • In single transfer mode - Master: A data transmit/receive interrupt request signal (INTIICBTIAn) is output and the IICBn enters the wait state upon detection of the falling edge of the 9th clock. - Slave: During an address match, the INTIICBTIAn signal is output and the IICBn enters the wait state upon detection of the falling edge of the 9th clock. During address mis-match, the INTIICBTIAn signal is not output and the IICBn does not enter the wait state. • In continuous transfer mode In the continuous transfer mode, transition to the wait state is not affected by the setting of the IICBnSLWT bit. - Reception: The IICBn enters the wait state at the falling edge of the 8th clock. - Transmission: The IICBn enters the wait state at the falling edge of the 9th clock. **Caution** During data transmission, rewriting the IICBnSLWT bit is prohibited.

Table 29-11 IICBnCTL0 register contents (3/3)

Bit position	Bit name	Function
0	IICBnSLAC	Controls acknowledge signal output. 0: Disables acknowledge signal output. Master:The acknowledge signal is not output during data reception (SDAn = “H”). Slave:The acknowledge signal is not output during data transfer when an address match occurs (SDAn = “H”). 1: Enables acknowledge signal output. Master:The acknowledge signal is output during data reception (SDAn = “L”). Slave:The acknowledge signal is output during data transfer when an address match occurs (SDAn =“L”). When the IICBn is operating as a slave, in the case of an address match, an acknowledge signal is output during address transfer regardless of the value of the IICBnSLAC bit (SDAn = “L”). also, no acknowledge signal is output (SDAn = “H”) while the IICBn is transmitting data or when it does not participate in communications.

(4) IICBnCTL1 – IICBn control register 1

This register controls the operation of the IICBn.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 0020_H

Initial Value 0000_H

Caution Write access to the IICBnCTL1 register is prohibited when the value of the IICBnCTL0.IICBnIICE is 1.

7	6	5	4	3	2	1	0
IICBn MDSC	IICBn LGDF2	IICBn LGDF1	IICBn LGDF0	0	0	IICBn SLSE	IICBn SLRS
R/W	R/W	R/W	R/W	R	R	R/W	R/W

Table 29-12 IICBnCTL1 register contents (1/2)

Bit position	Bit name	Function
7	IICBnMDSC	Specifies the operation mode for the IICBn. 0: Standard mode (SCL clock frequency: 100 kbps max.) 1: Fast mode (SCL clock frequency: 400 kbps max.)
6 to 4	IICBnLGDF[2:0]	Specify the digital filter sampling frequency. Note that the digital filter can be used only in the fast mode. 000:Does not use digital filter. SCLn and SDAn are used without passing through the digital filter in the IICBn. The digital filter circuit operations are stopped. Other than above:Uses digital filter. SCLn and SDAn are used passing through the digital filter in the IICBn. When using a digital filter, set bits IICBnLGDF2 to IICBnLGDF0 as follows.

IICBnLGDF2 to IICBnLGDF0 bits	Frequency
001	Minimum frequency [a] ≤ PCLK ≤ 20 MHz
010	20 MHz < PCLK ≤ 40 MHz
011	40 MHz < PCLK ≤ 60 MHz
100	60 MHz < PCLK ≤ 80 MHz
101	80 MHz < PCLK ≤ 100 MHz
110, 111	Setting prohibited

a) A list of the minimum frequencies by setting is shown below.

Operation mode (IICBnMDSC)	When digital filter used (IICBnLGDF bits = 000)	When digital filter used (IICBnLGDF bits ≠ 000)
Standard mode (0)	1.0 MHz	Use prohibited
Fast mode (1)	3.5 MHz	4.0 MHz

Table 29-12 IICBnCTL1 register contents (2/2)

Bit position	Bit name	Function
1	IICBnSLSE	Enables/disables start condition output in the initial communication state. 0: Disables start condition output in the initial communication state. 1: Enables start condition output in the initial communication state. If the IICBnSLSE bit is set to 1, a start condition can be output by setting the IICBnTRG.IICBnSTT bit to 1 in the initial communication state (from when the IICBnCTL0.IICBnIICE bit is set to 1 until detection of a stop condition). The IICBnSLSE bit is automatically cleared to 0 upon detection of a start condition (even without a 0 write operation). **Caution** Clear the IICBnSLSE bit to 0 when participating in communications after other communications have started. When other communications are being performed, if the IICBnSTT has been set to 1 with the IICBnSLSE bit set to 1, the other communications may be damaged.
0	IICBnSLRS	Enables/disables the communication reserve function. 0: Enables communication reserve function. 1: Disables communication reserve function. Communication reserve function enabled state: If the IICBnSLRS bit is cleared to 0 while the IICBn is not operating as a master, the communication reserve state can be set by setting the IICBnSTT bit to 1 while the bus is being used. Whether the communication reserve state is set can be confirmed by checking the IICBnSTR0.IICBnSSRS bit. Communication reserve function disabled state: If the IICBnSTT bit is set to 1 while the IICBn is not participating in communications as a master and the bus is being used, the value of the IICBnSTR0.IICBnSTCF becomes 1 and communication reservation is not done.

(5) IICBnWL – IICBn low level width setting register

This register is used to set the low level width of the serial clock register (SCLn).

Access This register can be read/written in 16-bit units.

Address <IICBn_base> + 0024_H

Initial Value $03FF_H$

Caution Write access to the IICBnWL register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	IICBnWL9	IICBnWL8
R	R	R	R	R	R	R/W	R/W

7	6	5	4	3	2	1	0
IICBnWL7	IICBnWL6	IICBnWL5	IICBnWL4	IICBnWL3	IICBnWL2	IICBnWL1	IICBnWL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 29-13 IICBnWL register contents

Bit position	Bit name	Function
9 to 0	IICBnWL[9:0]	Specify the t_{LOW} period (low level width of the SCLn clock) of the I^2C bus specification. The value of the IICBnWL register is used to determine the serial output timing of other I^2C bus specifications. For the serial output timing setting conditions, refer to *a "Setting transfer clock by using IICBnWL and IICBnWH registers" on page 2254* .

(a) Setting transfer clock by using IICBnWL and IICBnWH registers

The various timings in compliance with the I^2C bus specifications can be set by setting the IICBnWL register and IICBnWH register.

- Setting transfer clock on master side

$$\text{Transfer clock (Hz)} = \frac{\text{PCLK}}{\text{IICBnWL} + \text{IICBnWH} + \text{PCLK}\ (t_R + t_F)}$$

At this time, the optimal setting values of IICBnWL and IICBnWH are as follows.
(The fractional parts of all setting values are rounded up.)

– When the fast mode

$$\text{IICBnWL} = \frac{0.52}{\text{Transfer clock}} \times \text{PCLK}$$

$$\text{IICBnWH} = \left(\frac{0.48}{\text{Transfer clock}} - t_R - t_F\right) \times \text{PCLK}$$

– When the normal mode

$$\text{IICBnWL} = \frac{0.47}{\text{Transfer clock}} \times \text{PCLK}$$

$$\text{IICBnWH} = \left(\frac{0.53}{\text{Transfer clock}} - t_R - t_F\right) \times \text{PCLK}$$

Caution The data hold time must be 0.9 µs or less in the fast mode and 3.45 µs or less in the standard mode.

Note The data hold time is determined by the IICBWL register setting as follows:

Data hold time = IICBnWL.IICBnWL[9:2] / PCLK

- Setting IICBnWL and IICBnWH on slave side
  (The fractional parts of all setting values are rounded up.)

  – When the fast mode

    IICBnWL = 1.3 µs × PCLK

    IICBnWH = (1.2 µs – t_R – t_F) × PCLK

  – When the normal mode

    IICBnWL = 4.7 µs × PCLK

    IICBnWH = (5.3 µs – t_R – t_F) × PCLK

Note IICBnWL: IICBn low-level width setting register
IICBnWH: IICBn high-level width setting register
t_F: SDAn and SCLn signal falling times
t_R: SDAn and SCLn signal rising times
PCLK: Frequency of the clock supplied to the IICBn
f_{CLK}: SCL clock frequency

(6) IICBnWH – IICBn high-level width setting register

This register is used to set the high level width of the serial clock signal (SCLn).

Access This register can be read/written in 16-bit units.

Address <IICBn_base> + 0028_H

Initial Value $03FF_H$

Caution Write access to the IICBnWH register is prohibited when the value of the IICBnCTL0.IICBnIICE bit is 1.

15	14	13	12	11	10	9	8
0	0	0	0	0	0	IICBnWH9	IICBnWH8
R	R	R	R	R	R	R/W	R/W

7	6	5	4	3	2	1	0
IICBnWH7	IICBnWH6	IICBnWH5	IICBnWH4	IICBnWH3	IICBnWH2	IICBnWH1	IICBnWH0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 29-14 IICBnWH register contents

Bit position	Bit name	Function
9 to 0	IICBnWH[9:0]	Specify the t_{HIGH} period (high level width of the SCLn clock) of the I^2C bus specification. The value of the IICBnWH register is used to determine the serial output timing of other I^2C bus specifications. For the serial output timing setting conditions, refer to *a "Setting transfer clock by using IICBnWL and IICBnWH registers" on page 2254* .

(7) IICBnTRG – IICBn trigger register

This register is used to set the IICBn trigger.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + $000C_H$

Initial Value 0000_H

7	6	5	4	3	2	1	0
0	0	0	0	IICBn LRET	IICBn WRET	IICBn STT	IICBn SPT
R	R	R	R	R/W	R/W	R/W	R/W

Table 29-15 IICBnTRG register contents (1/4)

Bit position	Bit name	Function
3	IICBnLRET	Communication exit trigger bit 0: Normal operation 1: The IICBn exits the current communication and enters the wait state. This bit is automatically cleared to 0 following execution. The following occurs when IICBnLRET is 1. - SCLn and SDAn each go into high impedance (communication wait state). - Bits IICBnSSMS, IICBnSSDR, IICBnSSWT, IICBnSSEX, IICBnSSC0, IICBnSSTR, IICBnSSAC, IICBnSSRS, and IICBnSSST of the IICBnSTR0 register are cleared to 0. - When IICBnSTT = 1 (start condition output preparation) or IICBnSPT =1 (stop condition output preparation) has been set, output of a start condition or stop condition is stopped. The communication reserved state is released if the IICBn exits the communication in the communication reserved state. If it is necessary for the IICBn to operate a master again after this, the IICBnSTT bit must be set to 1 again. **Caution** If IICBnLRET is set to 1 during master operation (IICBnSTR0.IICBnSSMS = 1), the bus is released. Since serial clock output stops, problems occur during communication on the slave side.
2	IICBnWRET	This is the trigger bit for exiting the wait state. 0: Does not exit the wait state. 1: Exits the wait state and resumes communication. This bit is automatically cleared following execution. If the IICBn have exited the wait state by setting the IICBnWRET bit to 1 during the wait state triggered by the falling edge of the 9th clock, the IICBnSTR0.IICBnSSTR bit is cleared to 0 and SDAn goes into high impedance (this enables the external master to output a stop condition or start condition.) If the IICBn is not in the wait state (IICBnSTR0.IICBnSSWT = 0), setting this bit to 1 has no meaning. There are other conditions for exiting the wait state in addition to the setting of this bit. For details, refer to *29.7.4 "Entering and exiting wait state"*.

Table 29-15 IICBnTRG register contents (2/4)

Bit position	Bit name	Function
1	IICBnSTT	Start condition trigger bit 0: Does not output a start condition. 1: Outputs a start condition (This bit is automatically cleared to 0 after it has been set to 1. The IICBnSTT bit can be set to 1 under the following conditions: [1] IICBnSTR0.IICBnSSMS bit = Master state (1) • Single transfer mode - During wait state triggered by the falling edge of the 9th clock (both address transfer and data transfer) - During data reception, only after clearing the ACKEn bit to 0 to report the end of reception to the slave • Continuous transfer mode - During wait state triggered by the falling edge of the 9th clock of address transfer - During data transfer - During data reception, only after clearing the ACKEn bit to 0 to report the end of reception to the slave After exiting the wait state that was triggered by the falling edge of the 9th clock, and upon detection of the falling edge of the 9th clock in all other cases, SDAn and SCLn are set to high level after the t_{LOW} (low-level width of SCLn clock) period, and when SDAn is set to the low level after waiting for $t_{SU:STA}$ of the I^2C bus specification (setup time of start/restart condition), a start condition is output. Then, SCLn is set to low level after the $t_{HD:STA}$ time (hold time) of the I^2C bus specification has elapsed. [2] Slave state or communication wait state (IICBnSTR0.IICBnSSMS = 0) • IICBnSTR0.IICBnSSBS bit = 0 (bus release state) Outputs a start condition. Following verification of the t_{BUF} (bus free time (between stop/start conditions) of the I^2C specification (if this time has not elapsed, following lapse, and if this time has elapsed, immediately), a start condition is output when SDAn is changed from high level to low level while SCLn is high level. (At this time, SCLn outputs a high level.) Then, SCLn is set to low level after the $t_{HD:\ STA}$ time of the I^2C bus specification has elapsed. • IICBnSSBS bit = 1 (bus communication state) This status indicates that communication is performed on the bus while the IICBn is not operating as a master. - When communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0): A start condition is output after the bus has been released (stop condition has been detected) and the t_{BUF} time of the I^2C bus specification has elapsed. However, even if the t_{BUF} time has not elapsed, upon detection of a start condition, SDAn is immediately set to low level without waiting for the t_{BUF} time. - When communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1): The IICBnSTR0.IICBnSTCF bit is set to 1 and a start condition is not output. **Caution** [2] shows the operations according to the value of the IICBnSSBS bit when the IICBnSTT bit is 0. Even if the IICBnSTT bit is set to 1 after checking the value of the IICBnSSBS bit through register read, the value of IICBnSSBS may differ from its value when it was checked.

Table 29-15 IICBnTRG register contents (3/4)

Bit position	Bit name	Function
1	IICBnSTT	The output processing of the start condition is started by setting the IICBnSTT bit to 1, but upon detection of the following states, output processing of the start condition is stopped and the start condition is not output. - When 0 is written to the IICBnCTL0.IICBnIICE bit - When 1 is written to the IICBnLRET bit - Upon detection of arbitration loss - When 1 is written to the IICBnSPT bit after 1 is written to the IICBnSTT bit while the IICBn is operating as a master in the continuous transfer mode - When 1 is written to the IICBnSTT and IICBnSPT bits during the same data transfer period while the IICBn is operating as a master in the continuous transfer mode (In this case, writing 1 to the IICBnSTT bit is enabled.) **Caution** When start in the initial communication state is enabled (IICBnCTL1.IICB0SLSE bit = 1), the start condition is output regardless of the bus status when the IICBnSTT bit is set to 1. If other communications are performed at that time, they may be damaged. **Note** Setting the IICBnSTT bit at the same time as the IICBnSPT bit is prohibited.
0	IICBnSPT	Stop condition trigger bit 0: Does not output a stop condition. 1: Outputs a stop condition (This bit is automatically cleared after it has been set to 1). The IICBnSPT bit can be set to 1 under the following conditions while the IICBn is performing communication as a master. • Single transfer mode - Wait state triggered by the falling edge of the 9th clock (both address transfer and data transfer) - During data reception, only after clearing the ACKEn bit to 0 to report the end of reception to the slave • Continuous transfer mode The IICBnSPT bit can be set to 1 in the following states. - During the wait state triggered by the falling edge of the 9th clock of address transfer - During data transfer - Detection of a NACK signal (IICBnSTR0.IICBnSSAC bit = 0) during the wait state triggered by the falling edge of the 9th clock for during data reception A stop condition can be output with the following procedure. (If the IICBn is in the wait state, after exiting the wait state) SCLn is released when SDAn has output a low level, and SCLn = high level, SDAn is low level are waited for. Then, following the lapse of the $t_{SU:STO}$ time, a stop condition is output by setting SDAn to high level.

Table 29-15 IICBnTRG register contents (4/4)

Bit position	Bit name	Function
0	IICBnSPT	The output processing of the stop condition is started by setting the IICBnSPT bit to 1, but upon detection of the following states, output processing of the stop condition is stopped and the stop condition is not output. - When 0 is written to the IICBnIICE bit - When 1 is written to the IICBnLRET bit - Upon detection of a stop condition - Upon detection of arbitration loss - When 1 is written to the IICBnSTT bit after IICBnSPT has been set to 1 while the IICBn is operating as a master in the continuous transfer mode **Caution** Setting the IICBnSPT bit to 1 is prohibited during slave operation (IICBnSTR0.IICBnSSMS bit = 0) **Note** Setting the IICBnSPT bit to 1 at the same time as the IICBnSTT bit is prohibited.

(8) IICBnSTR0 – IICBn status register 0

This register indicates the statuses of the IICBn and the bus.

Access This register can be read only in 16-bit units. However, when IICBnIICE is 0, this register can also be write accessed.

Address <IICBn_base> + 0010_H

Initial Value 0000_H. This register is also initialized by changing the value of the IICBnCTL0.IICBnIICE bit from 1 to 0 or from 0 to 1.

15	14	13	12	11	10	9	8
IICBn SSMS	0	IICBn SSDR	IICBn SSWT	IICBn SSEX	IICBn SSCO	IICBn SSTR	IICBn SSAC
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
IICBn SSRS	IICBn SSBS	IICBn SSST	IICBn SSSP	0	0	IICBn STCF	IICBn ALDF
R	R	R	R	R	R	R	R

Table 29-16 IICBnSTR0 register contents (1/6)

Bit position	Bit name	Function
15	IICBnSSMS	Master state check flag: Indicates that the IICBn is operating as a master. 1: Indicates that the IICBn is operating as a master. Setting condition: Upon detection of a start condition after 1 is written to the IICBnTRG.IICBnSTT bit Clearing conditions: • When 1 is written to the IICBnTRG.IICBnLRET bit • Upon detection of a stop condition • Upon detection of arbitration loss If a setting condition coincides with a clearing condition, the clearing condition takes priority.
13	IICBnSSDR	IICBnDAT register status flag 1: Indicates that data in the IICBnDAT register remains unprocessed. During reception operation: Received data remains unread in the IICBnDAT register. During transmission operation: Data written to the IICBnDAT register has not been transferred to the IICBnDAT register. Setting condition: • When the IICBnDAT register is written during address transfer and data transfer while the IICBnSSWT bit is 0 (Note that the IICBnSSDR bit is not set to 1 if address data is written to the IICBnDAT register while the IICBn is operating as a master, because the address data is directly transferred to the IICBnDAT register in this event.) • At the falling edge of the 9th clock after an address match with a slave • When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data reception • When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data reception

Table 29-16 IICBnSTR0 register contents (2/6)

Bit position	Bit name	Function
13	IICBnSSDR	Clearing conditions: • Clearing conditions given priority over setting conditions - When 1 is written to the IICBnLRET bit - Upon detection of arbitration loss - At the falling edge of the 9th clock during address transfer while the IICBn is operating as a master - When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data transmission - When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data transmission • Clearing condition for which setting conditions are given priority - When the IICBnDAT register is read while the IICBnDAT register does not have any received data that must be transferred to the IICBnDAT register
12	IICBnSSWT	Wait state flag 1: Indicates that the IICBn is in the wait state. Setting condition: ■ In single transfer mode <Common to master/slave> • During data transfer, upon detection of the falling edge of the 8th clock with IICBnSLWT = 0 • During data transfer, upon detection of the falling edge of the 9th clock with IICBnSLWT = 1 <Master> • When the IICBn becomes a master (IICBnSSMS = 1) after 1 is written to the IICBnSTT bit, and the falling edge of the first SCLn is detected without the IICBnDAT register being written • Upon detection of the falling edge of the 9th clock during address transfer <Slave> • Upon detection of the falling edge of the 9th clock during address transfer when an address match occurred ■ In continuous transfer mode <During data transfer period, common to master/slave> • During data transmission, when the data to be transmitted next has not been written - When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data transmission with IICBnSSDR = 0 - When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data transmission with IICBnSSDR = 0 • During data reception, when the previous received data has not been read - When IICBnCTL0.IICBnSLWT = 0, at the falling edge of the 8th clock during data reception with IICBnSSDR = 1 - When IICBnCTL0.IICBnSLWT = 1, at the falling edge of the 9th clock during data reception with IICBnSSDR = 1 - Upon NACK detection (However, only if 1 has not been written to IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT while the IICBn is operating as a master)

Table 29-16 IICBnSTR0 register contents (3/6)

Bit position	Bit name	Function
12	IICBnSSWT	During address transfer period, operating as master> - When the IICBn becomes a master (IICBnSSMS = 1) after 1 is written to the IICBnSTT bit, and the first falling edge is detected without the IICBnDAT register being written - Upon NACK detection (However, only if 1 has not been written to IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT) <During address transfer period, operating as slave> - Upon detection of the falling edge of the 9th clock while IICBnSSTR bit is 0 during address transfer when an address match occurred - Upon NACK detection Clearing conditions: • Clearing conditions given priority over setting conditions - When 1 is written to the IICBnLRET bit - When 1 is written to the IICBnSTT bit while the IICBn is operating as a master in the continuous transfer mode - When 1 is written to the IICBnSPT bit while the IICBn is operating as a master in the continuous transfer mode - When the IICBnDAT register is written while the IICBn is performing transmission in the continuous transfer mode - During the wait state triggered by the falling edge of the 8th clock, when the IICBnDAT register is read while reception is performed in the continuous transfer mode - During the wait state triggered by the falling edge of the 9th clock, when the IICBnDAT register is read while the IICBn is performing reception in the continuous transfer mode and an acknowledge signal ($\overline{ACK}$) has been received • Clearing condition for which setting conditions are given priority - When 1 is written to the IICBnTRG.IICBnWRET bit - When 1 is written to the IICBnSTT bit while the IICBn is operating as a master in the single transfer mode - When 1 is written to the IICBnSPT bit while the IICBn is operating as a master in the single transfer mode - When the IICBnDAT register is written while the IICBn is performing reception in the single transfer mode **Caution** If the IICBn exits the wait state that was triggered by the falling edge of the 9th clock by writing 1 to the IICBnWRET bit, the IICBnSSTR bit is cleared to 0 and the bus is released (both SCLn and SDAn go into high impedance).

Table 29-16 IICBnSTR0 register contents (4/6)

Bit position	Bit name	Function
11	IICBnSSEX	Expansion code reception detection flag 1: Indicates that an expansion code has been received. Setting condition: Upon detection of the falling edge of the 8th clock while transferring received address data whose higher 4 bits are either 0000 or 1111 Clearing conditions: • When 1 is written to the IICBnLRET bit • Upon detection of a stop condition • Upon detection of a start condition **Caution** When the expansion codes match, the processing after the interrupt differs according to the ensuing data, and therefore is dependent on software processing.
10	IICBnSSCO	Address match detection flag 1: Indicates that an address that matches the IICBnSVA register has been detected. Setting condition: Upon detection of the falling edge of the 8th clock while transferring a received address that matches the IICBnSVA register Clearing conditions: • When 1 is written to the IICBnLRET bit • Upon detection of a stop condition • Upon detection of a start condition
9	IICBnSSTR	Transmission status detection flag 1: Indicates that data is being transmitted to the serial data bus. Setting condition: <Master> - Upon detection of a start condition after 1 is written to the IICBnSTT bit <Slave> - Upon detection of the falling edge of the 8th clock following reception of 1 to R/$\overline{W}$ bit during address transfer when an address match occurred Clearing conditions: <Common to master/slave> - When 1 is written to the IICBnLRET bit - Upon detection of a stop condition - When 1 is written to the IICBnWRET bit during the wait state triggered by the falling edge of the 9th clock <Master> - Upon detection of the falling edge of the 8th clock following reception of 1 to R/$\overline{W}$ bit during address transfer - Upon detection of arbitration loss <Slave> - Upon detection of a start condition

Table 29-16 IICBnSTR0 register contents (5/6)

Bit position	Bit name	Function
8	IICBnSSAC	Acknowledge ($\overline{ACK}$) detection flag 1:Indicates that an acknowledge signal has been detected. Setting condition: Upon detection of the falling edge of SCLn when a low level has been received at the $\overline{ACK}$ bit during participation in communications Clearing conditions: • When 1 is written to the IICBnLRET bit • Upon detection of the rising edge of SCLn **Caution** The value of the IICBnSSAC bit changes regardless of whether or not an interrupt has occurred.
7	IICBnSSRS	Communication reserve state flag 0: Not communication reserve state 1: Communication reserve state Setting condition: When 1 is written to the IICBnSTT bit during bus communication while the IICBn is not operating as a master, in the communication reserve function enabled state (IICBnCTL1.IICBnSLRS = 0) Clearing conditions: • When 1 is written to the IICBnLRET bit • When IICBnSSMS = 1
6	IICBnSSBS	IICBn bus status flag 0: Bus released state (initial communication state when IICBnCTL1.IICBnSLSE = 1) 1: Bus communication state (initial communication state when IICBnCTL1.IICBnSLSE = 0) Setting condition: • Upon detection of a start condition • When 1 is written to the IICBnCTL0.IICBnIICE bit when IICBnSLSE = 0 Clearing conditions: Upon detection of a stop condition **Note** The IICBnSSBS bit operates whether or not the IICBn is participating in communications.
5	IICBnSSST	Start condition detection flag 1: Indicates that a start condition has been detected. Setting condition: Upon detection of a start condition Clearing conditions: • When 1 is written to the IICBnLRET bit • Upon detection of a stop condition • Upon detection of the rising edge of SCLn following the end of address transfer **Note** The IICBnSSST bit operates whether or not the IICBn is participating in communications.
4	IICBnSSSP	Stop condition detection flag 1: Indicates that a stop condition has been detected. Setting condition: Upon detection of a stop condition Clearing conditions: Upon detection of the falling edge of the first SCLn following start condition detection **Note** The IICBnSSSP bit operates whether or not the IICBn is participating in communications.

Table 29-16 IICBnSTR0 register contents (6/6)

Bit position	Bit name	Function
1	IICBnSTCF	IICBnSTT bit clear flag 1: Indicates that the IICBnSTT bit has been cleared because start condition output failed. Setting condition: When 1 is written to the IICBnSTT bit during bus communication when the IICBn is not operating as a master, in the communication reserve function disabled state (IICBnSLRS = 1) **Caution** Even if the bus is released in the external bus state, this bit is set to 1 when 1 is written to the IICBnSTT bit if the communication reserve function is disabled, unless the IICBn recognizes the bus release state (IICBnSSBS = 1). Clearing condition: When 1 is written to the IICBnSTRC.IICBnCLSF bit
0	IICBnALDF	Arbitration loss detection flag 1: Indicates that an arbitration loss has been detected. Setting condition: Upon detection of arbitration loss Clearing condition: When 1 is written to the IICBnSTRC.IICBnCLAF bit If a setting condition coincides with a clearing condition, the setting condition takes priority. Upon detection of arbitration loss, the IICBnSSMS and IICBnSSTR bits are cleared to 0. (SCLn and SDAn become high level and the bus is released.) **Caution** When the IICBnALDF bit is set to 1 due to arbitration loss, the INTIICBTIA or INTIICBTISn interrupt request signal is output. After confirming that the IICBnALDF bit has been set to 1 with an interrupt request signal, clear the IICBnALDF bit with the IICBnSTRC.IICBnCLAF bit. If the value of the IICBnALDF is not cleared and remains 1, the INTIICBTISn interrupt request signal will be output at the interrupt timing, even during unrelated communication.

(9) IICBnSTR1 – IICBn status register 1

This register indicates the status of the serial bus.

Access This register is read-only, in 8-bit units.

Address <IICBn_base> + 0014_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IICBn SSCL	IICBn SSDA
R	R	R	R	R	R	R	R

Table 29-17 IICBnSTR1 register contents

Bit position	Bit name	Function
1	IICBnSSCL	Indicates the level of the SCLn pin (input). 0: Low level 1: High level
0	IICBnSSDA	Indicates the level of the SDAn pin (input). 0: Low level 1: High level

(10) IICBnSTRC – IICBn status clear register

This register clears the IICBnSTCF and IICBnALDF bits of the IICBnSTR0 register.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 0018_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	IICBnCLSF	IICBnCLAF
R	R	R	R	R	R	R/W	R/W

Table 29-18 IICBnSTRC register contents

Bit position	Bit name	Function
1	IICBnCLSF	Clears the IICBnSTR0.IICBnSTCF bit. 1: Clears the IICBnSTCF bit. **Note** If the IICBnCLSF bit is read after setting data, 0 is returned.
0	IICBnCLAF	Clears the IICBnSTR0.IICBnALDF bit. 1: Clears the IICBnALDF bit. **Caution** If writing 1 to the IICBnCLAF bit and the setting condition of the IICBnALDF bit occur at the same time, the setting condition of the IICBnALDF takes priority. **Note** If the IICBnCLAF bit is read after data setting, 0 is returned.

(11) IICBnEMU - IICBn emulation register

This register controls whether the IICBn can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <IICBn_base> + 34_H

Initial Value 0000_H

7	6	5	4	3	2	1	0
IICBn SVSDIS	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R

Table 29-19 IICBnEMU register contents

Bit position	Bit name	Function
7	IICBn SVSDIS	Emulation control 0: IICBn can be stopped during emulation 1: IICBn continuous operating during emulation

29.7 Operation

The IICBn supports two transfer modes, single transfer mode and continuous transfer mode.

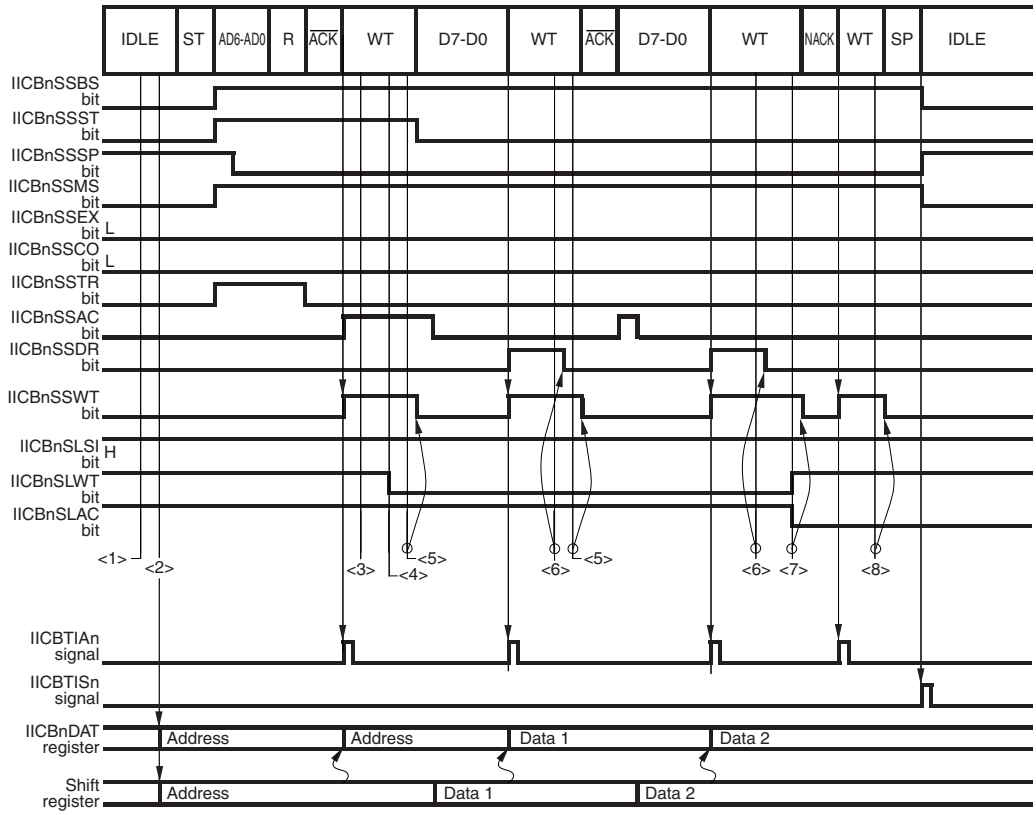
The transfer mode when the addresses of the master device and the slave device match is selected with the IICBnCTL0.IICBnMDTX0 bit, and the transfer mode when the extension code is detected by the slave device is selected with the IICBnMDTX1 bit.

29.7.1 Single transfer mode

In the single transfer mode, a data transmit/receive interrupt request signal is output at the timing specified using the IICBnCTL0.IICBnSLW bit to make the IICBn enter the wait state, and transmit/receive data processing is performed during this wait state.

The various processing operations are described below.

(1) Example of communication in single transfer mode (master reception)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.

<4> Wait timing setting
During data reception, clear the IICBnCTL0.IICBnSLWT bit (to 0) so that the IICBn enters the wait state at the falling edge of the 8th clock.

<5> Data reception
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1) to start reception.

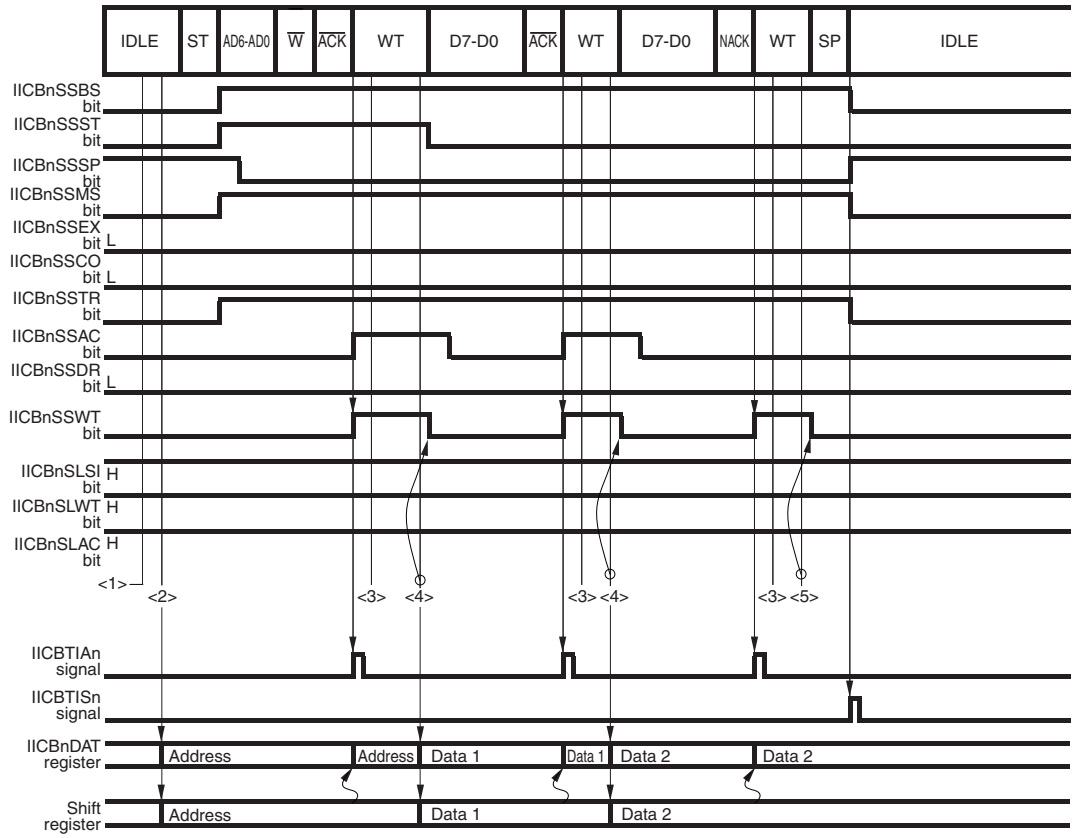
<6> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<7> Data reception completion processing

- Set the IICBnCTL0.IICBnSLWT bit to 1 and the IICBnSLAC bit to 0.
- Next, exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1). The end of the data is notified to the transmitting device without outputting $\overline{\text{ACK}}$.

<8> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(2) Example of communication in single transfer mode (master transmission)



Note During data transmission, set the IICBnCTL0.IICBnSLWT bit (to 1) so that the IICBn enters the wait state at the falling edge of the 9th clock.

<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

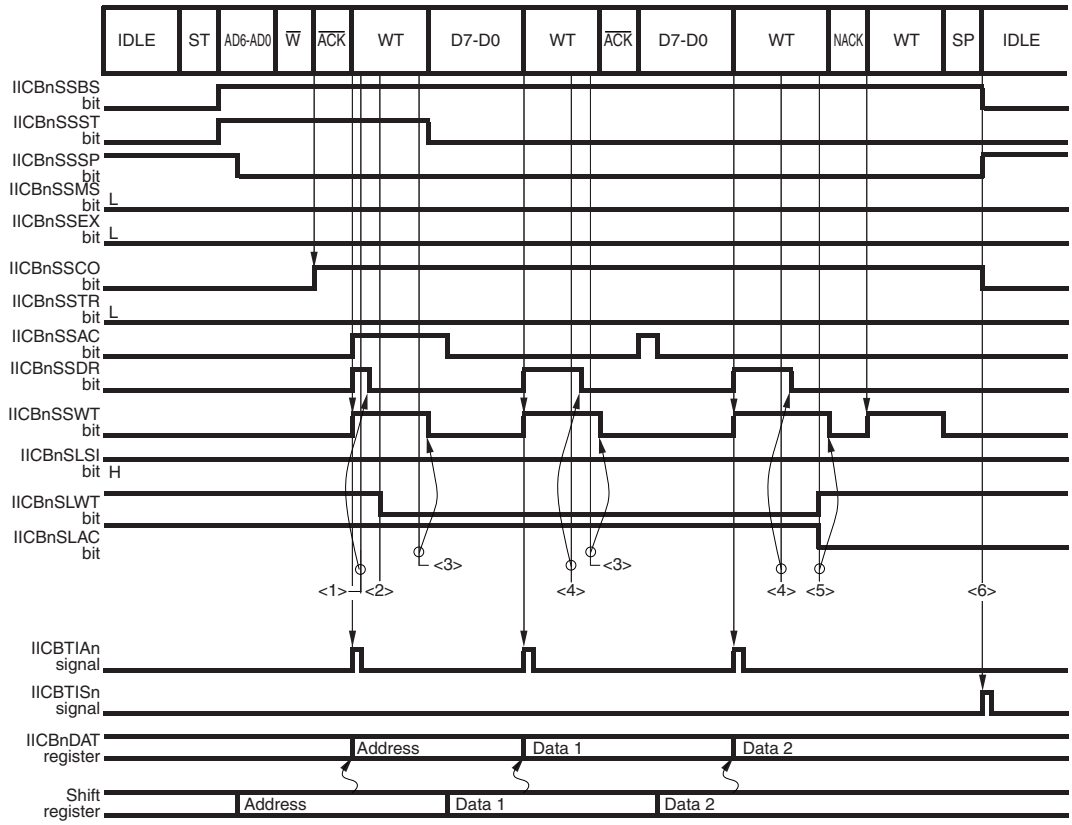
<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.

<4> Data transmission
Exit the wait state by setting the transmit data into the IICBnDAT register to start transmission.

<5> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(3) Example of communication in single transfer mode (slave reception)



<1> Operation mode check in slave mode

- Check the operation mode using the IICBTIAn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSSCO, and IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

<2> Wait timing setting
During data reception, clear the IICBnCTL0.IICBnSLWT bit (to 0) so that the IICBn enters the wait state at the falling edge of the 8th clock.

<3> Data reception
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1) to start reception.

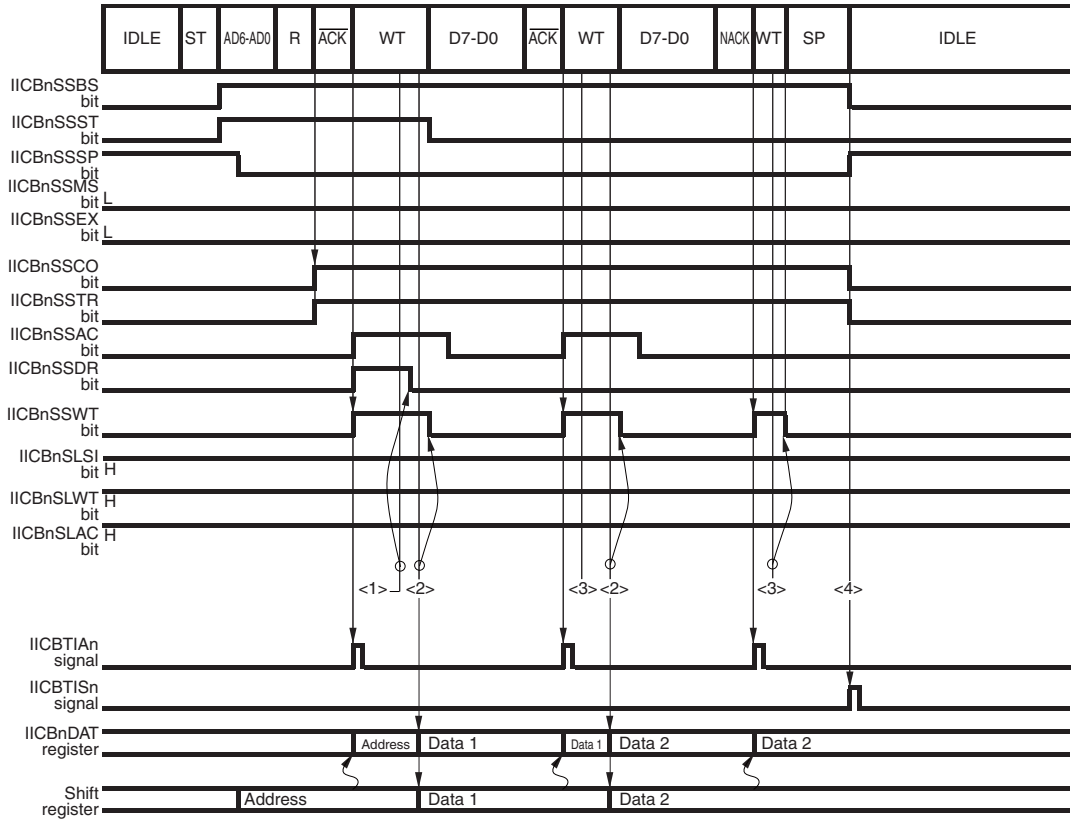
<4> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<5> Data reception completion processing

- Set the IICBnCTL0.IICBnSLWT bit to 1 and the IICBnSLAC bit to 0.
- Next, exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1). The end of the data is notified to the transmitting device without outputting $\overline{\text{ACK}}$.

<6> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

(4) Example of communication in single transfer mode (slave transmission)



Note During data transmission, set the IICBnCTL0.IICBnSLWT bit (to 1) so that the IICBn enters the wait state at the falling edge of the 9th clock.

<1> Operation mode check in slave mode

- Check the operation mode using the IICBTIAn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSSCO, and IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

<2> Data transmission
Exit the wait state by setting the transmit data into the IICBnDAT register to start transmission.

<3> Acknowledge result check
Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit using the IICBTIAn interrupt.
If $\overline{\text{ACK}}$ is not output, the transmission is judged to have been completed, and the IICBn exits the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

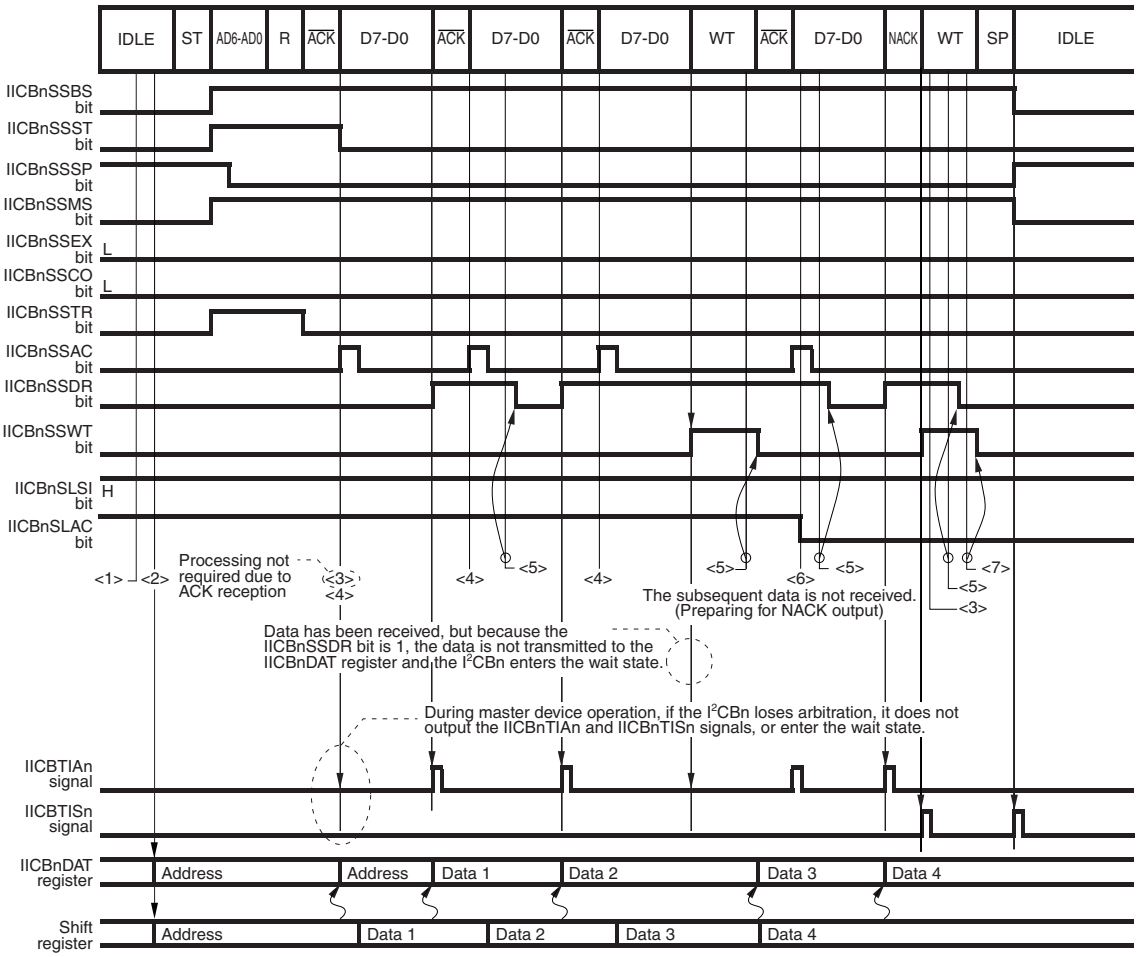
<4> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

29.7.2 Continuous transfer mode

The continuous transfer mode allows continuous communication without entering the wait state by reading/writing data to/from the IICBnDAT register each time the data transmit/receive interrupt request signal (IICBTIAn) is output.

The processing operations are described below.

(1) Example of communication in continuous transfer mode (master reception)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.

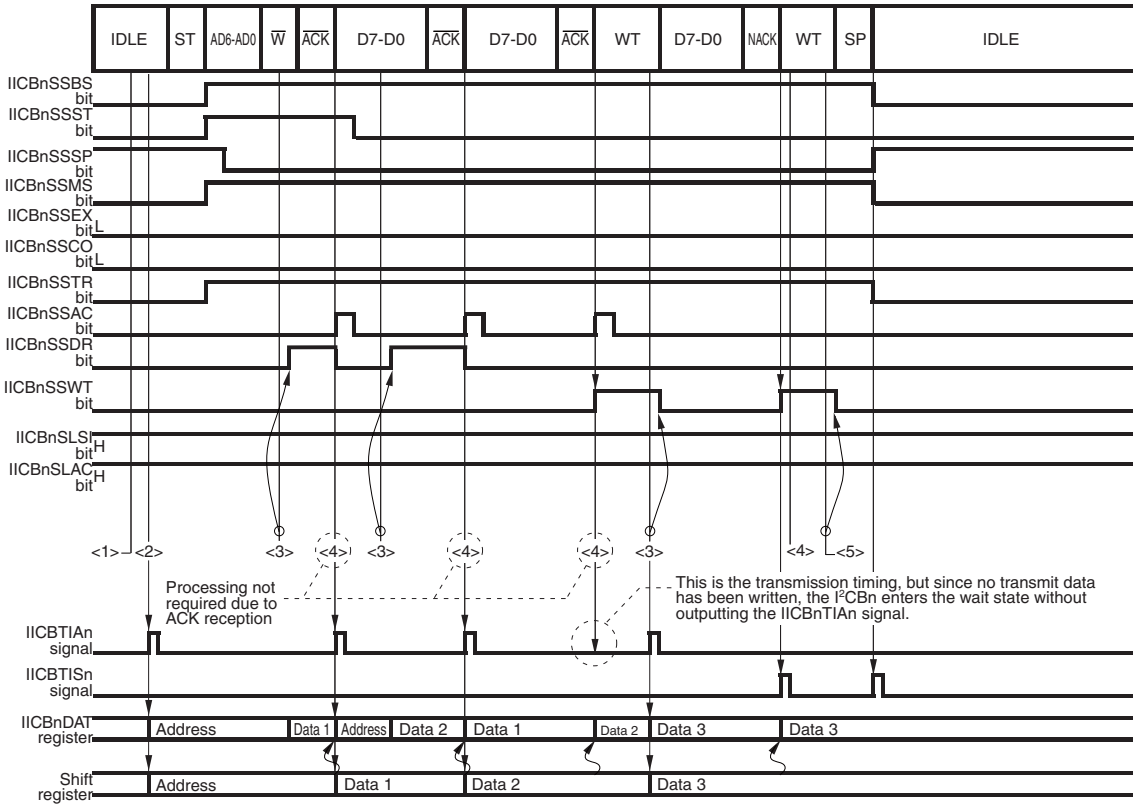
<4> Acknowledge result check
If there is no unread data in the IICBnDAT register by the time reception starts, the IICBn starts reception without entering the wait state.

<5> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<6> Data reception completion processing
By clearing the IICBnCTL0.IICBnSLAC bit (to 0) before reading the receive data immediately preceding the final receive data, the next $\overline{\text{ACK}}$ is not output and the end of the data is notified to the transmitting device.

<7> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(2) Example of communication in continuous transfer mode (master transmission)



<1> Start condition output
Set the IICBnTRG.IICBnSTT bit (to 1).

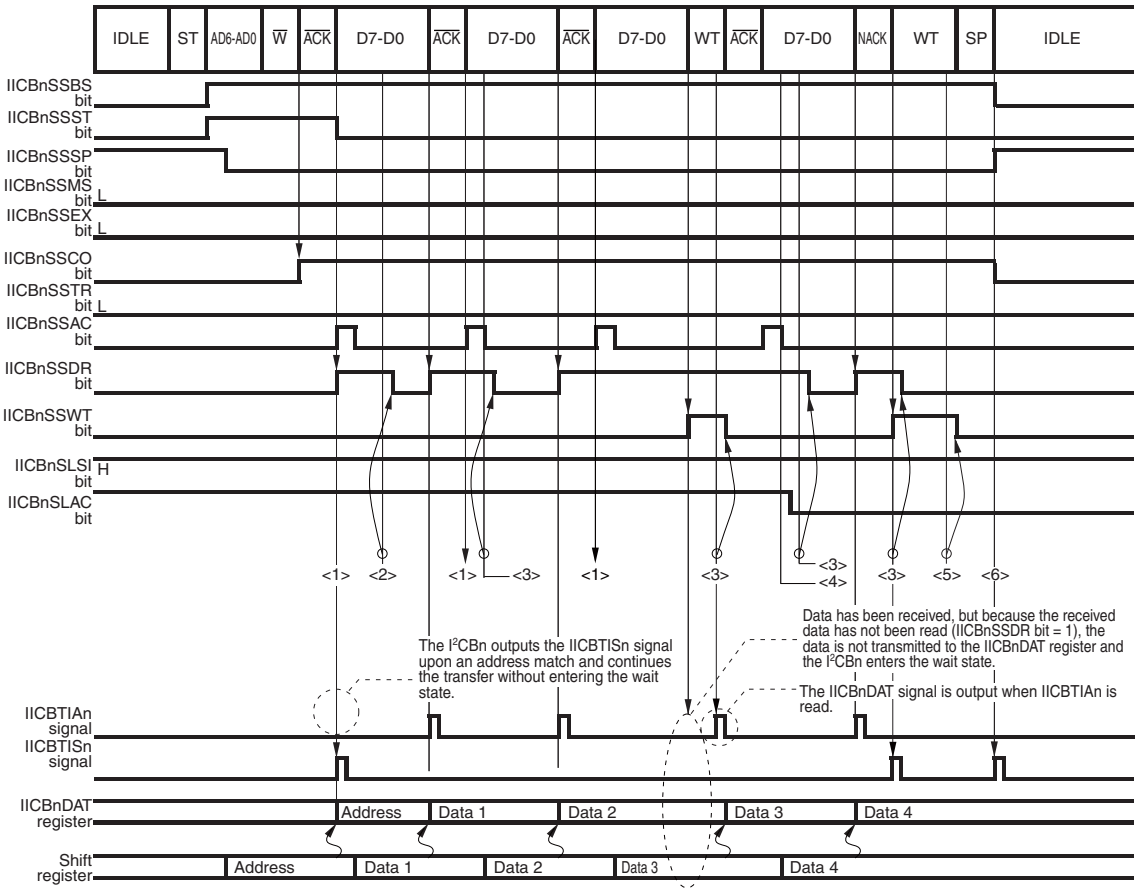
<2> Address and transfer direction specification output
Set the address of the slave device and the transfer direction as 8 bits into the IICBnDAT register.

<3> Data transmission
Set the transmit data to the IICBnDAT register using the IICBTIAn interrupt.

<4> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.

<5> Stop condition output
Set the IICBnTRG.IICBnSPT bit (to 1).

(3) Example of communication in continuous transfer mode (slave reception)



<1> Data reception
If there is no unread data in the IICBnDAT register by the time reception starts, the IICBn starts reception without entering the wait state.

<2> Operation mode check in slave mode

- Check the operation mode using the IICBTISn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSSCO, and IICBnSSTR bits.
- Read the IICBnDAT register (empty read).

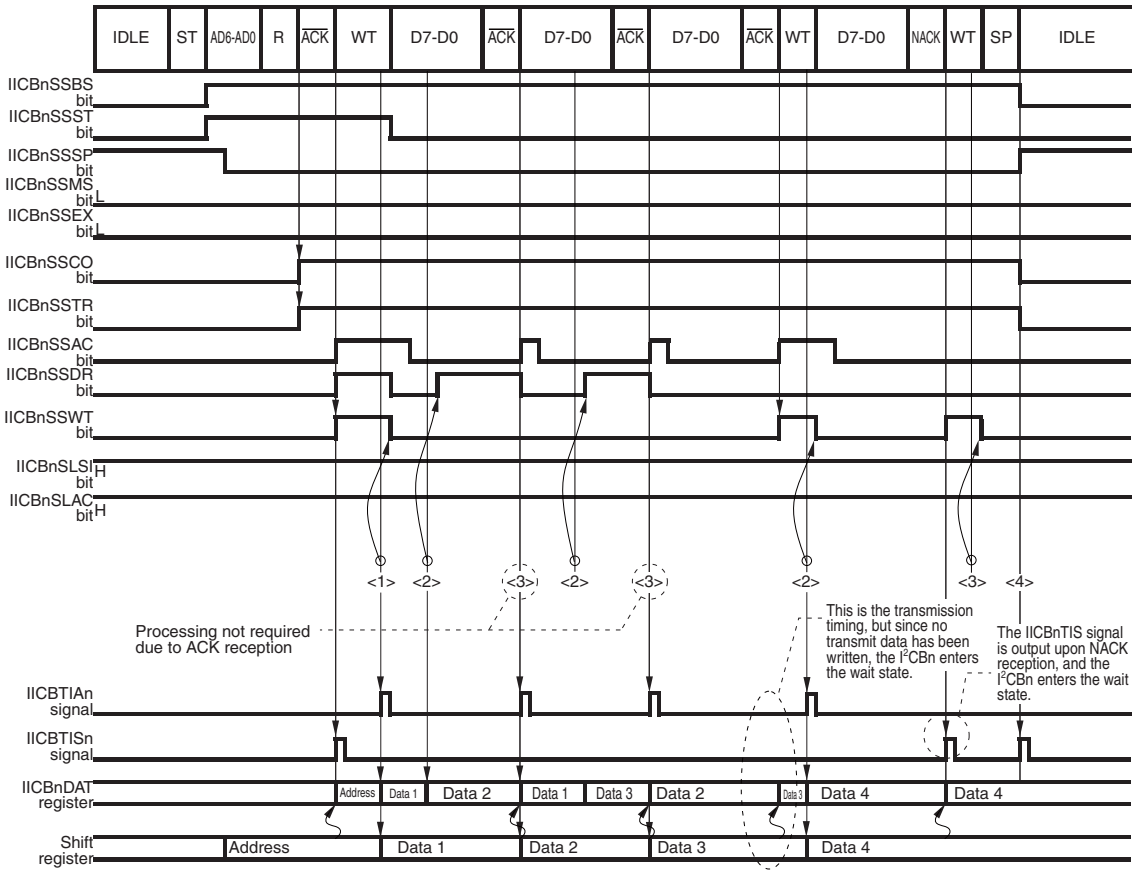
<3> Receive data load
Read the receive data from the IICBnDAT register using the IICBTIAn interrupt.

<4> Data reception completion processing <1>
By clearing the IICBnCTL0.IICBnSLAC bit (to 0) before reading the receive data immediately preceding the final receive data, the next $\overline{\text{ACK}}$ is not output and the end of the data is notified to the transmitting device.

<5> Data reception completion processing <2>
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$.
Exit the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

<6> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

(4) Slave transmission in continuous transfer mode



<1> Operation mode check in slave mode

- Check the operation mode using the IICBTISn interrupt.
- Check the address transfer, address match, and reception operation with the IICBnSTR0.IICBnSSST, IICBnSSCO, and IICBnSSTR bits.
- After reading (empty read) the IICBnDAT register, set the first transmit data to the IICBnDAT register.

<2> Data transmission
Set the transmit data to the IICBnDAT register using the IICBTIAn interrupt.

<3> Acknowledge result check
The IICBTISn interrupt occurs only if the slave device does not return $\overline{\text{ACK}}$. Check the acknowledge result by reading the IICBnSTR0.IICBnSSAC bit.
If $\overline{\text{ACK}}$ is not output, the transmission is judged to have been completed, and the IICBn exits the wait state by setting the IICBnTRG.IICBnWRET bit (to 1).

<4> Stop condition detection
Detect the stop condition using the IICBTISn interrupt.

29.7.3 Arbitration

When the IICBn operates as the master device and loses arbitration, it enters the slave standby state by setting both SCLn and SDAn to high level upon detection of the arbitration loss, and then the IICBnSTR0.IICBnALDF bit is set (to 1) each time the status interrupt request signal (IICBTISn) is output.

(1) Status upon occurrence of arbitration

The statuses upon occurrence of arbitration during master device operation (IICBnSTR0.IICBnSSMS bit = 1) are listed below.

[12]Address transmission

[13]R/$\overline{W}$ bit transmission of address transfer

[14]Extension code transmission

[15]R/$\overline{W}$ bit transmission of extension code transfer

[16]Data transmission

[17]$\overline{ACK}$ bit transmission after data reception

[18]Start condition detection during address transfer or data transfer

[19]Stop condition detection during address transfer or data transfer

[20]The SDAn signal is low when the IICBn is attempting to output a restart condition

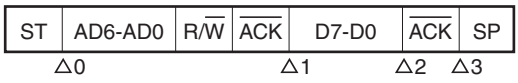
[21]The SDAn signal is low when the IICBn is attempting to output a stop condition

[22]The falling edge of the SCLn signal is detected when the IICBn is attempting to output a restart condition

29.7.4 Entering and exiting wait state

The IICBn enters the wait state at the following timings.

Table 29-20 Wait state transit timings



Timing	Description	Refer to:
Δ0	Upon detection of the first falling edge of the SCLn, following detection of start condition as the master device	*1 "Wait state at falling edge of first SCLn after IICBn became master" on page 2282*
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer after the start condition	*2 "Wait state upon detection of the falling edge of the 9th SCLn during address transfer after the start condition" on page 2283*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*3 "Wait state upon detection of the falling edge of the 8th SCLn during data transfer" on page 2284*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*4 "Wait state upon detection of the falling edge of the 9th SCLn during data transfer" on page 2285*

Note
ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

The method to exit the wait state differs according to the wait state.

Exit the wait state by applying the appropriate method for each of the four wait states as described below.

(1) Wait state at falling edge of first SCLn after IICBn became master

Δ0 indicates the wait state when the data to be transferred has not been written (to the IICBnDAT register) when the falling edge of the first SCLn after the IICBn became the master is detected, after 1 was written to the IICBnTRG.IICBnSTT bit.

(a) Wait state transit condition

The IICBn enters the wait state if data is not written to the IICBnDAT register in the period from when the IICBnSTT bit becomes 1 until the Δ0 timing, upon detection of the first falling edge of SCLn after the IICBn became master, after 1 was written to the IICBnSTT bit.

However, the valid times to write data to the IICBnDAT register (without entering the wait state) after 1 was written to the IICBnSTT bit differ depending on whether the communication reservation function is enabled. The valid times to write to the IICBnDAT register for each of these cases are shown in *Figure 29-12 "Valid times to write to IICBnDAT register"*.

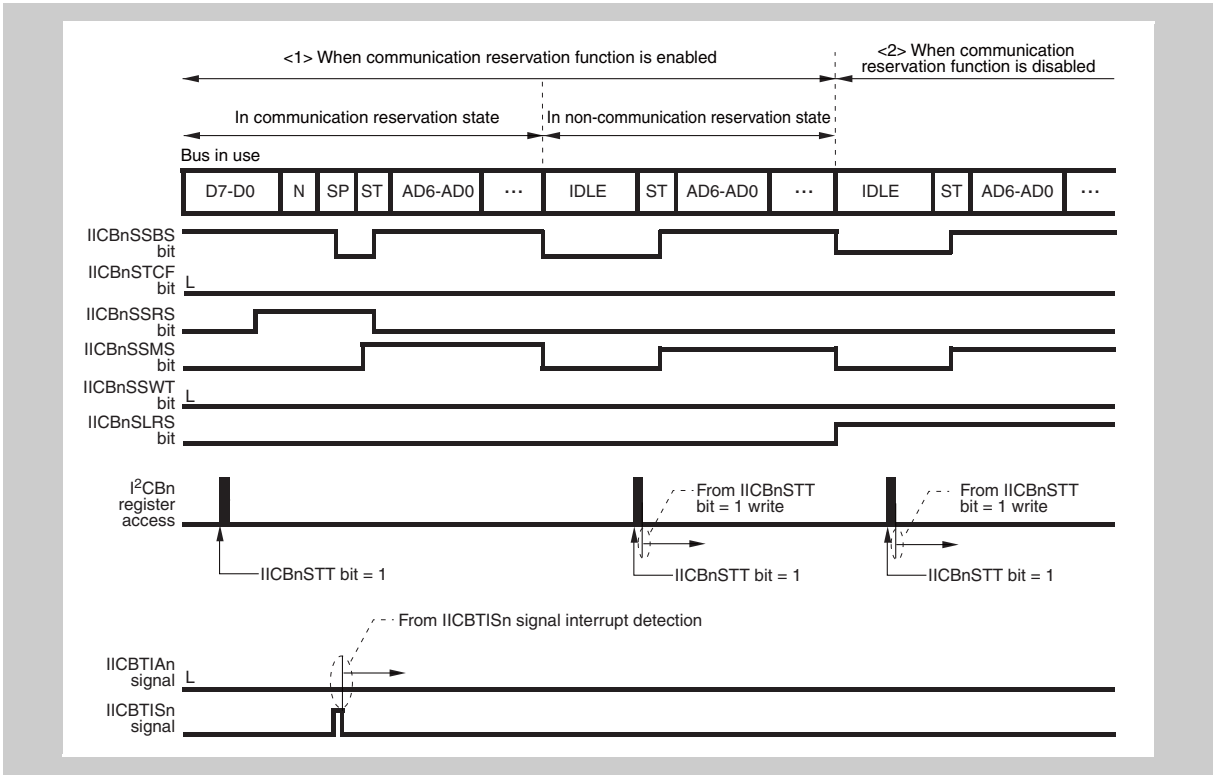


Figure 29-12 Valid times to write to IICBnDAT register

Caution The communication reservation function is disabled (<2> in the above figure) while the IICBnSTR0.IICBnSTCF bit is 0.
When the IICBnSTCF bit becomes 1, setting from IICBnSTCF bit = 1 write is required again.

(b) Wait state exit conditions

Exit the wait state by writing to the IICBnDAT register.

(2) Wait state upon detection of the falling edge of the 9th SCLn during address transfer after the start condition

Δ1 indicates the wait state entered upon completion of address transfer.

(a) Wait state transit condition

<Single transfer mode>

In the single transfer mode, the IICBn always enters the wait state while it operates as the master.

While the IICBn operates as a slave, it enters the wait state upon an address match, or upon extension code detection while the IICBnSLWT bit is 1.

<Continuous transfer mode>

In the continuous transfer mode, the IICBn enters the wait state in the following cases.

- Upon detection of NACK
- When the IICBn operates as the master and transmits data, if the data to be transferred next has not been written
- When the IICBn operates as a slave, if the received data has not been read, or during transmission

(b) Wait state exit conditions

<Single transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by writing 1 to the IICBnWRET bit during reception. When the IICBn operates as the master and the IICBnSTR0.IICBnSSAC bit is 0 or the IICBn is at the transmission side, the wait state can be exited by writing 1 to the IICBnTRG.IICBnSTT or the IICBnTRG.IICBnSPT bit.

<Continuous transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by reading the IICBnDAT register during reception. When the IICBn operates as the master and the IICBnSSAC bit is 0, the wait state can be exited by writing 1 to the IICBnSTT or IICBnSPT bit.

(3) Wait state upon detection of the falling edge of the 8th SCLn during data transfer

Δ2 indicates the wait state entered upon detection of the falling edge of the 8th SCLn during data transfer.

(a) Wait state transit condition

<Single transfer mode>

When the IICBn participates in communications and the IICBnCTL0.IICBnSLWT bit is 0, the IICBn enters the wait state if the falling edge of the 8th SCLn is detected.

<Continuous transfer mode>

When the IICBn participates in communications and the IICBnSTR0.IICBnSSTR bit is 0, the IICBn enters the wait state if processing of the previous data (read from the IICBnDAT register) has not completed and 1 has not been written to the IICBnTRG.IICBnSTT and IICBnTRG.IICBnSPT bits before the falling edge of the 8th SCLn.

(b) Wait state exit conditions

<Single transfer mode>

Exit the wait state by writing to the IICBnDAT register during transmission, or by writing 1 to the IICBnTRG.IICBnWRET bit during reception.

<Continuous transfer mode>

Exit the wait state by reading the IICBnDAT register.

(4) Wait state upon detection of the falling edge of the 9th SCLn during data transfer

Δ3 indicates the wait state entered upon detection of the falling edge of the 9th SCLn during data transfer.

During the continuous transfer mode, the IICBn enters the wait state upon NACK reception.

(a) Wait state transit condition

<Single transfer mode>

When the IICBn participates in communications and the IICBnCTL0.IICBnSLWT bit is 1, the IICBn enters the wait state if the falling edge of the 9th SCLn is detected.

<Continuous transfer mode>

When the IICBn participates in communications, it enters the wait state in the following three cases during data transmission:

- Upon reception of NACK by ACK bit while the IICBnSLWT bit is 1
- When transmit data is not written to the data register
- When the previous received data is not read

(b) Wait state exit conditions

The wait state exit conditions are listed for each transfer mode in *Table 29-21 "Wait state exit conditions"*.

Table 29-21 Wait state exit conditions

Master/ slave	Transfer mode	Transfer direction	IICBnSTR0. IICBnSSAC bit	Exit conditions
Master	Single transfer mode	Reception	0	IICBnTRG.IICBnSTT bit = 1 or IICBnTRG.IICBnSPT bit = 1
			1	IICBnTRG.IICBnLRET bit = 1
		Transmission	0	IICBnSTT bit = 1 or IICBnSPT bit = 1
			1	Write to IICBnDAT register or IICBnSTT bit = 1 or IICBnSPT bit = 1
	Continuous transfer mode	Reception	0	IICBnSTT bit = 1 or IICBnSPT bit = 1
			1	Read from IICBnDAT register[a]
		Transmission	0	IICBnSTT bit = 1 or IICBnSPT bit = 1
			1	Write to IICBnDAT register[b]
Slave	Single transfer mode	Reception	-	IICBnWRET bit = 1
		Transmission	0	IICBnWRET bit = 1
			1	Write to IICBnDAT register[a]
	Continuous transfer mode	Reception	0	IICBnWRET bit = 1
		Transmission	0	IICBnWRET bit = 1
			1	Write to IICBnDAT register

a) Condition for exiting the wait state that was entered when no transmit data has been written to the data register
b) Condition for exiting the wait state that was entered when the received data has not been read

29.7.5 Extension code

The processing when the extension code is received differs according to the data after the extension code and thus must be executed through the user's software.

Therefore, the operation differs from that during normal slave address reception. These differences are described below.

(1) When the upper 4 bits of the received address are 0000 or 1111, the extension code reception flag (IICBnSTR0.IICBnSSEX bit) is set to 1 to indicate that an extension code has been received. The status interrupt request signal (IICBTISn) is output at the falling edge of the 8th clock, and the IICBn enters the wait state (IICBnTRG.IICBnSSWT = 1). The IICBnSTR0.IICBnSSDR bit is then set (to 1).

(2) During address transfer, the acknowledge output can be controlled by setting the IICBnCTL0.IICBnLAC bit. (Note that an acknowledge is always output upon an address match, regardless of the setting of this bit, during address transfer for normal slave address reception.)

(3) The method for exiting the wait state entered upon extension code detection depends on the setting of the IICBnMDTX1 bit as follows.

<When IICBnMDTX1 bit is 0>

During transmission while the IICBnCTL0.IICBnSLWT bit is 0, exit the wait state by writing to the IICBnDAT register. During transmission while the IICBnSLWT bit is 1, or during reception, exit the wait state by writing 1 to the IICBnTRG.IICBnWRET bit.

<When IICBnMDTX1 bit is 1>

During transmission, exit the wait state by writing to the IICBnDAT register, and, during reception, exit the wait state by reading from the IICBnDAT register.

(4) At the falling edge of the 9th clock, if the IICBnSLWT bit is 1, the interrupt request signal (IICBTIAn) is output and the IICBn enters the wait state (IICBnTRG.IICBnSSWT = 1). If the IICBnCTL0.IICBnSLWT bit is 0, the interrupt request signal (IICBTIAn) is not output and the IICBn does not enter the wait state.

(5) If the IICBn receives an extension code, it participates in communications even if the addresses do not match.

For example, to avoid operating the IICBn as a slave device after receiving an extension code, set the IICBnTRG.IICBnLRET bit to 1. The IICBn enters the standby state for the next communication.

29.8 Interrupt Request Signals

Caution In this section, the operation when an extension code is received is omitted. For details about the extension code, refer to *29.7.5 "Extension code"*.

The IICBn has two interrupt request signals, the data transmit/receive interrupt request signal (IICBTIAn) and the status interrupt request signal (IICBTISn). Both signals are pulses of one PCLK clock width. The interrupt request signal output timing differs according to the transfer mode set using the IICBnCTL0.IICBnMDTX1 and IICBnMDTX0 bits. The interrupt request signals are explained below for each transfer mode.

To perform transfer with an address match between the master device and the slave device, select the single transfer mode or continuous transfer mode with the IICBnMDTX0 bit, and to perform transfer with extension code detection by the slave, select the single transfer mode or continuous transfer mode using the IICBnMDTX1 bit.

29.8.1 Single transfer mode

The interrupt request signal timing in the single transfer mode is described below.
During the single transfer mode, for the IICBTIAn and IICBTISn interrupt request signals, whether to output an interrupt is judged based on the IICBn state when the falling edge of SCLn is detected during the bus cycle. Note, however, that whether to output an interrupt is judged based on the IICBn state when a stop condition is detected at the Δ4 timing.

Table 29-22 Interrupt request signal output timing (single transfer mode)

Output timing	Description	Refer to:
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer	*1 "Interrupt request signal output conditions and output interrupt request signals during address transfer" on page 2289*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*2 "Interrupt request signal output conditions and interrupt request signals output during data transfer" on page 2290*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*2 "Interrupt request signal output conditions and interrupt request signals output during data transfer" on page 2290*
Δ4	Upon detection of a stop condition	*3 "Interrupt request signal output upon stop condition detection" on page 2290*

Note
ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

(1) Interrupt request signal output conditions and output interrupt request signals during address transfer

Δ1 in *Table 29-22 "Interrupt request signal output timing (single transfer mode)"* indicates the interrupt request signal output timing during address transfer. *Table 29-23 "Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)"* lists the interrupt request signal output conditions and the interrupt request signals output (IICBTIAn or IICBTISn) at the Δ1 timing.

Table 29-23 Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)

IICBn SSMS	IICBn ALDF	IICBn SLWT	IICBn SSCO	Δ1		Remark
				Interrupt	Wait	
1	0	×	×	IICBTIAn	Wait	-
1	1	×	×	This state does not exist.		-
0	0	×	0	IICBTISn[a]	-	After restart, non-participation in communications
0	0	×	1	IICBTIAn	Wait	-
0	1	×	0	IICBTISn	-	After arbitration loss, non-participation in communications
0	1	×	1	IICBTIAn	Wait	-

a) In case of an address match or extension code detection, before the restart condition

Note ×: don't care

(2) Interrupt request signal output conditions and interrupt request signals output during data transfer

Δ2 and Δ3 in *Table 29-22 "Interrupt request signal output timing (single transfer mode)"* indicate the interrupt request signal output timings during data transfer. The interrupt request signal output timing at Δ2 or Δ3 is determined by the setting of the IICBnCTL0.IICBnSLWT bit. *Table 29-24 "Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)"* lists the interrupt request signal output conditions and the interrupt request signals (IICBTIAn or IICBTISn) output at the Δ2 and Δ3 timings.

Table 29-24 Interrupt request signal output conditions and interrupt request signals output during address transfer (single transfer mode)

IICBn SSMS	IICBn ALDF	IICBn SLWT	IICBn SSCO	Δ2		Δ3		Remark
				Interrupt	Wait	Interrupt	Wait	
1	0	0	×	IICBTIAn	Wait	-	-	-
1	0	1	×	-	-	IICBTIAn	Wait	-
1	1	×	×	This state does not exist.				-
0	0	×	0	-	-	-	-	Non-participation in communications
0	0	0	1	IICBTIAn	Wait	-	-	-
0	0	1	1	-	-	IICBTIAn	Wait	-
0	1	0	0	IICBTISn	-	-	-	Non-participation in communications after arbitration loss
0	1	1	0	-	-	IICBTISn	-	Non-participation in communications after arbitration loss
0	1	0	1	IICBTIAn	Wait	-	-	-
0	1	1	1	-	-	IICBTIAn	Wait	-

Note ×: don't care

(3) Interrupt request signal output upon stop condition detection

Δ4 in *Table 29-22 "Interrupt request signal output timing (single transfer mode)"* indicates the interrupt request signal output timing upon detection of a stop condition.

Interrupt request signal output is controlled according to the IICBnCTL0.IICBnSLSI bit. If a stop condition is detected while the IICBnSLSI bit is 1, the status interrupt request signal (IICBTISn) is output.

29.8.2 Continuous transfer mode

(1) Data transmit/receive interrupt request signal (IICBTIAn)

The conditions for outputting an IICBTIAn signal in the continuous transfer mode are described below.

- Interrupt request signal output condition during reception

  When receive data is saved from the shift register to the IICBnDAT register (timing <1> in *Figure 29-13 "IICBTIAn signal output timing (reception in continuous transfer mode)"*)

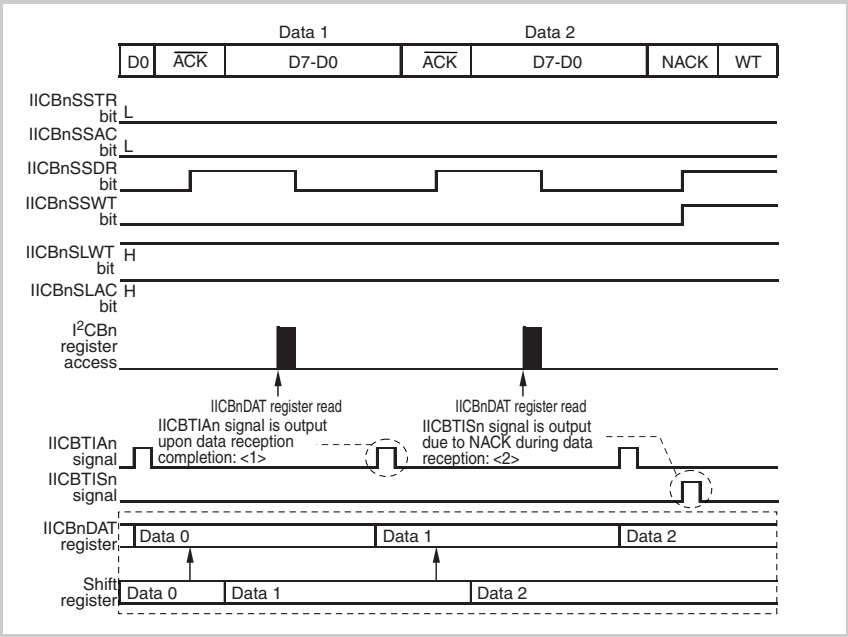


Figure 29-13 IICBTIAn signal output timing (reception in continuous transfer mode)

- Interrupt request signal output condition during transmission

  When data is written to the IICBnDAT register while there is no transmit data in the shift register and IICBnDAT register (timing <2> in *Figure 29-14 "IICBTIAn signal output timing (transmission in continuous transfer mode)"*).

  When data is saved from the IICBnDAT register to the shift register (timing <1> in *Figure 29-14 "IICBTIAn signal output timing (transmission in continuous transfer mode)"*).

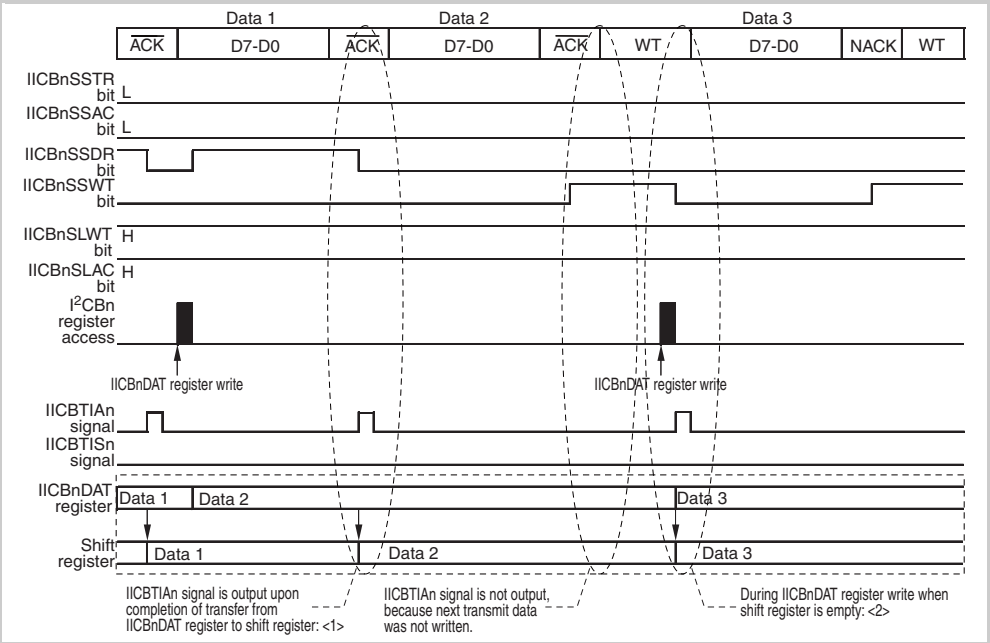
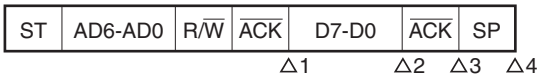


Figure 29-14 IICBTIAn signal output timing (transmission in continuous transfer mode)

(2) Status interrupt request signal (IICBTISn)

The IICBTISn signal output timing in the continuous transfer mode is the same as that in the single transfer mode.

Table 29-25 IICBTISn signal output timing



Output timing	Description	Refer to:
Δ1	Upon detection of the falling edge of the 9th SCLn during address transfer after the start condition	*a "IICBTISn signal output conditions during address transfer" on page 2294*
Δ2	Upon detection of the falling edge of the 8th SCLn during data transfer	*b "IICBTISn signal output conditions during data transfer" on page 2295*
Δ3	Upon detection of the falling edge of the 9th SCLn during data transfer	*b "IICBTISn signal output conditions during data transfer" on page 2295*
Δ4	Upon detection of a stop condition	*c "IICBTISn signal output upon detection of stop condition" on page 2296*

Note
ST: Start condition
AD6 to AD0: Address
R/$\overline{W}$: Transfer direction specification
$\overline{ACK}$: Acknowledge
D7 to D0: Data
SP: Stop condition

(a) IICBTISn signal output conditions during address transfer

Δ1 in *Table 29-25 "IICBTISn signal output timing"* indicates the IICBTISn signal output timing during address transfer. *Table 29-26 "IICBTISn signal output conditions during address transfer (continuous transfer mode)"* lists the IICBTISn signal output conditions at the Δ1 timing.

Table 29-26 IICBTISn signal output conditions during address transfer (continuous transfer mode)

IICBn SSMS	IICBn SSCO	IICBn ALDF	Transfer direction	IICBn SSDR	IICBn SSAC	Δ1	
						Interrupt	Wait
1	×	0	Transmission	0	1	-	Wait
1	×	0	Transmission	0	0	IICBTISn	Wait
1	×	0	Transmission	1	1	-	-
1	×	0	Transmission	1	0	IICBTISn	Wait
1	×	0	Reception	0	1	-	-
1	×	0	Reception	0	0	-	Wait
1	×	0	Reception	1	1	IICBTISn during IICBnDAT read	Wait
1	×	0	Reception	1	0	IICBTISn during IICBnDAT read	Wait
1	×	1	×	×	×	This state does not exist.	
0	0	0	×	×	×	IICBTISn[a]	-
0	0	1	×	×	×	IICBTISn	-
0	1	×	Transmission	×	1	IICBTISn	Wait
0	1	×	Reception	0	1	IICBTISn	-
0	1	×	Reception	1	1	IICBTISn during IICBnDAT read	Wait

a) Upon an address match, before restart condition

Caution For Δ1, the IICBnSSAC bit is always 0.

Note ×: don't care

(b) IICBTISn signal output conditions during data transfer

Δ2 and Δ3 in *Table 29-25 "IICBTISn signal output timing"* indicate the IICBTISn signal output timings during data transfer. *Table 29-27 "IICBTISn signal output conditions during data transfer (continuous transfer mode)"* lists the IICBTISn signal output conditions at the Δ2 and Δ3 timings.

Table 29-27 IICBTISn signal output conditions during data transfer (continuous transfer mode)

IICBn SSMS	IICBn SSCO	IICBn SLWT	IICBn ALDF	Transfer direction	IICBn SSDR	IICBn SSAC	IICBnSTT or IICBnSPT	Δ2		Δ3	
								Interrupt	Wait	Interrupt	Wait
1	×	0	×	Transmission	0	1	a	-	-	-	-
1	×	0	×	Transmission	0	0	a	-	-	IICBTISn	Wait
1	×	0	×	Transmission	1	1	a	-	-	-	-
1	×	0	×	Transmission	1	0	a	-	-	IICBnDAT during IICBTISn write	Wait
1	×	0	×	Reception	0	1	a	-	-	-	-
1	×	0	×	Reception	0	0	a	-	-	IICBTISn	Wait
1	×	0	×	Reception	1	1	a	-	-	-	-
1	×	0	×	Reception	1	0	a	-	-	IICBnDAT after IICBTISn read	Wait
1	×	×	×	×	×	0	b		-	IICBTISn	-
1	×	×	×	×	×	1	b	-	-	-	-
0	0	×	0	×	×	×	×	-	-	-	-
0	0	0	1	Reception	×	×	×	IICBTIS	-	-	-
0	0	1	1	Transmission	×	×	×	-	-	IICBTISn	-
0	1	0	×	Transmission	0	1	a	-	-	-	Wait
0	1	0	×	Transmission	0	0	a	-	-	IICBnDAT during IICBTISn write	Wait
0	1	0	×	Transmission	1	1	a	-	-	-	-
0	1	0	×	Transmission	1	0	a	-	-	IICBTISn	Wait
0	1	0	×	Reception	0	1	a	-	-	-	-
0	1	0	×	Reception	0	0	a	-	-	IICBTISn	Wait
0	1	0	×	Reception	1	1	a	-	-	-	-
0	1	0	×	Reception	1	0	a	-	-	IICBnDAT during IICBTISn read	Wait

a) When 1 has not been written to the IICBnTRG.IICBnSTT or IICBnTRG.IICBnSPT bit
b) When 1 has been written to the IICBnSTT or IICBnSPT bit

Note ×: don't care

(c) IICBTISn signal output upon detection of stop condition

Δ4 in *Table 29-25 "IICBTISn signal output timing"* indicates the IICBTISn signal output timing upon detection of a stop condition.

IICBTISn signal output is controlled according to the IICBnCTL0.IICBnSLSI bit. If a stop condition is detected while the IICBnSLSI bit is 1, the IICBTISn signal is output.

29.9 Interrupt Outputs and Statuses

This section describes the statuses of the IICBnSTR0 register during interrupt output by communication flow.

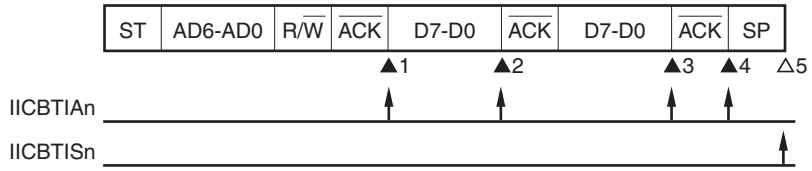
The meanings of the symbols used in the figures are as follows.

ST:	Start condition
AD6-AD0:	Address
R, $\overline{W}$, R/$\overline{W}$:	Transfer direction specification
$\overline{ACK}$:	Acknowledge
NACK:	Not acknowledge
D7-D0:	Data
SP:	Stop condition

29.9.1 Single transfer mode (master device operation)

(1) Start ~ Address ~ Data ~ Data ~ Stop (normal transmission/reception)

<1> When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B

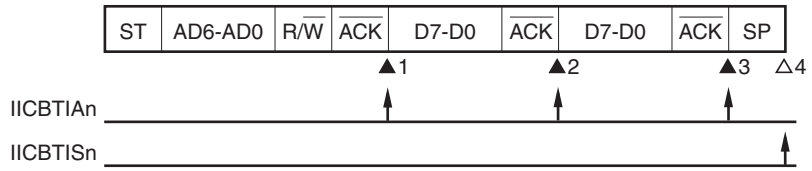
▲3: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnSLWT bit = 1)

▲4: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B

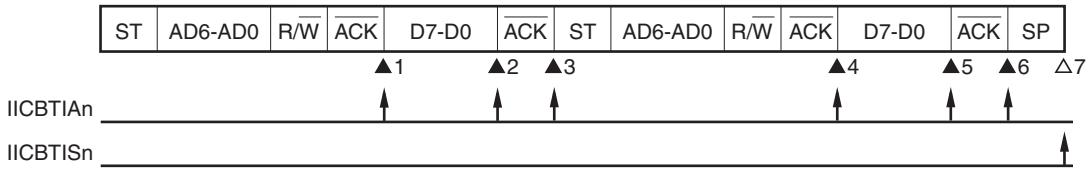
▲3: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnSPT bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

<1> When IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnSLWT bit = 1)

▲3: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnTRG.IICBnSTT bit = 1, IICBnSLWT bit = 0)

▲4: IICBnSTR0 register = 1-0100X1 0110--00B

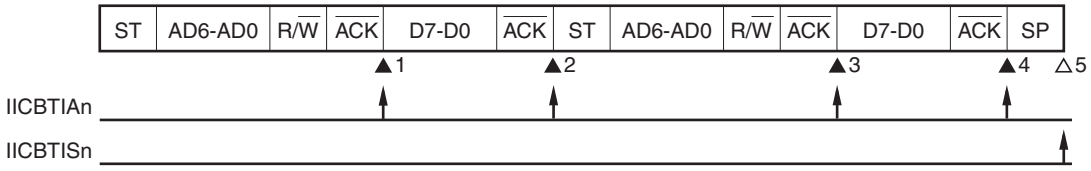
▲5: IICBnSTR0 register = 1-0100X0 0100--00B (IICBnSLWT bit = 1)

▲6: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnSPT bit = 1)

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0100X1 0110--00B

▲2: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnSTT bit = 1)

▲3: IICBnSTR0 register = 1-0100X1 0110--00B

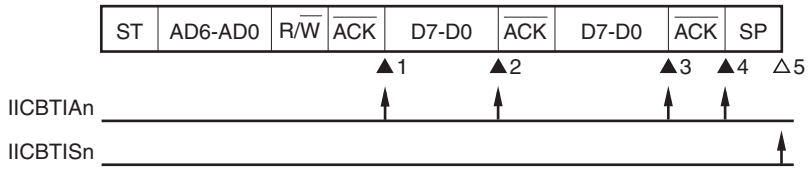
▲4: IICBnSTR0 register = 1-0100XX 0100--00B (IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(3) Start ~ ExCode ~ Data ~ Data ~ Stop (extension code transmission)

<1> When IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-0110X1 0110--00B

▲2: IICBnSTR0 register = 1-0110X0 0100--00B

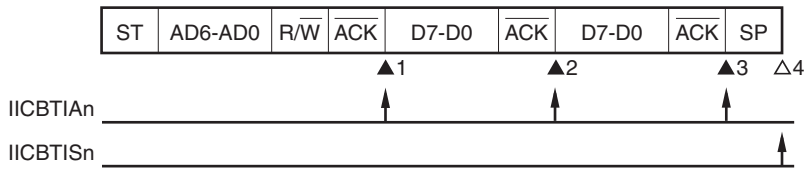
▲3: IICBnSTR0 register = 1-0110X0 0100--00B (IICBnSLWT bit = 1)

▲4: IICBnSTR0 register = 1-0110XX 0100--00B (IICBnSPT bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-0110X1 0110--00B

▲2: IICBnSTR0 register = 1-0110X1 0100--00B

▲3: IICBnSTR0 register = 1-0110XX 0100--00B (IICBnSPT bit = 1)

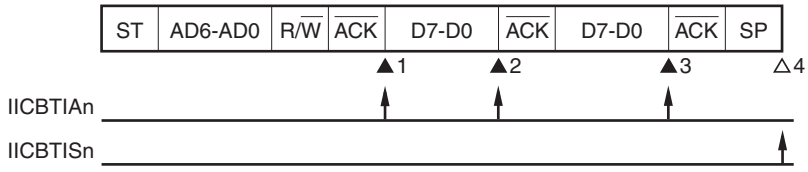
△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.2 Single transfer mode (slave device operation: during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

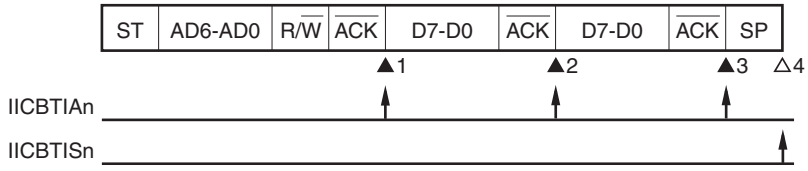
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0101X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X1 0100--00B

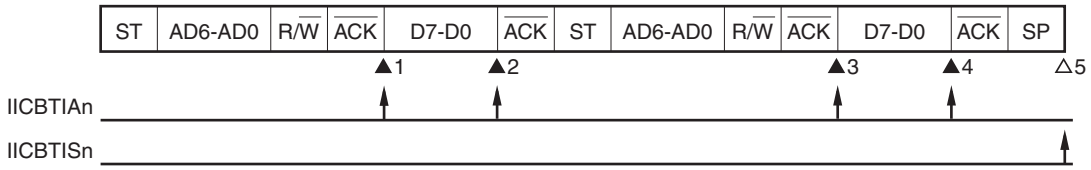
▲3: IICBnSTR0 register = 0-0101XX 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address match)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

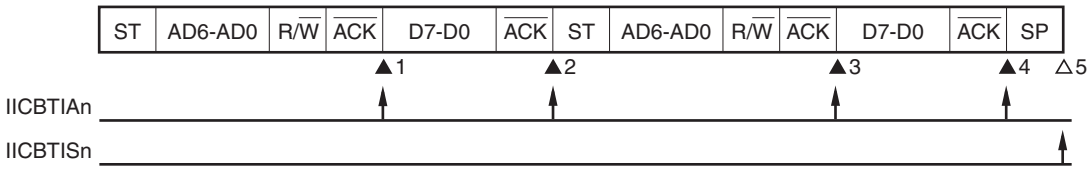
▲3: IICBnSTR0 register = 0-0101X1 0110--00B

▲4: IICBnSTR0 register = 0-0101X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101XX 0100--00B

▲3: IICBnSTR0 register = 0-0101X1 0110--00B

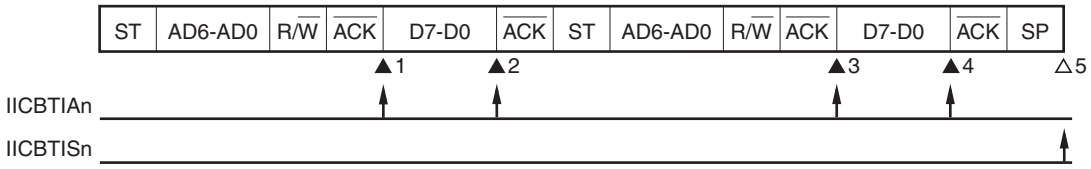
▲4: IICBnSTR0 register = 0-0101XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

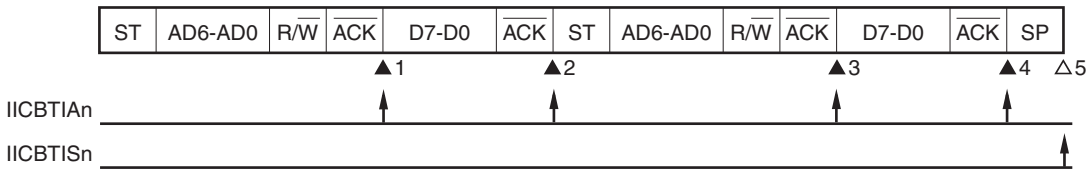
▲3: IICBnSTR0 register = 0-0110X1 0110--00B

▲4: IICBnSTR0 register = 0-0110X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101XX 0100--00B

▲3: IICBnSTR0 register = 0-0110X1 0110--00B

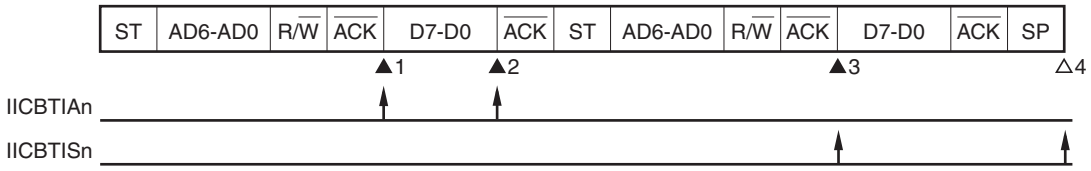
▲4: IICBnSTR0 register = 0-0110XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

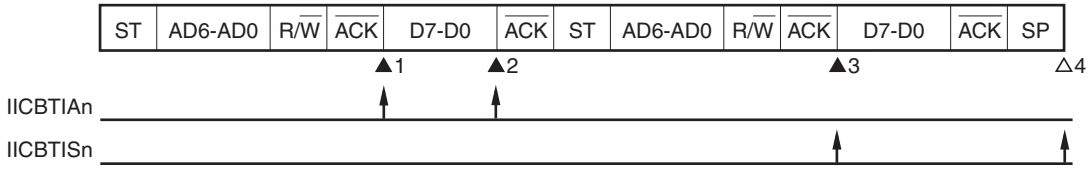
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0101X1 0110--00B

▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

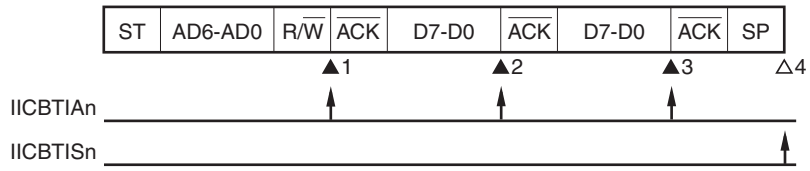
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.3 Single transfer mode (slave device operation: during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

The IICBn always participates in communications when it receives an extension code.

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

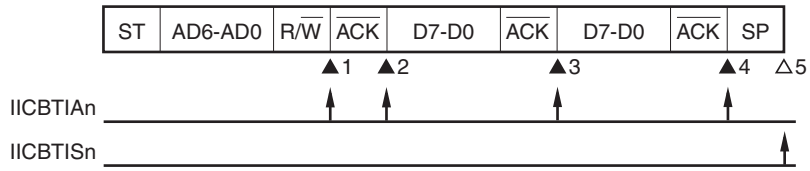
▲2: IICBnSTR0 register = 0-0110X0 0100--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

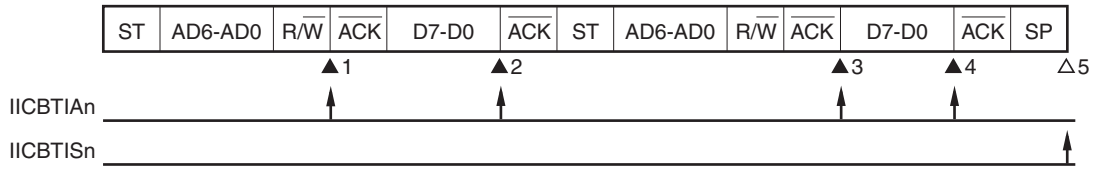
▲4: IICBnSTR0 register = 0-0110XX 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address match)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

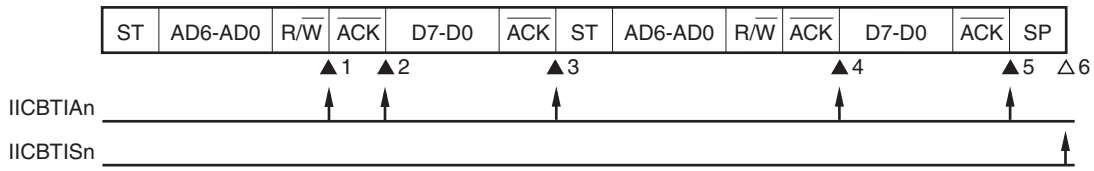
▲3: IICBnSTR0 register = 0-0101X1 0110--00B

▲4: IICBnSTR0 register = 0-0101X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

▲4: IICBnSTR0 register = 0-0101X1 0110--00B

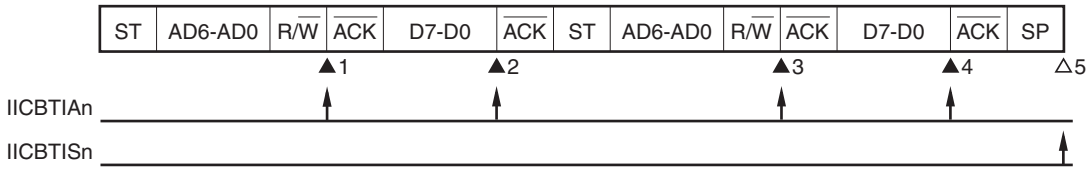
▲5: IICBnSTR0 register = 0-0101XX 0100--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

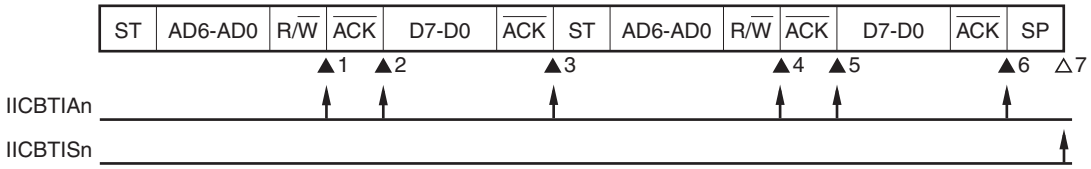
▲3: IICBnSTR0 register = 0-0110X0 0110--00B

▲4: IICBnSTR0 register = 0-0110X0 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110XX 0100--00B

▲4: IICBnSTR0 register = 0-0110X0 0110--00B

▲5: IICBnSTR0 register = 0-0110X1 0110--00B

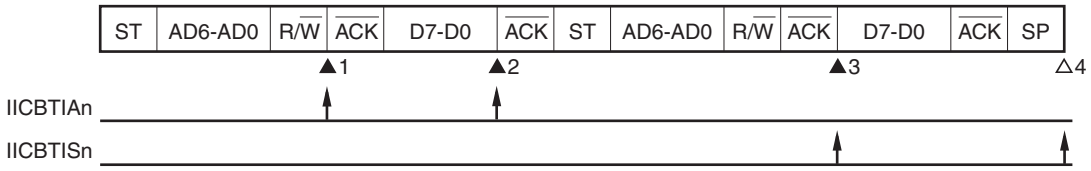
▲6: IICBnSTR0 register = 0-0110XX 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

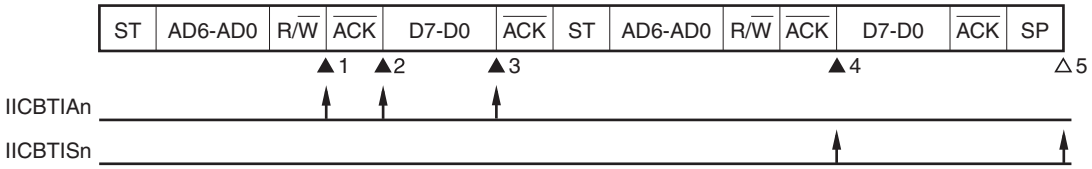
▲2: IICBnSTR0 register = 0-0110X0 0100--00B

▲3: IICBnSTR0 register = 0-0000X0 0110--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-0110X0 0110--00B

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0000X0 0100--00B

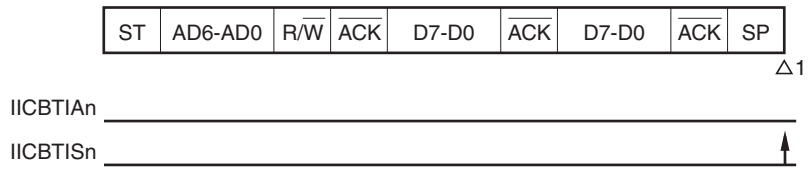
▲4: IICBnSTR0 register = 0-0000X0 0110--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.4 Single transfer mode (non-participation in communications)

(1) Start ~ Code ~ Data ~ Data ~ Stop



△1: IICBnSTR0 register = 0-000000 0001--00B

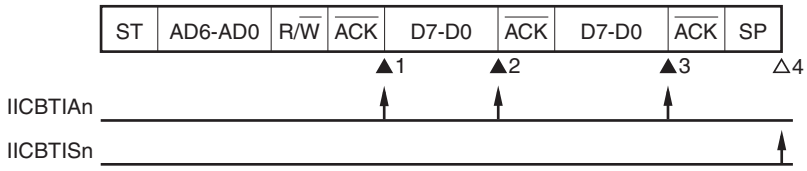
Note △ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

29.9.5 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): operation as slave after arbitration loss)

When the IICBn operates as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Address match after arbitration loss

<1> When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0101X1 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

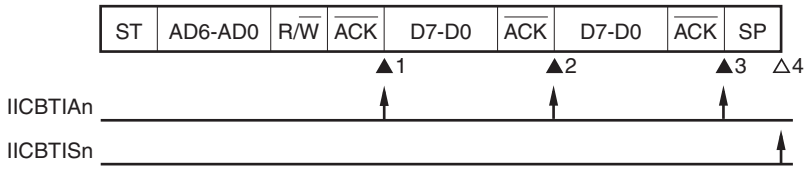
▲2: IICBnSTR0 register = 0-0101X0 0100--00B

▲3: IICBnSTR0 register = 0-0101X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note
- ▲ Always output
- △ Output only when IICBnCTL0.IICBnSLSI = 1
- \- Undefined
- X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0101X1 0110--01B (IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0101X1 0100--00B

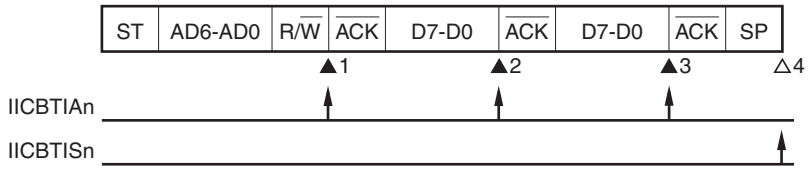
▲3: IICBnSTR0 register = 0-0101XX 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Note
- ▲ Always output
- △ Output only when IICBnSLSI = 1
- \- Undefined
- X don't care

(2) Upon extension code detection after arbitration loss

<1> When IICBnCTL0.IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 0-0110X0 0110--01B (IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0110X0 0100--00B

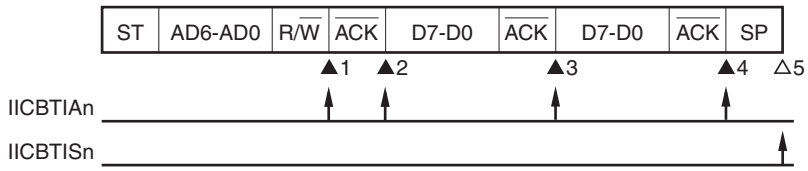
▲3: IICBnSTR0 register = 0-0110X0 0100--00B

△4: IICBnSTR0 register = 0-000000 0001--00B

Notes **1.** ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

2.n = 0 to 5

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-0110X0 0110--01B (IICBnCLAF bit = 1)

▲2: IICBnSTR0 register = 0-0110X1 0110--00B

▲3: IICBnSTR0 register = 0-0110X0 0100--00B

▲4: IICBnSTR0 register = 0-0110XX 0100--00B

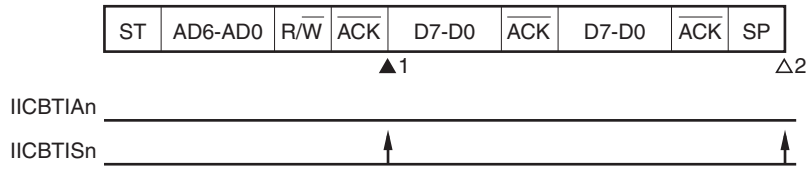
△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

29.9.6 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss)

When the IICBn is used as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Arbitration loss during transmission of slave address



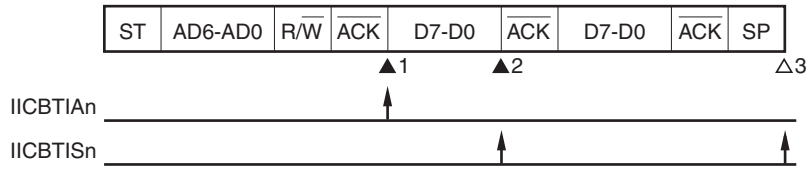
▲1: IICBnSTR0 register = 0-0000X1 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Arbitration loss during data transfer

<1> When IICBnSLWT bit is 0



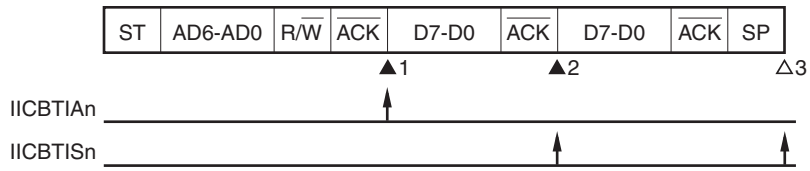
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

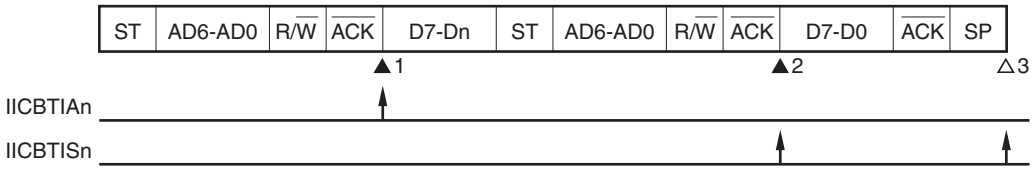
▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(3) Arbitration loss for the restart condition during data transfer

<1> When IICBnSLWT bit is 1 (extension code mismatch, address mismatch)



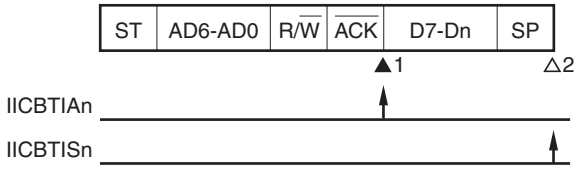
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnCLAF bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(4) Arbitration loss for the stop condition during data transfer



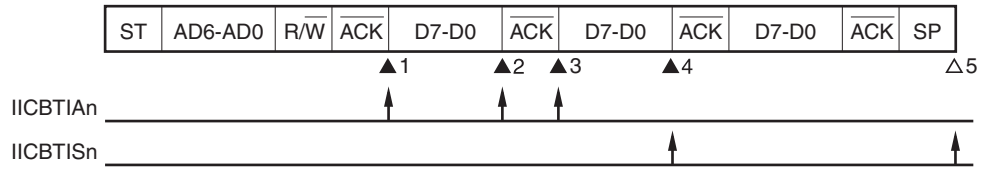
▲1: IICBnSTR0 register = 1-1000X1 0110--00B

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output regardless of setting of IICBnSLSI bit
\- Undefined
X don't care

(5) Arbitration loss because the SDAn signal is low level when attempting to output restart condition

<1> When IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnSLWT bit = 1)

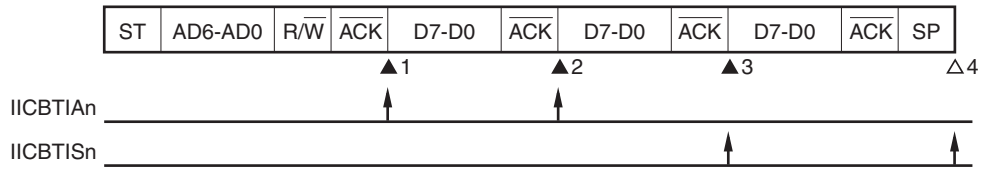
▲3: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnSLWT bit = 0, IICBnTRG.IICBnSTT bit = 1)

▲4: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnCLAF bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnSLWT bit = 0, IICBnSTT bit = 1)

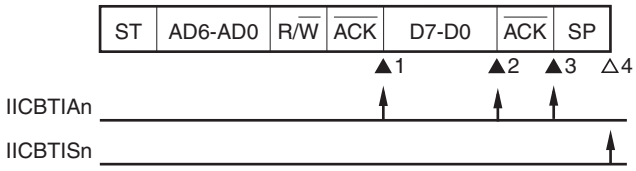
▲3: IICBnSTR0 register = 0-0000X0 0100--01B (IICBnCLAF bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(6) Arbitration loss for the stop condition when attempting to output restart condition

<1> When IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

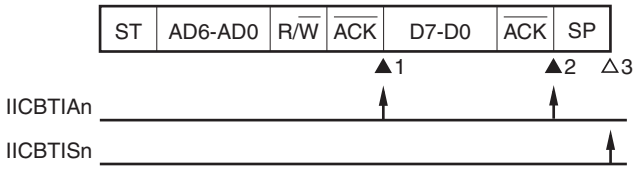
▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnSLWT bit = 0)

▲3: IICBnSTR0 register = 1-0000XX 0100--00B (IICBnSTT bit = 1)

△4: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output regardless of setting of IICBnSLSI bit
- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

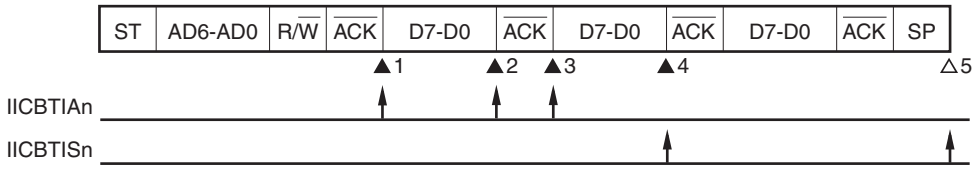
▲2: IICBnSTR0 register = 1-0000XX 0100--00B (IICBnSTT bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output regardless of setting of IICBnSLSI bit
- Undefined
X don't care

(7) Arbitration loss because the SDAn signal is low level when attempting to output stop condition

<1> When IICBnSLWT bit is 0



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000X0 0100--00B (IICBnSLWT bit = 1)

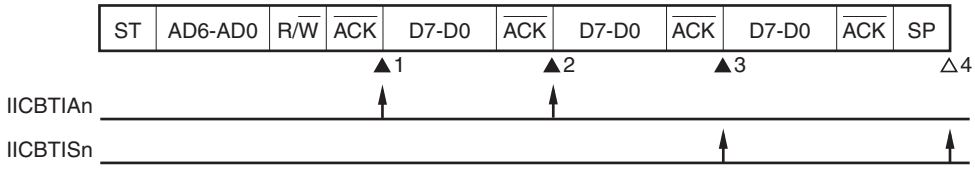
▲3: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnSLWT bit = 0, ICBnTRG.IICBnSPT bit = 1)

▲4: IICBnSTR0 register = 0-0000XX 0100--01B (IICBnCLAF bit = 1)

△5: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

<2> When IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 1-1000XX 0100--00B (IICBnSPT bit = 1)

▲3: IICBnSTR0 register = 0-0000XX 0100--01B (IICBnCLAF bit = 1)

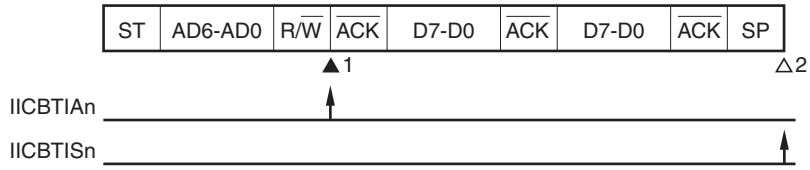
△4: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.7 Single transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1): non-participation in communications after arbitration loss (during extension code transfer))

When the IICBn is used as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Arbitration loss during extension code transfer

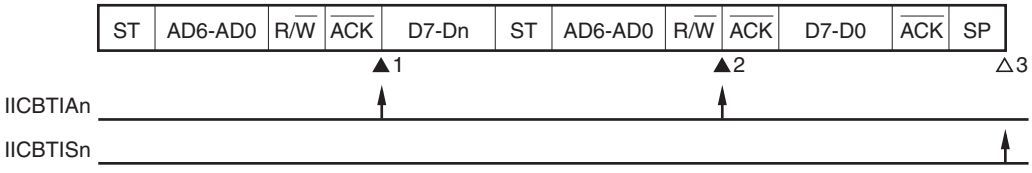


▲1: IICBnSTR0 register = 0-1100X0 0110--01B (IICBnSTRC.IICBnCLAF bit = 1, IICBnTRG.IICBnLRET bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don't care

(2) Arbitration loss for the restart condition during data transfer (extension code match)



▲1: IICBnSTR0 register = 1-1000X1 0110--00B

▲2: IICBnSTR0 register = 0-1100X0 0100--01B (IICBnCLAF bit = 1, IICBnLRET bit = 1)

△3: IICBnSTR0 register = 0-000000 0001--01B

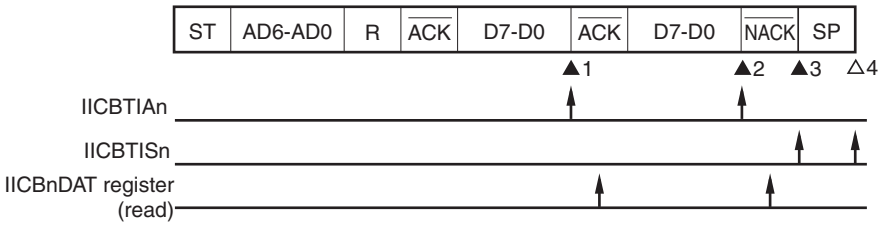
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.8 Continuous transfer mode (master device operation (reception))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICBnSTR0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-100000 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 1-100000 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 1-000000 0100--00B

▲3: IICBnSTR0 register = 1-010000 0100--00B

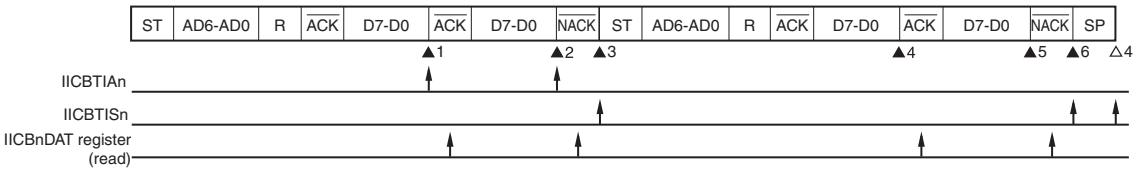
→ IICBnTRG.IICBnSPT register = 1

△4: IICBnTRG.IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(2) Start ~ Address ~ Data × 2 ~ Start ~ Address ~ Data × 2 ~ Stop

<1> When IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-100001 0100--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 1-100000 0100--00B]

IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-010000 0100--00B

▲3: IICBnSTR0 register = 1-010000 0100--00B

→ IICBnTRG.IICBnSTT bit = 1

[▲4: IICBnSTR0 register = 1-100000 0100--00B]

IICBnDAT register read

[▲5: IICBnSTR0 register = 1-100000 0100--00B]

IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-000000 0100--00B

▲6: IICBnSTR0 register = 1-010000 0100--00B

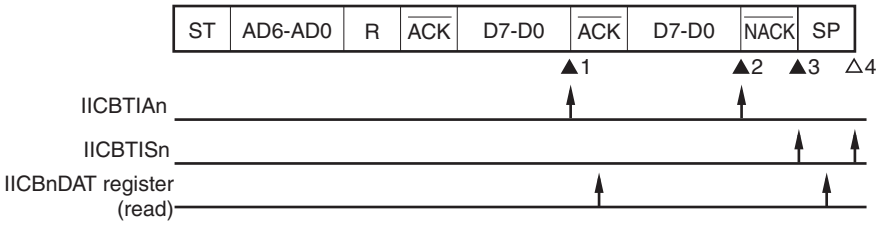
→ IICBnSTT bit = 1

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined

(3) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 1-101001 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 1-0010001 0100--00B

[▲2: IICBnSTR0 register = 1-101000 0100--00B]

IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 1-011000 0100--00B

▲3: IICBnSTR0 register = 1-01000 0100--00B

→ IICBnSPT register = 1

△4: IICBnTRG.IICBnSTR0 register = 0-000000 0001--00B

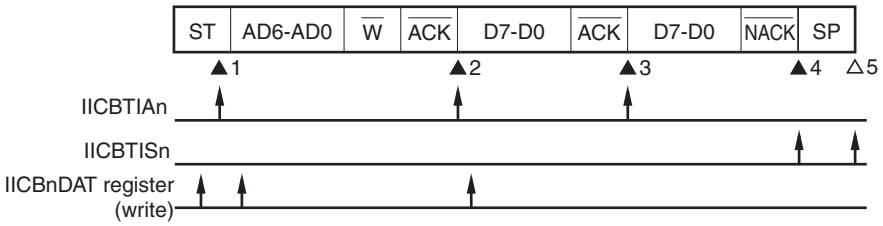
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

29.9.9 Continuous transfer mode (master device operation (transmission))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICBnSTR0.IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

▲4: IICBnSTR0 register = 1-010010 0100--00B

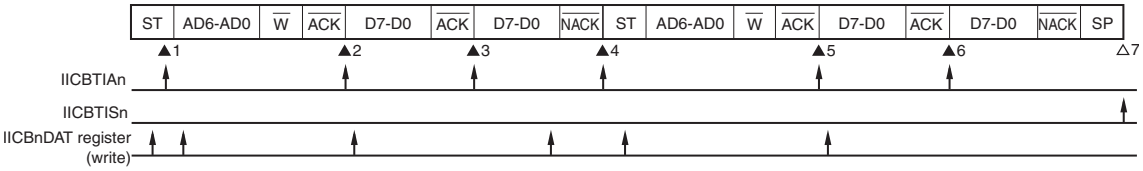
IICBnTRG.IICBnSPT register = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined
X don’t care

(2) Start ~ Address ~ Data × 2 ~ Start ~ Address ~ Data × 2 ~ Stop

<1> When IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

IICBnSTT bit = 1

IICBnDAT register write (address)

[▲4: IICBnSTR0 register = 1-000010 010X--00B]

IICBnDAT register write

[▲5: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲6: IICBnSTR0 register = 1-000011 0110--00B]

IICBnSPT register = 1

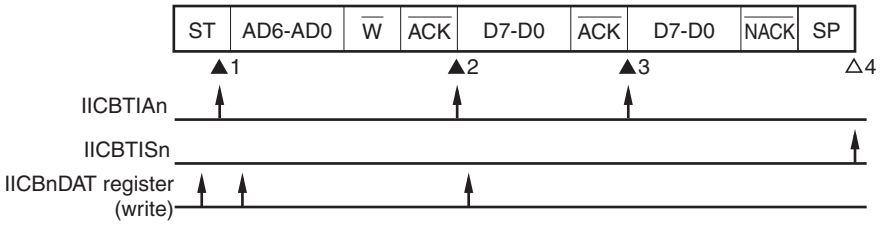
IICBnDAT register write

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(3) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICBnSLWT bit is 1



IICBnDAT register write (address)

[▲1: IICBnSTR0 register = X-0000X0 0X0X--00B]

IICBnDAT register write

[▲2: IICBnSTR0 register = 1-000011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 1-000011 0100--00B]

IICBnSPT register = 1

△4: IICBnSTR0 register = 0-000000 0001--00B

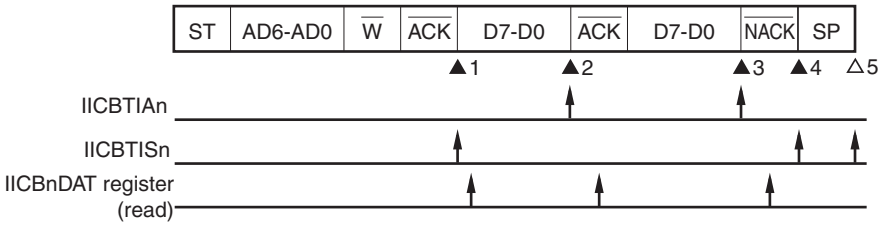
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.10 Continuous transfer mode (slave device operation (reception): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-100101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0100--00B

[▲3: IICBnSTR0 register = 0-100100 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0100-00B

▲4: IICBnSTR0 register = 0-010100 0100-00B

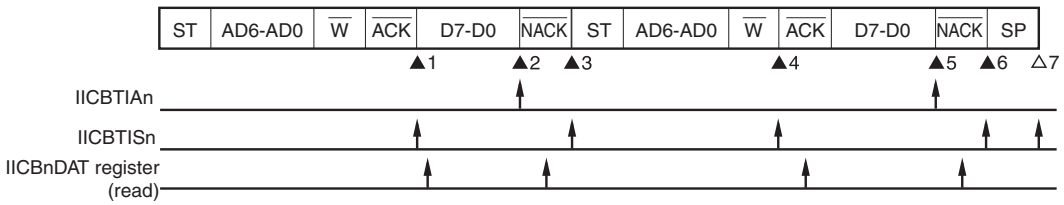
IICBnTRG.IICBnWRET register = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
- Undefined

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address match)



[▲1: IICBnSTR0 register = 0-110101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100101 0100--00B]

IICBnSLAC bit = 0

→ IICBnDAT register read

▲3: IICBnSTR0 register = 0-110101 0110--00B

IICBnWRET register = 1

[▲4: IICBnSTR0 register = 0-100100 0110--00B]

IICBnSLAC bit = 0

IICBnDAT register read

→ IICBnSTR0 register = 0-000100 0110--00B

[▲5: IICBnSTR0 register = 0-100100 0100--00B]

▲6: IICBnSTR0 register = 0-010100 0100--00B

IICBnWRET register = 1

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, extension code reception)

[▲1: IICBnSTR0 register = 0-100101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-100100 0110--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-100100 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-111000 0100--00B

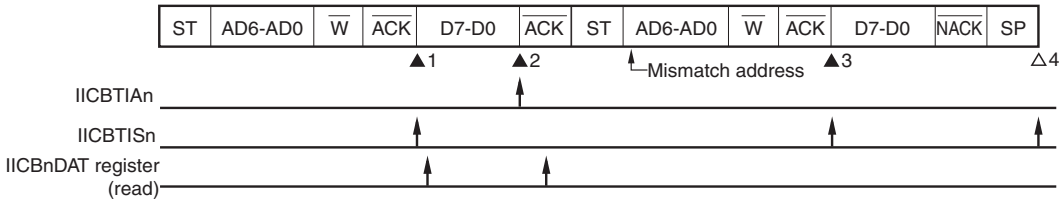
IICBnWRET register = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



[▲1: IICBnSTR0 register = 0-000101 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-000000 0110--00B]

△4: IICBnSTR0 register = 0-000000 0001--00B

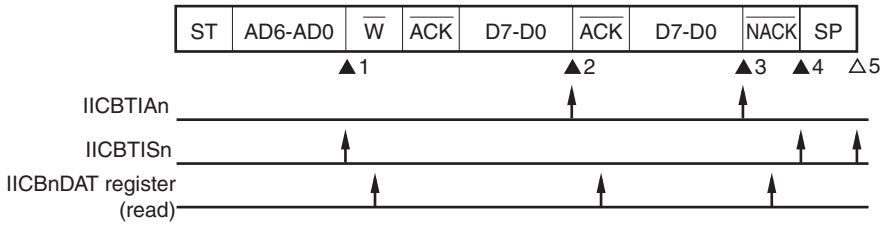
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.11 Continuous transfer mode (slave device operation (reception): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101001 0110--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-10001 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-111000 0100--00B

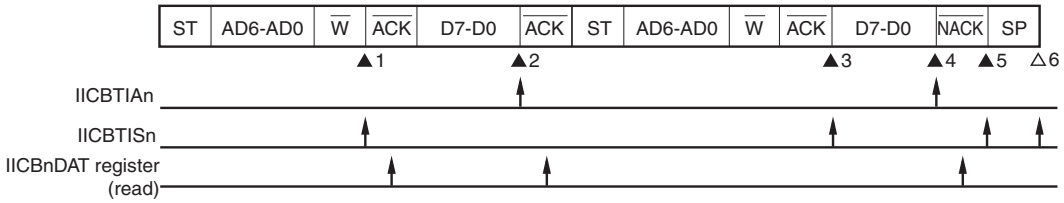
IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address match)



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-011000 0110--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-111001 0100--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-010100 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-110100 0100--00B

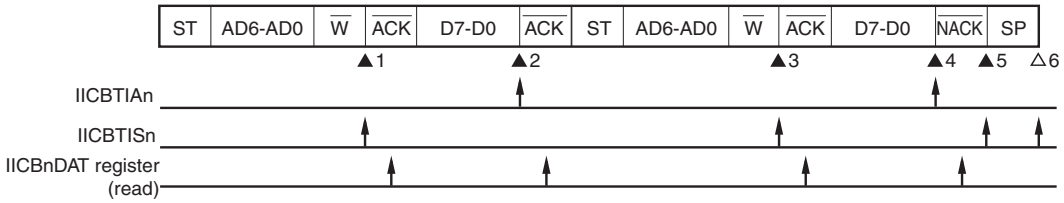
IICBnWRET bit = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, extension code reception)



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-101000 0110--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲4: IICBnSTR0 register = 0-101001 0110--00B]

IICBnDAT register read

▲5: IICBnSTR0 register = 0-011000 0100--00B

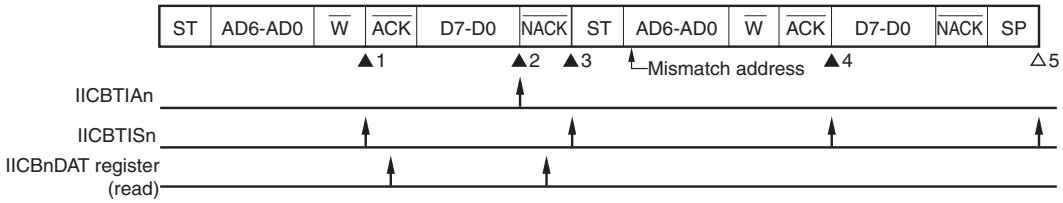
IICBnWRET bit = 1

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 0 (after restart, address mismatch (extension code mismatch))



[▲1: IICBnSTR0 register = 0-101000 0110--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101001 0110--00B]

IICBnSLAC bit = 0

▲3: IICBnSTR0 register = 0-010000 0100--00B

IICBnWRET bit = 1

[▲4: IICBnSTR0 register = 0-000000 0110--00B]

△5: IICBnSTR0 register = 0-000000 0001--00B

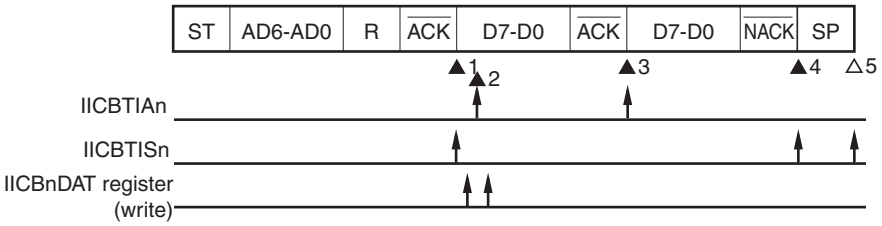
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.12 Continuous transfer mode (slave device operation (transmission): during slave address reception (IICBnSTR0.IICBnSSC0 bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Address ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.ICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-00011X 0100--00B]

IICBnDAT register write

→ IICBnSTR0 register = 0-100011X 0100--00B

▲3: IICBnSTR0 register = 0-000111 0100--00B

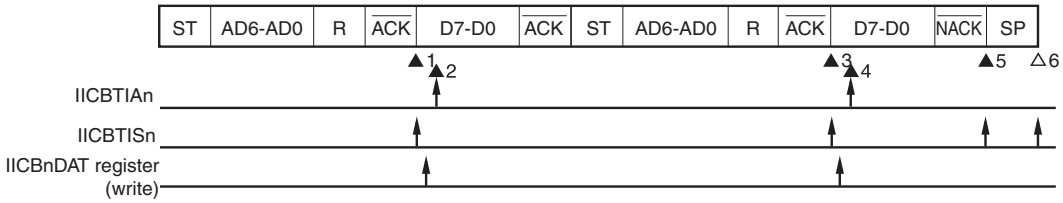
▲4: IICBnSTR0 register = 0-010110 0100--00B

△5: IICBnSTR0 register = 0-000000 0001--00B

Note
- ▲ Always output
- △ Output only when IICBnCTL0.IICBnSLSI = 1
- \- Undefined
- X don't care

(2) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-010111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-00111X 01X0--00B]

▲3: IICBnSTR0 register = 0-010111 0110--00B

IICBnDAT register write

[▲4: IICBnSTR0 register = 0-100101 01X0--00B]

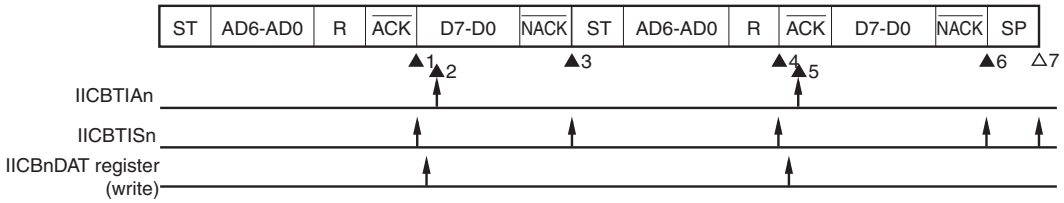
▲5: IICBnSTR0 register = 0-110100 0100--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

(3) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-100111 0100--00B]

▲3: IICBnSTR0 register = 0-111010 0110--00B

▲4: IICBnSTR0 register = 0-111010 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-111011 0110--00B]

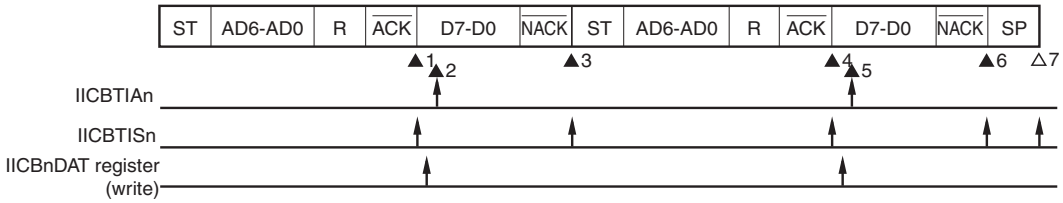
▲6: IICBnSTR0 register = 0-111010 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(4) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-110111 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-100111 0100--00B]

▲3: IICBnSTR0 register = 0-000010 0100--00B

▲4: IICBnSTR0 register = 0-000011 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-00001X 0100--00B]

▲6: IICBnSTR0 register = 0-000010 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

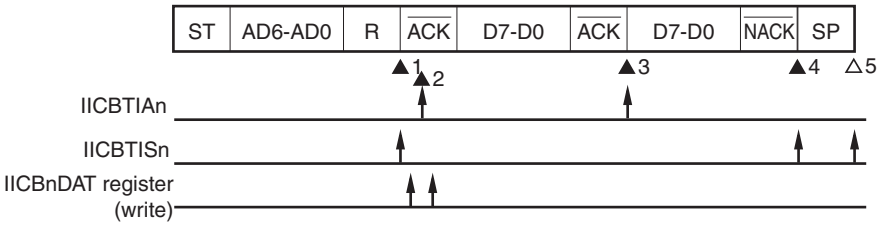
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don't care

29.9.13 Continuous transfer mode (slave device operation (transmission): during extension code reception (IICBnSTR0.IICBnSSEX bit = 1))

Note The interrupts enclosed in brackets ([ ]) do not make the IICBn enter the wait state. Note that these interrupts are not output when a stop condition is detected.

(1) Start ~ Code ~ Data ~ Data ~ Stop

<1> When IICBnCTL0.IICBnSLWT bit is 1



▲1: IICBnSTR0 register = 0-011010 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011011 0110--00B]

IICBnDAT register write

[▲3: IICBnSTR0 register = 0-011011 0100--00B]

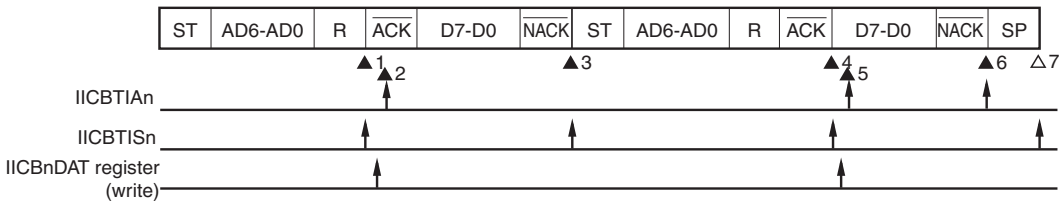
▲4: IICBnSTR0 register = 0-111010 0100--00B

△5: IICBnSTR0 register = 0-000010 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, address match)



▲1: IICBnSTR0 register = 0-011000 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

▲3: IICBnSTR0 register = 0-011000 0100--00B

▲4: IICBnSTR0 register = 0-010101 0110--00B

IICBnDAT register write

[▲5: IICBnSTR0 register = 0-010101 0110--00B]

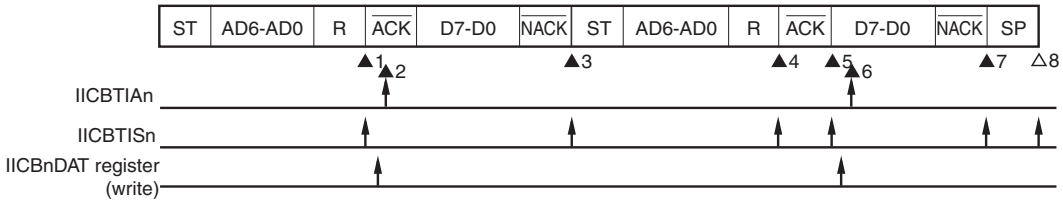
▲6: IICBnSTR0 register = 0-010100 0100--00B

△7: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(3) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, extension code reception)



▲1: IICBnSTR0 register = 0-011000 0110--00B

IICBnDAT register write

[▲2: IICBnSTR0 register = 0-011001 0110--00B]

▲3: IICBnSTR0 register = 0-011000 0100--00B

▲4: IICBnSTR0 register = 0-011000 0110--00B

▲5: IICBnSTR0 register = 0-011001 0110--00B

IICBnDAT register write

[▲6: IICBnSTR0 register = 0-011001 0110--00B]

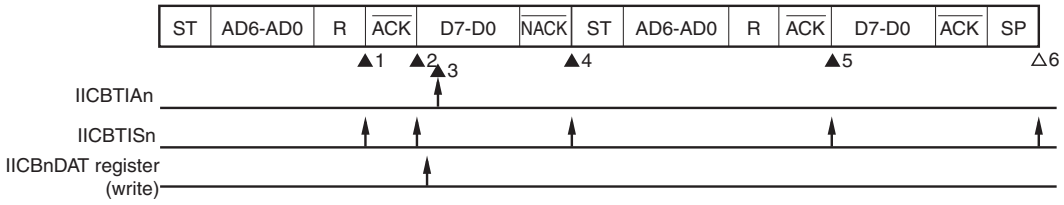
▲7: IICBnSTR0 register = 0-011000 0100--00B

△8: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(4) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop

<1> When IICBnSLWT bit is 1 (after restart, address mismatch (extension code mismatch))



▲1: IICBnSTR0 register = 0-011000 0110--00B

▲2: IICBnSTR0 register = 0-011001 0110--00B

IICBnDAT register write

[▲3: IICBnSTR0 register = 0-011010 0100--00B]

▲4: IICBnSTR0 register = 0-000000 0100--00B

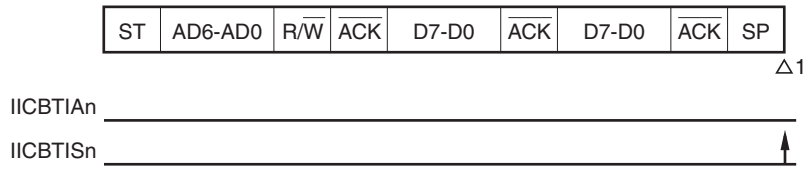
▲5: IICBnSTR0 register = 0-000000 0110--00B

△6: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

29.9.14 Continuous transfer mode (non-participation in communications)

(1) Start ~ Code ~ Data ~ Data ~ Stop



△1: IICBnSTR0 register = 0-0000X0 0001--00B

Note △ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

29.9.15 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): operation as slave after arbitration loss)

When the IICBn is used as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Address match after arbitration loss

<1> During reception, when IICBnCTL0.IICBnSLWT bit is 0

[▲1: IICBnSTR0 register = 0-100101 0110--01B]

IICBnSTRC.IICBnCLAF bit = 1

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-100101 0100--00B]

IICBnCTL0.IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-100100 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-010100 0100--00B

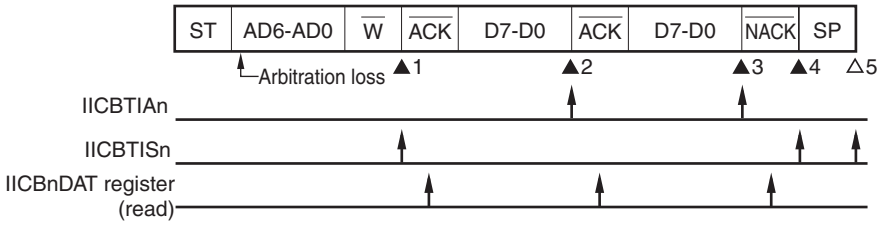
IICBnTRG.IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Upon extension code detection after arbitration loss

<1> During reception, when IICBnSLWT bit is 0



[▲1: IICBnSTR0 register = 0-101000 0110--01B]

IICBnCLAF bit = 1

IICBnDAT register read

[▲2: IICBnSTR0 register = 0-101000 0110--00B]

IICBnSLAC bit = 0

IICBnDAT register read

[▲3: IICBnSTR0 register = 0-101000 0100--00B]

IICBnDAT register read

▲4: IICBnSTR0 register = 0-011000 0100--00B]

IICBnWRET bit = 1

△5: IICBnSTR0 register = 0-000000 0001--00B

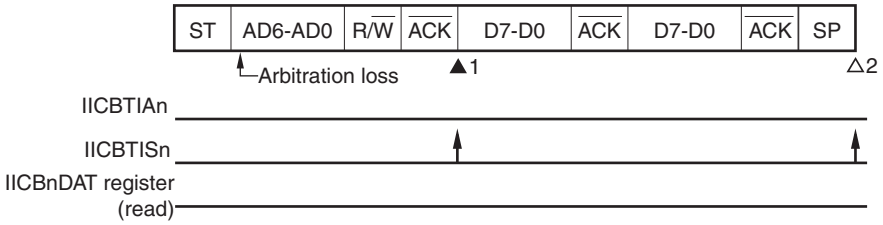
Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

29.9.16 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss)

When the IICBn is used as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Arbitration loss during slave address transmission

<1> During reception, when IICBnSLWT bit is 0



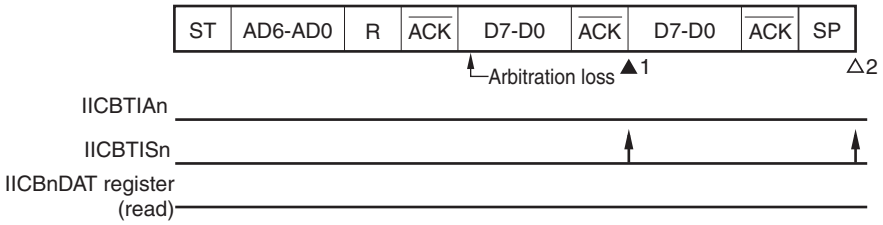
▲1: IICBnSTR0 register = 0-000001 0110--01B (IICBnSTRC.IICBnCLAF bit = 1)

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined

(2) Arbitration loss during data transfer

<1> During reception, when IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 0-000000 0100--01B]

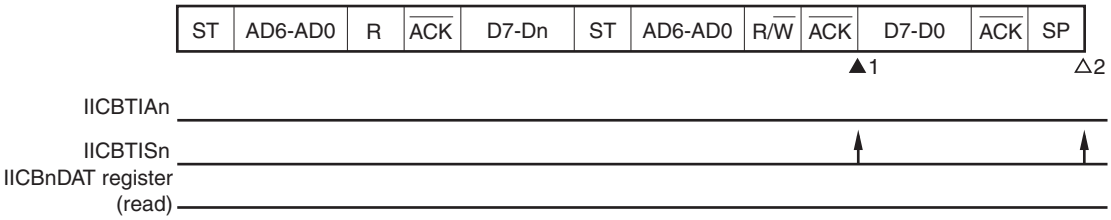
IICBnCLAF bit = 1

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined

(3) Arbitration loss for the restart condition during data transfer

<1> During reception, when IICBnSLWT bit is 1 (extension code mismatch, address mismatch)



[▲1: IICBnSTR0 register = 0-000001 0100--01B]

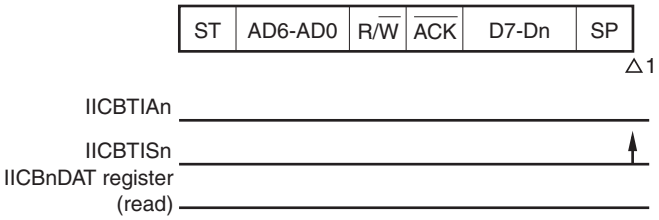
IICBnCLAF bit = 1

△2: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined

(4) Arbitration loss for the stop condition during data transfer

<1> During reception, when IICBnSLWT bit is 1



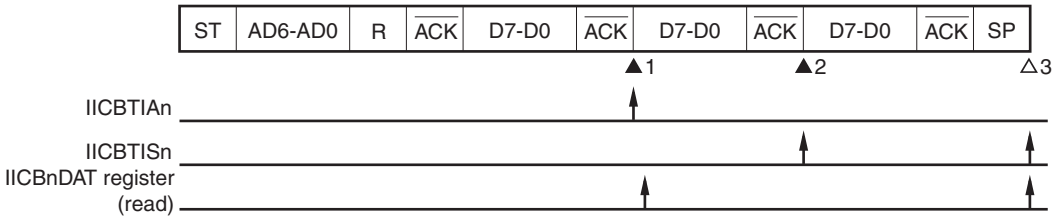
△1: IICBnSTR0 register = 0-000000 0001--01B

IICBnCLAF bit = 1

Note △ Output only when IICBnSLSI = 1
- Undefined

(5) Arbitration loss because the SDAn signal is low level when attempting to output restart condition

<1> When IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-1000XX 0100--00B]

IICBnDAT register read

IICBnTRG.IICBnSTT bit = 1

▲2: IICBnSTR0 register = 0-000000 0100--01B

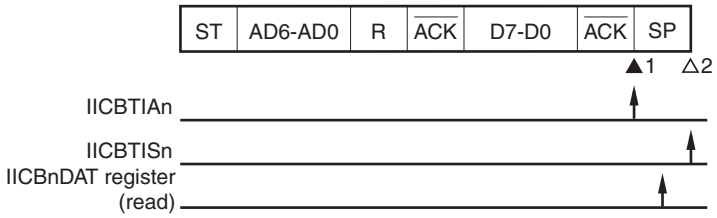
IICBnCLAF bit = 1

△3: IICBnSTR0 register = 0-000000 0001--00B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

(6) Arbitration loss for the stop condition when attempting to output restart condition

<1> When IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-000001 0100--00B]

IICBnDAT register read

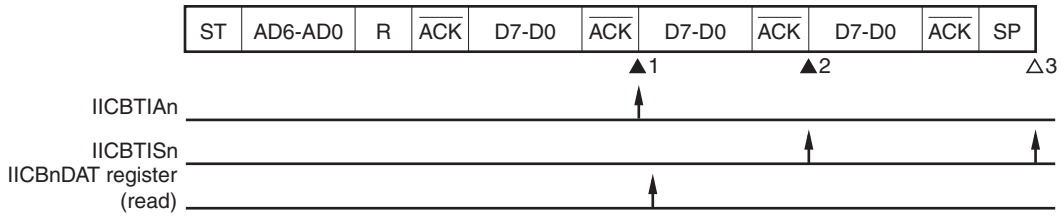
IICBnSTT bit = 1

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined

(7) Arbitration loss because the SDAn signal is low level when attempting to output stop condition

<1> When IICBnSLWT bit is 1



[▲1: IICBnSTR0 register = 1-1000XX 0100--00B]

IICBnDAT register read

IICBnTRG.IICBnSPT bit = 1

[▲2: IICBnSTR0 register = 0-0000XX 0100--01B(IICBnCLAF bit = 1)

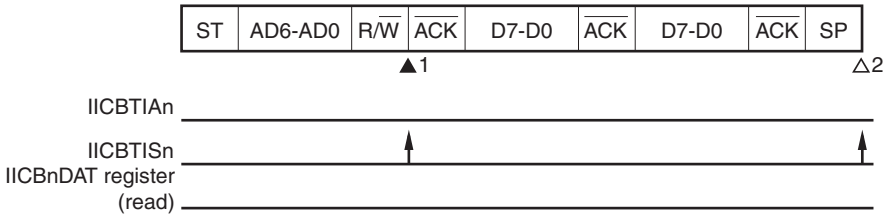
△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
- Undefined
X don't care

29.9.17 Continuous transfer mode (arbitration loss operation (IICBnSTR0.IICBnALDF bit = 1) (when address was transferred during reception): non-participation in communications after arbitration loss (during extension code transfer))

When the IICBn is used as a master in a multi-master system, read the IICSn.MSTSn bit to check the arbitration result each time the INTIICn interrupt is output.

(1) Arbitration loss during extension code transfer



[▲1: IICBnSTR0 register = 0-1000X0 0110--01B]

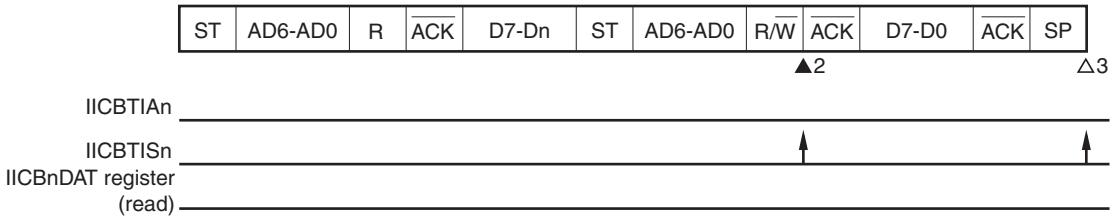
IICBnSTRC.IICBnCLAF bit = 1

IICBnTRG.IICBnLRET bit = 1

△2: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnCTL0.IICBnSLSI = 1
\- Undefined
X don’t care

(2) Arbitration loss for the restart condition during data transfer (extension code match)



▲1: IICBnSTR0 register = 1-0000X1 0110--00B

▲2: IICBnSTR0 register = 0-0100X0 0100--01B (IICBnCLAF bit = 1, IICBnLRET bit = 1)

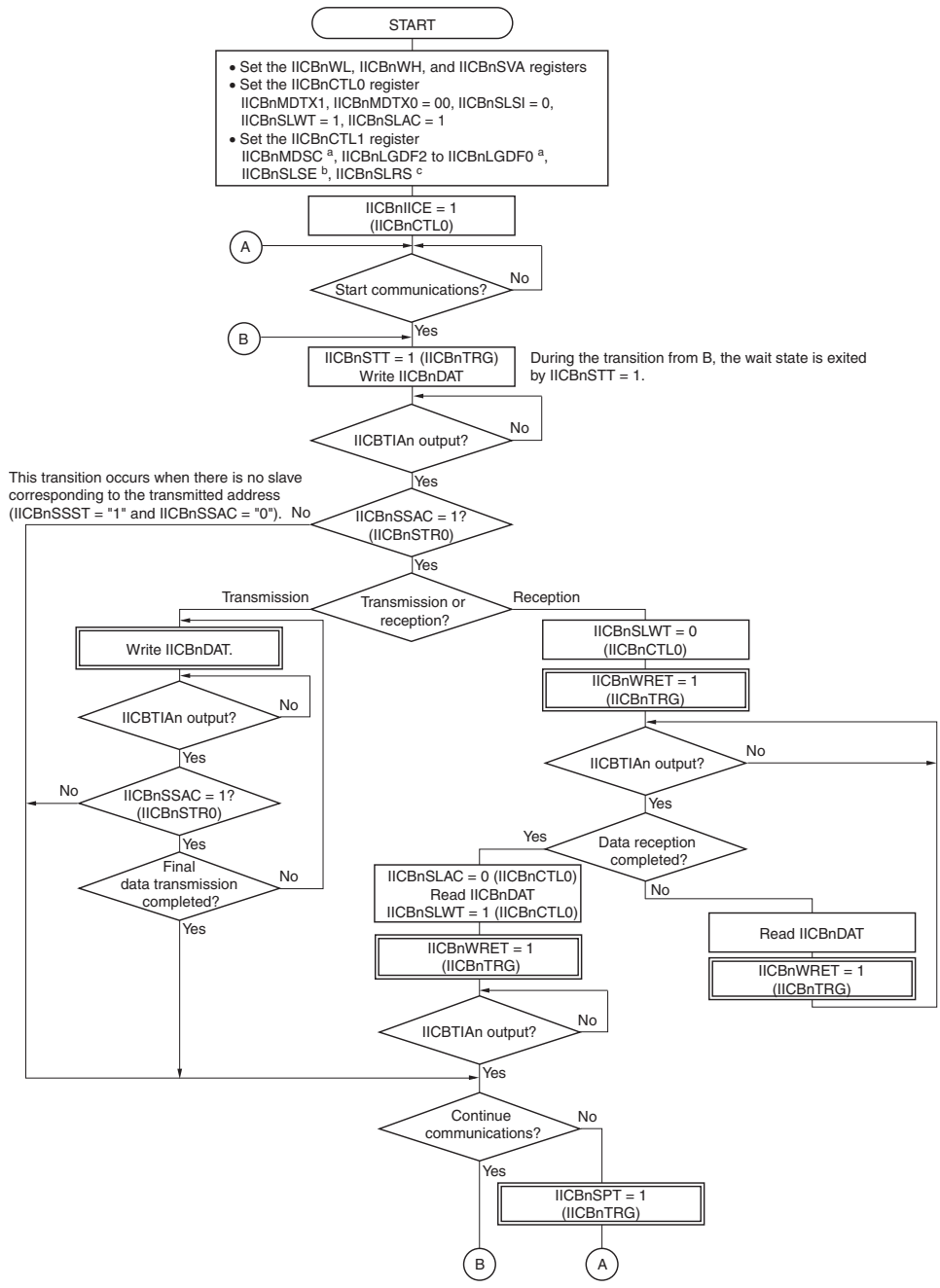
△3: IICBnSTR0 register = 0-000000 0001--01B

Note ▲ Always output
△ Output only when IICBnSLSI = 1
\- Undefined
X don’t care

29.10 Setting Sequence

29.10.1 Single master environment

(1) Master operate setting sequence during single transfer mode

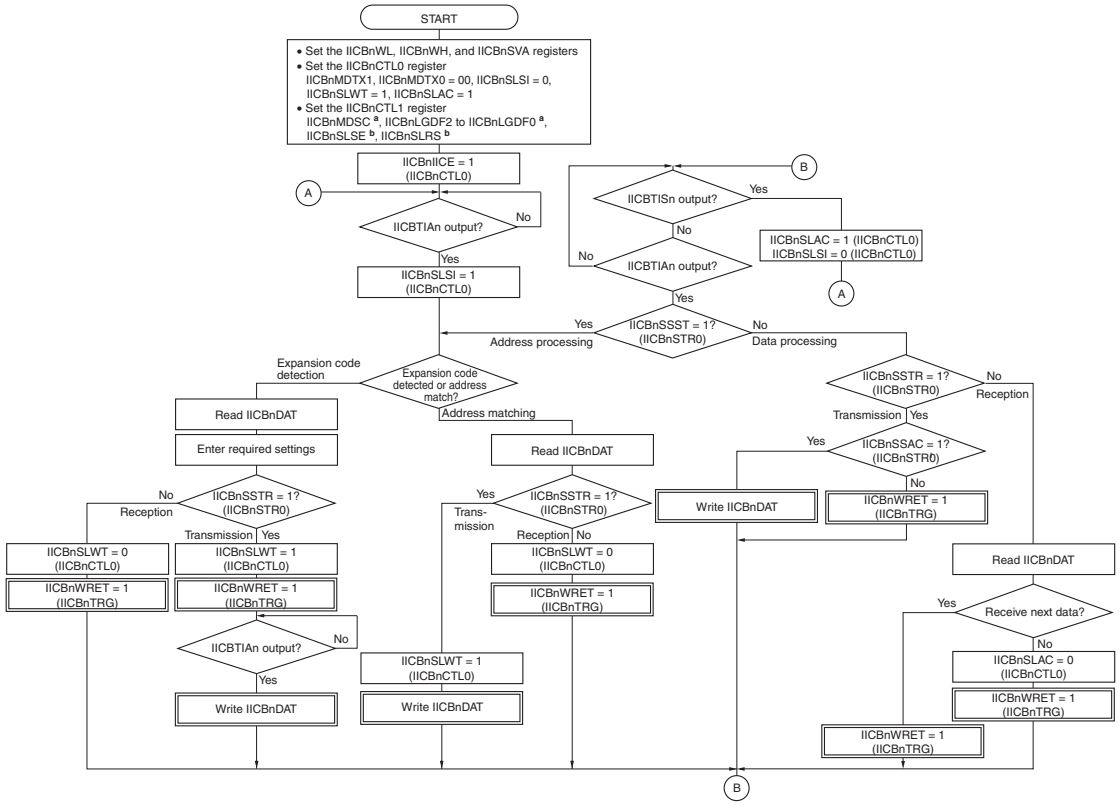


a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 29-15 Master operate setting sequence during single transfer mode (single master environment)

(2) Slave operate setting sequence during single transfer mode

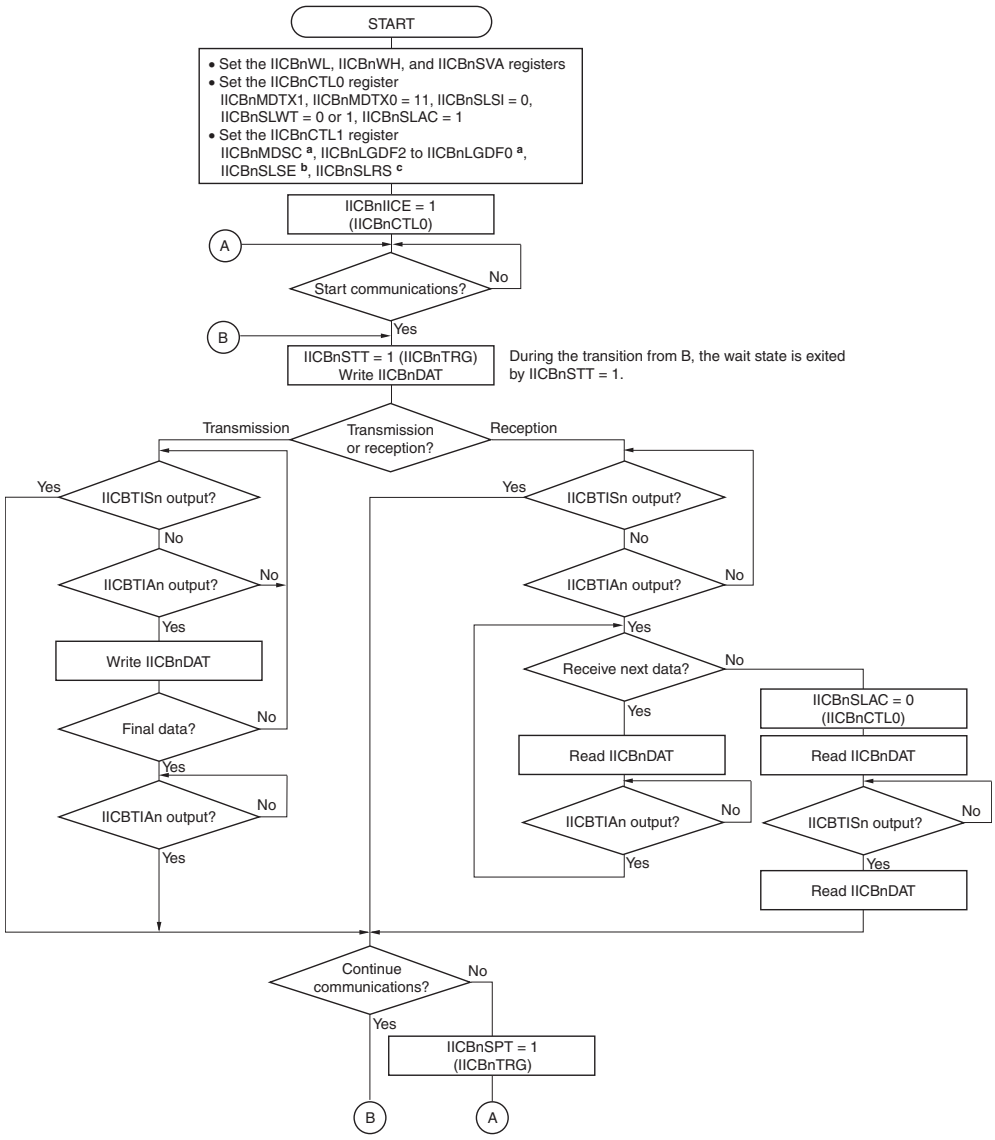


a) Enter settings according to the environment.
b) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 29-16 Slave operate setting sequence during single transfer mode (single master environment)

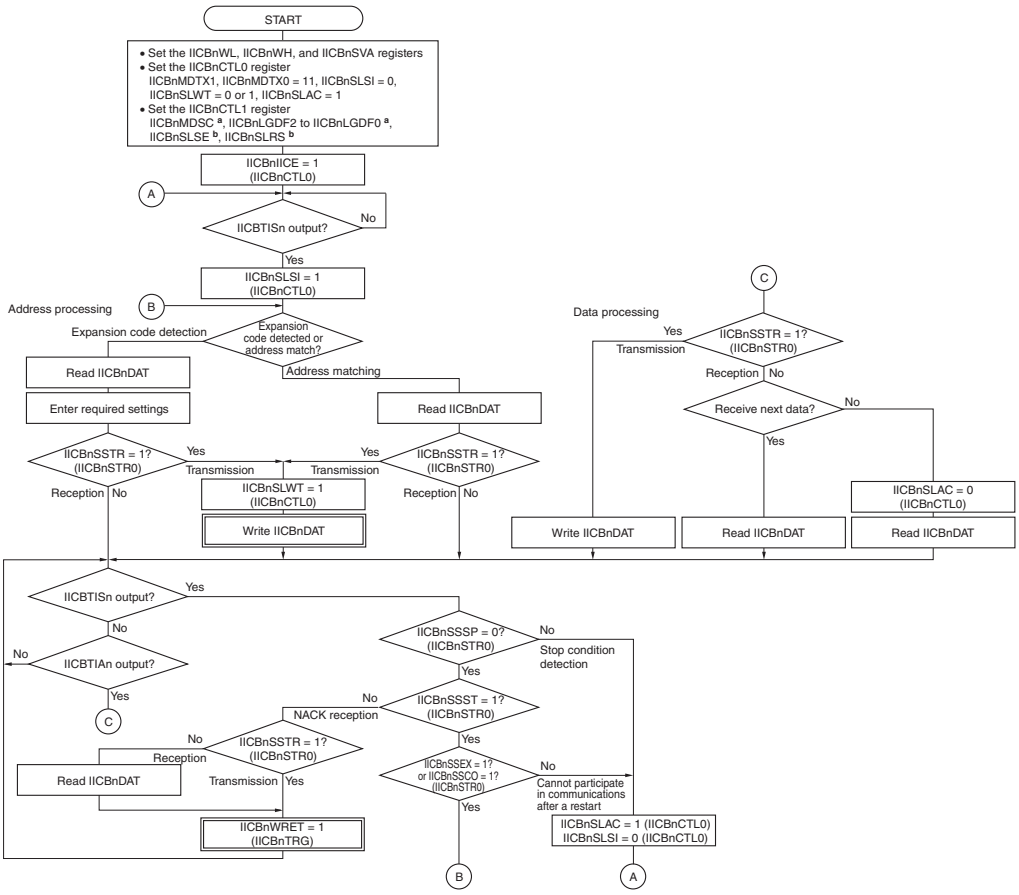
(3) Master operate setting sequence during continuous transfer mode



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) Values are not referenced in this environment.

Figure 29-17 Master operate setting sequence during continuous transfer mode (single master environment)

(4) Slave operate setting sequence during continuous transfer mode



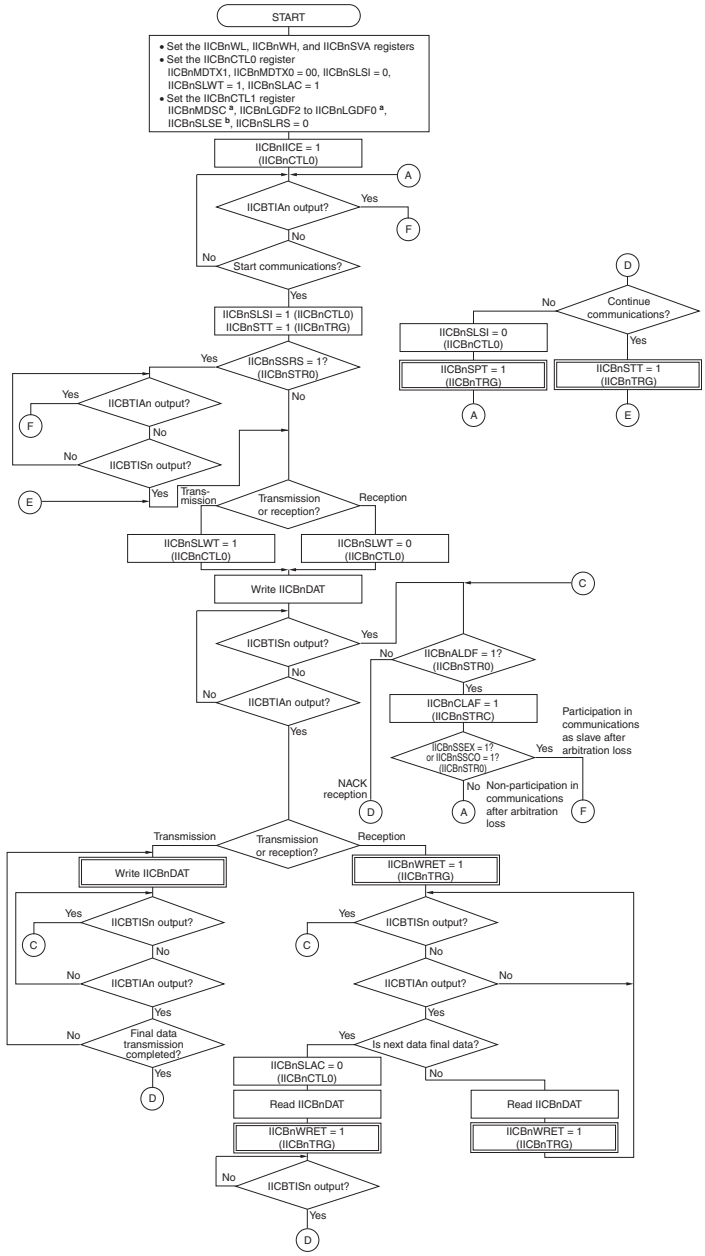
a) Enter settings according to the environment.
b) Values are not referenced in this environment.

Note The double-boxed items indicate processing that exits the wait state.

Figure 29-18 Slave operate setting sequence during continuous transfer mode (single master environment)

29.10.2 Multi-master environment

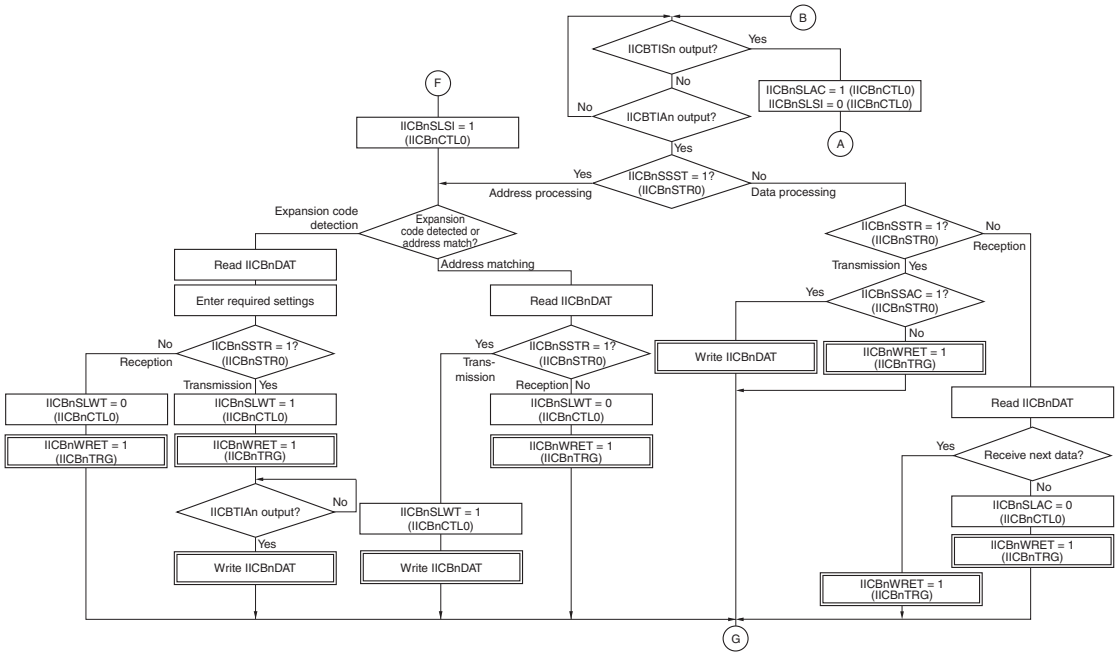
(1) Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

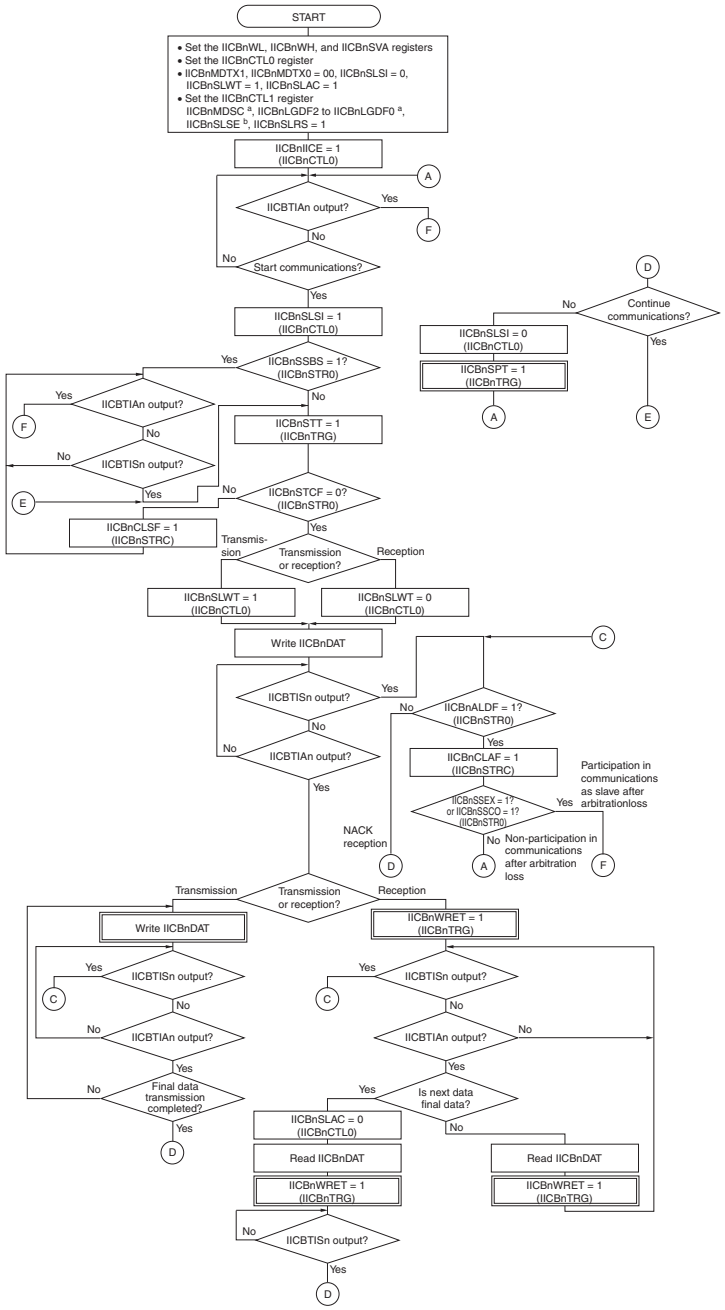
Figure 29-19 Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exiting the wait state.

Figure 29-19 Single transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (2/2)

(2) Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1)

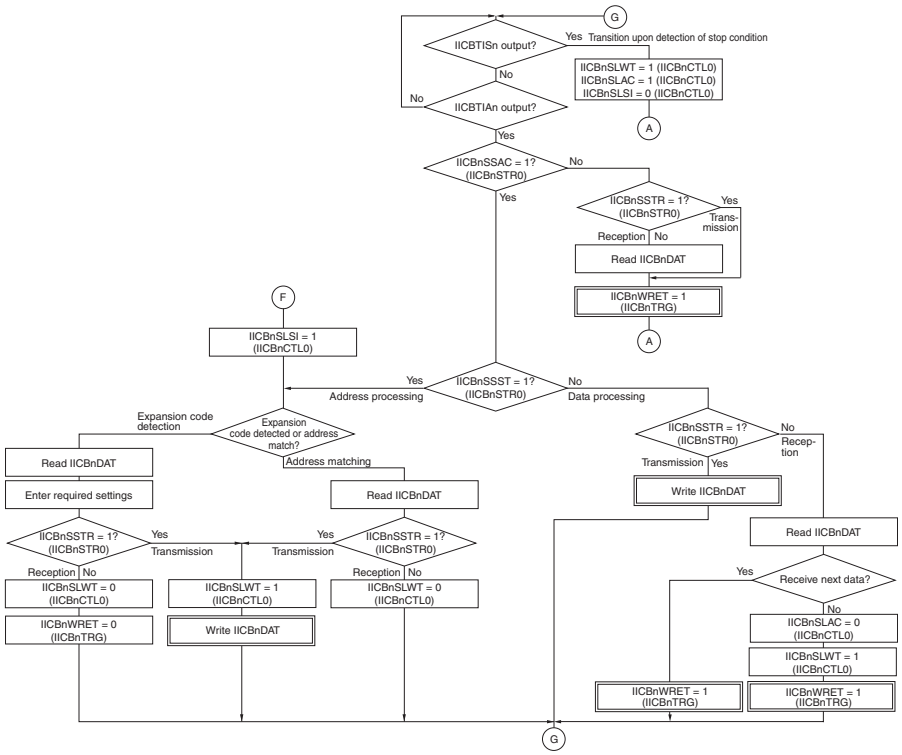


a) Enter settings according to the environment.

b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

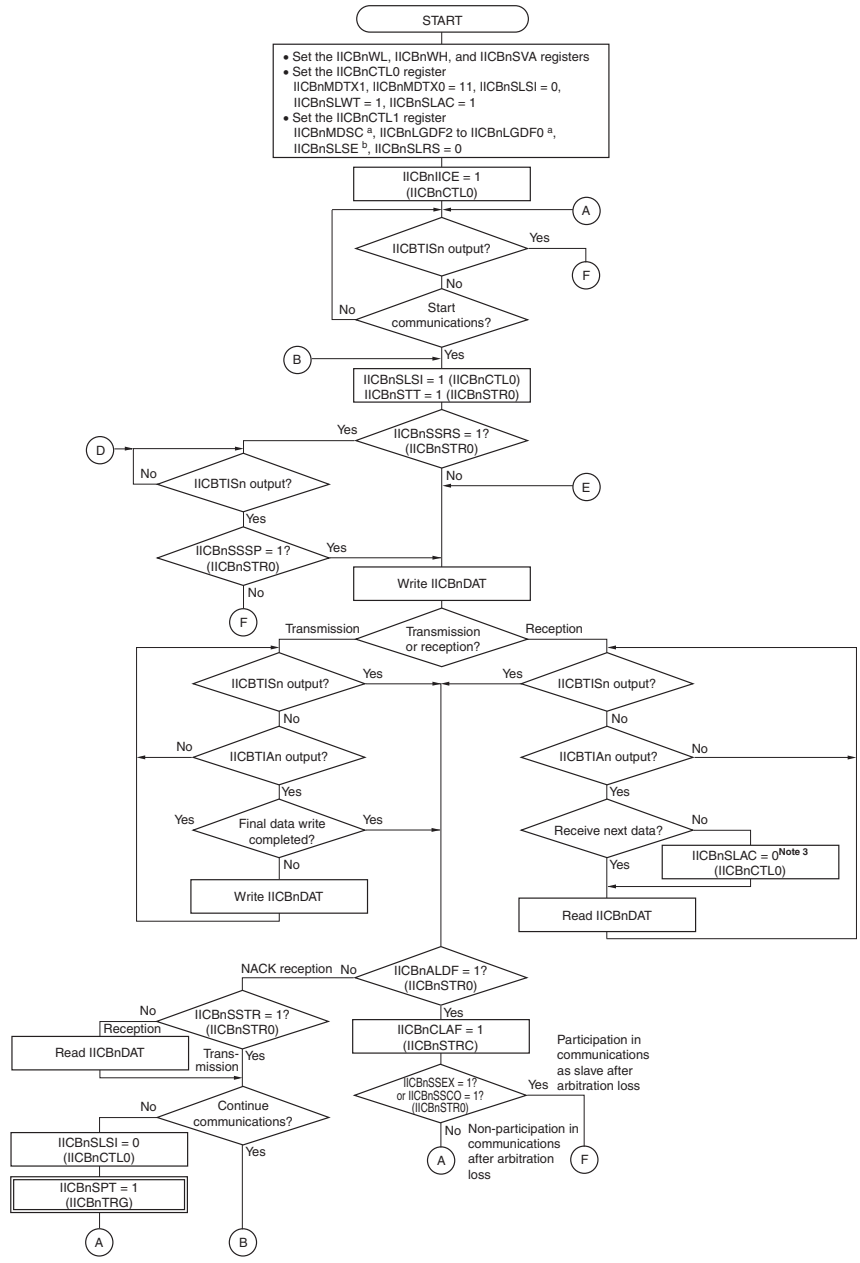
Figure 29-20 Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 29-20 Single transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (2/2)

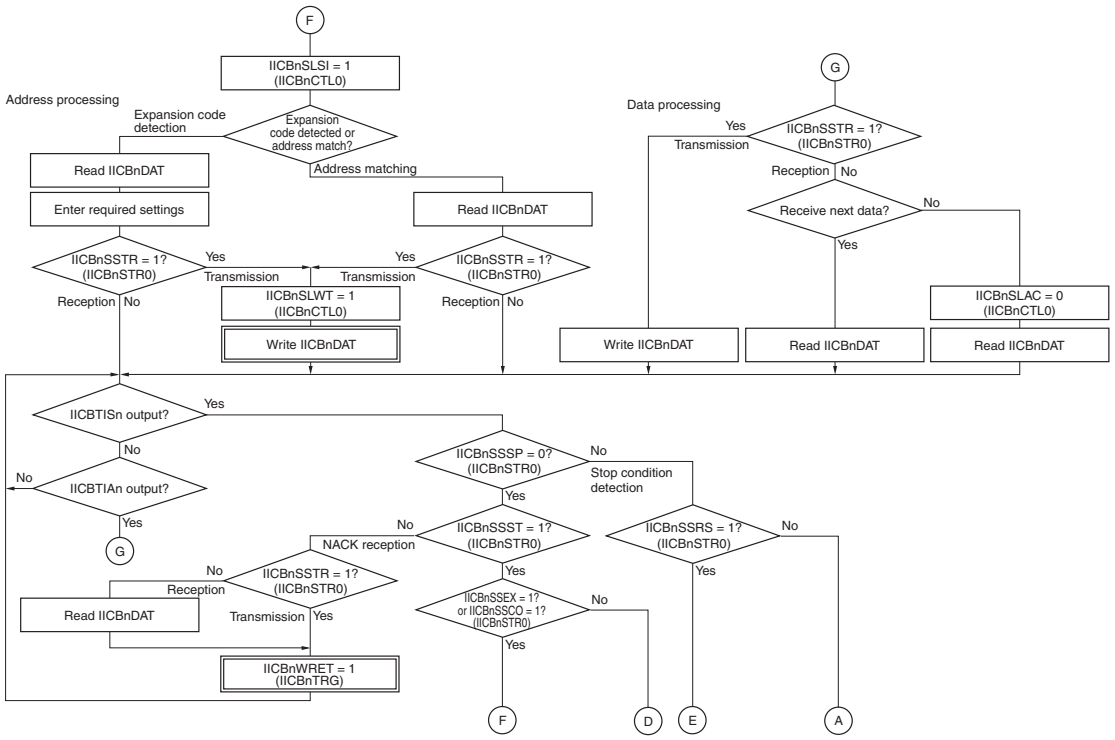
(3) Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.
c) The IICBnCTL0.IICBnSLAC bit must be cleared to 0 before the 8th SCLn clock.

Note The double-boxed items indicate processing that exits the wait state.

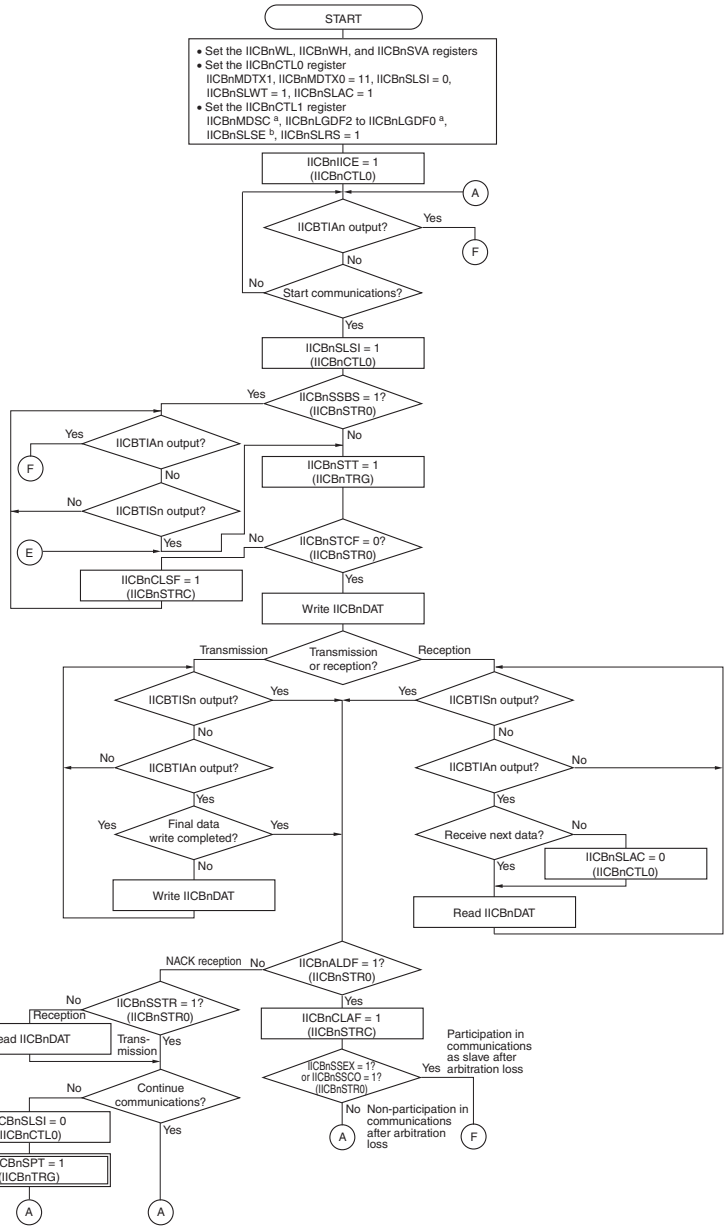
Figure 29-21 Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 29-21 Continuous transfer mode setting sequence when communication reserve function is enabled (IICBnCTL1.IICBnSLRS bit = 0) (multi-master environment) (2/2)

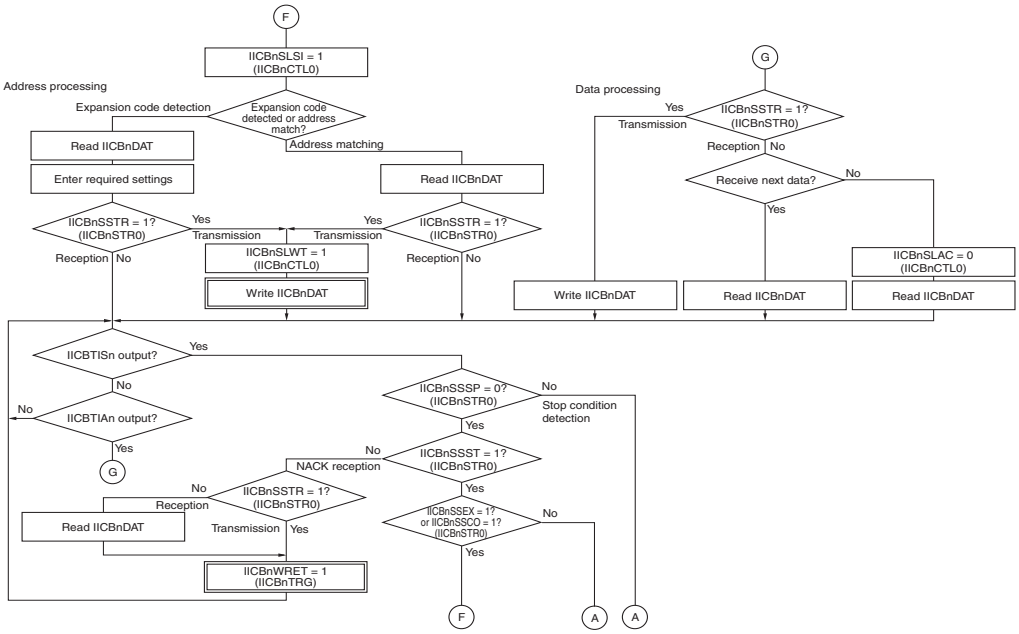
(4) Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1)



a) Enter settings according to the environment.
b) Set 1 to enable output of a start condition in the initial communication state. Otherwise, set 0.

Note The double-boxed items indicate processing that exits the wait state.

Figure 29-22 Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (1/2)



Note The double-boxed items indicate processing that exits the wait state.

Figure 29-22 Continuous transfer mode setting sequence when communication reserve function is disabled (IICBnCTL1.IICBnSLRS bit = 1) (multi-master environment) (2/2)

Chapter 30 FlexRay™ (FLX)

Note FlexRay™ is a trademark of DaimlerChrysler AG.

This chapter contains a generic description of the FlexRay module.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

30.1 V850E2/Fx4-H FLXn Features

Instances This microcontroller has following number of instances of the FlexRay interface FLXn.

Table 30-1 Instances of FLX

FlexRay Interface	
Instance	1
Name	FLX0

Instances index n Throughout this chapter, the instance of the FlexRay module is identified by the index "n" (n = 0), for example, FLXnSUCC1 for the FLXn SUC configuration register 1.

Register addresses All FlexRay module register addresses are given as address offsets to the individual base address <FLXn_base>.
The <FLXn_base> address of FLXn are listed in the following table:

Table 30-2 Register base addresses <FLXn_base>

FLXn instance	<FLXn_base> address
FLX0	FF58 0000$_H$

The FlexRay module FLXn comprises the FlexRay core module E-Ray as the major function. An additional message buffer with the error correction module ECC completes the FlexRay module.

Vendor information The reset value of the FLXnVI register contains product specific vendor information. The vendor information of the V850E2/Fx4-H is given in the following table:

Table 30-3 Vendor information register FLXnVI value

FLXn instance	FLXnVI
FLX0	0105 00xx$_H$

Message buffer size The FlexRay module is equipped with a message buffer of the following size:

Table 30-4 Message buffer size

FLXn instance	Message buffer size
FLX0	8 KB

Clock supply All FlexRay modules provides two clock inputs.

Table 30-5 FLXn clock supply

FLXn instance	FLXn clock	Connected to
FLX0	eray_sclk	Clock Controller CKSCLK_103
	eray_bclk	Clock Controller CKSCLK_102

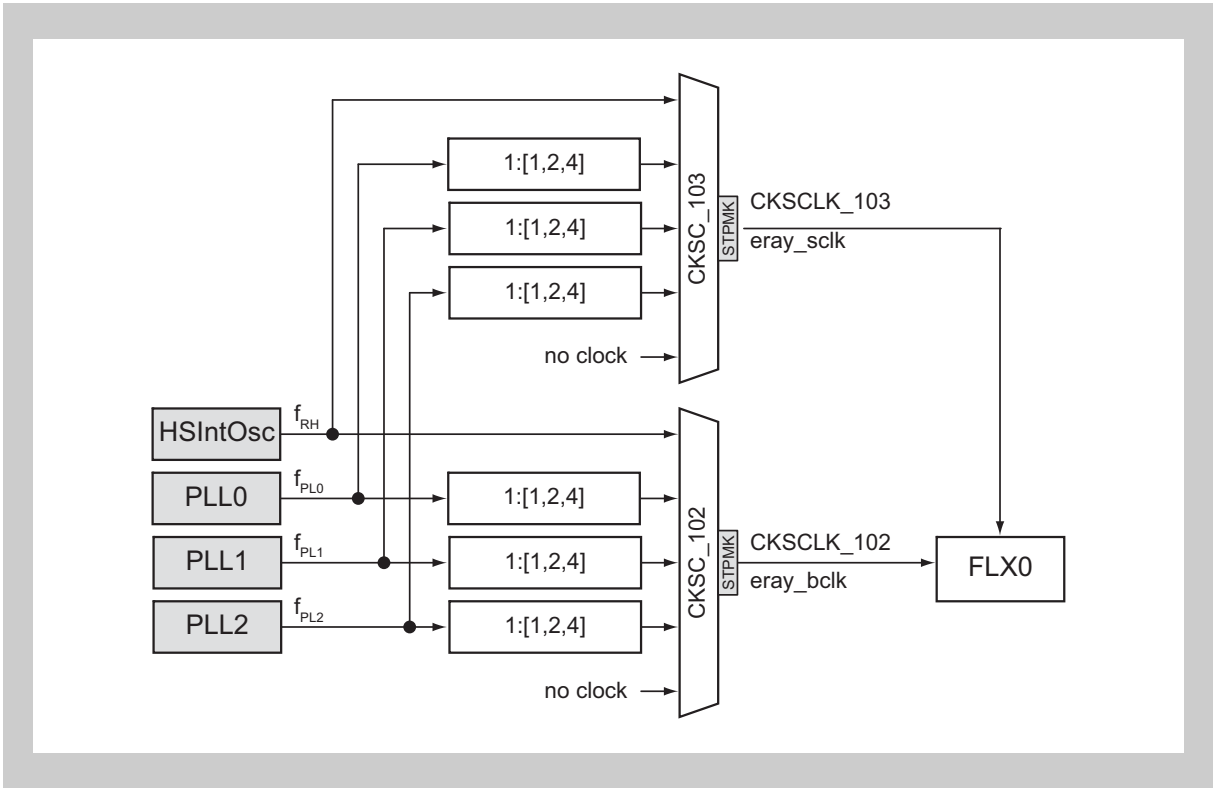


Figure 30-1 FLX clock supply

In the description of the E-Ray module special signal names are used, which differ from the names, used in other chapters of this document. In the following tables are provided to assign the E-Ray signal names to those used in the other chapters.

Interrupts and DMA/DTS

The FlexRay module can generate following interrupt and DMA/DTS requests:

Table 30-6 FLXn interrupt and DMA/DTS requests

FLXn signals	Function	Connected to
eray_int0	Interrupt 0	Interrupt Controller INTFLXA0I0 DMA Controller trigger 85 DTS Controller trigger 16
eray_int1	Interrupt 1	Interrupt Controller INTFLXA0I1 DMA Controller trigger 86 DTS Controller trigger 17
eray_tint0	Timer interrupt 0	Interrupt Controller INTFLXA0I2
eray_tint1	Timer interrupt 1	Interrupt Controller INTFLXA0I3
eray_ibusy	Transfer IBF to MBF busy	not connected
eray_obusy	Transfer MBF to OBF busy	not connected

FLX H/W reset

The FlexRay modules and their registers are initialized by the following reset signal:

Table 30-7 FLXn reset signal

ADCAn	Reset signal
FLX0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals

The I/O signals of the FlexRay module are listed in the table below.
All signals are connected to ports.

Table 30-8 FLXn I/O signals

FLXn signals	Function	Connected to
eray_txd1	Data transmitter output 1	Port FTXDA
eray_rxd1	Data receiver input 1	Port FRXDA
eray_txen1	Transmit enable signal 1	Port FTXENA
eray_txd2	Data transmitter output 2	Port FTXDB
eray_rxd2	Data receiver input 2	Port FRXDB
eray_txen2	Transmit enable signal 2	Port FTXENB
eray_stpwt	Stop watch trigger	Port FSTPWT

30.2 E-Ray Overview

30.2.1 Conventions

The following conventions are used within this document:

- CAPITALS POC states and CHI commands

30.2.2 Definition

- FlexRay Frame: Header Segment + Payload Segment
- Message Buffer: Header Section + Data Section
- Message RAM: Header Partition + Data Partition
- Data Frame: FlexRay frame that is not a null frame

30.2.3 References

This chapter refers to the following documents:

Table 30-9 References

Ref	Author(s)	Title
1	FlexRay Group	FlexRay Communication System Protocol Specification v2.1 Revision A (05/12/22)
2	BOSCH AE/ EIP	Addendum to E-Ray FlexRay IP-Module Specification Revision 1.2.6
3	BOSCH AE/ EIP	E-Ray FlexRay IP-Module Specification Revision 1.2.6

30.2.4 Terms and abbreviations

This chapter uses the following terms and abbreviations:

Table 30-10 Terms and abbreviations (1/2)

Term	Meaning
AP	Action Point
BD	Bus Drivert
BSS	Byte Start Sequence
CAS	Collision Avoidance Symbol
CC	Communication Controller
CHI	Controller Host Interface
CIF	Customer Interface
CRC	Cyclic Redundancy Check
FES	Frame End Sequence
FIFO	First In First Out (message buffer structure)
FSM	Finite State Machine

Table 30-10 Terms and abbreviations (2/2)

Term	Meaning
FSP	Frame and Symbol Processing Block
FSS	Frame Start Sequence
FTM	Fault Tolerant Midpoint
GTU	Global Time Unit Block
HIF	Host CPU Interface Block
IBF	Input Buffer
INT	Interrupt Control Block
MHD	Message Handler
MT	Macrotick
MTS	Media Access Test Symbol
NCT	Network Communication Time
NEM	Network Management Block
NIT	Network Idle Time
NM	Network Management
OBF	Output Buffer
POC	Protocol Operation Control
PRT	Protocol Controller Block
SDL	Specification and Description Language
SUC	System Universal Control Block
TBF	Transient Buffer
TDMA	Time Division Multiple Access (media access method)
TSS	Transmission Start Sequence
TT-D	Time Triggered Distributed Synchronization
µT	Microtick
WUP	Wakeup Pattern
WUS	Wakeup Symbol

30.2.5 Functional overview

The V850E2/Fx4-H includes a FlexRay-IP called ‘E-Ray’ version R1.0.2, which is designed by BOSCH. The E-Ray IP-module performs communication according to the FlexRay protocol specification v2.1. With maximum specified sample clock the bitrate can be programmed to values up to 10 MBit/s. Additional bus driver (BD) hardware is required for connection to the physical layer.

For communication on a FlexRay network, individual message buffers with up to 254 data bytes are configurable. The message storage consists of a single-ported Message RAM that holds up to 128 message buffers. All functions concerning the handling of messages are implemented in the Message Handler. Those functions are the acceptance filtering, the transfer of messages between the two FlexRay Channel Protocol Controllers and the Message RAM, maintaining the transmission schedule as well as providing message status information.

The register set of the E-Ray IP-module can be accessed directly by the V850E2/Fx4-H. These registers are used to control/configure/monitor the FlexRay Channel Protocol Controllers, Message Handler, Global Time Unit, System Universal Control, Frame and Symbol Processing, Network Management, Interrupt Control, and to access the Message RAM via Input / Output Buffer.

The E-Ray IP-module R1.2 supports the following features:

- Conform to FlexRay protocol specification v2.1
- Conform to E-Ray specification v1.2.6
- Data rates of up to 10 Mbit/s on each channel
- Up to 128 message buffers configurable
- 8 Kbyte of Message RAM for storage of e.g.
  - 128 messages buffers with max. 48 byte data section or up to
  - 30 messages buffers with max. 254 byte data section
- Configuration of message buffers with different payload lengths possible
- One configurable receive FIFO
- Each message buffer can be configured as receive buffer, as transmit buffer or as part of the receive FIFO
- Host access to message buffers via Input and Output Buffer
  Input Buffer: holds message to be transferred to the Message RAM
  Output Buffer: holds message read from the Message RAM
- Filtering for slot counter, cycle counter, and channel
- Maskable module interrupts
- Network Management supported

30.2.6 Block diagram

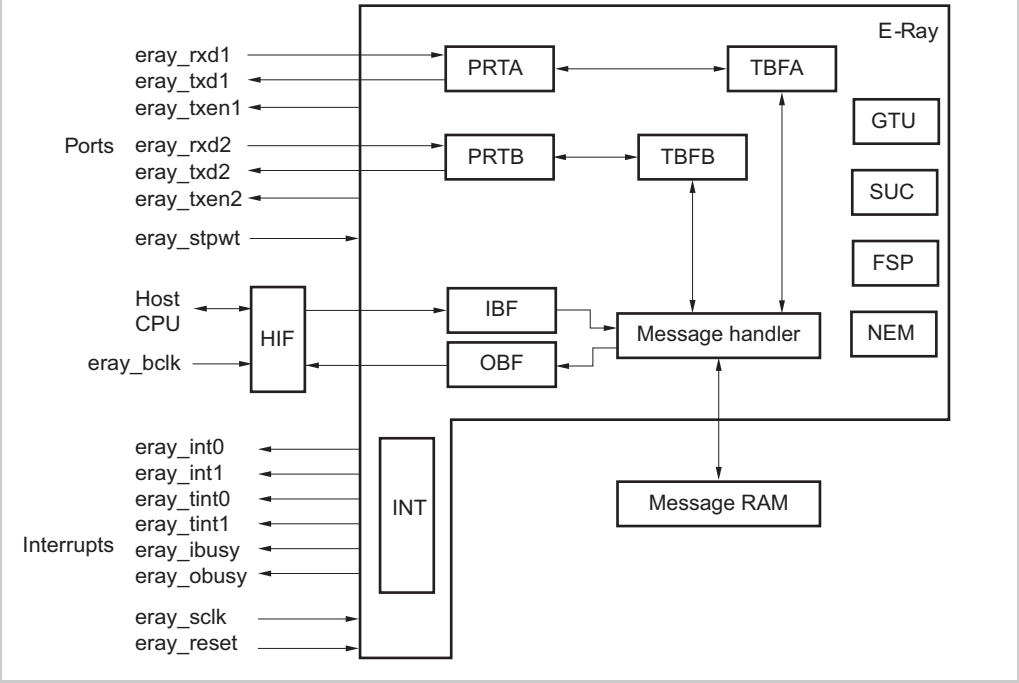


Figure 30-2 E-Ray block diagram

(1) Host Interface (HIF)

The Host Interface connects the CPU to the E-Ray IP-module. Configuration registers, status registers, and interrupt registers are attached to the respective blocks and can be accessed via the Host Interface.

(2) Input Buffer (IBF)

For write access to the message buffers configured in the Message RAM, the Host CPU can write the header and data section for a specific message buffer to the Input Buffer. The Message Handler then transfers the data from the Input Buffer to the selected message buffer in the Message RAM.

(3) Output Buffer (OBF)

For read access to a message buffer configured in the Message RAM the Message Handler transfers the selected message buffer to the Output Buffer. After the transfer has completed, the Host CPU can read the header and data section of the transferred message buffer from the Output Buffer.

(4) Message Handler (MHD)

The E-Ray Message Handler controls data transfers between the following components:

- Input / Output Buffer and Message RAM
- Transient Buffer RAMs of the two FlexRay Protocol Controllers and Message RAM

(5) TBF A/B (Transient Buffer RAM A/B)

The Transient Buffer stores the data section of two complete messages.

(6) PRT A/B (FlexRay Channel Protocol Controllers)

The FlexRay Channel Protocol Controllers consist of shift register and FlexRay protocol FSM. They are connected to the Transient Buffer RAMs for intermediate message storage and to the physical layer via bus driver BD.

They perform the following functionality:

- Control and check of bit timing
- Reception / transmission of FlexRay frames and symbols
- Check of header CRC
- Generation / check of frame CRC
- Interfacing to bus driver

The FlexRay Channel Protocol Controllers have interfaces to:

- Physical Layer (bus driver)
- Transient Buffer RAM
- Message Handler
- Global Time Unit
- System Universal Control
- Frame and Symbol Processing
- Network Management
- Interrupt Control

(7) Global Time Unit (GTU)

The Global Time Unit performs the following functions:

- Generation of microtick
- Generation of macrotick
- Fault tolerant clock synchronization by FTM algorithm
  - rate correction
  - offset correction
- Cycle counter
- Timing control of static segment
- Timing control of dynamic segment (minislotting)
- Support of external clock correction

(8) System Universal Control (SUC)

The System Universal Control controls the following functions:

- Configuration
- Wakeup
- Startup
- Normal Operation
- Passive Operation
- Monitor Mode

(9) Frame and Symbol Processing (FSP)

The Frame and Symbol Processing controls the following functions:

- Checks the correct timing of frames and symbols
- Tests the syntactical and semantical correctness of received frames
- Sets the slot status flags

(10) Network Management (NEM)

The Network Management performs the following functions:

- Handling of the network management vector

(11) Interrupt Control (INT)

The Interrupt Controller performs the following functions:

- Provides error and status interrupt flags
- Controls enable / disable of interrupt sources
- Controls assignment of interrupt sources to the two module interrupt lines
- Enable / disable of the two module interrupt lines
- Manages the two interrupt timers
- Stop watch time capturing

30.2.7 Host CPU interface timing

The numbers of eray_bclk clock cycles for single CPU read or write accesses are given in the table below:

Table 30-11 Host CPU access cycles

Access	Access cycle eray_bclk
Register/RAM write	1
Register read	1
RAM read	2

30.2.8 Reset timing

When leaving hardware reset, an internal procedure is started that initializes the module-internal RAM blocks to zero. The module-internal RAMs can also be cleared by CHI command CLEAR_RAMS (FLXnSUCC1.FLXnCMD[3:0] = "1100") when the CC is in DEFAULT_CONFIG or CONFIG state. The initialization of the E-Ray internal RAM blocks requires 2048 eray_bclk cycles.

No Host CPU access to IBF or OBF is possible during initialization of the internal RAM blocks after hardware reset or after assertion of CHI command CLEAR_RAMS.

After hardware reset all registers hold the reset values as summarized in *Table 30-13 “E-Ray register map” on page 2373* . Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS.

During the initialization of the RAMs, FLXnSUCC1.FLXnPBSY will show POC busy.

30.3 Programmer's Model

30.3.1 Register map

The E-Ray module allocates an address space of 2 Kbytes (0000_H to $07FF_H$). The registers are organized as 32-bit registers. Host access to the Message RAM is done via the Input and Output Buffers. They buffer data to be transferred to and from the Message RAM under control of the Message Handler, avoiding conflicts between Host accesses and message reception / transmission.

The assignment of the message buffers is done according to the scheme shown in *Table 30-12 "Assignment of message buffers"* below. The number N of available message buffers depends on the payload length of the configured message buffers. The maximum number of message buffers is 128. The maximum payload length supported is 254 bytes.

Message buffer size

The size of the implemented message buffer is defined in the first section of this chapter under the key word "Message buffer size".

The message buffers are separated into three consecutive groups:

Static Buffers	Transmit / receive buffers static segment
Static + Dynamic Buffers	Transmit / receive buffers static + dynamic segment
FIFO	Receive FIFO

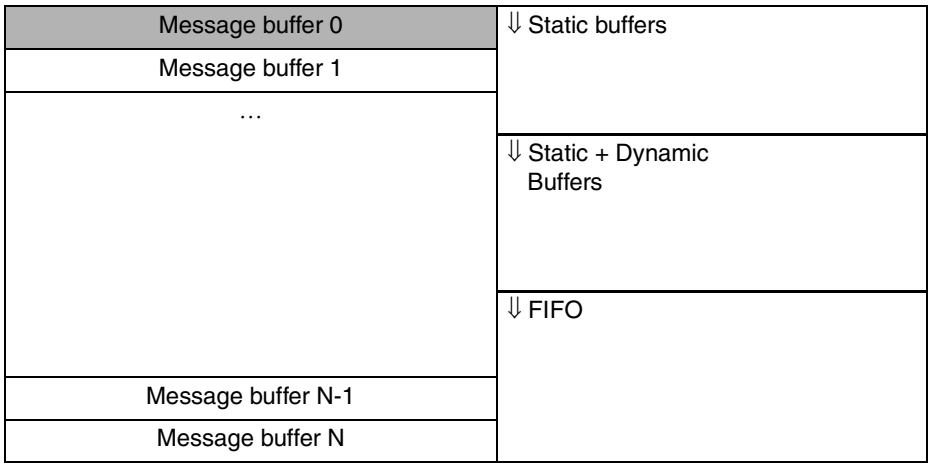
The message buffer separation configuration can be changed in DEFAULT_CONFIG or CONFIG state only by programming the Message RAM Configuration register.

The first group starts with message buffer 0 and consists of static message buffers only. Message buffer 0 is dedicated to hold the startup / sync frame or the single slot frame, if the node transmits one, as configured by FLXnTXST, FLXnTXSY, and FLXnTSM in the SUC Configuration Register 1. In this case message buffer 0 has to be configured with the key slot ID and can only be (re)configured in DEFAULT_CONFIG or CONFIG state.

The second group consists of message buffers assigned to the static or dynamic segment.

The message buffers belonging to the third group are concatenated to a single receive FIFO.

Table 30-12 Assignment of message buffers



Offset address The addresses in the table below are given as offsets to the base address <FLXn_base>. <FLXn_base> of the FLXn is defined in the first section of this chapter under the key word “Register addresses”.

Table 30-13 E-Ray register map (1/4)

Offset address	Symbol	Name	Reset	Acc	Block
E-Ray registers:					
0000_H	FLXnCI	Controller Information	$6572\ 6179_H$	R	VIF
0004_H	FLXnVI	Vendor Information	[a]	R	
0008_H	FLXnCS	Control Setting	$0000\ 0002_H$	R/W	
$000C_H$	–	reserved (1)	$0000\ 0000_H$	R	
Special registers:					
0010_H	FLXnTEST1	Test Register 1	$0000\ 0300_H$	R/W	HIF
0014_H	FLXnTEST2	Test Register 2	$0000\ 0000_H$	R/W	
0018_H	–	reserved (1)	$0000\ 0000_H$	R	
$001C_H$	FLXnLCK	Lock Register	$0000\ 0000_H$	R/W	
Interrupt registers:					
0020_H	FLXnEIR	Error Interrupt Register	$0000\ 0000_H$	R/W	INT
0024_H	FLXnSIR	Status Interrupt Register	$0000\ 0000_H$	R/W	
0028_H	FLXnEILS	Error Interrupt Line Select	$0000\ 0000_H$	R/W	
$002C_H$	FLXnSILS	Status Interrupt Line Select	$0303\ FFFF_H$	R/W	
0030_H	FLXnEIES	Error Interrupt Enable Set	$0000\ 0000_H$	R/W	
0034_H	FLXnEIER	Error Interrupt Enable Reset	$0000\ 0000_H$	R/W	
0038_H	FLXnSIES	Status Interrupt Enable Set	$0000\ 0000_H$	R/W	
$003C_H$	FLXnSIER	Status Interrupt Enable Reset	$0000\ 0000_H$	R/W	
0040_H	FLXnILE	Interrupt Line Enable	$0000\ 0000_H$	R/W	
0044_H	FLXnT0C	Timer 0 Configuration	$0000\ 0000_H$	R/W	
0048_H	FLXnT1C	Timer 1 Configuration	$0002\ 0000_H$	R/W	
$004C_H$	FLXnSTPW1	Stop Watch Register 1	$0000\ 0000_H$	R/W	
0050_H	FLXnSTPW2	Stop Watch Register 2	$0000\ 0000_H$	R/W	

Table 30-13 E-Ray register map (2/4)

Offset address	Symbol	Name	Reset	Acc	Block
0054_H - $007C_H$	–	reserved (11)	$0000\ 0000_H$	R	
CC control registers:					
0080_H	FLXnSUCC1	SUC Configuration Register 1	$0C40\ 1080_H$	R/W	SUC
0084_H	FLXnSUCC2	SUC Configuration Register 2	$0100\ 0504_H$	R/W	
0088_H	FLXnSUCC3	SUC Configuration Register 3	$0000\ 0011_H$	R/W	
$008C_H$	FLXnNEMC	NEM Configuration Register	$0000\ 0000_H$	R/W	NEM
0090_H	FLXnPRTC1	PRT Configuration Register 1	$084C\ 0633_H$	R/W	PRT
0094_H	FLXnPRTC2	PRT Configuration Register 2	$0F2D\ 0A0E_H$	R/W	
0098_H	FLXnMHDC	MHD Configuration Register	$0000\ 0000_H$	R/W	MHD
$009C_H$		reserved (1)	$0000\ 0000_H$	R	
$00A0_H$	FLXnGTUC01	GTU Configuration Register 1	$0000\ 0280_H$	R/W	GTU
$00A4_H$	FLXnGTUC02	GTU Configuration Register 2	$0002\ 000A_H$	R/W	
$00A8_H$	FLXnGTUC03	GTU Configuration Register 3	$0202\ 0000_H$	R/W	
$00AC_H$	FLXnGTUC04	GTU Configuration Register 4	$0008\ 0007_H$	R/W	
$00B0_H$	FLXnGTUC05	GTU Configuration Register 5	$0E00\ 0000_H$	R/W	
$00B4_H$	FLXnGTUC06	GTU Configuration Register 6	$0002\ 0000_H$	R/W	
$00B8_H$	FLXnGTUC07	GTU Configuration Register 7	$0002\ 0004_H$	R/W	GTU
$00BC_H$	FLXnGTUC08	GTU Configuration Register 8	$0000\ 0002_H$	R/W	
$00C0_H$	FLXnGTUC09	GTU Configuration Register 9	$0000\ 0101_H$	R/W	
$00C4_H$	FLXnGTUC10	GTU Configuration Register 10	$0002\ 0005_H$	R/W	
$00C8_H$	FLXnGTUC11	GTU Configuration Register 11	$0000\ 0000_H$	R/W	
$00CC_H$ - $00FC_H$	–	reserved (13)	$0000\ 0000_H$	R	
CC status registers:					
0100_H	FLXnCCSV	CC Status Vector	$0010\ 4000_H$	R	SUC
0104_H	FLXnCCEV	CC Error Vector	$0000\ 0000_H$	R	
0108_H - $010C_H$	–	reserved (2)	$0000\ 0000_H$	R	
0110_H	FLXnSCV	Slot Counter Value	$0000\ 0000_H$	R	GTU
0114_H	FLXnMTCCV	Macrotick and Cycle Counter Value	$0000\ 0000_H$	R	
0118_H	FLXnRCV	Rate Correction Value	$0000\ 0000_H$	R	
$011C_H$	FLXnOCV	Offset Correction Value	$0000\ 0000_H$	R	
0120_H	FLXnSFS	Sync Frame Status	$0000\ 0000_H$	R	
0124_H	FLXnSWNIT	Symbol Window and NIT Status	$0000\ 0000_H$	R	
0128_H	FLXnACS	Aggregated Channel Status	$0000\ 0000_H$	R/W	
$012C_H$	–	reserved (1)	$0000\ 0000_H$	R	
0130_H - 0168_H	FLXnESIDm	Even Sync ID [01...15]	$0000\ 0000_H$	R	GTU
$016C_H$	–	reserved (1)	$0000\ 0000_H$	R	
0170_H - $01A8_H$	FLXnOSIDm	Odd Sync ID [01...15]	$0000\ 0000_H$	R	GTU

Table 30-13 E-Ray register map (3/4)

Offset address	Symbol	Name	Reset	Acc	Block
$01AC_H$	–	reserved (1)	$0000\ 0000_H$	R	
$01B0_H$ - $01B8_H$	FLXnNMVm	Network Management Vector [1...3]	$0000\ 0000_H$	R	NEM
$01BC_H$ - $02FC_H$	–	reserved (81)	$0000\ 0000_H$	R	
Message buffer control registers:					
0300_H	FLXnMRC	Message RAM Configuration	$0180\ 0000_H$	R/W	MHD
0304_H	FLXnFRF	FIFO Rejection Filter	$0180\ 0000_H$	R/W	
0308_H	FLXnFRFM	FIFO Rejection Filter Mask	$0000\ 0000_H$	R/W	
$030C_H$	FLXnFCL	FIFO Critical Level	$0000\ 0080_H$	R/W	
Message buffer status registers:					
0310_H	FLXnMHDS	Message Handler Status	$0000\ 0080_H$	R/W	MHD
0314_H	FLXnLDTS	Last Dynamic Transmit Slot	$0000\ 0000_H$	R	
0318_H	FLXnFSR	FIFO Status Register	$0000\ 0000_H$	R	
$031C_H$	FLXnMHDF	Message Handler Constraints Flags	$0000\ 0000_H$	R/W	
0320_H	FLXnTXRQ1	Transmission Request 1	$0000\ 0000_H$	R	
0324_H	FLXnTXRQ2	Transmission Request 2	$0000\ 0000_H$	R	
0328_H	FLXnTXRQ3	Transmission Request 3	$0000\ 0000_H$	R	
$032C_H$	FLXnTXRQ4	Transmission Request 4	$0000\ 0000_H$	R	MHD
0330_H	FLXnNDAT1	New Data 1	$0000\ 0000_H$	R	
0334_H	FLXnNDAT2	New Data 2	$0000\ 0000_H$	R	
0338_H	FLXnNDAT3	New Data 3	$0000\ 0000_H$	R	
$033C_H$	FLXnNDAT4	New Data 4	$0000\ 0000_H$	R	
0340_H	FLXnMBSC1	Message Buffer Status Changed 1	$0000\ 0000_H$	R	
0344_H	FLXnMBSC2	Message Buffer Status Changed 2	$0000\ 0000_H$	R	
0348_H	FLXnMBSC3	Message Buffer Status Changed 3	$0000\ 0000_H$	R	
$034C_H$	FLXnMBSC4	Message Buffer Status Changed 4	$0000\ 0000_H$	R	
0350_H - $03EC_H$	–	reserved (40)	$0000\ 0000_H$	R	
Identification Registers:					
$03F0_H$	FLXnCREL	Core Release Register	$1027\ 1031_H$	R	GIF
$03F4_H$	FLXnENDN	Endian Register	$8765\ 4321_H$	R	
$03F8_H$ - $03FC_H$	–	reserved (2)	$0000\ 0000_H$	R	

Table 30-13 E-Ray register map (4/4)

Offset address	Symbol	Name	Reset	Acc	Block
Input buffer:					
0400_H - $04FC_H$	FLXnWRDSm	Write Data Section [01...64]	$0000\ 0000_H$	R/W	IBF
0500_H	FLXnWRHS1	Write Header Section 1	$0000\ 0000_H$	R/W	
0504_H	FLXnWRHS2	Write Header Section 2	$0000\ 0000_H$	R/W	
0508_H	FLXnWRHS3	Write Header Section 3	$0000\ 0000_H$	R/W	
$050C_H$	–	reserved (1)	$0000\ 0000_H$	R/W	
0510_H	FLXnIBCM	Input Buffer Command Mask	$0000\ 0000_H$	R/W	
0514_H	FLXnIBCR	Input Buffer Command Request	$0000\ 0000_H$	R/W	
0518_H - $05FC_H$	–	reserved (58)	$0000\ 0000_H$	R	
Output buffer:					
0600_H - $06FC_H$	FLXnRDDSm	Read Data Section [01...64]	$0000\ 0000_H$	R	OBF
0700_H	FLXnRDHS1	Read Header Section 1	$0000\ 0000_H$	R	
0704_H	FLXnRDHS2	Read Header Section 2	$0000\ 0000_H$	R	
0708_H	FLXnRDHS3	Read Header Section 3	$0000\ 0000_H$	R	
$070C_H$	FLXnMBS	Message Buffer Status	$0000\ 0000_H$	R	
0710_H	FLXnOBCM	Output Buffer Command Mask	$0000\ 0000_H$	R/W	
0714_H	FLXnOBCR	Output Buffer Command Request	$0000\ 0000_H$	R/W	
0718_H - $07FC_H$	–	reserved (58)	$0000\ 0000_H$	R	

a) The vendor information of register FLXnVI is defined in the first section of this chapter under the key word "Vendor information"

30.3.2 E-Ray registers

(1) FLXnCI - Controller information

This register contains FlexRay controller information.

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0000_H

Initial Value $6572\ 6179_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnMN[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMN[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-14 FLXnCI register contents

Bit position	Bit name	Function
31 to 0	FLXn MN[31:00]	Magic Number This field contains a code which shows the name of the Bosch IP. The value is "6572 6179" which shows "eray" as ASCII-codes.

(2) FLXnVI - Vendor information

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0004_H

Initial Value Refer to the keyword “Vendor information” in the first section of this chapter.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnVI[7:0]								FLXnFMR[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	FLXnPCN[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-15 FLXnVI register contents

Bit position	Bit name	Function
31 to 24	FLXn VI[7:0]	Vendor ID This field contains a code which shows the producer of the macro.
23 to 16	FLXn FMR[7:0]	FlexRay Macro Release Number This field contains a code which shows the current release number of the IP.
7 to 0	FLXn PCN[7:0]	Product Code Number This field contains a code which shows the chip-product.

Vendor Information The vendor information of register FLXnVI is defined in the first section of this chapter under the key word "Vendor information".

(3) FLXnCS - Control settings

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0008_H

Initial Value $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnCSLK[7:0]								0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn MD	FLXn SR
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 30-16 FLXnCS register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>31 to 24</td><td>FLXn
CSLK[7:0]</td><td>Control Setting Lock Key
To set and clear FLXnSR bit and FLXnMD bit, the write operation has to be directly preceded by two consecutive write accesses to the Control Setting Lock Key. If the host access to other E-Ray's register in between this consecutive sequence, writing FLXnCS register is simply ignored.
<table>
<tr><th></th><th>set FLXnSR bit</th><th>clear FLXnSR bit</th><th>set FLXnMD bit</th><th>clear FLXnMD bit</th></tr>
<tr><td>First write to FLXnCS Register</td><td>$67xx\_xxxx_H$</td><td>$67xx\_xxxx_H$</td><td>$67xx\_xxxx_H$</td><td>$67xx\_xxxx_H$</td></tr>
<tr><td>Second write to FLXnCS Register</td><td>$98xx\_xxx1_H$</td><td>$98xx\_xxx0_H$</td><td>$98xx\_xxx2_H$</td><td>$98xx\_xxx0_H$</td></tr>
</table>
Reading these bits returns 0.</td></tr>
<tr><td>1</td><td>FLXn
MD</td><td>Macro Disable
Macro disable bit. This bit is connected to the clock module. If this bit is set, internal clock distribution (eray_sclk, eray_bclk) is halted. Therefore software can NOT access registers/RAM of FlexRay Macro when FLXnMD=1. In case of that, write access will be simply ignored and read access will get incorrect value. Exceptionally microcontroller specifc E-Ray registers (i.e. FLXnCI / FLXnVI / FLXnCS) can be accessed even when FLXnMD=1.Unlock sequence is required for setting and clearing FLXnMD bit. Macro disable bit does not affect to host IF logic itself.
The user must set FLXnMD only in CONFIG state and must wait a transfer between RAMs to complete. Otherwise the last received message can be destroy due to sudden clock stopping.
Note that the read access to FLXnTEST1 during FLXnMD=1 doesn't cause clearing FLXnTEST1.FLXnCERA and FLXnCERB.
A module reset will be performed after setting FLXnMD and the RAMs can't be accessed until the FLXnSUCC1.FLXnPBSY is cleared.
Note: After power up of the microcontroller this bit is set and the FlexRay macro can't be access until this bit is set to 0.</td></tr>
<tr><td>0</td><td>FLXn
SR</td><td>Software Reset
Software Reset bit. This bit is connected to the E-Ray reset line. If this bit is set, a reset will be executed immediately. The user must clear FLXnSR bit in order to release a reset. Unlock sequence is required for setting and clearing FLXnSR bit. Software reset does not affect to host IF logic itself.
Note: After a software reset the RAMs can't be accessed until the FLXnSUCC1.FLXnPBSY is cleared.</td></tr>
</table>

Accessing to E-RAY registers and RAM is restricted by the combination of FLXnSR and FLXnMD bits.

FLXn SR	FLXn MD	CIF registers (FLXnCI, FLXnVI, FLXnCS)	E-Ray registers	E-Ray RAM
0	0	R/W possible	R/W possible	R/W possible
0	1	R/W possible	Read only[a]	NOT possible[b]
1	0	R/W possible	NOT possible[b]	NOT possible[b]
1	1	R/W possible	NOT possible[b]	NOT possible[b]

a) Write access: simply ignored
b) Read access: capture invalid data, write access: simply ignored

The table below explains the possible bit transition of FLXnSR and FLXnMD bits.

FLXnSR	FLXnMD	Bit transition
0 → 1	0	allowed
0	0 → 1	allowed
0 → 1	0 → 1	ignored
1	0 → 1	allowed
0 → 1	1	ignored
0 → 1	1 → 0	allowed

There is no restriction to clear SR and FLXnMD bits.

30.3.3 Special registers

(1) FLXnTEST1 - Test register 1

The Test Register 1 holds the control bits to configure the test modes of the E-Ray module. Write access to these bits is only possible if bit FLXnWRTEN is set to '1'.

When the E-Ray IP is operated in one of its test modes that requires FLXnWRTEN to be set (RAM Test Mode, I/O Test Mode, Asynchronous Transmit Mode, and Loop Back Mode) only the selected test mode functionality is available.

The test functions are not available in addition to the normal operational mode functions, they change the functions of parts of the E-Ray module. Therefore normal operation as specified outside this chapter and as required by the FlexRay protocol specification and the FlexRay conformance test is not possible. Test mode functions may not be combined with each other or with FlexRay protocol functions.

The test mode features are intended for hardware testing or for FlexRay bus analyzer tools. They are not intended to be used in FlexRay applications.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0010_H

Initial Value $0000\ 0300_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnCERB[3:0]				FLXnCERA[3:0]				0	0	FLXn TXENB	FLXn TXENA	FLXn TXB	FLXn TXA	FLXn RXB	FLXn RXA
R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	FLXn AOB	FLXn AOA	0	0	FLXn TMC1	FLXn TMC0	0	0	FLXn ELBE	FLXn WRTEN
R	R	R	R	R	R	R	R	R	R	R/W	R/W	R	R	R/W	R/W

Table 30-17 FLXnTEST1 register contents (1/2)

Bit position	Bit name	Function
31 to 28	FLXn CERB[3:0]	Coding Error Report Channel B Set when a coding error is detected on channel B. Reset to zero when register FLXnTEST1 is read or written. Once the FLXnCERB[3:0] is set it will remain unchanged until FLXnTEST1 register is accessed. 0000 = No coding error detected 0001 = Header CRC error detected 0010 = Frame CRC error detected 0011 = Frame Start Sequence FSS too long 0100 = First bit of Byte Start Sequence BSS seen LOW 0101 = Second bit of Byte Start Sequence BSS seen HIGH 0110 = First bit of Frame End Sequence FES seen HIGH 0111 = Second bit of Frame End Sequence FES seen LOW 1000 = CAS / MTS symbol seen too short 1001 = CAS / MTS symbol seen too long 1010...1111 = reserved
27 to 24	FLXn CERA[3:0]	Coding Error Report Channel A Set when a coding error is detected on channel A. Reset to zero when register FLXnTEST1 is read or written. Once the FLXnCERA[3:0] is set it will remain unchanged until FLXnTEST1 register is accessed. 0000 = No coding error detected 0001 = Header CRC error detected 0010 = Frame CRC error detected 0011 = Frame Start Sequence FSS too long 0100 = First bit of Byte Start Sequence BSS seen LOW 0101 = Second bit of Byte Start Sequence BSS seen HIGH 0110 = First bit of Frame End Sequence FES seen HIGH 0111 = Second bit of Frame End Sequence FES seen LOW 1000 = CAS / MTS symbol seen too short 1001 = CAS / MTS symbol seen too long 1010...1111 = reserved
21	FLXn TXENB	Control of Channel B Transmit Enable Pin 0 = eray_txen2_n pin drives a ’0’ 1 = eray_txen2_n pin drives a ’1
20	FLXn TXENA	Control of Channel A Transmit Enable Pin 0 = ray_txen1_n pin drives a ’0’ 1 = eray_txen1_n pin drives a ’1’
19	FLXn TXB	Control of Channel B Transmit Pin 0 = eray_txd2 pin drives a ’0’ 1 = eray_txd2 pin drives a ’1’
18	FLXn TXA	Control of Channel A Transmit Pin 0 = eray_txd1 pin drives a ’0’ 1 = eray_txd1 pin drives a ’1’
17	FLXn RXB	Mon itor Channel B Receive Pin 0 = eray_rxd2 = ’0’ 1 = eray_rxd2 = ’1’ The FLXnRXB pin has to be fixed to high or low, otherwise the value is undefined.
16	FLXn RXA	Monitor Channel A Receive Pin 0 = eray_rxd1 = ’0’ 1 = eray_rxd1 = ’1’ The FLXnRXA pin has to be fixed to high or low, otherwise the value is undefined.

Table 30-17 FLXnTEST1 register contents (2/2)

Bit position	Bit name	Function
9	FLXn AOB	Activity on B FLXnAOB is set when there is activity on channel B. It is reset when 11 consecutive '1' bits are detected or if the POC state is DEFAULT_CONFIG or CONFIG. During STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE the function of FLXnAOB is the inverse of zChannel-Idle as specified in the FlexRay protocol spec v2.1, chapter 3, BITSTRB process. FLXnAOB shouldbe ignored in all other POC states. 1 = Activity detected, channel B not idle 0 = No activity detected, channel B idle
8	FLXn AOA	Activity on A FLXnAOA is set when there is activity on channel A. It is reset when 11 consecutive '1' bits are detected or if the POC state is DEFAULT_CONFIG or CONFIG. During STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE the function of FLXnAOA is the inverse of zChannelIdle as specified in the FlexRay protocol spec v2.1, chapter 3, BITSTRB process. FLXnAOA should be ignored in all other POC states. 1 = Activity detected, channel A not idle 0 = No activity detected, channel A idle
5 to 4	FLXn TMC[1:0]	Test Multiplexer Control 00= Normal signal path (default) 01 = RAM Test Mode - Internal busses are multiplexed to make all RAM blocks of the E-Ray module directly accessible by the Host. This mode is intended to enable testing of the embedded RAM blocks during production testing. 10 = I/O Test Mode - Output pins eray_txd1, eray_txd2, eray_txen1_n, eray_txen2_n, are driven to the values defined by bits FLXnTXA, FLXnTXB, FLXnTXENA, FLXnTXENB. The values applied to the input pins eray_rxd1, eray_rxd2 can be read from register bits FLXnRXA, FLXnRXB. 11= Normal signal path (default)
1	FLXn ELBE	External Loop Back Enable There are two possibilities to perform a loop back test. External loop back via physical layer or internal loop back for in-system self-test (default). In case of an internal loop back pins eray_txen1,2_n are in their inactive state, pins eray_txd1,2 are set to HIGH, pins eray_rxd1,2 are not evaluated. Bit FLXnELBE is evaluated only when POC is in loop back mode and test multiplexer control is in non-multiplexing mode FLXnTMC[1:0] = "00". 1 = External loop back 0 = Internal loop back (default)
0	FLXn WRTEN	Write Test Register Enable Enables write access to the test registers. To set the bit from '0' to '1' the test mode key has to be written as defined in section Lock Register (FLXnLCK). The unlock sequence is not required when FLXnWRTEN is kept at '1' while other bits of the register are changed. The bit can be reset to '0' at any time. 1 =Write access to the Test Register is enabled 0 =Write access to the Test Register is disabled

Notes 1. Coding errors are signalled in all states where frame decoding is possible. FLXnCERA[3:0] and FLXnCERB[3:0] should be ignored in all other states. The error codes regarding CAS / MTS symbols concern only the monitored bit pattern, irrelevant whether those bit patterns are seen in the symbol window or elsewhere.

2. The following Test Register 1 bits are used to test the interface to the physical layer (connectivity test) by driving / reading the respective pins: FLXnTXENB, FLXnTXENA, FLXn TXB, FLXn TXA, FLXn RXB, FLXn RXA

Asynchronous Transmit Mode (ATM)

The asynchronous transmit mode is entered by writing FLXnSUCC1.FLXnCMD[3:0] = "1110" while the CC is in CONFIG state and bit FLXnTEST1.FLXnWRTEN is set to '1'. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when bit FLXnTEST1.FLXnWRTEN is not set, FLXnSUCC1.FLXnCMD[3:0] will be reset to "0000" = command_not_accepted. Reading FLXnCCSV.FLXnPOCS[5:0] will return "00 1110" while the E-Ray module is in ATM mode. Asynchronous transmit mode can be left by writing FLXnSUCC1.FLXnCMD[3:0] = "0001" (FLXnCHI command: CONFIG).

In ATM mode transmission of a FlexRay frame is triggered by writing the number of the respective message buffer to FLXnIBCR.FLXnIBRH[6:0] while FLXnIBCM.STXR is set to '1'. In this mode wakeup, startup, and clock synchronization are bypassed. The CHI command SEND_MTS results in the immediate transmission of an MTS symbol.

The cycle counter value of frames send in ATM mode can be programmed via FLXnMTCCV.FLXnCCV[5.0] (writeable in ATM and loop back mode only).

Loop Back Mode

The loop back mode is entered by writing FLXnSUCC1.FLXnCMD[3:0] = "1111" when CONFIG state is entered after transition to DEFAULT_CONFIG state is made by resetting the FlexRay Controller (FLXnCS.FLXnSR) and bit FLXnTEST1.FLXnWRTEN is set to '1'. This write operation has to be directly preceded by two consecutive write accesses to the Configuration Lock Key (unlock sequence). When called in any other state or when FLXnTEST1.FLXnWRTEN is not set, FLXnSUCC1.FLXnCMD[3:0] will be reset to "0000" = command_not_accepted. Reading FLXnCCSV.FLXnPOCS[5:0] will return "00 1101" while the E-Ray module is in loop back mode.

Loop back mode can be left by writing FLXnSUCC1.FLXnCMD[3:0] = "0001" (CHI command: CONFIG).

The loop back mode is intended to check the module's internal data paths. Normal, time triggered operation is not possible in loop back mode.

There are two possibilities to perform a loop back test. External loop back via physical layer (FLXnTEST1.FLXnELBE = '1') or internal loop back for in-system self-test (FLXnTEST1.FLXnELBE = '0'). In case of an internal loop back pins eray_txen1,2_n are in their inactive state, pins eray_txd1,2 are set to HIGH, pins eray_rxd1,2 are not evaluated.

When the CC is in loop back mode, a loop back test is started by the Host writing a message to the Input Buffer and requesting the transmission by writing to register FLXnIBCR. The Message Handler will transfer the message into the Message RAM and then into the Transient Buffer of the selected channel. The Channel Protocol Controller (PRT) will read (in 32-bit words) the message from the transmit part of the Transient Buffer and load it into its Rx / Tx shift register. The serial transmission is looped back into the shift register; its content is written into the receive part of the channels's Transient Buffer before the next word is loaded.

The PRT and the Message Handler will then treat this transmitted message like a received message, perform an acceptance filtering on frame ID and receive channel, and store the message into the Message RAM if it passed acceptance filtering. The loop back test ends with the Host requesting this received message from the Message RAM and then checking the contents of the Output Buffer.

Each FlexRay channel is tested separately. The E-Ray cannot receive messages from the FlexRay bus while it is in the loop back mode.

The cycle counter value of frames used in loop back mode can be programmed via FLXnMTCCV.FLXnCCV[5.0] (writeable in ATM and loop back mode only).

Note that in case of an odd payload the last two bytes of the looped-back payload will be shifted by 16 bits to the right inside the last 32-bit data word.

(2) FLXnTEST2 - Test register 2

The Test Register 2 holds all bits required for the RAM test of the seven embedded RAM blocks of the E-Ray module. Write access to this register is only possible when FLXnTEST1.WRTEN is set to '1'.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0014_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	FLXn SSEL[2:0]			0	FLXn RS[2:0]		
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Table 30-18 FLXnTEST2 register contents

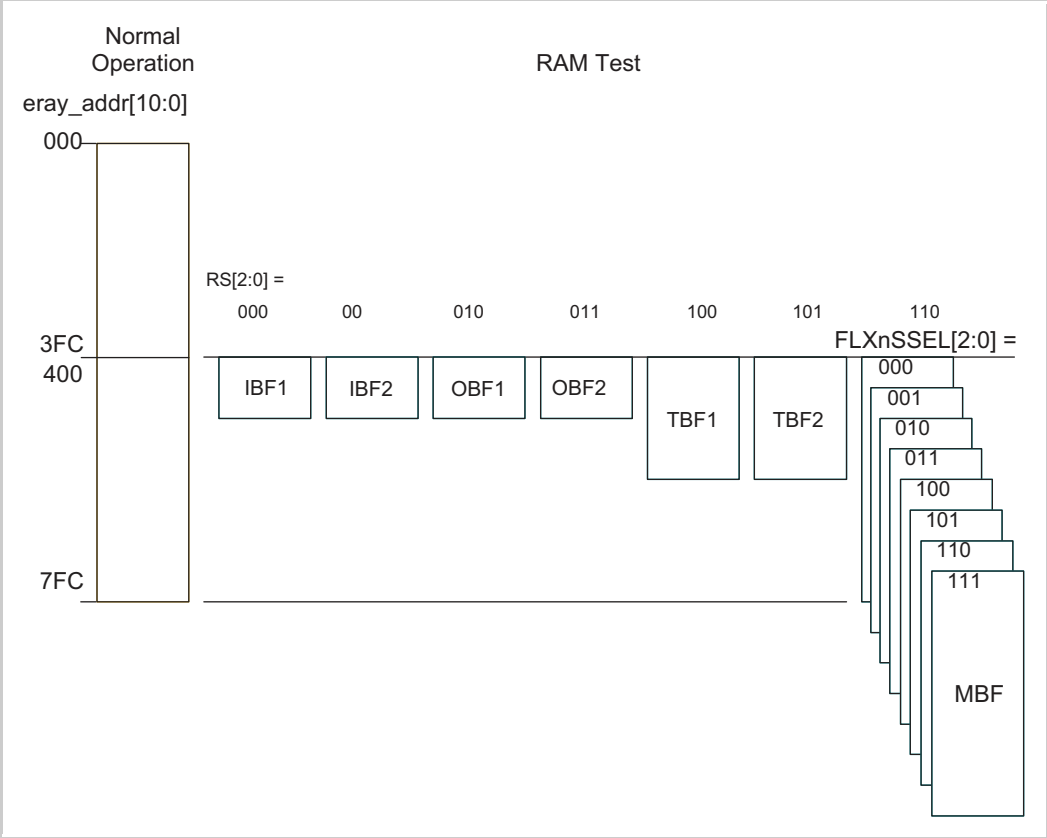
Bit position	Bit name	Function
6 to 4	FLXn SSEL[2:0]	RAM Select In RAM Test mode the RAM blocks selected by RS[2:0] are mapped to module address 400_H to $7FF_H$ (1024 byte addresses). 000 = Input Buffer RAM 1 (IBF1) 001 = Input Buffer RAM 2 (IBF2) 010 = Output Buffer RAM 1 (OBF1) 011 = Output Buffer RAM 2 (OBF2) 100 = Transient Buffer RAM A (TBF1) 101 = Transient Buffer RAM B (TBF2) 110 = Message RAM (MBF) 111 = unused
2 to 0	FLXn RS[2:0]	Segment Select To enable access to the complete Message RAM (8192 byte addresses) the Message RAM is segmented 000 = access to RAM bytes 0000h to 03FFh enabled 001 = access to RAM bytes 0400h to 07FFh enabled 010 = access to RAM bytes 0800h to 0BFFh enabled 011 = access to RAM bytes 0C00h to 0FFFh enabled 100 = access to RAM bytes 1000h to 13FFh enabled 101 = access to RAM bytes 1400h to 17FFh enabled 110 = access to RAM bytes 1800h to 1BFFh enabled 111 = access to RAM bytes 1C00h to 1FFFh enabled

RAM Test Mode In RAM test mode (FLXnTEST1.FLXnTMC[1:0] = "01"), one of the seven RAM blocks can be selected for direct RD/WR access by programming FLXnTEST2.FLXnRS[2:0].

For external access the selected RAM block is mapped to address space 400_H to $7FF_H$ (1024 byte addresses or 256 word addresses).

Because the length of the Message RAM exceeds the available address space, the Message RAM is segmented into segments of 1024 bytes. The segments can be selected by programming FLXnTEST2.FLXnSSEL[2:0].

Note In case of accessing MBF, do not access out of the connected MBF area.



(3) FLXnLCK - Lock register

The Lock Register is write-only. Reading the register will return 0000_H.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $001C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnTMK[7:0]								FLXnCLK[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-19 FLXnLCK register contents

Bit position	Bit name	Function
15 to 8	FLXn TMK[7:0]	Test Mode Key To write bit FLXnTEST1.FLXnWRTEN, the write operation has to be directly preceded by two consecutive write accesses to the Test Mode Key (unlock sequence). If the write sequence below is interrupted by other write accesses between the second write to the Test Mode Key and the write access to the FLXnTEST1 register, FLXnTEST1.FLXnWRTEN is not set to '1' and the sequence has to be repeated. First write: FLXnLCK.TMK[7:0] = "0111 0101" (75_H) Second write: FLXnLCK.TMK[7:0] = "1000 1010" ($8A_H$) Third write: FLXnTEST1.FLXnWRTEN = 1
7 to 0	FLXn CLK[7:0]	Configuration Lock Key To leave CONFIG state by writing FLXnSUCC1.FLXnCMD[3:0] (commands READY, MONITOR_MODE, ATM, LOOP_BACK), the write operation has to be directly preceded by two write accesses to the Configuration Lock Key (unlock sequence). If the write sequence below is interrupted by other write accesses between the second write to the Configuration Lock Key and the write access to the FLXnSUCC1 register, the CC remains in CONFIG state and the sequence has to be repeated. First write: FLXnLCK.CLK[7:0] = "1100 1110" (CE_H) Second write: FLXnLCK.CLK[7:0] = "0011 0001" (31_H) Third write: FLXnSUCC1.FLXnCMD[3:0]

30.3.4 Interrupt registers

(1) FLXnEIR - Error interrupt register

The flags are set when the CC detects one of the listed error conditions. They remain set until the Host clears them. A flag is cleared by writing a '1' to the corresponding bit position. After writing '1', read the register again to confirm that the corresponding bit has actually been cleared. If it has not been cleared, perform the operation to clear it again. Writing a '0' has no effect on the flag. A hard reset will also clear the register.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0020_H

Initial Value $0000\ 0000_H$

(a) FLXnEIR read

31	30	29	28	27	26	25	24
0	0	0	0	0	FLXn TABBE	FLXn LTVBE	FLXn EDBE
23	22	21	20	19	18	17	16
0	0	0	0	0	FLXn TABAE	FLXn LTVAE	FLXn EDAE
15	14	13	12	11	10	9	8
0	0	0	0	FLXn MHFE	FLXn IOBAE	FLXn IIBAE	FLXn EFAE
7	6	5	4	3	2	1	0
FLXn RFOE	0	FLXn CCLE	FLXn CCFE	FLXn SFOE	FLXn SFBME	FLXn CNAE	FLXn PEMCE

Table 30-20 FLXnEIR read register contents (1/3)

Bit position	Bit name	Function
Channel-specific error flags channel B:		
26	FLXn TABBE	The flag signals to the Host that a transmission across a slot boundary occurred for channel B. 1 = Transmission across slot boundary detected on channel B 0 = No transmission across slot boundary detected on channel B
25	FLXn LTVBE	Latest Transmit Violation Channel B The flag signals a latest transmit violation on channel B to the Host. 1 = Latest transmit violation detected on channel B 0 = No latest transmit violation detected on channel B
24	FLXn EDBE	Error Detected on Channel B This bit is set whenever one of the flags FLXnACS.FLXnSEDBE, FLXnACS.FLXnCEDBE, FLXnACS.FLXnCIBE, FLXnACS.FLXnSBVBE changes from '0' to '1'. 1 = Error detected on channel B 0 = No error detected on channel B

Table 30-20 FLXnEIR read register contents (2/3)

Bit position	Bit name	Function
Channel-specific error flags channel A:		
18	FLXn TABAE	The flag signals to the Host that a transmission across a slot boundary occurred for channel A. 1 = Transmission across slot boundary detected on channel A 0 = No transmission across slot boundary detected on channel A
17	FLXn LTVAE	Latest Transmit Violation Channel A The flag signals a latest transmit violation on channel A to the Host. 1 = Latest transmit violation detected on channel A 0 = No latest transmit violation detected on channel A
16	FLXn EDAE	Error Detected on Channel A This bit is set whenever one of the flags FLXnACS.FLXnSEDAE, FLXnACS.FLXnCEDAE, FLXnACS.FLXnCIAE, FLXnACS.FLXnSBVAE changes from ’0’ to ’1’. 1 = Error detected on channel A 0 = No error detected on channel A
Flags for all channels:		
11	FLXn MHFE	Message Handler Constraints Flag The flag signals a Message Handler constraints violation condition. It is set whenever one of the flags FLXnMHDF.FLXnSNUAE, FLXnMHDF.FLXnSNUBE, FLXnMHDF.FLXnFNFAE, MHDF.FLXnFNFBE, FLXnMHDF.FLXnDTBFAE, FLXnMHDF.FLXnDTBFBE, FLXnMHDF.FLXnWAHPE changes from ’0’ to ’1’. 1 = Message Handler failure detected 0 = No Message Handler failure detected
10	FLXn IOBAE	Illegal Output buffer Access This flag is set by the CC when the Host requests the transfer of a message buffer from the Message RAM to the Output Buffer while FLXnOBCR.FLXnOBSYS is set to ’1’. 1 = Illegal Host access to Output Buffer occurred 0 = No illegal Host access to Output Buffer occurred
9	FLXn IIBAE	Illegal Input Buffer Access This flag is set by the CC when the Host wants to modify a message buffer via Input Buffer and one of the following conditions applies: • The CC is not in CONFIG or DEFAULT_CONFIG state and the Host writes to the Input Buffer Command Request register to modify the - Header section of message buffer 0, 1 if configured for transmission in key slot - Header section of static message buffers with buffer number < FLXnMRC.FLXnFDB[7:0] while FLXnMRC.FLXnSEC[1:0] = "01" - Header section of any static or dynamic message buffer while FLXnMRC.FLXnSEC[1:0] = "1x" - Header and / or data section of any message buffer belonging to the receive FIFO • The Host writes to any register of the Input Buffer while FLXnIBCR.FLXnIBSYH is set to ’1’. 1 = Illegal Host access to Input Buffer occurred 0 = No illegal Host access to Input Buffer occurred
8	FLXn EDAE	Empty FIFO Access This flag is set by the CC when the Host requests the transfer of a message from the receive FIFO via Output Buffer while the receive FIFO is empty. 1 = Host access to empty FIFO occurred 0 = No Host access to empty FIFO occurred

Table 30-20 FLXnEIR read register contents (3/3)

Bit position	Bit name	Function
7	FLXn RFOE	Receive FIFO Overrun The flag is set by the CC when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. The actual state of the FIFO is monitored in register FLXnFSR. 1 = A receive FIFO overrun has been detected 0 = No receive FIFO overrun detected
5	FLXn CCLE	CHI Command Locked The flag signals that the write access to the CHI command vector FLXnSUCC1.FLXnCMD[3:0] was not successful because the execution of the previous CHI command has not yet completed. In this case bit FLXnCNAE is also set to '1'. 1 = CHI command not accepted 0 = CHI command accepted
4	FLXn CCFE	Clock Correction Failure This flag is set at the end of the cycle whenever one of the following errors occurred: • Missing offset and / or rate correction • Clock correction limit reached The clock correction status is monitored in registers FLXnCCEV and FLXnSFS. A failure may occur during startup, therefore bit FLXnCCFE should be cleared by the Host after the CC entered NORMAL_ACTIVE state. 1 = Clock correction failed 0 = No clock correction error
3	FLXn SFOE	Sync Frame Overflow Set when either the number of sync frames received during the last communication cycle or the total number of different sync frame IDs received during the last double cycle exceeds the maximum number of sync frames as defined by FLXnGTUC02.FLXnSNM[3:0]. 1 = More sync frames received than configured by FLXnGTUC02.FLXnSNM[3:0] 0 = Number of received sync frames FLXnGTUC02.FLXnSNM[3:0]
2	FLXn SFBME	Sync Frames Below Minimum This flag signals that the number of sync frames received during the last communication cycle was below the limit required by the FlexRay protocol. May be set during startup and therefore should be cleared by the Host after the CC entered NORMAL_ACTIVE state. 1 = Less than the required minimum of sync frames received 0 = Sync node: 1 or more sync frames received Non-sync node:2 or more sync frames received
1	FLXn CNAE	Command Not Accepted The flag signals that the write access to the CHI command vector FLXnSUCC1.FLXnCMD[3:0] was not successful because the requested command was not valid in the actual POC state, or because the CHI command was locked (FLXnCCLE = '1'). 1 = CHI command not accepted 0 = CHI command accepted
0	FLXn PEMCE	POC Error Mode Changed This flag is set whenever the error mode signalled by FLXnCCEV.FLXnERRM[1:0] has changed. 1 = Error mode has changed 0 = Error mode has not changed

(b) FLXnEIR write

31	30	29	28	27	26	25	24
–	–	–	–	–	FLXnCL TABB	FLXnCL LTVB	FLXnCL EDB
23	22	21	20	19	18	17	16
–	–	–	–	–	FLXnCL TABA	FLXnCL LTVA	FLXnCL EDA
15	14	13	12	11	10	9	8
–	–	–	–	FLXnCL MHF	FLXnCL IOBA	FLXnCL IIBA	FLXnCL EFA
7	6	5	4	3	2	1	0
FLXnCL RFO	–	FLXnCL CCL	FLXnCL CCF	FLXnCL SFO	FLXnCL SFBM	FLXnCL CNA	FLXnCL PEMC

Table 30-21 FLXnEIR write register contents

Bit position	Bit name	Function
31 to 0	see above	Clears the corresponding flags described in *Table 30-20 "FLXnEIR read register contents" on page 2389* : 1 = Clears the corresponding flag. 0 = No function

Note Writing to bits 31 to 27, 23 to 19, 15 to 12 and 6 is ignored.

(2) FLXnSIR - Status interrupt register

The flags are set when the CC detects one of the listed events. The flags remain set until the Host clears them. A flag is cleared by writing a ’1’ to the corresponding bit position. After writing ‘1’, read the register again to confirm that the corresponding bit has actually been cleared. If it has not been cleared, perform the operation to clear it again. Writing a ’0’ has no effect on the flag. A hard reset will also clear the register.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0024_H

Initial Value $0000\ 0000_H$

(a) FLXnSIR read

31	30	29	28	27	26	25	24
0	0	0	0	0	0	FLXn MTSBF	FLXn WUPBF
23	22	21	20	19	18	17	16
0	0	0	0	0	0	FLXn MTSAF	FLXn WUPAF
15	14	13	12	11	10	9	8
FLXn SDSF	FLXn MBSIF	FLXn SUCSF	FLXn SWEF	FLXn TOBCF	FLXn TIBCF	FLXn TI1F	FLXn TI0F
7	6	5	4	3	2	1	0
FLXn NMVCF	FLXn RFCLF	FLXn RFNEF	FLXn RXIF	FLXn TXIF	FLXn CYCSF	FLXn CASF	FLXn WSTF

Table 30-22 FLXnSIR read register contents (1/3)

Bit position	Bit name	Function
Channel-specific status flags:		
25	FLXn MTSBF	MTS Received on Channel B (*vSS!ValidMTSB*) Media Access Test symbol received on channel B during the preceding symbol window. Updated by the CC for each channel at the end of the symbol window. 1 = MTS symbol received on channel B 0 = No MTS symbol received on channel B
24	FLXn WUPBF	Wakeup Pattern Channel B This flag is set by the CC when a wakeup pattern was received on channel B. Only set when the CC is in WAKEUP, READY, or STARTUP state, or when in Monitor mode. 1 = Wakeup pattern received on channel B 0 = No wakeup pattern received on channel B
17	FLXn MTSAF	MTS Received on Channel A (*vSS!ValidFLXnMTSA*) Media Access Test symbol received on channel A during the preceding symbol window. Updated by the CC for each channel at the end of the symbol window. 1 = MTS symbol received on channel A 0 = No MTS symbol received on channel A
16	FLXn WUPAF	Wakeup Pattern Channel A This flag is set by the CC when a wakeup pattern was received on channel A. Only set when the CC is in WAKEUP, READY, or STARTUP state, or when in Monitor mode. 1 = Wakeup pattern received on channel A 0 = No wakeup pattern received on channel A

Table 30-22 FLXnSIR read register contents (2/3)

Bit position	Bit name	Function
Flags for all channels:		
15	FLXn SDSF	Start of Dynamic Segment This flag is set by the CC when the dynamic segment starts. 1 = Dynamic segment started 0 = Dynamic segment not yet started
14	FLXn MBSIF	Message Buffer Status Interrupt This flag is set by the CC when the message buffer status MBS has changed and if bit FLXnRHMBI of that message buffer is set. 1 = Message buffer status of at least one message buffer with FLXnRHMBI = '1' has changed 0 = No message buffer status change of message buffer with FLXnRHMBI = '1'
13	FLXn SUCSF	Startup Completed Successfully This flag is set whenever a startup completed successfully and the CC entered NORMAL_ACTIVE state. 1 = Startup completed successfully 0 = No startup completed successfully
12	FLXn SWEF	Stop Watch Event This flag is set after a stop watch activation when the actual cycle counter and macrotick value are stored in the Stop Watch register (FLXnSTPW1)). 1 = Stop Watch Event occurred 0 = No Stop Watch Event
11	FLXn TOBCF	Transfer Output Buffer Completed This flag is set whenever a transfer from the Message RAM to the Output Buffer has completed and bit FLXnOBCR.FLXnOBSYS in the Output Buffer Command Request register has been reset by the Message Handler. 1 = Transfer between Message RAM and Output Buffer completed 0 = No transfer completed since bit was reset
10	FLXn TIBCF	Transfer Input Buffer Completed This flag is set whenever a transfer from Input Buffer to the Message RAM has completed and bit FLXnIBCR.FLXnIBSYS in the Input Buffer Command Request register has been reset by the Message Handler. 1 = Transfer between Input Buffer and Message RAM completed 0 = No transfer completed since bit was reset
9	FLXn TI1F	Timer Interrupt 1 This flag is set whenever timer 1 matches the conditions configured in register FLXnT1C. A Timer Interrupt 1 is also signalled on pin eray_tint1. 1 = Timer Interrupt 1 occurred 0 = No Timer Interrupt 1
8	FLXn TI0F	Timer Interrupt 0 This flag is set whenever timer 0 matches the conditions configured in register FLXnT0C. A Timer Interrupt 0 is also signalled on pin eray_tint0. 1 = Timer Interrupt 0 occurred 0 = No Timer Interrupt 0
7	FLXn NMVCF	Network Management Vector Changed This interrupt flag signals a change in the Network Management Vector visible to the Host. 1 = Network management vector changed 0 = No change in the network management vector
6	FLXn RFCLF	Receive FIFO Critical Level This flag is set when the receive FIFO fill level FLXnFSR.FLXnRFFL[7:0] is equal or greater than the critical level as configured by FLXnFCL.FLXnCL[7:0]. 1 = Receive FIFO critical level reached 0 = Receive FIFO below critical level

Table 30-22 FLXnSIR read register contents (3/3)

Bit position	Bit name	Function
5	FLXn RFNEF	Receive FIFO Not Empty This flag is set by the CC when a received valid frame was stored into the empty receive FIFO. The actual state of the receive FIFO is monitored in register FLXnFSR. 1 = Receive FIFO is not empty 0 = Receive FIFO is empty
4	FLXn RXIF	Receive Interrupt This flag is set by the CC whenever the set condition of a message buffers ND flag is fulfilled (see New Data 1/2/3/4 (FLXnNDAT1/2/3/4)), and if bit FLXnRHMBI of that message buffer is set to '1'. 1 = At least one data section has been updated 0 = No data section has been updated
3	FLXn TXIF	Transmit Interrupt This flag is set by the CC after successful frame transmission if bit FLXnWHMBI in the respective message buffer is set to '1'. 1 = At least one frame was transmitted successfully 0 = No frame transmitted
2	FLXn CYCSF	Cycle Start Interrupt This flag is set by the CC when a communication cycle starts 1 = Communication cycle started 0 = No communication cycle started
1	FLXn CASF	Collision Avoidance Symbol This flag is set by the CC during STARTUP state when a CAS or a potential CAS was received. 1 = Bit pattern matching the CAS symbol received 0 = No bit pattern matching the CAS symbol received
0	FLXn WSTF	Wakeup Status This flag is set when the wakeup status vector FLXnCCSV.FLXnWSV[2:0] is changed by a protocol event. 1 = Wakeup status changed 0 = Wakeup status unchanged

(b) FLXnSIR write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	FLXnCL MTSB	FLXnCL WUPB
23	22	21	20	19	18	17	16
–	–	–	–	–	–	FLXnCL MTSA	FLXnCL WUPA
15	14	13	12	11	10	9	8
FLXnCL SDS	FLXnCL MBSI	FLXnCL SUCS	FLXnCL SWE	FLXnCL TOBC	FLXnCL TIBC	FLXnCL TI1	FLXnCL TI0
7	6	5	4	3	2	1	0
FLXnCL NMVC	FLXnCL RFCL	FLXnCL RFNE	FLXnCL RXI	FLXnCL TXI	FLXnCL CYCS	FLXnCL CAS	FLXnCL WST

Table 30-23 FLXnSIR write register contents

Bit position	Bit name	Function
31 to 0	see above	Clears the corresponding flags described in *Table 30-22 "FLXnSIR read register contents" on page 2393* : 1 = Clears the corresponding flag. 0 = No function

Note Writing to bits 31 to 26 and 23 to 18 is ignored.

(3) FLXnEILS - Error interrupt line select

The Error Interrupt Line Select register assigns an interrupt generated by a specific error interrupt flag from register FLXnEIR to one of the two module interrupt lines:

1 = Interrupt assigned to interrupt line eray_int1

0 = Interrupt assigned to interrupt line eray_int0

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0028_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXn TABBL	FLXn LTVBL	FLXn EDBL	0	0	0	0	0	FLXn TABAL	FLXn LTVAL	FLXn EDAL
R	R	R	R	R	R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	FLXn MHFL	FLXn IOBAL	FLXn IIBAL	FLX nEFAL	FLXn RFOL	0	FLXn CCLL	FLXn CCFL	FLXn SFOL	FLXn SFBML	FLXn CNAL	FLXn PEMCL
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-24 FLXnEILS register contents (1/2)

Bit position	Bit name	Function
26	FLXn TABBL	Transmission Across Boundary Channel B Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
25	FLXn LTVBL	Latest Transmit Violation Channel B Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
24	FLXn EDBL	Error Detected on Channel B Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
18	FLXn TABAL	Transmission Across Boundary Channel A Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
17	FLXn LTVAL	Latest Transmit Violation Channel A Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
16	FLXn EDAL	Error Detected on Channel A Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
11	FLXn MHFL	Message Handler Constraints Flag Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
10	FLXn IOBAL	Illegal Output Buffer Access Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
9	FLXn IIBAL	Illegal Output Buffer Access Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0

Table 30-24 FLXnEILS register contents (2/2)

Bit position	Bit name	Function
8	FLXn EFAL	Empty FIFO Access Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
7	FLXn RFOL	Receive FIFO Overrun Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
5	FLXn CCLL	CHI Command Locked Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
4	FLXn CCFL	Clock Correction Failure Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
3	FLXn SFOL	Sync Frame Overflow Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
2	FLXn SFBML	Sync Frames Below Minimum Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
1	FLXn CNAL	Command Not Accepted Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
0	FLXn PEMCL	POC Error Mode Changed Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0

(4) FLXnSILS - Status Interrupt Line Select

The settings in the Status Interrupt Line Select register assign an interrupt generated by a specific status interrupt flag to one of the two module interrupt lines (eray_int0 or eray_int1).

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $002C_H$

Initial Value $0303\ FFFF_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	FLXn MTSBL	FLXn WUPBL	0	0	0	0	0	0	FLXn MTSAL	FLXn WUPAL
R	R	R	R	R	R	R/W	R/W	R	R	R	R	R	R	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXn SDSL	FLXn MBSIL	FLXn SUCSL	FLXn SWEL	FLXn TOBCL	FLXn TIBCL	FLXn TI1L	FLXn TI0L	FLXn NMVCL	FLXn RFCLL	FLXn RFNEL	FLXn RXIL	FLXn TXIL	FLXn CYCSL	FLXn CASL	FLXn WSTL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-25 FLXnSILS register contents (1/2)

Bit position	Bit name	Function
25	FLXn MTSBL	Media Access Test Symbol Channel B Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
24	FLXn WUPBL	Wakeup Pattern Channel B Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
17	FLXn MTSAL	Media Access Test Symbol Channel A Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
16	FLXn WUPAL	Wakeup Pattern Channel A Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
15	FLXn SDSL	Start of Dynamic Segment Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
14	FLXn MBSIL	Message Buffer Status Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
13	FLXn SUCSL	Startup Completed Successfully Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
12	FLXn SWEL	Stop Watch Event Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
11	FLXn TOBCL	Transfer Output Buffer Completed Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0

Table 30-25 FLXnSILS register contents (2/2)

Bit position	Bit name	Function
10	FLXn TIBCL	Transfer Input Buffer Completed Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
9	FLXn TI1L	Timer Interrupt 1 Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
8	FLXn TI0L	Timer Interrupt 0 Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
7	FLXn NMVCL	Network Management Vector Changed Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
6	FLXn RFCLL	Receive FIFO Critical Level Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
5	FLXn RFNEL	Receive FIFO Not Empty Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
4	FLXn RXIL	Receive Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
3	FLXn TXIL	Transmit Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
2	FLXn CYCSL	Cycle Start Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
1	FLXn CASL	Collision Avoidance Symbol Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0
0	FLXn WSTL	Wakeup Status Interrupt Line 1 = Interrupt assigned to interrupt line eray_int1 0 = Interrupt assigned to interrupt line eray_int0

(5) FLXnEIES/FLXnEIER - Error interrupt enable set / reset

The settings in the Error Interrupt Enable register determine which status changes in the Error Interrupt Register will result in an interrupt.

The enable bits are set by writing to FLXnEIES and reset by writing to FLXnEIER. Writing a '1' sets / resets the specific enable bit, writing a '0' has no effect. Reading from both addresses will result in the same value.

Access This register can be read/written in 32-bit units.

Address FLXnEIES: <FLXn_base> + 0030_H
FLXnEIER: <FLXn_base> + 0034_H

Initial Value $0000\ 0000_H$

(a) FLXnEIES:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnSETABBE	FLXnSELTVBE	FLXnSEEDBE	0	0	0	0	0	FLXnSETABAE	FLXnSELTVAE	FLXnSEEDAE
R	R	R	R	R	R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	FLXnSEMHFE	FLXnSEIOBAE	FLXnSEIIBAE	FLXnSEEFAE	FLXnSERFOE	0	FLXnSECCLE	FLXnSECCFE	FLXnSESFOE	FLXnSESFBME	FLXnSECNAE	FLXnSEPEMCE
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

(b) FLXnEIER:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnCLTABBE	FLXnCLLTVBE	FLXnCLEDBE	0	0	0	0	0	FLXnCLTABAE	FLXnCLLTVAE	FLXnCLEDAE
R	R	R	R	R	R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	FLXnCLMHFE	FLXnCLIOBAE	FLXnCLIIBAE	FLXnCLEFAE	FLXnCLRFOE	0	FLXnCLCCLE	FLXnCLCCFE	FLXnCLSFOE	FLXnCLSFBME	FLXnCLCNAE	FLXnCLPEMCE
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

All bits of these registers behave as follows:

- write FLXnEIES.FLXnSE... = 1: enable interrupt
- write FLXnEIER.FLXnCL... = 1: disable interrupt
- read FLXnEIES.FLXnSE... or FLXnEIER.FLXnCL... = 1: interrupt enabled
- read FLXnEIES.FLXnSE... or FLXnEIER.FLXnCL... = 0: interrupt disabled

Table 30-26 FLXnEIES/FLXnEIER register contents

Bit position	Bit name	Function
26	FLXnSETABBE FLXnCLTABBE	Transmission Across Boundary Channel B Interrupt
25	FLXnSELTVBE FLXnCLLTVBE	Latest Transmit Violation Channel B Interrupt
24	FLXnSEEDBE FLXnCLEDBE	Error Detected on Channel B Interrupt
18	FLXnSETABAE FLXnCLTABAE	Transmission Across Boundary Channel A Interrupt
17	FLXnSELTVAE FLXnCLLTVAE	Latest Transmit Violation Channel A Interrupt
16	FLXnSEEDAE FLXnCLEDAE	Error Detected on Channel A Interrupt
11	FLXnSEMHFE FLXnCLMHFE	Message Handler Constraints Flag Interrupt
10	FLXnSEIOBAE FLXnCLIOBAE	Illegal Output Buffer Access Interrupt
9	FLXnSEIIBAE FLXnCLIIBAE	Illegal Input Buffer Access Interrupt
8	FLXnSEEFAE FLXnCLEFAE	Empty FIFO Access Interrupt
7	FLXnSERFOE FLXnCLRFOE	Receive FIFO Overrun Interrupt
5	FLXnSECCLE FLXnCLCCLE	CHI Command Locked Interrupt
4	FLXnSECCFE FLXnCLCCFE	Clock Correction Failure Interrupt
3	FLXnSESFOE FLXnCLSFOE	Sync Frame Overflow Interrupt
2	FLXnSESFBME FLXnCLSFBME	Sync Frames Below Minimum Interrupt
1	FLXnSECNAE FLXnCLCNAE	Command Not Accepted Interrupt
0	FLXnSEPEMCE FLXnCLPEMCE	POC Error Mode Changed Interrupt

(6) FLXnSIES/FLXnSIER - Status interrupt enable set / reset

The settings in the Status Interrupt Enable register determine which status changes in the Status Interrupt Register will result in an interrupt.

The enable bits are set by writing to FLXnSIES and reset by writing to FLXnSIER. Writing a '1' sets / resets the specific enable bit, writing a '0' has no effect. Reading from both addresses will result in the same value.

Access This register can be read/written in 32-bit units.

Address FLXnSIES: <FLXn_base> + 0038_H
FLXnSIER: <FLXn_base> + $003C_H$

Initial Value $0000\ 0000_H$

(a) FLXnSIES:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	FLXnSEMTSBE	FLXnSEWUPBE	0	0	0	0	0	0	FLXnSEMTSAE	FLXnSEWUPAE
R	R	R	R	R	R	R/W	R/W	R	R	R	R	R	R	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnSESDSE	FLXnSEMBSIE	FLXnSESUCSE	FLXnSESWEE	FLXnSETOBCE	FLXnSETIBCE	FLXnSETI1E	FLXnSETI0E	FLXnSENMVCE	FLXnSERFCLE	FLXnSERFNEE	FLXnSERXIE	FLXnSETXIE	FLXnSECYCSE	FLXnSECASE	FLXnSEWSTE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

(b) FLXnSIER:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	FLXnCLMTSBE	FLXnCLWUPBE	0	0	0	0	0	0	FLXnCLMTSAE	FLXnCLWUPAE
R	R	R	R	R	R	R/W	R/W	R	R	R	R	R	R	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnCLSDSE	FLXnCLMBSIE	FLXnCLSUCSE	FLXnCLSWEE	FLXnCLTOBCE	FLXnCLTIBCE	FLXnCLTI1E	FLXnCLTI0E	FLXnCLNMVCE	FLXnCLRFCLE	FLXnCLRFNEE	FLXnCLRXIE	FLXnCLTXIE	FLXnCLCYCSE	FLXnCLCASE	FLXnCLWSTE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

All bits of these registers behave as follows:

- write FLXnSIES.FLXnSE... = 1: enable interrupt
- write FLXnSIER.FLXnCL... = 1: disable interrupt
- read FLXnSIES.FLXnSE... or FLXnSIER.FLXnCL... = 1: interrupt enabled
- read FLXnSIES.FLXnSE... or FLXnSIER.FLXnCL... = 0: interrupt disabled

Table 30-27 FLXnSIES/FLXnSIER register contents

Bit position	Bit name	Function
25	FLXnSEMTSBE FLXnCLMTSBE	MTS Received on Channel B Interrupt
24	FLXnSEWUPBE FLXnCLWUPBE	Wakeup Pattern Channel B Interrupt
17	FLXnSEMTSAE FLXnCLMTSAE	MTS Received on Channel A Interrupt
18	FLXnSEWUPAE FLXnCLWUPAE	Wakeup Pattern Channel A Interrupt
17	FLXnSESDSE FLXnCLSDSE	Start of Dynamic Segment Interrupt
16	FLXnSEMBSIE FLXnCLMBSIE	Message Buffer Status Interrupt
13	FLXnSESUCSE FLXnCLSUCSE	Startup Completed Successfully Interrupt
12	FLXnSESWEE FLXnCLSWEE	Stop Watch Event Interrupt
11	FLXnSETOBCE FLXnCLTOBCE	Transfer Output Buffer Completed Interrupt
10	FLXnSETIBCE FLXnCLTIBCE	Transfer Input Buffer Completed Interrupt
9	FLXnSETI1E FLXnCLTI1E	Timer Interrupt 1
8	FLXnSETI0E FLXnCLTI0E	Timer Interrupt 0
7	FLXnSENMVCE FLXnCLNMVCE	Network Management Vector Changed Interrupt
6	FLXnSERFCLE FLXnCLRFCLE	Receive FIFO Critical Level Interrupt
5	FLXnSERFNEE FLXnCLRFNEE	Receive FIFO Not Empty Interrupt
4	FLXnSERXIE FLXnCLRXIE	Receive Interrupt
3	FLXnSETXIE FLXnCLTXIE	Transmit Interrupt
2	FLXnSECYCSE FLXnCLCYCSE	Cycle Start Interrupt
1	FLXnSECASE FLXnCLCASE	Collision Avoidance Symbol Interrupt
0	FLXnSEWSTE FLXnCLWSTE	Wakeup Status Interrupt

(7) FLXnILE - Interrupt line enable

Each of the two interrupt lines to the Host CPU (eray_int0, eray_int1) can be enabled / disabled separately by programming bit FLXnEINTL0 and FLXnEINTL1.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0040_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn EINTL1	FLXn EINTL0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 30-28 FLXnILE register contents

Bit position	Bit name	Function
1	FLXn EINTL1	Enable Interrupt Line 1 1 = Interrupt line eray_int1 enabled 0 = Interrupt line eray_int1 disabled
0	FLXn EINTL0	Enable Interrupt Line 0 1 = Interrupt line eray_int0 enabled 0 = Interrupt line eray_int0 disabled

(8) FLXnT0C - Timer 0 configuration

Absolute timer. Specifies in terms of cycle count and macrotick the point in time when the timer 0 interrupt occurs. When the timer 0 interrupt is asserted, output signal eray_tint0 is set to ’1’ for the duration of one macrotick and FLXnSIR.FLXnTI0F is set to ’1’.

Timer 0 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 0 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit FLXnT0RC to ’0’.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0044_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnT0MO[13:00]													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FLXnT0CC[6:0]							0	0	0	0	0	0	FLXn T0MS	FLXn T0RC
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R/W	R/W

Table 30-29 FLXnT0C register contents

Bit position	Bit name	Function
29 to 16	FLXn T0MO[13:00]	Timer 0 Macrotick Offset Configures the macrotick offset from the beginning of the cycle where the interrupt is to occur. The Timer 0 Interrupt occurs at this offset for each cycle in the cycle set.
14 to 8	FLXn T0CC[6:0]	Timer 0 Cycle Code The 7-bit timer 0 cycle code determines the cycle set used for generation of the timer 0 interrupt. For details about the configuration of the cycle code see 5.7.3 Cycle Counter Filtering.
1	FLXn T0MS	Timer 0 Mode Select 1 = Continuous mode 0 = Single-shot mode
0	FLXn T0RC	Timer 0 Run Control 1 = Timer 0 running 0 = Timer 0 halted

Note The configuration of timer 0 is compared against the macrotick counter value, there is no separate counter for timer 0. In case the CC leaves NORMAL_ACTIVE or NORMAL_PASSIVE state, or if timer 0 is halted by Host command, output signal eray_tint0 is reset to ’0’ immediately.

(9) FLXnT1C - Timer 1 configuration

Relative timer. After the specified number of macroticks has expired, the timer 1 interrupt is asserted, output signal eray_tint1 is set to '1' for the duration of one macrotick and FLXnSIR.FLXnTI1F is set to '1'.

Timer 1 can be activated as long as the POC is either in NORMAL_ACTIVE state or in NORMAL_PASSIVE state. Timer 1 is deactivated when leaving NORMAL_ACTIVE state or NORMAL_PASSIVE state except for transitions between the two states.

Before reconfiguration of the timer, the timer has to be halted first by writing bit FLXnT1RC to '0'.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0048_H

Initial Value $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnT1MC[13:00]													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn T1MS	FLXn T1RC
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 30-30 FLXnT1C register contents

Bit position	Bit name	Function
29 to 16	FLXn T1MC[13:00]	Timer 1 Macrotick Count When the configured macrotick count is reached the timer 1 interrupt is generated. Valid values are: 2 to 16383 MT in continuous mode 1 to 16383 MT in single-shot mode
1	FLXn T1MS	Timer 1 Mode Select 1 = Continuous mode 0 = Single-shot mode
0	FLXn T1RC	Timer 1 Run Control 1 = Timer 1 running 0 = Timer 1 halted

Note In case the CC leaves NORMAL_ACTIVE or NORMAL_PASSIVE state, or if timer 1 is halted by Host command, output signal eray_tint1 is reset to '0' immediately.

(10) FLXnSTPW1 - Stop watch register

The stop watch is activated by a rising or falling edge on pin eray_stpwt, by an interrupt 0,1 event (rising edge on pin eray_int0 or eray_int1) or by the Host by writing bit FLXnSSWT to '1'. With the macrotick counter increment following next to the stop watch activation the actual cycle counter and macrotick values are captured in register FLXnSTPW1 while the slot counter values for channel A and B are captured in register FLXnSTPW2.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $004C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnSMTV[13:00]													
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnSCCV[5:0]						0	FLXn EINT1	FLXn EINT0	FLXn EETP	FLXn SSWT	FLXn EDGE	FLXn SWMS	FLXn ESWT
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-31 FLXnSTPW1 register contents (1/2)

Bit position	Bit name	Function
29 to 16	FLXn SMTV[13:00]	Stopped Macrotick Value State of the macrotick counter when the stop watch event occurred. Valid values are 0 to 15,999.
13 to 8	FLXn SCCV[5:0]	Stopped Cycle Counter Value State of the cycle counter when the stop watch event occurred. Valid values are 0 to 63
6	FLXn EINT1	Enable Interrupt 1 Trigger Enables stop watch trigger by interrupt 1event if FLXnESWT = '1'. 1 = Interrupt 1 event triggers stop watch 0 = Stop watch trigger by interrupt 1 disabled
5	FLXn EINT0	Enable Interrupt 0 Trigger Enables stop watch trigger by interrupt 0 event if FLXnESWT = '1'. 1 = Interrupt 0 event triggers stop watch 0 = Stop watch trigger by interrupt 0 disabled
4	FLXn EETP	Enable External Trigger Pin Enables stop watch trigger event via pin eray_stpwt if FLXnESWT = '1'. 1 = Edge on pin eray_stpwt triggers stop watch 0 = Stop watch trigger via pin eray_stpwt disabled
3	FLXn SSWT	Software Stop Watch Trigger When the Host writes this bit to '1' the stop watch is activated. After the actual cycle counter and macrotick value are stored in the Stop Watch register this bit is reset to '0'. The bit is only writeable while FLXnESWT = '0'. 1 = Stop watch activated by software trigger 0 = Software trigger reset

Table 30-31 FLXnSTPW1 register contents (2/2)

Bit position	Bit name	Function
2	FLXn EDGE	Stop Watch Trigger Edge Select 1 = Rising edge 0 = Falling edge
1	FLXn SWMS	Stop Watch Mode Select 1 = Continuous mode 0 = Single-shot mode
0	FLXn ESWT	External Stop Watch Trigger If enabled an edge on input eray_stpwt or an interrupt 0,1 event (rising edge on pin eray_int0 or eray_int1) activates the stop watch. In single-shot mode this bit is reset to '0' after the actual cycle counter and macrotick value are stored in the Stop Watch register. 1 = Stop watch trigger enabled 0 = Stop watch trigger disabled

Note Bits FLXnESWT and FLXnSSWT cannot be set to '1' simultaneously. In this case the write access to the register is ignored, and both bits keep their previous values. Either the external stop watch trigger or the software stop watch trigger may be used.

(11) FLXnSTPW2 - Stop watch register

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0050_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnSSCVB[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnSSCVA[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-32 FLXnSTPW2 register contents

Bit position	Bit name	Function
26 to 16	FLXn SSCVB[10:00]	Stop Watch Captured Slot Counter Value Channel B State of the slot counter for channel B when the stop watch event occurred. Valid values are 0 to 2047.
10 to 0	FLXn SSCVA[10:00]	Stop Watch Captured Slot Counter Value Channel A State of the slot counter for channel A when the stop watch event occurred. Valid values are 0 to 2047.

30.3.5 CC control registers

This section describes the registers provided by the CC to allow the Host to control the operation of the CC. The FlexRay protocol specification requires the Host to write application configuration data in CONFIG state only. Please consider that the configuration registers are not locked for writing in DEFAULT_CONFIG state.

The configuration data is reset when DEFAULT_CONFIG state is entered from hard reset. To change POC state from DEFAULT_CONFIG to CONFIG state the Host has to apply CHI command CONFIG. If the Host wants the CC to leave CONFIG state, the Host has to proceed as described in Section Lock Register (LCK).

All bits marked with an asterisk * can be updated in DEFAULT_CONFIG or CONFIG state only!

(1) FLXnSUCC1 - SUC configuration register 1

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0080_H

Initial Value $0C40\ 1080_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	FLXn CCHB*	FLXn CCHA*	FLXn MTSB*	FLXn MTSA*	FLXn HCSE*	FLXn TSM*	FLXn WUCS*	FLXnPTA[4:0]*				
R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnCSA[4:0]*					0	FLXn TXSY*	FLXn TXST*	FLXn PBSY	0	0	0	FLXnCMD[3:0]			
R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

Table 30-33 FLXnSUCC1 register contents (1/3)

Bit position	Bit name	Function
27	FLXn CCHB	Connected to Channel B (*pChannels*) Configures whether the node is connected to channel B. 1 = Node connected to channel B (default by hardware reset) 0 = Not connected to channel B
26	FLXn CCHA	Connected to Channel A (*pChannels*) Configures whether the node is connected to channel A. 1 = Node connected to channel A (default by hardware reset) 0 = Not connected to channel A

Table 30-33 FLXnSUCC1 register contents (2/3)

Bit position	Bit name	Function
25	FLXn MTSB	Select Channel B for MTS Transmission The bit selects channel B for MTS symbol transmission. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state. 1 = Channel B selected for MTS transmission 0 = Channel B disabled for MTS transmission Note: FLXnMTSA,B may also be changed outside DEFAULT_CONFIG or CONFIG state when the write to FLXnSUCC1 register is directly preceded by the unlock sequence for the Configuration Lock Key as described in 21.4.3.2 Lock Register (FLXnLCK). This may be combined with CHI command SEND_MTS. If both bits FLXnMTSA and FLXnMTSB are set to '1' an MTS symbol will be transmitted on both channels when requested by writing FLXnCMD[3:0] = "1000".
24	FLXn MTSA	Select Channel A for MTS Transmission The bit selects channel A for MTS symbol transmission. The flag is reset by default and may be modified only in DEFAULT_CONFIG or CONFIG state. 1 = Channel A selected for MTS transmission 0 = Channel A disabled for MTS transmission
23	FLXn HCSE	Halt due to Clock Sync Error (*pAllowHaltDueToClock*) Controls reaction of the CC to a clock synchronization error. The bit can be modified in DEFAULT_CONFIG or CONFIG state only. 1 = CC will enter HALT state 0 = CC will enter/remain in NORMAL_PASSIVE
22	FLXn TSM	Transmission Slot Mode (*pSingleSlotEnabled*) Selects the initial transmission slot mode. In SINGLE slot mode the CC may only transmit in the preconfigured key slot. The key slot ID is configured in the header section of message buffer 0 respectively message buffers 0 and 1 depending on bit FLXnMRC.FLXnSPLM. In case FLXnTSM = '1', message buffer 0 respectively message buffers 0,1 can be (re)configured in DEFAULT_CONFIG or CONFIG state only. In ALL slot mode the CC may transmit in all slots. FLXnTSM is a configuration bit which can only be set / reset by the Host. The bit can be written in DEFAULT_CONFIG or CONFIG state only. The CC changes to ALL slot mode when the Host successfully applied the ALL_SLOTS command by writing FLXnCMD[3:0] = "0101" in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. The actual slot mode is monitored by FLXnCCSV.FLXnSLM[1:0]. 1 = SINGLE Slot Mode (default by hardware reset) 0 = ALL Slot Mode
21	FLXn WUCS	Wakeup Channel Select (*pWakeupChannel*) With this bit the Host selects the channel on which the CC sends the Wakeup pattern. The CC ignores any attempt to change the status of this bit when not in DEFAULT_CONFIG or CONFIG state. 1 = Send wakeup pattern on channel B 0 = Send wakeup pattern on channel A
20 to 16	FLXn PTA[4:0]	Passive to Active (*pAllowPassiveToActive*) Defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the CC is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If set to "00000" the CC is not allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. It can be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 31 even/odd cycle pairs.

Table 30-33 FLXnSUCC1 register contents (3/3)

Bit position	Bit name	Function
15 to 11	FLXn CSA[4:0]	Cold Start Attempts (*gColdStartAttempts*) Configures the maximum number of attempts that a cold starting node is permitted to try to start up the network without receiving any valid response from another node. It can be modified in DEFAULT_CONFIG or CONFIG state only. Must be identical in all nodes of a cluster. Valid values are 2 to 31.
9	FLXn TXSY	Transmit Sync Frame in Key Slot *(pKeySlotUsedForSync)* Defines whether the key slot is used to transmit sync frames. The bit can be modified in DEFAULT_CONFIG or CONFIG state only. 1 = Key slot used to transmit sync frame, node is sync node 0 = No sync frame transmission in key slot, node is neither sync nor coldstart node
8	FLXn TXST	Transmit Startup Frame in Key Slot *(pKeySlotUsedForStartup)* Defines whether the key slot is used to transmit startup frames. The bit can be modified in DEFAULT_CONFIG or CONFIG state only. 1 = Key slot used to transmit startup frame, node is leading or following coldstarter 0 = No startup frame transmission in key slot, node is non-coldstarter
7	FLXn PBSY	POC Busy Signals that the POC is busy and cannot accept a command from the Host. FLXnCMD[3:0] is locked against write accesses. Set to '1' after hard reset during initialization of internal RAM blocks. 1 = POC is busy, FLXnCMD[3:0] locked 0 = POC not busy, FLXnCMD[3:0] writeable
3 to 0	FLXn CMD[3:0]	CHI Command Vector The Host may write any CHI command at any time, but certain commands are enabled only in certain POC states. If a command is not enabled, it will not be executed, the CHI command vector FLXnCMD[3:0] will be reset to "0000" = command_not_accepted, and flag FLXnEIR.FLXnCNAE will be set to '1'. In case the previous CHI command has not yet completed, FLXnEIR.FLXnCCLE is set to '1' together with FLXnEIR.FLXnCNAE; the CHI command needs to be repeated. Except for HALT state, a POC state change command applied while the CC is already in the requested POC state neither causes a state change nor will FLXnEIR.FLXnCNAE be set. For details refer to *Table 30-34 "FLXnSUCC1.FLXnCMD[3:0] function" on page 2413*

Reading FLXnCMD[3:0] shows whether the last CHI command was accepted. The actual POC state is monitored by FLXnCCSV.FLXnPOCS[5:0]. The reserved CHI commands belong to the hardware test functions. In general the Host must check FLXnSUCC1.FLXnPBSY before writing a new CHI command.

Table 30-34 FLXnSUCC1.FLXnCMD[3:0] function (1/2)

FLXn CMD[3:0]	Function
0000	command_not_accepted FLXnCMD[3:0] is reset to "0000" due to one of the following conditions: • Illegal command applied by the Host • Host applied command to leave CONFIG state without preceding config lock key • Host applied new command while execution of the previous Host command has not completed • Host writes command_not_accepted When FLXnCMD[3:0] is reset to "0000", FLXnEIR.FLXnCNAE is set, and - if enabled - an interrupt is generated. Commands which are not accepted are not executed.
0001	CONFIG Go to POC state CONFIG when called in POC states DEFAULT_CONFIG, READY, or in MONITOR_MODE. When called in HALT state the CC transits to POC state DEFAULT_CONFIG. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0010	READY Go to POC state READY when called in POC states CONFIG, NORMAL_ACTIVE, NORMAL_PASSIVE, STARTUP, or WAKEUP. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0011	WAKEUP Go to POC state WAKEUP when called in POC state READY. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0100	RUN Go to POC state STARTUP when called in POC state READY. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0101	ALL_SLOTS Leave single slot mode after successful startup / integration at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0110	HALT Set halt request FLXnCCSV.FLXnHRQ and go to POC state HALT at the next end of cycle when called in POC states NORMAL_ACTIVE or NORMAL_PASSIVE. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
0111	FREEZE Set the freeze status indicator CCSV.FLXnFSI and go to POC state HALT immediately. Can be called from any state.
1000	SEND_MTS Send single MTS symbol during the next following symbol window on the channel configured by FLXnMTSA, FLXnMTSB, when called in POC state NORMAL_ACTIVE after CC entered ALL slot mode (FLXnCCSV.FLXnSLM[1:0] = "11"). When called in any other state, or when called while a previously requested MTS has not yet been transmitted, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
1001	ALLOW_COLDSTART The command resets FLXnCCSV.FLXnCSI to enable the node to become leading coldstarter. When called in states DEFAULT_CONFIG, CONFIG, HALT, or MONITOR_MODE, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted. To become leading coldstarter it is also required that both FLXnTXST and FLXnTXSY are set.

Table 30-34 FLXnSUCC1.FLXnCMD[3:0] function (2/2)

FLXn CMD[3:0]	Function
1010	RESET_STATUS_INDICATORSFLXnCSNI Resets status flags FLXnCCSV.FLXnCSNI, FLXnCCSV.FLXnCSAI, and FLXnCCSV.FLXnWSV[2:0] to their default values. May be called in POC states READY and STARTUP.When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted.
1011	MONITOR_MODE Enter MONITOR_MODE when called in POC state CONFIG. In this mode the CC is able to receive FlexRay frames and wakeup pattern. It is also able to detect coding errors. The temporal integrity of received frames is not checked. This mode can be used for debugging purposes, e.g. in case that the startup of a FlexRay network fails. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted. For details see MONITOR_MODE
1100	CLEAR_RAMS Sets FLXnMHDS.FLXnCRAM when called in DEFAULT_CONFIG or CONFIG state. When called in any other state, FLXnCMD[3:0] will be reset to "0000" = command_not_accepted. FLXnMHDS.FLXnCRAM is also set when the CC leaves hard reset. By setting FLXnMHDS.FLXnCRAM all internal RAM blocks are initialized to zero. During the initialization of the RAMs, FLXnPBSY will show POC busy. Access to the configuration and status registers is possible during execution of CHI command CLEAR_RAMS. The initialization of the E-Ray internal RAM blocks requires 2048 eray_bclk cycles. There should be no Host access to IBF or OBF during initialization of the internal RAM blocks after hard reset or after assertion of CHI command CLEAR_RAMS. Before asserting CHI command CLEAR_RAMS the Host should make sure that no transfer between Message RAM and IBF / OBF or the Transient Buffer RAMs is ongoing. This command also resets the Message Buffer Status registers FLXnMHDS, FLXnLDTS, FLXnFSR, FLXnMHDF, FLXnTXRQ1/2/3/4, FLXnNDAT1/2/3/4, and FLXnMBSC1/2/3/4.

Note All accepted commands with exception of CLEAR_RAMS and SEND_MTS will cause a change of the POC state in the eray_sclk domain after at most 8 cycles of the slower of the two clocks eray_bclk and eray_sclk, counted from the falling edge of the CHI input signal eray_select, assumed that POC was not busy when the command was applied and that no POC state change was forced by bus activity in that time frame. Reading register CCSV will show data that is additionally delayed by synchronization from eray_sclk to eray_bclk domain and by the Host-specific CPU interface. The maximum additional delay is 12 cycles of the slower of the two clocks eray_bclk and eray_sclk.

Table 30-35 "Reference to CHI Host command summary from FlexRay protocol specification" below references the CHI commands from the FlexRay Protocol Specification V2.1 (section 2.2.1.1, Table 2-2) to the E-Ray CHI command vector FLXnCMD[3:0].

Table 30-35 Reference to CHI Host command summary from FlexRay protocol specification (1/2)

CHI command	Where processed (POC States)	CHI command vector FLXnCMD[3:0]
ALL_SLOTS	POC: normal active POC: normal passive	ALL_SLOTS
ALLOW_COLDSTART	All except POC: default config POC: config POC: halt	ALLOW_COLDSTART
CONFIG	POC default config, POC: ready	CONFIG

Table 30-35 Reference to CHI Host command summary from FlexRay protocol specification (2/2)

CHI command	Where processed (POC States)	CHI command vector FLXnCMD[3:0]
CONFIG_COMPLETE	POC: config	Unlock sequence & READY
DEFAULT_CONFIG	POC: halt	CONFIG
FREEZE	All	FREEZE
HALT	POC: normal active, POC: normal passive	HALT
READY	All except POC:default config POC:config, POC:ready POC:halt	READY
RUN	POC: ready	RUN
WAKEUP	POC: ready	WAKEUP

(2) FLXnSUCC2 - SUC configuration register 2

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0084_H

Initial Value $0100\ 0504_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	FLXnLTN[3:0]				0	0	0	FLXnLT[20:16]				
R	R	R	R	R/W	R/W	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnLT[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-36 FLXnSUCC2 register contents

Bit position	Bit name	Function
27 to 24	FLXn LTN[3:0]	Listen Timeout Noise (*gListenNoise* - 1) Configures the upper limit for startup and wakeup listen timeout in the presence of noise expressed as a multiple of pdListenTimeout. The range for gListenNoise is 2 to 16. FLXnLTN[3:0] must be configured identical in all nodes of a cluster.
20 to 0	FLXn LT[20:00]	Listen Timeout (*pdListenTimeout*) Configures wakeup / startup listen timeout in T. The range for pdListenTimeout is 1284 to 1283846 uT.

Note The wakeup / startup noise timeout is calculated as follows:
pdListenTimeout • gListenNoise = FLXnLT[20:00] • (FLXnLTN[3:0] + 1)

(3) FLXnSUCC3 - SUC configuration register 3

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0088_H

Initial Value $0000\ 0011_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	FLXn WCF[3:0]*				FLXn WCP[3:0]*			
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-37 FLXnSUCC3 register contents

Bit position	Bit name	Function
7 to 4	FLXn WCF[3:0]	Maximum Without Clock Correction Fatal (*gMaxWithoutClockCorrectionFatal*) Defines the number of consecutive even/odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE or NORMAL_PASSIVE to HALT state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 cycle pairs.
3 to 0	FLXn WCP[3:0]	Maximum Without Clock Correction Passive (*gMaxWithoutClockCorrectionPassive*) Defines the number of consecutive even/odd cycle pairs with missing clock correction terms that will cause a transition from NORMAL_ACTIVE to NORMAL_PASSIVE state. Must be identical in all nodes of a cluster. Valid values are 1 to 15 cycle pairs.

Note The transition to HALT state is prevented if FLXnSUCC1.FLXnHCSE is not set.

(4) FLXnNEMC - NEM configuration register

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $008C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	FLXnNML[3:0]*			
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

Table 30-38 FLXnNEMC register contents

Bit position	Bit name	Function
3 to 0	FLXn NML[3:0]	Network Management Vector Length (*gNetworkManagementVectorLength*) These bits configure the length of the NM vector. The configured length must be identical in all nodes of a cluster. Valid values are 0 to 12 bytes.

(5) FLXnPRTC1 - PRT configuration register 1

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0090_H

Initial Value $084C\ 0633_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnRWP[5:0]*						0	FLXnRXW[8:0]*								
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnBRP[1:0]*		FLXnSPP[1:0]*		0	FLXnCASM[6:0]*							FLXnTSST[3:0]*			
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-39 FLXnPRTC1 register contents (1/2)

Bit position	Bit name	Function
31 to 26	FLXn RWP[5:0]	Repetitions of Tx Wakeup Pattern (*pWakeupPattern*) Configures the number of repetitions (sequences) of the Tx wakeup symbol. Valid values are 2 to 63.
24 to 16	FLXn RXW[8:0]	Wakeup Symbol Receive Window Length (*gdWakeupSymbolRxWindow*) Configures the number of bit times used by the node to test the duration of the received wakeup pattern. Must be identical in all nodes of a cluster. Valid values are 76 to 301 bit times.
15 to 14	FLXn BRP[1:0]	Baud Rate Prescaler (*gdSampleClockPeriod*, *pSamplePerMicrotick*) The Baud Rate Prescaler configures the baud rate on the FlexRay bus. The baud rates listed below are valid with a sample clock eray_sclk = 80 MHz. One bit time always consists of 8 samples independent of the configured baud rate. 00 = 10 MBit/s (default) gdSampleClockPeriod = 12.5 ns = 1 • eray_sclk pSamplesPerMicrotick = 2 (1 T = 25 ns) 01 = 5 MBit/s gdSampleClockPeriod = 25 ns = 2 eray_sclk pSamplesPerMicrotick = 1 (1 T = 25 ns) 10, 11 = 2.5 MBit/s gdSampleClockPeriod = 50 ns = 4 eray_sclk pSamplesPerMicrotick = 1 (1 T = 50 ns)

Table 30-39 FLXnPRTC1 register contents (2/2)

Bit position	Bit name	Function
13 to 12	FLXn SPP[1:0]	Strobe Point Position Defines the sample count value for strobing. The strobed bit value is set to the voted value when the sample count is incremented to the value configured by FLXnSPP[1:0]. 00, 11= Sample 5 (default) 01 = Sample 4 10 = Sample 6 **Note:** The current revision 2.1 of the FlexRay protocol requires that FLXnSPP[1:0] = "00". The alternate strobe point positions could be used to compensate for asymmetries in the physical layer.
10 to 4	FLXn CASM[6:0]	Collision Avoidance Symbol Max (*gdCASRxLowMax*) Configures the upper limit of the acceptance window for a collision avoidance symbol (CAS). FLXnCASM06 is fixed to '1'. Valid values are 67 to 99 bit times.
3 to 0	FLXn TSST[3:0]	Transmission Start Sequence Transmitter (*gdTSSTransmitter*) Configures the duration of the Transmission Start Sequence (TSS) in terms of bit times (1 bit time = 4 T = 100ns@10Mbps). Must be identical in all nodes of a cluster. Valid values are 3 to 15 bit times.

(6) FLXnPRTC2 - PRT configuration register 2

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0094_H

Initial Value $0F2D\ 0A0E_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnTXL[5:0]*						FLXnTXI[7:0]*							
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnRXL[5:0]*						0	0	FLXnRXI[5:0]*					
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-40 FLXnPRTC2 register contents

Bit position	Bit name	Function
29 to 24	FLXn TXL[5:0]	Wakeup Symbol Transmit Low (*gdWakeupSymbolTxLow*) Configures the number of bit times used by the node to transmit the low phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 15 to 60 bit times.
23 to 16	FLXn TXI[7:0]	Wakeup Symbol Transmit Idle (*gdWakeupSymbolTxIdle*) Configures the number of bit times used by the node to transmit the idle phase of the wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 45 to 180 bit times.
13 to 8	FLXn RXL[5:0]	Wakeup Symbol Receive Low (*gdWakeupSymbolRxLow*) Configures the number of bit times used by the node to test the duration of the low phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 10 to 55 bit times.
5 to 0	FLXn RXI[5:0]	Wakeup Symbol Receive Idle (*gdWakeupSymbolRxIdle*) Configures the number of bit times used by the node to test the duration of the idle phase of the received wakeup symbol. Must be identical in all nodes of a cluster. Valid values are 14 to 59 bit times.

(7) FLXnMHDC - MHD configuration register

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0098_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	FLXnSLT[12:00]*												
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	FLXnSFDL[6:0]*						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-41 FLXnMHDC register contents

Bit position	Bit name	Function
28 to 16	FLXn SLT[12:00]	Start of Latest Transmit (*pLatestTx*) Configures the maximum minislot value allowed before inhibiting frame transmission in the dynamic segment of the cycle. There is no transmission in dynamic segment if FLXnSLT[12:00] is set to zero. Valid values are 0 to 7981 minislots.
6 to 0	FLXn SFDL[6:0]	Static Frame Data Length (*gPayloadLengthStatic*) Configures the cluster-wide frame length for all frames sent in the static segment in double bytes. The frame length must be identical in all nodes of a cluster. Valid values are 0 to 127.

(8) FLXnGTUC01 - GTU configuration register 1

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00A0_H$

Initial Value $0000\ 0280_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	FLXnUT[19:16]*			
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnUT[15:00]*															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-42 FLXnGTUC01 register contents

Bit position	Bit name	Function
19 to 0	FLXn UT[19:00]	Microtick per Cycle (*pMicroPerCycle*) Configures the duration of the communication cycle in microticks. Valid values are 640 to 640000 µT.

(9) FLXnGTUC02 - GTU configuration register 2

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00A4_H$

Initial Value $0002\ 000A_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	FLXnSNM[3:0]*			
R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnMPC[13:00]*													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-43 FLXnGTUC02 register contents

Bit position	Bit name	Function
19 to 16	FLXn SNM[3:0]	Sync Node Max (*gSyncNodeMax*) Maximum number of frames within a cluster with sync frame indicator bit FLXnRHSYN set to '1'. Must be identical in all nodes of a cluster. Valid values are 2 to 15.
13 to 0	FLXn MPC[13:00]	Macrotick Per Cycle (*gMacroPerCycle*) Configures the duration of one communication cycle in macroticks. The cycle length must be identical in all nodes of a cluster. Valid values are 10 to 16000 MT.

(10) FLXnGTUC03 - GTU configuration register 3

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00A8_H$

Initial Value $0202\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	FLXnMIOB[6:0]*							0	FLXnMIOA[6:0]*						
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnUIOB[7:0]*								FLXnUIOA[7:0]*							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-44 FLXnGTUC03 register contents

Bit position	Bit name	Function
30 to 24	FLXn MIOB[6:0]	Macrotick Initial Offset Channel B *(pMacroInitialOffset[B])* Configures the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 MT.
22 to 16	FLXn MIOA[6:0]	Macrotick Initial Offset Channel A *(pMacroInitialOffset[A])* Configures the number of macroticks between the static slot boundary and the subsequent macrotick boundary of the secondary time reference point based on the nominal macrotick duration. Must be identical in all nodes of a cluster. Valid values are 2 to 72 MT.
15 to 8	FLXn UIOB[7:0]	Microtick Initial Offset Channel B (*pMicroInitialOffset[B]*) Configures the number of microticks which describe the distance between the macrotick boundary described by *gMacroInitialOffset* and the exact secondary time reference point. The parameter depends on *pDelayCompensation[B]* and therefore has to be set for each channel independently. Valid values are 0 to 240 µT.
7 to 0	FLXn UIOA[7:0]	Microtick Initial Offset Channel A (*pMicroInitialOffset[A]*) Configures the number of microticks which describe the distance between the macrotick boundary described by *gMacroInitialOffset* and the exact secondary time reference point. The parameter depends on *pDelayCompensation[A]* and therefore has to be set for each channel independently. Valid values are 0 to 240 µT.

(11) FLXnGTUC04 - GTU configuration register 4

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only. For details about configuration of FLXnNIT[13:00] and FLXnOCS[13:00] see Section Configuration of NIT Start and Offset Correction Start.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00AC_H$

Initial Value $0008\ 0007_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnOCS[13:00]													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnNIT[13:00]													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-45 FLXnGTUC04 register contents

Bit position	Bit name	Function
29 to 16	FLXn NIT[13:00]	Network Idle Time Start (*gMacroPerCycle - gdNIT - 1*) Configures the starting point of the Network Idle Time NIT at the end of the communication cycle expressed in terms of macroticks from the beginning of the cycle. The start of NIT is recognized if Macrotick = gMacroPerCycle - gdNIT -1 and the increment pulse of Macrotick is set. Must be identical in all nodes of a cluster. Valid values are 7 to 15997 MT.
13 to 0	FLXn OCS[13:00]	Offset Correction Start (*gOffsetCorrectionStart - 1*) Determines the start of the offset correction within the NIT phase, calculated from start of cycle. Must be identical in all nodes of a cluster. For cluster consisting of E-Ray implementations only, it is sufficient to program FLXnOCS = FLXnNIT + 1. Valid values are 8 to 15998 MT.

(12) FLXnGTUC05 - GTU configuration register 5

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00B0_H$

Initial Value $00E00\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnDEC[7:0]*								0	0	0	FLXnCDD[4:0]*				
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnDCB[7:0]*								FLXnDCA[7:0]*							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-46 FLXnGTUC05 register contents

Bit position	Bit name	Function
31 to 24	FLXn DEC[7:0]	Decoding Correction (*pDecodingCorrection*) Configures the decoding correction value used to determine the primary time reference point. Valid values are 14 to 143 T.
20 to 16	FLXn CDD[4:0]	Cluster Drift Damping (*pClusterDriftDamping*) Configures the cluster drift damping value used in clock synchronization to minimize accumulation of rounding errors. Valid values are 0 to 20 T.
15 to 8	FLXn DCB[7:0]	Delay Compensation Channel B (*pDelayCompensation[B]*) Used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 to 0.05s. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 T.
7 to 0	FLXn DCA[7:0]	Delay Compensation Channel A (*pDelayCompensation[A]*) Used to compensate for reception delays on the indicated channel. This covers assumed propagation delay up to cPropagationDelayMax for microticks in the range of 0.0125 to 0.05s. In practice, the minimum of the propagation delays of all sync nodes should be applied. Valid values are 0 to 200 T.

(13) FLXnGTUC06 - GTU configuration register 6

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00B4_H$

Initial Value $0002\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnMOD[10:00]*										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnASR[10:00]*										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-47 FLXnGTUC06 register contents

Bit position	Bit name	Function
10 to 0	FLXnMOD [10:00]	Maximum Oscillator Drift (*pdMaxDrift*) Maximum drift offset between two nodes that operate with unsynchronized clocks over one communication cycle in µT. Valid values are 2 to 1923 µT.
26 to 16	FLXnASR [10:00]	Accepted Startup Range (*pdAcceptedStartupRange*) Number of microticks constituting the expanded range of measured deviation for startup frames during integration. Valid values are 0 to 1875 T.

(14) FLXnGTUC07 - GTU configuration register 7

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00B8_H$

Initial Value $0002\ 0004_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	FLXnNSS[9:0]*									
R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	FLXnSSL[9:0]*									
R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-48 FLXnGTUC07 register contents

Bit position	Bit name	Function
25 to 16	FLXn NSS[9:0]	Number of Static Slots (*gNumberOfStaticSlots*) Configures the number of static slots in a cycle. At least 2 coldstart nodes must be configured to startup a FlexRay network. The number of static slots must be identical in all nodes of a cluster. Valid values are 2 to 1023.
9 to 0	FLXn SSL[9:0]	Static Slot Length (*gdStaticSlot*) Configures the duration of a static slot in macroticks. The static slot length must be identical in all nodes of a cluster. Valid values are 4 to 659 MT.

(15) FLXnGTUC08 - GTU configuration register 8

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00BC_H$

Initial Value $0000\ 0002_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	FLXnNMS[12:00]*												
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	FLXnMSL[5:0]*					
R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-49 FLXnGTUC08 register contents

Bit position	Bit name	Function
28 to 16	FLXn NMS[12:0]	Number of Minislots (*gNumberOfMinislots*) Configures the number of minislots within the dynamic segment of a cycle. The number of minislots must be identical in all nodes of a cluster. Valid values are 0 to 7986.
5 to 0	FLXn MSL[5:0]	Minislot Length (*gdMinislot*) Configures the duration of a minislot in macroticks. The minislot length must be identical in all nodes of a cluster. Valid values are 2 to 63 MT.

(16) FLXnGTUC09 - GTU configuration register 9

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00C0_H$

Initial Value $0000\ 0101_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn DSI[1:0]*	
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	FLXnMAPO[4:0]*					0	0	FLXnAPO[5:0]*					
R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-50 FLXnGTUC09 register contents

Bit position	Bit name	Function
17 to 16	FLXn DSI[1:0]	Dynamic Slot Idle Phase (*gdDynamicSlotIdlePhase*) The duration of the dynamic slot idle phase has to be greater or equal than the idle detection time. Must be identical in all nodes of a cluster. Valid values are 0 to 2 Minislot.
12 to 8	FLXn MAPO[4:0]	Minislot Action Point Offset (*gdMinislotActionPointOffset*) Configures the action point offset in macroticks. Must be identical in all nodes of a cluster. Valid values are 1 to 31 MT.
5 to 0	FLXn APO[5:0]	Action Point Offset (*gdActionPointOffset*) Configures the action point offset in macroticks within static slots and symbol window. Must be identical in all nodes of a cluster. Valid values are 1 to 63 MT.

(17) FLXnGTUC10 - GTU configuration register 10

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00C4_H$

Initial Value $0002\ 0005_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnMRC[10:00]*										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnMOC[13:00]*													
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-51 FLXnGTUC10 register contents

Bit position	Bit name	Function
26 to 16	FLXn MRC[10:00]	Maximum Rate Correction (*pRateCorrectionOut*) Holds the maximum permitted rate correction value to be applied by the internal clock synchronization algorithm. The CC checks only the internal rate correction value against the maximum rate correction value (absolute value). Valid values are 2 to 1923 T.
1 to 0	FLXn MOC[13:00]	Maximum Offset Correction (*pOffsetCorrectionOut*) Holds the maximum permitted offset correction value to be applied by the internal clock synchronization algorithm (absolute value). The CC checks only the internal offset correction value against the maximum offset correction value. Valid values are 5 to 15266 µT.

(18) FLXnGTUC01 - GTU configuration register 11

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $00C8_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnERC[2:0]*			0	0	0	0	0	FLXnEOC[2:0]*		
R	R	R	R	R	R/W	R/W	R/W	R	R	R	R	R	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	FLXn ERCC[1:0]		0	0	0	0	0	0	FLXn EOCC[1:0]	
R	R	R	R	R	R	R/W	R/W	R	R	R	R	R	R	R/W	R/W

Table 30-52 FLXnGTUC01 register contents

Bit position	Bit name	Function
26 to 24	FLXn ERC[2:0]	External Rate Correction (*pExternRateCorrection*) Holds the external clock rate correction value in microticks to be applied by the internal clock synchronization algorithm. The value is subtracted/added from/to the calculated rate correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 7 µT.
18 to 16	FLXn EOC[2:0]	External Offset Correction (*pExternOffsetCorrection*) Holds the external clock offset correction value in microticks to be applied by the internal clock synchronization algorithm. The value is subtracted/added from/to the calculated offset correction value. The value is applied during NIT. May be modified in DEFAULT_CONFIG or CONFIG state only. Valid values are 0 to 7 µT.
9 to 8	FLXn ERCC[1:0]	External Rate Correction Control *(vExternRateControl)* By writing to FLXnERCC[1:0] the external rate correction is enabled as specified below. Should be modified only outside NIT. 00, 01 = No external rate correction 10 = External rate correction value subtracted from calculated rate correction value 11 = External rate correction value added to calculated rate correction value
1 to 0	FLXn EOCC[1:0]	External Offset Correction Control *(vExternOffsetControl)* By writing to FLXnEOCC[1:0] the external offset correction is enabled as specified below. Should be modified only outside NIT. 00, 01 = No external offset correction 10 = External offset correction value subtracted from calculated offset correction value 11 = External offset correction value added to calculated offset correction value

30.3.6 CC status registers

The status vector may change faster than the Host can poll the status vector,depending on eray_bclk frequency.

(1) FLXnCCSV - CC status vector

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0100_H

Initial Value $0010\ 4000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnPSL[5:0]						FLXnRCA[4:0]					FLXnWSV[2:0]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	FLXn CSI	FLXn CSAI	FLXn CSNI	0	0	FLXn SLM[1:0]		FLXn HRQ	FLXn FSI	FLXnPOCS[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-53 FLXnCCSV register contents (1/3)

Bit position	Bit name	Function
29 to 24	FLXn PSL[5:0]	POC Status Log Status of FLXnPOCS[5:0] immediately before entering HALT state. Set to HALT when FREEZE command is applied during HALT state and FLXnFSI is not already set i.e. the HALT state was not reached by FREEZE command. Reset to "00 0000" when leaving HALT state.
23 to 19	FLXn RCA[4:0]	Indicates the number of remaining coldstart attempts. The RUN command resets this counter to the maximum number of coldstart attempts as configured by FLXnSUCC1.FLXnCSA[4:0]. The initial value of FLXnRCA[4:0] during CONFIG and DEFAULT_CONFIG state is also FLXnSUCC1.FLXnCSA[4:0].
18 to 16	FLXn WSV[2:0]	Wakeup Status (*vPOC!WakeupStatus*) Indicates the status of the current wakeup attempt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state. 000 = UNDEFINED. Wakeup not yet executed by the CC. 001 = RECEIVED_HEADER. Set when the CC finishes wakeup due to the reception of a frame header without coding violation on either channel in WAKEUP_LISTEN state. 010 = RECEIVED_WUP. Set when the CC finishes wakeup due to the reception of a valid wakeup pattern on the configured wakeup channel in WAKEUP_LISTEN state. 011 = COLLISION_HEADER. Set when the CC stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid header on either channel. 100 = COLLISION_WUP. Set when the CC stops wakeup due to a detected collision during wakeup pattern transmission by receiving a valid wakeup pattern on the configured wakeup channel. 101 = COLLISION_UNKNOWN. Set when the CC stops wakeup by leaving WAKEUP_DETECT state after expiration of the wakeup timer without receiving a valid wakeup pattern or a valid frame header. 110 = TRANSMITTED. Set when the CC has successfully completed the transmission of the wakeup pattern. 111 = reserved

Table 30-53 FLXnCCSV register contents (2/3)

Bit position	Bit name	Function
14	FLXn CSI	Cold Start Inhibit (*vColdStartInhibit*) Indicates that the node is disabled from cold starting. The flag is set whenever the POC enters READY state due to CHI command READY. The flag has to be reset under control of the Host by CHI command ALLOW_COLDSTART (FLXnCMD[3:0] = "1001"). 1 = Cold starting of node disabled 0 = Cold starting of node enabled
13	FLXn CSAI	Coldstart Abort Indicator Coldstart aborted. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY state to STARTUP state.
12	FLXn CSNI	Coldstart Noise Indicator (*vColdStartNoise*) Indicates that the cold start procedure occurred under noisy conditions. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or from READY to STARTUP state.
9 to 8	FLXn SLM[1:0]	Slot Mode (*vPOC!SlotMode*) Indicates the actual slot mode of the POC in states READY, WAKEUP, STARTUP, NORMAL_ACTIVE, and NORMAL_PASSIVE. Default is SINGLE. Changes to ALL, depending on FLXnSUCC1.FLXnTSM. In NORMAL_ACTIVE or NORMAL_PASSIVE state the CHI command ALL_SLOTS will change the slot mode from SINGLE over ALL_PENDING to ALL. Set to SINGLE in all other states. 00 = SINGLE 01 = reserved 10 = ALL_PENDING 11 = ALL

Table 30-53 FLXnCCSV register contents (3/3)

Bit position	Bit name	Function
7	FLXn HRQ	Halt Request (*vPOC!CHIHaltRequest*) Indicates that a request from the Host has been received to halt the POC at the end of the communication cycle. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.
6	FLXn FSI	Freeze Status Indicator (*vPOC!Freeze*) Indicates that the POC has entered the HALT state due to CHI command FREEZE or due to an error condition requiring an immediate POC halt. Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state.
5 to 0	FLXn POCS[5:0]	Protocol Operation Control Status Indicates the actual state of operation of the CC Protocol Operation Control 00 0000 = DEFAULT_CONFIG state 00 0001 = READY state 00 0010 = NORMAL_ACTIVE state 00 0011 = NORMAL_PASSIVE state 00 0100 = HALT state 00 0101 = MONITOR_MODE state 00 0110…00 1110 = reserved 00 1111 =CONFIG state Indicates the actual state of operation of the POC in the wakeup path 01 0000 = WAKEUP_STANDBY state 01 0001 = WAKEUP_LISTEN state 01 0010 = WAKEUP_SEND state 01 0011 = WAKEUP_DETECT state 01 0100…01 1111 = reserved Indicates the actual state of operation of the POC in the startup path 10 0000 = STARTUP_PREPARE state 10 0001 = COLDSTART_LISTEN state 10 0010 = COLDSTART_COLLISION_RESOLUTION state 10 0011 = COLDSTART_CONSISTENCY_CHECK state 10 0100 = COLDSTART_GAP state 10 0101 = COLDSTART_JOIN State 10 0110 = INTEGRATION_COLDSTART_CHECK state 10 0111 = INTEGRATION_LISTEN state 10 1000 = INTEGRATION_CONSISTENCY_CHECK state 10 1001 = INITIALIZE_SCHEDULE state 10 1010 = ABORT_STARTUP state 10 1011 = STARTUP_SUCCESS state 10 1100…11 1111 = reserved

(2) FLXnCCEV - CC error vector

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0104_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	FLXn PTAC[4:0]					FLXn ERRM[1:0]		0	0	FLXn CCFC[3:0]			
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Reset by CHI command RESET_STATUS_INDICATORS or by transition from HALT to DEFAULT_CONFIG state or when entering READY state.

Table 30-54 FLXnCCEV register contents

Bit position	Bit name	Function
12 to 8	FLXn PTAC[4:0]	Passive to Active Count (*vAllowPassiveToActive*) Indicates the number of consecutive even / odd cycle pairs that have passed with valid rate and offset correction terms, while the node is waiting to transit from NORMAL_PASSIVE state to NORMAL_ACTIVE state. The transition takes place when FLXnPTAC[4:0] equals FLXnSUCC1.FLXnPTA[4:0] -1.
7 to 6	FLXn ERRM[1:0]	Error Mode (*vPOC!ErrorMode*) Indicates the actual error mode of the POC. 00 = ACTIVE (green) 01 = PASSIVE (yellow) 10 = COMM_HALT (red) 11 = reserved
3 to 0	FLXn CCFC[3:0]	Clock Correction Failed Counter (*vClockCorrectionFailed*) The Clock Correction Failed Counter is incremented by one at the end of any odd communication cycle where either the Missing Offset Correction error or Missing Rate Correction error are active. The Clock Correction Failed Counter is reset to '0' at the end of an odd communication cycle if neither the Offset Correction Failed nor the Rate Correction Failed errors are active. The Clock Correction Failed Counter stops at 15.

(3) FLXnSCV - Slot counter value

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0110_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnSCCB[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnSCCA[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-55 FLXnSCV register contents

Bit position	Bit name	Function
26 to 16	FLXn SCCB[10:00]	Slot Counter Channel B (*vSlotCounter[B]*) Current slot counter value on channel B. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 2047.
10 to 0	FLXn SCCA[10:00]	Slot Counter Channel A (*vSlotCounter[A]*) Current slot counter value on channel A. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 2047.

(4) FLXnMTCCV - Macrotick and cycle counter value

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0114_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	FLXnCCV[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	FLXnMTV[13:00]													
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-56 FLXnMTCCV register contents

Bit position	Bit name	Function
21 to 16	FLXn CCV[5:0]	Cycle Counter Value (*vCycleCounter*) Current cycle counter value. The value is incremented by the CC at the start of a communication cycle. Valid values are 0 to 63.
13 to 0	FLXn MTV[13:0]	Macrotick Value (*vMacrotick*) Current Macrotick value. The value is incremented by the CC and reset at the start of a communication cycle. Valid values are 0 to 15999.

(5) FLXnRCV - Rate correction value

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0118_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	FLXnRCV[11:00]											
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-57 FLXnRCV register contents

Bit position	Bit name	Function
11 to 0	FLXn RCV[11:00]	Rate Correction Value (*vRateCorrection*) Rate correction value (two's complement). Calculated internal rate correction value before limitation. If the FLXnRCV value exceeds the limits defined by FLXnGTUC010.FLXnMRC[10:00], flag FLXnSFS.FLXnRCLR is set to '1'.

(6) FLXnOCV - Offset correction value

Access This register can be read in 32-bit units.

Address <FLXn_base> + $011C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	FLXnOCV[18:16]		
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnOCV[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-58 FLXnOCV register contents

Bit position	Bit name	Function
18 to 0	FLXn OCV[18:00]	Offset Correction Value *(vOffsetCorrection)* Offset correction value (two's complement). Calculated internal offset correction value before limitation. If the FLXnOCV value exceeds the limits defined by FLXnGTUC010.FLXnMOC[13:00], flag FLXnSFS.FLXnOCLR is set to '1'.

Note The external rate / offset correction value is added to the limited rate / offset correction value.

(7) FLXnSFS - Sync frame status

The maximum number of valid sync frames in a communication cycle is 15.

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0120_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	FLXn RCLR	FLXn MRCS	FLXn OCLR	FLXn MOCS
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnVSBO[3:0]				FLXnVSBE[3:0]				FLXnVSAO[3:0]				FLXnVSAE[3:0]			
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-59 FLXnSFS register contents (1/2)

Bit position	Bit name	Function
19	FLXn RCLR	Rate Correction Limit Reached The Rate Correction Limit Reached flag signals to the Host, that the rate correction value has exceeded its limit as defined by FLXnGTUC10.FLXnMRC[10:00]. The flag is updated by the CC at start of offset correction phase. 1 = Rate correction limit reached 0 = Rate correction below limit
18	FLXn MRCS	Missing Rate Correction Signal The Missing Rate Correction flag signals to the Host, that no rate correction calculation can be performed because no pairs of even / odd sync frames were received. The flag is updated by the CC at start of offset correction phase. 1 = Missing rate correction signal 0 = Rate correction signal valid
17	FLXn OCLR	Offset Correction Limit Reached The Offset Correction Limit Reached flag signals to the Host, that the offset correction value has exceeded its limit as defined by FLXnGTUC10.FLXnMOC[13:00]. The flag is updated by the CC at start of offset correction phase. 1 = Offset correction limit reached 0 = Offset correction below limit
16	FLXn MOCS	Missing Offset Correction Signal The Missing Offset Correction flag signals to the Host, that no offset correction calculation can be performed because no sync frames were received. The flag is updated by the CC at start of offset correction phase. 1 = Missing offset correction signal 0 = Offset correction signal valid
15 to 12	FLXn VSBO[3:0]	Valid Sync Frames Channel B, *odd* communication cycle Holds the number of valid sync frames received on channel B in the odd communication cycle. If transmission of sync frames is enabled by FLXnSUCC1.FLXnTXSY the value is incremented by one. The value is updated during the NIT of each odd communication cycle.

Table 30-59 FLXnSFS register contents (2/2)

Bit position	Bit name	Function
11 to 8	FLXn VSBE[3:0]	Valid Sync Frames Channel B, *even* communication cycle Holds the number of valid sync frames received on channel B in the even communication cycle. If transmission of sync frames is enabled by FLXnSUCC1.FLXnTXSY the value is incremented by one. The value is updated during the NIT of each even communication cycle.
7 to 4	FLXn VSAO[3:0]	Valid Sync Frames Channel A, *odd* communication cycle Holds the number of valid sync frames received on channel A in the odd communication cycle. If transmission of sync frames is enabled by FLXnSUCC1.FLXnTXSY the value is incremented by one. The value is updated during the NIT of each odd communication cycle.
3 to 0	FLXn VSAE[3:0]	Valid Sync Frames Channel A, *even* communication cycle Holds the number of valid sync frames received on channel A in the even communication cycle. If transmission of sync frames is enabled by FLXnSUCC1.FLXnTXSY the value is incremented by one. The value is updated during the NIT of each even communication cycle.

Note Bits FLXnVSAE[3:0], FLXnVSAO[3:0], FLXnVSBE[3:0], and FLXnVSBO[3:0] are only valid if the respective channel is assigned to the CC by FLXnSUCC1.FLXnCCHA or FLXnSUCC1.FLXnCCHB.

(8) FLXnSWNIT - Symbol window and NIT status

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0124_H

Initial Value $000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	FLXn SWNIT SBNB	FLXn SWNIT SENB	FLXn SWNIT SBNA	FLXn SWNIT SENA	FLXn SWNIT MTSB	FLXn SWNIT MTSA	FLXn SWNIT TCSB	FLXn SWNIT SBSB	FLXn SWNIT SESB	FLXn SWNIT TCSA	FLXn SWNIT SBSA	**FLXn SWNIT SESA**
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Symbol window related status information. Updated by the CC at the end of the symbol window for each channel. During startup the status data is not updated. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Table 30-60 FLXnSWNIT register contents (1/2)

Bit position	Bit name	Function
11	FLXn SWNIT SBNB	Slot Boundary Violation during NIT Channel B (*vSS!BViolationB*) 1 = Slot boundary violation during NIT detected on channel B 0 = No slot boundary violation detected
10	FLXn SWNIT SENB	Syntax Error during NIT Channel B (*vSS!SyntaxErrorB*) 1 = Syntax error during NIT detected on channel B 0 = No syntax error detected
9	FLXn SWNIT SBNA	Slot Boundary Violation during NIT Channel A (*vSS!BViolationA*) 1 = Slot boundary violation during NIT detected on channel A 0 = No slot boundary violation detected
8	FLXn SWNIT SENA	Syntax Error during NIT Channel A (*vSS!SyntaxErrorA*) 1 = Syntax error during NIT detected on channel A 0 = No syntax error detected
7	FLXn SWNIT MTSB	MTS Received on Channel B *(vSS!ValidMTSB)* Media Access Test symbol received on channel B during the preceding symbol window. Updated by the CC for each channel at the end of the symbol window. When this bit is set to '1', also interrupt flag FLXnSIR.FLXnMTSBF is set to '1'. 1 = MTS symbol received on channel B 0 = No MTS symbol received on channel B
6	FLXn SWNIT MTSA	MTS Received on Channel A *(vSS!ValidMTSA)* Media Access Test symbol received on channel A during the preceding symbol window. Updated by the CC for each channel at the end of the symbol window. When this bit is set to '1', also interrupt flag FLXnSIR.FLXnMTSAF is set to '1'. 1 = MTS symbol received on channel A 0 = No MTS symbol received on channel A
5	FLXn SWNIT TCSB	Transmission Conflict in Symbol Window Channel B (*vSS!TxConflictB*) 1 = Transmission conflict in symbol window detected on channel B 0 = No transmission conflict detected
4	FLXn SWNIT SBSB	Slot Boundary Violation in Symbol Window Channel B (*vSS!BViolationB*) 1 = Slot boundary violation during symbol window detected on channel B 0 = No slot boundary violation detected

Table 30-60 FLXnSWNIT register contents (2/2)

Bit position	Bit name	Function
3	FLXn SWNIT SESB	Syntax Error in Symbol Window Channel B (*vSS!SyntaxErrorB*) 1 = Syntax error during symbol window detected on channel B 0 = No syntax error detected
2	FLXn SWNIT TCSA	Transmission Conflict in Symbol Window Channel A (*vSS!TxConflictA*) 1 = Transmission conflict in symbol window detected on channel A 0 = No transmission conflict detected
1	FLXn SWNIT SBSA	Slot Boundary Violation in Symbol Window Channel A (*vSS!BViolationA*) 1 = Slot boundary violation during symbol window detected on channel A 0 = No slot boundary violation detected
0	FLXn SWNIT SESA	Syntax Error in Symbol Window Channel A (*vSS!SyntaxErrorA*) 1 = Syntax error during symbol window detected on channel A 0 = No syntax error detected

Note Bits FLXnSWNITSENA, FLXnSWNITSBNA, FLXnSWNITSENB, and FLXnSWNITSBNB are NIT related status information. Updated by the CC at the end of the NIT for each channel.

(9) FLXnACS - Aggregated channel status

The aggregated channel status provides the Host with an accrued status of channel activity for all communication slots regardless of whether they are assigned for transmission or subscribed for reception. The aggregated channel status also includes status data from the symbol window and the network idle time. The status data is updated (set) after each slot and aggregated until it is reset by the Host. During startup the status data is not updated. A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect on the flag. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0128_H

Initial Value $0000\ 0000_H$

(a) FLXnACS read

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
0	0	0	FLXn SBVBE	FLXn CIBE	FLXn CEDBE	FLXn SEDBE	FLXn VFRBF
7	6	5	4	3	2	1	0
0	0	0	FLXn SBVAE	FLXn CIAE	FLXn CEDAE	FLXn SEDAE	FLXn VFRAF

Table 30-61 FLXnACS read register contents (1/2)

Bit position	Bit name	Function
12	FLXn SBVBE	Slot Boundary Violation on Channel B (*vSS!BViolationB*) One or more slot boundary violations were observed on channel B at any time during the observation period (static or dynamic slots including symbol window and NIT). 1 = Slot boundary violation(s) observed on channel B 0 = No slot boundary violation observed
11	FLXn CIBE	Communication Indicator Channel B One or more valid frames were received on channel B in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation. 1 = Valid frame(s) received on channel B in slots containing any additional communication 0 = No valid frame(s) received in slots containing any additional communication
10	FLXn CEDBE	Content Error Detected on Channel B (*vSS!ContentErrorB*) One or more frames with a content error were received on channel B in any static or dynamic slot during the observation period. 1 = Frame(s) with content error received on channel B 0 = No frame with content error received

Table 30-61 FLXnACS read register contents (2/2)

Bit position	Bit name	Function
9	FLXn SEDBE	Syntax Error Detected on Channel B (*vSS!SyntaxErrorB*) One or more syntax errors in static or dynamic slots including symbol window and NIT were observed on channel B. 1 = Syntax error(s) observed on channel B 0 = No syntax error observed
8	FLXn VFRBF	Valid Frame Received on Channel B (*vSS!ValidFrameB*) One or more valid frames were received on channel B in any static or dynamic slot during the observation period. Reset under control of the Host. 1 = Valid frame(s) received on channel B 0 = No valid frame received
4	FLXn SBVAE	Slot Boundary Violation on Channel A (*vSS!BViolationA*) One or more slot boundary violations were observed on channel A at any time during the observation period (static or dynamic slots including symbol window and NIT). 1 = Slot boundary violation(s) observed on channel A 0 = No slot boundary violation observed
3	FLXn CIAE	Communication Indicator Channel A One or more valid frames were received on channel A in slots that also contained any additional communication during the observation period, i.e. one or more slots received a valid frame AND had any combination of either syntax error OR content error OR slot boundary violation. 1 = Valid frame(s) received on channel A in slots containing any additional communication 0 = No valid frame(s) received in slots containing any additional communication
2	FLXn CEDAE	Content Error Detected on Channel A (*vSS!ContentErrorA*) One or more frames with a content error were received on channel A in any static or dynamic slot during the observation period. 1 = Frame(s) with content error received on channel A 0 = No frame with content error received
1	FLXn SEDAE	Syntax Error Detected on Channel A (*vSS!SyntaxErrorA*) One or more syntax errors in static or dynamic slots including symbol window and NIT were observed on channel A. 1 = Syntax error(s) observed on channel A 0 = No syntax error observed
0	FLXn VFRAF	Valid Frame Received on Channel A (*vSS!ValidFrameA*) One or more valid frames were received on channel A in any static or dynamic slot during the observation period. 1 = Valid frame(s) received on channel A 0 = No valid frame received

Note The set condition of flags FLXnCIAE and FLXnCIBE is also fulfilled if there is only one single frame in the slot and the slot boundary at the end of the slot is reached during the frames channel idle recognition phase.

When one of the flags FLXnSEDBE, FLXnCEDBE, FLXnCIBE, FLXnSBVBE changes from ‘0‘ to ‘1‘, interrupt flag FLXnEIR.FLXnEDBE is set to ’1’.When one of the flags FLXnSEDAE, FLXnCEDAE, FLXnCIAE, FLXnSBVAE changes from ‘0‘ to ‘1‘, interrupt flag FLXnEIR.FLXnEDAE is set to ’1’.

(b) FLXnACS write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	FLXnCL SBVB	FLXnCL CIB	FLXnCL CEDB	FLXnCL SEDB	FLXnCL VFRB
7	6	5	4	3	2	1	0
–	–	–	FLXnCL SBVA	FLXnCL CIA	FLXnCL CEDA	FLXnCL SEDA	FLXnCL VFRA

Table 30-62 FLXnACS write register contents

Bit position	Bit name	Function
31 to 0	see above	Clears the corresponding flags described in *Table 30-61 "FLXnACS read register contents" on page 2445* : 1 = Clears the corresponding flag. 0 = No function

Note Writing to bits 31 to 13 and 7 to 5 is ignored.

(10) FLXnESIDm - Even sync ID (m = 01 to 15]

Registers FLXnESID01 to FLXnESID15 hold the frame IDs of the sync frames received in *even* communication cycles used for clock synchronisation up to the limit of gSyncNodeMax. The values are sorted in ascending order, with register FLXnESID01 holding the lowest received sync frame ID. If the node itself transmits a sync frame in an even communication cycle, register FLXnESID01 holds the respective sync frame ID as configured in message buffer 0 and flags FLXnRXEA, FLXnRXEB are set. The value is updated during the NIT of each even communication cycle. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0130_H - <FLXn_base> + 0168_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXn RXEB	FLXn RXEA	0	0	0	0	FLXn EIDm[9:0]									
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-63 FLXnESIDm register contents

Bit position	Bit name	Function
15	FLXn RXEB	Received / Configured Even Sync ID on Channel B Signals that a sync frame corresponding to the stored even sync ID was received on channel B or that the node is configured to be a sync node with key slot = FLXnEIDm[9:0] (FLXnESID1 only). 1 = Sync frame received on channel B / node configured to transmit sync frames 0 = No sync frame received on channel B / node not configured to transmit sync frames
14	FLXn RXEA	Received / Configured Even Sync ID on Channel A Signals that a sync frame corresponding to the stored even sync ID was received on channel A or that the node is configured to be a sync node with key slot = FLXnEIDm[9:0] (FLXnESID1 only). 1 = Sync frame received on channel A / node configured to transmit sync frames 0 = No sync frame received on channel A / node not configured to transmit sync frames
9 to 0	FLXn EIDm[9:0]	Even Sync ID (*vsSyncIDListA,B even*) Sync frame ID even communication cycle.

(11) FLXnOSIDm - Odd sync ID (m = 01 to 15)

Registers FLXnOSID01 to FLXnOSID15 hold the frame IDs of the sync frames received in odd communication cycles used for clock synchronisation up to the limit of gSyncNodeMax. The values are sorted in ascending order, with register FLXnOSID01 holding the lowest received sync frame ID. If the node itself transmits a sync frame in an odd communication cycle, register FLXnOSID01 holds the respective sync frame ID as configured in message buffer 0 and flags FLXnRXOA, FLXnRXOB are set. The value is updated during the NIT of each odd communication cycle. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0170_H - <FLXn_base> + $01A8_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXn RXOB	FLXn RXOA	0	0	0	0	FLXnOIDm[9:0]									
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-64 FLXnOSIDm register contents

Bit position	Bit name	Function
15	FLXn RXOB	Received / Configured Odd Sync ID on Channel B Signals that a sync frame corresponding to the stored odd sync ID was received on channel B or that the node is configured to be a sync node with key slot = FLXnOIDm[9:0] (FLXnOSID1 only). 1 = Sync frame received on channel B / node configured to transmit sync frames 0 = No sync frame received on channel B / node not configured to transmit sync frames
14	FLXn RXOA	Received / Configured Odd Sync ID on Channel A Signals that a sync frame corresponding to the stored odd sync ID was received on channel A or that the node is configured to be a sync node with key slot = FLXnOIDm[9:0] (FLXnOSID1 only). 1 = Sync frame received on channel A / node configured to transmit sync frames 0 = No sync frame received on channel A / node not configured to transmit sync frames
9 to 0	FLXn OIDm[9:0]	Odd Sync ID (*vsSyncIDListA,B odd*) Sync frame ID odd communication cycle.

(12) FLXnNMVm - Network management vector (m = 1 to 3)

The three network management registers hold the accrued NM vector (configurable 0 to 12 bytes). The accrued NM vector is generated by the CC by bit-wise ORing each NM vector received (valid static frames with FLXnRHPPI = '1') on each channel (see chapter Network Management).

The CC updates the NM vector at the end of each communication cycle as long as the CC is either in NORMAL_ACTIVE or NORMAL_PASSIVE state.

FLXnNMVm-bytes exceeding the configured NM vector length are not valid.

Access This register can be read in 32-bit units.

Address <FLXn_base> + $01B0_H$- <FLXn_base> + $01B8_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnNMm[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnNMm[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The table below shows the assignment of the data bytes to the network management vector.

Table 30-65 Assignment of data bytes to network management vector

Bit Word	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnNMV1	Data3								Data2							
FLXnNMV2	Data7								Data6							
FLXnNMV3	Data11								Data10							

Table 30-66

Bit Word	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnNMV1	Data1								Data0							
FLXnNMV2	Data5								Data4							
FLXnNMV3	Data9								Data8							

30.3.7 Message buffer control registers

(1) FLXnMRC - Message RAM configuration

The Message RAM Configuration register defines the number of message buffers assigned to the static segment, dynamic segment, and FIFO. The register can be written during DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0300_H

Initial Value $0180\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXn SPLM*	FLXn SEC[1:0]*		FLXnLCB[7:0]*							
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnFFB[7:0]*								FLXnFDB[7:0]*							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

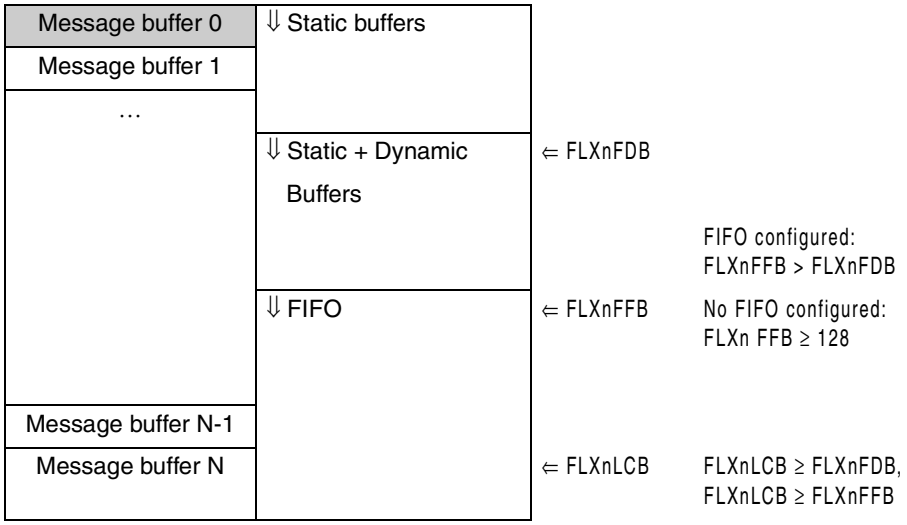
Table 30-67 FLXnMRC register contents (1/2)

Bit position	Bit name	Function
26	FLXn SPLM	Sync Frame Payload Multiplex This bit is only evaluated if the node is configured as sync node (FLXnSUCC1.FLXnTXSY = '1') or for single slot mode operation (FLXnSUCC1.FLXnTSM = '1'). When this bit is set to '1' message buffers 0 and 1 are dedicated for sync frame transmission with different payload data on channel A and B. When this bit is set to '0', sync frames are transmitted from message buffer 0 with the same payload data on both channels. Note that the channel filter configuration for message buffer 0 resp. message buffer 1 has to be chosen accordingly. 1 = Both message buffers 0 and 1 are locked against reconfiguration 0 = Only message buffer 0 locked against reconfiguration
25 to 24	FLXn SEC[1:0]	Secure Buffers Not evaluated when the CC is in DEFAULT_CONFIG or CONFIG state. 00 = Reconfiguration of message buffers enabled with numbers < FLXnFFB enabled Exception: In nodes configured for sync frame transmission or for single slot mode operation message buffer 0 (and if FLXnSPLM = '1', also message buffer 1) is always locked 01 = Reconfiguration of message buffers with numbers < FLXnFDB and with numbers FLXnFFB locked and transmission of message buffers for static segment with numbers FLXnFDB disabled 10 = Reconfiguration of all message buffers locked 11 = Reconfiguration of all message buffers locked and transmission of message buffers for static segment with numbers FLXnFDB disabled

Table 30-67 FLXnMRC register contents (2/2)

Bit position	Bit name	Function
23 to 16	FLXn LCB[7:0]	Last Configured Buffer 0º127 = Number of message buffers is FLXnLCB + 1 ≥1 28 = No message buffer configured
15 to 8	FLXn FFB[7:0]	First Buffer of FIFO 0 = All message buffers assigned to the FIFO 1...127 = Message buffers from FLXnFFB to FLXnLCB assigned to the FIFO ≥ 128 = No message buffer assigned to the FIFO
7 to 0	FLXn FDB[7:0]	First Dynamic Buffer 0 = No group of message buffers exclusively for the static segment configured 1...127 = Message buffers 0 to FLXnFDB - 1 reserved for static segment ≥ 128 = No dynamic message buffers configured

Note In case the node is configured as sync node (FLXnSUCC1.FLXnTXSY = ’1’) or for single slot mode operation (FLXnSUCC1.FLXnTSM = ’1’), message buffer 0 resp. 1 is reserved for sync frames or single slot frames and have to be configured with the node-specific key slot ID. In case the node is neither configured as sync node nor for single slot operation message buffer 0 resp. 1 is treated like all other message buffers.



The programmer has to ensure that the configuration defined by FLXnFDB[7:0], FLXnFFB[7:0], and FLXnLCB[7:0] is valid. *The CC does not check for erroneous configurations!*

Note The maximum number of header sections is 128. This means a maximum of 128 message buffers can be configured. The maximum length of a data section is 254 bytes. The length of the data section may be configured differently for each message buffer.
In case two or more message buffers are assigned to slot 1 by use of cycle filtering, all of them must be located either in the "Static Buffers" or at the beginning of the "Static + Dynamic Buffers" section.
The FlexRay protocol specification requires that each node has to send a frame in its key slot. Therefore at least message buffer 0 is reserved for transmission in the key slot. Due to this requirement a maximum number of 127 message buffers can be assigned to the FIFO. Nevertheless, a non protocol conform configuration without a transmission slot in the static segment would still be operational.
The payload length configured and the length of the data section need to be

configured identical for all message buffers belonging to the FIFO via FLXnWRHS2.FLXnWHPLC[6:0] and FLXnWRHS3.FLXnWHDP[10:00]. When the CC is not in DEFAULT_CONFIG or CONFIG state reconfiguration of message buffers belonging to the FIFO is locked.

(2) FLXnFRF - FIFO Rejection Filter

The FIFO Rejection Filter defines a user specified sequence of bits to which channel, frame ID, and cycle count of the incoming frames are compared. Together with the FIFO Rejection Filter Mask this register determines whether a message is rejected by the FIFO. The FLXnFRF register can be written during DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0304_H

Initial Value $0180\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	FLXn RNF*	FLXn RSS*	FLXnCYF[6:0]*						
R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	FLXnFID[10:00]*											FLXnCH[1:0]*	
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-68 FLXnFRF register contents

Bit position	Bit name	Function
24	FLXn RNF	Reject Null Frames If this bit is set, received null frames are not stored in the FIFO. 1 = Reject all null frames 0 = Null frames are stored in the FIFO
23	FLXn RSS	Reject in Static Segment If this bit is set, the FIFO is used only for the dynamic segment. 1 = Reject messages in static segment 0 = FIFO also used in static segment
22 to 16	FLXn CYF[6:0]	Cycle Counter Filter The 7-bit cycle counter filter determines the cycle set to which frame ID and channel rejection filter are applied. In cycles **not** belonging to the cycle set specified by FLXnCYF[6:0], all frames are rejected. For details about the configuration of the cycle counter filter see Section Cycle Counter Filtering.
12 to 2	FLXn FID[10:00]	Frame ID Filter Determines the frame ID to be rejected by the FIFO. With the additional configuration of register FLXnFRFM, the corresponding frame ID filter bits are ignored, which results in further rejected frame IDs. When FLXnFRFM.FLXnMFID[10:00] is zero, a frame ID filter value of zero means that **no** frame ID is rejected. 0...2047 = Frame ID filter values
1 to 0	FLXn CH[1:0]	Channel Filter 11 =no reception 10 =receive only on channel A 01 =receive only on channel B 00 =receive on both channels **Note:** If reception on both channels is configured, also in static segment always both frames (from channel A and B) are stored in the FIFO, even if they are identical.

(3) FLXnFRFM - FIFO rejection filter mask

The FIFO Rejection Filter Mask specifies which of the corresponding FLXnFRF bits are relevant for rejection filtering. If a bit is set, it indicates that the state of the corresponding bit in the FLXnFRF register will not affect whether or not the message is rejected by the FIFO. The FLXnFRFM register can be written during DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0308_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	FLXnMFID[10:00]*											0	0
R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 30-69 FLXnFRFM register contents

Bit position	Bit name	Function
12 to 2	FLXn MFID[10:00]	FIFO Rejection Filter Mask 1 = Ignore corresponding FIFO Rejection Filter bit. 0 = Corresponding FIFO Rejection Filter bit not ignored.

(4) FLXnFCL - FIFO Critical Level

The CC accepts modifications of the register in DEFAULT_CONFIG or CONFIG state only.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $030C_H$

Initial Value $0000\ 0080_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	FLXnCL[7:0]*							
R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-70 FLXnFCL register contents

Bit position	Bit name	Function
7 to 0	FLXn CL[7:0]	Critical Level When the receive FIFO fill level FLXnFSR.FLXnRFFL[7:0] is equal or greater than the critical level configured by FLXnCL[7:0], the receive FIFO critical level flag FLXnFSR.FLXnRFCL is set. If FLXnCL[7:0] is programmed to values > 128, bit FLXnFSR.FLXnRFCL is never set. When FLXnFSR.FLXnRFCL changes from '0' to '1' bit FLXnSIR.FLXnRFCLF is set to '1', and if enabled, an interrupt is generated.

30.3.8 Message buffer status registers

(1) FLXnMHDS - Message handler status

A flag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect on the flag. The register will also be cleared by hard reset or by CHI command CLEAR_RAMS.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0310_H

Initial Value $0000\ 0080_H$

(a) FLXnMHDS read

31	30	29	28	27	26	25	24
0	FLXnMBU[6:0]						
23	22	21	20	19	18	17	16
0	FLXnMBT[6:0]						
15	14	13	12	11	10	9	8
0	FLXnFMB[6:0]						
7	6	5	4	3	2	1	0
FLXn CRAM	FLXn MFMBE	FLXn FMBDE	FLXn DTBFBE	FLXn DTBFAE	FLXn DMRE	FLXn DOBFE	FLXn DIBFE

Table 30-71 FLXnMHDS read register contents (1/2)

Bit position	Bit name	Function
30 to 24	FLXn MBU[6:0]	Message Buffer Updated Number of message buffer that was updated last by the CC. For this message buffer the respective ND and / or MBC flag in the FLXnNDAT1/2/3/4 registers and the FLXnMBSC1/2/3/4 registers are also set.
22 to 16	FLXn MBT[6:0]	Message Buffer Transmitted Number of last successfully transmitted message buffer. If the message buffer is configured for single-shot mode, the respective FLXnTXR flag in the Transmission Request Register 1,2,3,4 was reset.
14 to 8	FLXn FMB[6:0]	Faulty Message Buffer Double bit error occurred when reading from or writing to the message buffer referenced by FLXnFMB[6:0]. Value only valid when one of the flags FLXnDIBFE, PMR, FLXnDTBFAE, FLXnDTBFBE, and flag FLXnFMBDE is set. Is not updated while flag FLXnFMBDE is set.
7	FLXn CRAM	Clear all internal RAM's Signals that execution of the CHI command CLEAR_RAMS is ongoing (all bits of all internal RAM blocks are written to '0'). The bit is set by hardware reset or by CHI command CLEAR_RAMS. 1 = Execution of the CHI command CLEAR_RAMS ongoing 0 = No execution of the CHI command CLEAR_RAMS
6	FLXn MFMBE	Multiple Faulty Message Buffers detected 1 = Another faulty message buffer was detected while flag FLXnFMBDE is set 0 = No additional faulty message buffer
5	FLXn FMBDE	Faulty Message Buffer Detected 1 = Message buffer referenced by FLXnFMB[6:0] holds faulty data due to a ECC double bit error 0 = No faulty message buffer

Table 30-71 FLXnMHDS read register contents (2/2)

Bit position	Bit name	Function
4	FLXn DTBFBE	Double bit Error Transient Buffer RAM B 1 = Double bit error occurred when reading Transient Buffer RAM B 0 = No double bit error
3	FLXn DTBFAE	Double bit Error Transient Buffer RAM A 1 = Double bit error occurred when reading Transient Buffer RAM A 0 = No double bit error
2	FLXn DMRE	Double bit error Message RAM 1 = Double bit occurred when reading the Message RAM 0 = No double bit error
1	FLXnDOBFE	Double bit Error Output Buffer RAM 1,2 1 = Double bit error occurred when reading Output Buffer RAM 1,2 0 = No double bit error
0	FLXnDIBFE	Double bit Error Input Buffer RAM 1,2 1 = Double bit error occurred when reading Input Buffer RAM 1,2 0 = No double bit error

Notes
1. When one of the following flags FLXnDIBFE, FLXnDOBFE, PMR, FLXnDTBFAE, FLXnDTBFBE changes from '0' to '1', the ECC double error detection interrupt INTFLXA0DED is asserted.
2. FLXnMBT[6:0] and FLXnMBU[6:0] are reset when the CC leaves CONFIG state or enters STARTUP state.

(b) FLXnMHDS write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	FLXnCL MFMB	FLXnCL FMBD	FLXnCL DTBFB	FLXnCL DTBFA	FLXnCL DMR	FLXnCL DOBF	FLXnCL DIBF

Table 30-72 FLXnMHDS write register contents

Bit position	Bit name	Function
31 to 0	see above	Clears the corresponding flags described in *Table 30-71 "FLXnMHDS read register contents" on page 2457* : 1 = Clears the corresponding flag. 0 = No function

Note Writing to bits 31 to 7 is ignored.

(2) FLXnLDTS - Last Dynamic Transmit Slot

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0314_H

Initial Value $0000\ 0000_H$
The register is reset when the CC leaves CONFIG state or enters STARTUP state.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	FLXnLDTB[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnLDTA[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-73 FLXnLDTS register contents

Bit position	Bit name	Function
26 to 16	FLXn LDTB[10:00]	Last Dynamic Transmission Channel B Value of vSlotCounter[B] at the time of the last frame transmission on channel B in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no frame was transmitted during the dynamic segment.
10 to 0	FLXn LDTA[10:00]	Last Dynamic Transmission Channel A Value of vSlotCounter[A] at the time of the last frame transmission on channel A in the dynamic segment of this node. It is updated at the end of the dynamic segment and is reset to zero if no frame was transmitted during the dynamic segment.

(3) FLXnFSR - FIFO Status Register

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0318_H

Initial Value $0000\ 0000_H$

The register is reset when the CC leaves CONFIG state or enters STARTUP state.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnRFFL[7:0]								0	0	0	0	0	FLXn RFO	FLXn RFCL	FLXn RFNE
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-74 FLXnFSR register contents

Bit position	Bit name	Function
15 to 8	FLXn RFFL[7:0]	Receive FIFO Fill Level Number of FIFO buffers filled with new data not yet read by the Host. Maximum value is 128.
2	FLXn RFO	Receive FIFO Overrun The flag is set by the CC when a receive FIFO overrun is detected. When a receive FIFO overrun occurs, the oldest message is overwritten with the actual received message. In addition, interrupt flag FLXnEIR.FLXnRFOE is set.The flag is cleared by the next FIFO read access issued by the Host. 1 = A receive FIFO overrun has been detected 0 = No receive FIFO overrun detected
1	FLXn RFCL	Receive FIFO Critical Level This flag is set when the receive FIFO fill level FLXnRFFL[7:0] is equal or greater than the critical level as configured by FLXnFCL.CL[7:0]. The flag is cleared by the CC as soon as FLXnRFFL[7:0] drops below FLXnFCL.CL[7:0]. When FLXnRFCL changes from '0' to '1' bit FLXnSIR.FLXnRFCLF is set to '1', and if enabled, an interrupt is generated. 1 = Receive FIFO critical level reached 0 = Receive FIFO below critical level
0	FLXn RFNE	Receive FIFO Not Empty This flag is set by the CC when a received valid frame (data or null frame depending on rejection mask) was stored in the FIFO. In addition, interrupt flag FLXnSIR.FLXnRFNEF is set. The bit is reset after the Host has read all message from the FIFO. 1 = Receive FIFO is not empty 0 = Receive FIFO is empty

(4) FLXnMHDF - Message Handler Constraints Flags

Some constraints exist for the Message Handler regarding eray_bclk frequency, Message RAM configuration, and FlexRay bus traffic (see Addendum to E-Ray FlexRay IP-Module Specification). To simplify software development, constraints violations are reported by setting flags in the FLXnMHDF.

A flag is cleared by writing a ’1’ to the corresponding bit position. Writing a ’0’ has no effect on the flag. A hard reset will also clear the register. The register is reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + $031C_H$

Initial Value $0000\ 0000_H$

(a) FLXnMHDF read

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	FLXn WAHPE
7	6	5	4	3	2	1	0
FLXn TNSBE	FLXn TNSAE	FLXn TBFBE	FLXn TBFAE	FLXn FNFBE	FLXn FNFAE	FLXn SNUBE	FLXn SNUAE

Table 30-75 FLXnMHDF read register contents (1/2)

Bit position	Bit name	Function
8	FLXn WAHPE	Write Attempt to Header Partition Outside DEFAULT_CONFIG and CONFIG state this flag is set by the CC when the message handler tries to write message data into the header partition of the Message RAM due to faulty configuration of a message buffer. The write attempt is not executed, to protect the header partition from unintended write accesses. 1 = Write attempt to header partition 0 = No write attempt to header partition
7	FLXn TNSBE	Transmission Not Started Channel B This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel B at the action point of the configured slot. 1 = Transmission not started on channel B 0 = No transmission not started on channel B
6	FLXn TNSAE	Transmission Not Started Channel A This flag is set by the CC when the Message Handler was not ready to start a scheduled transmission on channel A at the action point of the configured slot. 1 = Transmission not started on channel A 0 = No transmission not started on channel A

Table 30-75 FLXnMHDF read register contents (2/2)

Bit position	Bit name	Function
5	FLXn TBFBE	Transient Buffer Access Failure B This flag is set by the CC when a read or write access to TBF B requested by PRT B could not complete within the available time. 1 = TBF B access failure 0 = No TBF B access failure
4	FLXn TBFAE	Transient Buffer Access Failure A This flag is set by the CC when a read or write access to TBF A requested by PRT A could not complete within the available time. 1 = TBF A access failure 0 = No TBF A access failure
3	FLXn FNFBE	Find Sequence Not Finished Channel B This flag is set by the CC when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel B. 1 = Find sequence not finished for channel B 0 = No find sequence not finished for channel B
2	FLXn FNFAE	Find Sequence Not Finished Channel A This flag is set by the CC when the Message Handler, due to overload condition, was not able to finish a find sequence (scan of Message RAM for matching message buffer) with respect to channel A. 1 = Find sequence not finished for channel A 0 = No find sequence not finished for channel A
1	FLXn SNUBE	Status Not Updated Channel B This flag is set by the CC when the Message Handler, due to overload condition, was not able to update a message buffer's status MBS with respect to channel B. 1 = MBS for channel B not updated 0 = No overload condition occurred when updating MBS for channel B
0	FLXn SNUAE	Status Not Updated Channel A This flag is set by the CC when the Message Handler, due to overload condition, was not able to update a message buffer's status MBS with respect to channel A. 1 = MBS for channel A not updated 0 = No overload condition occurred when updating MBS for channel A

Note When one of the flags FLXnSNUAE, FLXnSNUBE, FLXnFNFAE, FLXnFNFBE, FLXnDTBFAE, FLXnDTBFBE, FLXnWAHPE changes from '0' to '1', interrupt flag FLXnEIR.FLXnMHFE is set to '1'.

(b) FLXnMHDF write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	FLXnCL WAHP
7	6	5	4	3	2	1	0
FLXnCL TNSB	FLXnCL TNSA	FLXnCL TBFB	FLXnCL TBFA	FLXnCL FNFB	FLXnCL FNFA	FLXnCL SNUB	FLXnCL SNUA

Table 30-76 FLXnMHDF write register contents

Bit position	Bit name	Function
31 to 0	see above	Clears the corresponding flags described in *Table 30-75 "FLXnMHDF read register contents" on page 2461* : 1 = Clears the corresponding flag. 0 = No function

Note Writing to bits 31 to 9 is ignored.

(5) FLXnTXRQ1/2/3/4 - Transmission request 1/2/3/4

The four registers reflect the state of the FLXnTXR flags of all configured message buffers. The flags are evaluated for transmit buffers only. If the number of configured message buffers is less than 128, the remaining FLXnTXR flags have no meaning

Access This register can be read in 32-bit units.

Address <FLXn_base> + $032C_H$
<FLXn_base> + 0328_H
<FLXn_base> + 0324_H
<FLXn_base> + 0320_H

Initial Value $0000\ 0000_H$

FLXnTXRQ4:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnTXR[127:112]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnTXR[111:096]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnTXRQ3:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnTXR[095:080]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnTXR[079:064]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnTXRQ2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnTXR[063:048]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnTXR[047:032]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnTXRQ1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnTXR[031:016]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnTXR[015:000]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-77 FLXnTXRQ1/2/3/4 register contents

Bit position	Bit name	Function
31 to 0 in all registers	FLXn TXR[127:000]	Transmission Request If the flag is set, the respective message buffer is ready for transmission respectively transmission of this message buffer is in progress. In single-shot mode the flags are reset after transmission has completed.

(6) FLXnNDAT1/2/3/4 - New data 1/2/3/4

The four registers reflect the state of the FLXnND flags of all configured message buffers. FLXnND flags belonging to transmit buffers have no meaning. If the number of configured message buffers is less than 128, the remaining FLXnND flags have no meaning. The registers are reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read in 32-bit units.

Address <FLXn_base> + $033C_H$
<FLXn_base> + 0338_H
<FLXn_base> + 0334_H
<FLXn_base> + 0330_H

Initial Value $0000\ 0000_H$

FLXnNDAT4**:**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnND[127:112]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnND[111:096]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnNDAT3:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnND[095:080]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnND[079:064]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnNDAT2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnND[063:048]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnND[047:032]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnNDAT1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnND[031:016]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnND[015:000]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-78 FLXnNDAT1/2/3/4 register contents

Bit position	Bit name	Function
31 to 0 in all registers	FLXn ND[127:000]	New Data The flags are set when a valid received data frame matches the message buffer's filter configuration, independent of the payload length received or the payload length configured for that message buffer. The flags are not set after reception of null frames except for message buffers belonging to the receive FIFO. An FLXnND flag is reset when the header section of the corresponding message buffer is reconfigured or when the data section has been transferred to the Output Buffer.

(7) FLXnMBSC1/2/3/4 - Message buffer status changed 1/2/3/4

The four registers reflect the state of the FLXnMBC flags of all configured message buffers. If the number of configured message buffers is less than 128, the remaining FLXnMBC flags have no meaning. The registers are reset when the CC leaves CONFIG state or enters STARTUP state.

Access This register can be read in 32-bit units.

Address <FLXn_base> + $034C_H$
<FLXn_base> + 0348_H
<FLXn_base> + 0344_H
<FLXn_base> + 0340_H

Initial Value $0000\ 0000_H$

FLXnMBSC4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnMBC[127:112]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMBC[111:096]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnMBSC3

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnMBC[095:080]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMBC[079:064]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnMBSC2

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnMBC[063:048]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMBC[047:032]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FLXnMBSC1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnMBC[031:016]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMBC[015:000]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-79 FLXnMBSC1/2/3/4 register contents

Bit position	Bit name	Function
31 to 0 in all registers	FLXn MBC[127:000]	Message Buffer Status Changed An MBC flag is set whenever the Message Handler changes one of the status flags FLXnMBSVFRA, FLXnMBSVFRB, FLXnMBSSEOA, FLXnMBSSEOB, FLXnMBSCEOA, FLXnMBSCEOB, FLXnMBSSVOA, FLXnMBSSVOB, FLXnMBSTCIA, FLXnMBSTCIB, FLXnMBSESA, FLXnMBSESB, FLXnMBSMLST, FLXnMBSFTA, FLXnMBSFTB in the header section (see 4.11.5 Message Buffer Status (MBS) and 5.12.1 Header Partition, header 4) of the respective message buffer. An MBC flag is reset when the header section of the corresponding message buffer is reconfigured or when it has been transferred to the Output Buffer.

30.3.9 Identification Registers

(1) FLXnCREL - Core Release Register

Access This register can be read in 32-bit units.

Address <FLXn_base> + $03F0_H$

Initial Value $1027\ 1031_H$. This register is initialized by any reset

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnREL[3:0]				FLXnSTEP[7:0]								FLXnYEAR[3:0]			
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMON[7:0]								FLXnDAY[7:0]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-80 FLXnCREL register contents

Bit position	Bit name	Function
31 to 28	FLXn REL[3:0]	Core Release One digit, BCD-coded. 1_H: The IP release is the Revision 1.0.0
27 to 20	FLXn STEP[7:0]	Step of Core Release Two digits, BCD-coded. 00_H: The IP release is the Revision 1.0.0
19 to 16	FLXn YEAR[3:0]	Design Time Stamp, Year One digit, BCD-coded. 6_H: The IP was designed in the year 2006.
15 to 8	FLXn MON[7:0]	Design Time Stamp, Month Two digits, BCD-coded. 05_H: The IP was designed in May.
7 to 0	FLXn DAY[7:0]	Design Time Stamp, Day Two digits, BCD-coded. 19_H: The IP was desinged on the 19th of May.

Table below shows how releases are coded in register FLXnCREL.

Table 30-81 Coding for releases

Release	Step	Sub-Step	Name
0	7	0	Beta2
0	7	1	Beta2ct
0	7	2	Revision 1.0 RC1
1	0	0	Revision 1.0.0

(2) FLXnENDN - Endian Register

The Message Handler Status register gives the Host CPU access to the actual state of the Message Handler.

Access This register can be read in 32-bit units.

Address <FLXn_base> + $03F4_H$

Initial Value $8765\ 4321_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnETV[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnETV[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-82 FLXnENDN register contents

Bit position	Bit name	Function
31 to 0	FLXn ETV[31:00]	Endianness Test Value The endianness test value is $8765\ 4321_H$.

30.3.10 Input buffer

Double buffer structure consisting of Input Buffer Host and Input Buffer Shadow. While the Host can write to Input Buffer Host, the transfer to the Message RAM is done from Input Buffer Shadow. The Input Buffer holds the header and data sections to be transferred to the selected message buffer in the Message RAM. It is used to configure the message buffers in the Message RAM and to update the data sections of transmit buffers.

When updating the header section of a message buffer in the Message RAM from the Input Buffer, the Message Buffer Status as described in Section 4.11.5 Message Buffer Status (MBS) is automatically reset to zero.

The header sections of message buffers belonging to the receive FIFO can only be (re)configured when the CC is in DEFAULT_CONFIG or CONFIG state. For those message buffers only the payload length configured and the data pointer need to be configured via FLXnWRHS2.FLXnWHPLC[6.0] and FLXnWRHS3.FLXnWHDP[10:00]. All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask.

The data transfer between Input Buffer (IBF) and Message RAM is described in detail in Section Data Transfer from Input Buffer to Message RAM.

(1) FLXnWRDSm - Write data section (m = 01 to 64)

Holds the data words to be transferred to the data section of the addressed message buffer. The data words (DWn) are written to the Message RAM in transmission order from DW1 (byte0, byte1) to DWPL (PL = number of data words as defined by the payload length configured FLXnWRHS2.FLXnWHPLC[6:0]).

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0400_H - <FLXn_base> + $04FC_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnWDMDm[31:16]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnWDMDm[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-83 FLXnWRDSm register contents

Bit position	Bit name	Function
31 to 0	FLXn WDMDm[31:00]	Message Data FLXnWDMDm[07:00] = DW2n-1, byte4n-4 FLXnWDMDm[15:08] = DW2n-1, byte4n-3 FLXnWDMDm[23:16] = DW2n, byte4n-2 FLXnWDMDm[31:24] = DW2n, byte4n-1

Note DW127 is located on FLXnWRDS64.FLXnWDMDm[15:00]. In this case FLXnWRDS64.FLXnWDMDm[31:16] is unused (no valid data). The Input Buffer RAMs are initialized to zero when leaving hard reset or by CHI command CLEAR_RAMS.
Transmission order on the FlexRay bus is FLXnWRDSm[7:0], FLXnWRDSm[15:8], FLXnWRDSm[23:16], FLXnWRDSm[31:24] with the most significant bit transmitted first. To check how the E-Ray's endianness matches with the Host CPU's endianness, read register FLXnENDN.

(2) FLXnWRHS1 - Write header section 1

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0500_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnWHMBI	FLXnWHTXM	FLXnWHPPIT	FLXnWHCFG	FLXnWHCHB	FLXnWHCHA	0	FLXnWHCYC[6:0]						
R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnWHFID[10:00]										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-84 FLXnWRHS1 register contents (1/2)

Bit position	Bit name	Function
29	FLXnWHMBI	Message Buffer Interrupt This bit enables the receive / transmit interrupt for the corresponding message buffer. After a dedicated receive buffer has been updated by the Message Handler, flag FLXnRXIF and /or FLXnMBSIF in the Status Interrupt register are set. After successful transmission the FLXnTXIF flag in the Status Interrupt register is set. 1 = The corresponding message buffer interrupt is enabled 0 = The corresponding message buffer interrupt is disabled
28	FLXnWHTXM	Transmission Mode This bit is used to select the transmission mode. 1 = Single-shot mode 0 = Continuous mode
27	FLXnWHPPIT	Payload Preamble Indicator Transmit This bit is used to control the state of the Payload Preamble Indicator in transmit frames. If the bit is set in a static message buffer, the respective message buffer holds network management information. If the bit is set in a dynamic message buffer the first two bytes of the payload segment may be used for message ID filtering by the receiver. Message ID filtering of received FlexRay frames is not supported by the E-Ray module, but can be done by the Host CPU. 1 = Payload Preamble Indicator set 0 = Payload Preamble Indicator not set
26	FLXnWHCFG	Message Buffer Configuration Bit This bit is used to configure the corresponding buffer as Transmit Buffer or as Receive Buffer. For message buffers belonging to the receive FIFO the bit is not evaluated. 1 = The corresponding buffer is configured as Transmit Buffer 0 = The corresponding buffer is configured as Receive Buffer

Table 30-84 FLXnWRHS1 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>25 to 24</td><td>FLXn WHCHB, FLXn WHCHA</td><td>Channel Filter Control
The 2-bit channel filtering field associated with each buffer serves as a filter for Receive Buffers, and as a control field for transmit buffers.

<table>
<tr><th>FLXn WH CHA</th><th>FLXn WH CHB</th><th>Transmit Buffer
transmit frame on</th><th>Receive Buffer
store frame received from</th></tr>
<tr><td>1</td><td>1</td><td>both channels
(static segment only)</td><td>channel A or B
(store first semantically valid frame, static segment only)</td></tr>
<tr><td>1</td><td>0</td><td>channel A</td><td>channel A</td></tr>
<tr><td>0</td><td>1</td><td>channel B</td><td>channel B</td></tr>
<tr><td>0</td><td>0</td><td>no transmission</td><td>ignore frame</td></tr>
</table>

Note: If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to ’1’, no frames are transmitted resp. received frames are ignored (same function as FLXnWHCHA = FLXnWHCHB = ’0’)</td></tr>
<tr><td>22 to 16</td><td>FLXn WHCYC[6:0]</td><td>Cycle Code
The 7-bit cycle code determines the cycle set used for cycle counter filtering. For details about the configuration of the cycle code see chapter Cycle Counter Filtering.</td></tr>
<tr><td>10 to 0</td><td>FLXn WHFID[10:00]</td><td>Frame ID
Frame ID of the selected message buffer. The frame ID defines the slot number for transmission / reception of the respective message. Message buffers with frame ID = ’0’ are considered as not valid.</td></tr>
</table>

(3) FLXnWRHS2 - Write header section 2

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0504_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	FLXnWHPLC[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnWHCRC[10:00]										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-85 FLXnWRHS2 register contents

Bit position	Bit name	Function
22 to 0	FLXn WHPLC[6:0]	Payload Length Configured Length of data section (number of 2-byte words) as configured by the Host. During static segment the static frame payload length as configured by FLXnMHDC.FLXnSFDL[6:0] defines the payload length for all static frames. If the payload length configured by FLXnWHPLC[6:0] is shorter than this value padding bytes are inserted to ensure that frames have proper physical length. The padding pattern is logical zero.
10 to 0	FLXn WHCRC[10:00]	Header FLXnWHCRC (*vRF!Header!HeaderCRC*) Receive Buffer: Configuration not required Transmit Buffer: Header CRC calculated and configured by the Host For calculation of the header CRC the payload length of the frame send on the bus has to be considered. In static segment the payload length of all frames is configured by FLXnMHDC.FLXnSFDL[6:0].

(4) FLXnWRHS3 - Write header section 3

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0508_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnWHDP[10:00]										
R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-86 FLXnWRHS3 register contents

Bit position	Bit name	Function
10 to 0	FLXn WHDP[10:00]	Data Pointer Pointer to the first 32-bit word of the data section of the addressed message buffer in the Message RAM.

(5) FLXnIBCM - Input buffer command mask

Configures how the message buffer in the Message RAM selected by the Input Buffer Command Request register is updated. When IBF Host and IBF Shadow are swapped, also mask bits FLXnLHSH, FLXnLDSH, and FLXnSTXRH are swapped with bits FLXnLHSS, FLXnLDSS, and FLXnSTXRS to keep them attached to the respective Input Buffer transfer.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0510_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn STXRS	FLXn LDSS	FLXn LHSS
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn STXRH	FLXn LDSH	FLXn LHSH
R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W

Table 30-87 FLXnIBCM register contents

Bit position	Bit name	Function
18	FLXn STXRS	Set Transmission Request Shadow 1 = Set FLXnTXR flag, transmit buffer released for transmission (operation ongoing or finished) 0 = Reset FLXnTXR flag
17	FLXn LDSS	Load Data Section Shadow 1 = Data section selected for transfer from Input Buffer to the Message RAM (transfer ongoing or finished) 0 = Data section is not updated
16	FLXn LHSS	Load Header Section Shadow 1 = Header section selected for transfer from Input Buffer to the Message RAM (transfer ongoing or finished) 0 = Header section is not updated
2	FLXn STXRH	Set Transmission Request Host If this bit is set to '1', the FLXnTXR flag for the selected message buffer is set in the FLXnTXRQ1/2/3/4 registers to release the message buffer for transmission. In single-shot mode the flag is cleared by the CC after transmission has completed. FLXnTXR is evaluated for transmit buffers only. 1 = Set FLXnTXR flag, transmit buffer released for transmission 0 = Reset FLXnTXR flag
1	FLXn LDSH	Load Data Section Host 1 = Data section selected for transfer from Input Buffer to the Message RAM 0 = Data section is not updated
0	FLXn LHSH	Load Header Section Host 1 = Header section selected for transfer from Input Buffer to the Message RAM 0 = Header section is not updated

(6) FLXnIBCR - Input buffer command request

When the Host writes the number of a target message buffer in the Message RAM to FLXnIBRH[6:0] in the Input Buffer Command Request register, IBF Host and IBF Shadow are swapped. In addition the message buffer numbers stored under FLXnIBRH[6:0] and FLXnIBRS[6:0] are also swapped (see also *2 “Host access to Message RAM” on page 2525*).

With this write operation the FLXnIBSYS bit in the Input Buffer Command Request register is set to ’1’. The Message Handler then starts to transfer the contents of IBF Shadow to the message buffer in the Message RAM selected by FLXnIBRS[6:0].

While the Message Handler transfers the data from IBF Shadow to the target message buffer in the Message RAM, the Host may configure the next message in the IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, the FLXnIBSYS bit is set back to ’0’ and the next transfer to the Message RAM may be started by the Host by writing the respective target message buffer number to FLXnIBRH[6:0].

If a write access to FLXnIBRH[6:0] occurs while FLXnIBSYS is ’1’, FLXnIBSYH is set to ’1’. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, FLXnIBSYH is reset to ’0’. FLXnIBSYS remains set to ’1’, and the next transfer to the Message RAM is started. In addition the message buffer numbers stored under FLXnIBRH[6:0] and FLXnIBRS[6:0] are also swapped.

Any write access to an Input Buffer register while both FLXnIBSYS and FLXnIBSYH are set will cause the error flag FLXnEIR.FLXnIIBAE to be set. In this case the Input Buffer will not be changed.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0514_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnIBSYS	0	0	0	0	0	0	0	0	FLXnIBRS[6:0]						
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnIBSYH	0	0	0	0	0	0	0	0	FLXnIBRH[6:0]						
R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-88 FLXnIBCR register contents

Bit position	Bit name	Function
31	FLXn IBSYS	Input Buffer Busy Shadow Set to '1' after writing FLXnIBRH[6:0]. When the transfer between IBF Shadow and the Message RAM has completed, FLXnIBSYS is set back to '0'. 1 = Transfer between IBF Shadow and Message RAM in progress 0 = Transfer between IBF Shadow and Message RAM completed
22 to 16	FLXn IBRS[6:0]	Input Buffer Request Shadow Number of the target message buffer actually updated / lately updated. Valid values are 0000_H to $007F_H$ (0...127).
15	FLXn IBSYH	Input Buffer Busy Host Set to '1' by writing FLXnIBRH[6:0] while FLXnIBSYS is still '1'. After the ongoing transfer between IBF Shadow and the Message RAM has completed, the FLXnIBSYH is set back to '0'. 1 = Request while transfer between IBF Shadow and Message RAM in progress 0 = No request pending
6 to 0	FLXn IBRH[6:0]	Input Buffer Request Host Selects the target message buffer in the Message RAM for data transfer from Input Buffer. Valid values are 0000_H to $007F_H$ (0...127).

30.3.11 Output buffer

Double buffer structure consisting of Output Buffer Host and Output Buffer Shadow. Used to read out message buffers from the Message RAM. While the Host can read from Output Buffer Host, the Message Handler transfers the selected message buffer from Message RAM to Output Buffer Shadow. The data transfer between Message RAM and Output Buffer (OBF) is described in Section Data Transfer from Message RAM to Output Buffer.

(1) FLXnRDDSm - Read data section (m = 01 to 64)

Holds the data words read from the data section of the addressed message buffer. The data words (DWn) are read from the Message RAM in reception order from DW1 (byte0, byte1) to DWPL (PL = number of data words as defined by the payload length configured FLXnRDHS2.FLXnRHPLC[6:0]).

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0600_H - <FLXn_base> + $06FC_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FLXnRDMDm[31:16]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnRDMDm[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-89 FLXnRDDSm register contents

Bit position	Bit name	Function
31 to 0	FLXn RD MDm[31:00]	Message Data FLXnRDMDm[07:00] = DW2n-1, byte4n-4 FLXnRDMDm[15:08] = DW2n-1, byte4n-3 FLXnRDMDm[23:16] = DW2n, byte4n-2 FLXnRDMDm[31:24] = DW2n, byte4n-1

Note DW127 is located on FLXnRDDS64.FLXnRDMD[15:0]. In this case FLXnRDDS64.FLXnRDMD[31:16] is unused (no valid data). The Output Buffer RAMs are initialized to zero when leaving hard reset or by CHI command CLEAR_RAMS.

(2) FLXnRDHS1 - Read header section 1

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0700_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnRHMBI	FLXnRHTXM	FLXnRHPPIT	FLXnRHCFG	FLXnRHCHB	FLXnRHCHA	0	FLXnRHCYC[6:0]						
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnRHFID[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Values as configured by the Host via FLXnWRHS1:

Table 30-90 FLXnRDHS1 register contents

Bit position	Bit name	Function
29	FLXnRHMBI	Message Buffer Interrupt
28	FLXnRHTXM	Transmission Mode
27	FLXnRHPPIT	Payload Preamble Indicator Transmit
26	FLXnRHCFG	Message Buffer Direction Configuration Bit
25	FLXnRHCHB	Channel Filter Control
24	FLXnRHCHA	Channel Filter Control
22 to 16	FLXnRHCYC[6:0]	Cycle Code
10 to 0	FLXnRHFID[10:00]	Frame ID

In case that the message buffer read from the Message RAM belongs to the receive FIFO, FLXnRHFID[10:00] holds the received frame ID, while FLXnRHCYC[6:0], FLXnRHCHA, FLXnRHCHB, FLXnRHCFG, FLXnRHPPIT, FLXnRHTXM, and FLXnRHMBI are reset to ’0’.

(3) FLXnRDHS2 - Read header section 2

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0704_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	FLXnRHPLR[6:0]							0	FLXnRHPLC[6:0]						
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnRHCRC[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-91 FLXnRDHS2 register contents

Bit position	Bit name	Function
30 to 24	FLXn RHPLR[6:0]	Payload Length Received (*vRF!Header!Length*) Payload length value updated from received data frames (exception: if message buffer belongs to the receive FIFO FLXnRHPLR[6:0] is also updated from received null frames)
22 to 16	FLXn RHPLC[6:0]	Payload Length Configured Length of data section (number of 2-byte words) as configured by the Host.
10 to 0	FLXn RHCRC[10:00]	Header CRC (*vRF!Header!HeaderCRC*) Receive Buffer: Header CRC updated from received frame Transmit Buffer: Header CRC calculated and configured by the Host

When a message is stored into a message buffer the following behaviour with respect to payload length received and payload length configured is implemented:

FLXnRHPLR[6:0] > FLXnRHPLC[6:0]:
The payload data stored in the message buffer is truncated to the payload length configured if FLXnRHPLC[6:0] even or else truncated to FLXnRHPLC[6:0] + 1.

FLXnRHPLR[6:0] FLXnRHPLC[6:0]:
The received payload data is stored into the message buffers data section. The remaining data bytes of the data section as configured by FLXnRHPLC[6:0] are filled with undefined data

FLXnRHPLR[6:0] = zero:
The message buffers data section is filled with undefined data

FLXnRHPLC[6:0] = zero:
Message buffer has no data section configured. No data is stored into the message buffers data section.

Note The Message RAM is organized in 4-byte words. When received data is stored into a message buffers data section, the number of 2-byte data words written into the message buffer is FLXnRHPLC[6:0] rounded to the next even value. FLXnRHPLC[6:0] should be configured identical for all message buffers belonging to the receive FIFO. Header 2 is updated from data frames only.

(4) FLXnRDHS3 - Read header section 3

Access This register can be read in 32-bit units.

Address <FLXn_base> + 0708_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnRHRES	FLXnRHPPI	FLXnRHNFI	FLXnRHSYN	FLXnRHSFI	FLXnRHRCI	0	0	FLXnRHRCC[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	FLXnRHDP[10:00]										
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 30-92 FLXnRDHS3 register contents

Bit position	Bit name	Function
29	FLXnRHRES	Reserved Bit (*vRF!Header!Reserved*) Reflects the state of the received reserved bit. The reserved bit is transmitted as '0'.
28	FLXnRHPPI	Payload Preamble Indicator (*vRF!Header!PPIndicator*) The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame. 1 = Static segment: Network management vector at the beginning of the payload Dynamic segment: Message ID at the beginning of the payload 0 = The payload segment of the received frame does not contain a network management vector or a message ID
27	FLXnRHNFI	Null Frame Indicator (*vRF!Header!NFIndicator*) Is set to '1' after storage of the first received data frame. 1 = Received frame is *not* a Null frame 0 = Received frame is a Null frame
26	FLXnRHSYN	Sync Frame Indicator (*vRF!Header!SyFIndicator*) A sync frame is marked by the sync frame indicator. 1 = The received frame is a sync frame 0 = No sync frame received
25	FLXnRHSFI	Startup Frame Indicator (*vRF!Header!SuFIndicator*) A startup frame is marked by the startup frame indicator. 1 = The received frame is a startup frame 0 = No startup frame received
24	FLXnRHRCI	Received on Channel Indicator (vSS!Channel) Indicates the channel from which the received data frame was taken to update the respective receive buffer. 1 = Frame received on channel A 0 = Frame received on channel B
21 to 16	FLXnRHRCC[5:0]	Receive Cycle Count (*vRF!Header!CycleCount*) Cycle counter value updated from received frame.
10 to 0	FLXnRHDP[10:00]	Data Pointer Pointer to the first 32-bit word of the data section of the addressed message buffer in the Message RAM.

Note Header 3 is updated from data frames only.

(5) FLXnMBS - Message buffer status

The message buffer status is updated by the CC with respect to the assigned channel(s) latest at the end of the slot following the slot assigned to the message buffer. The flags are updated only when the CC is in NORMAL_ACTIVE or NORMAL_PASSIVE state. If only one channel (A or B) is assigned to a message buffer, the channel-specific status flags of the other channel are written to zero. If both channels are assigned to a message buffer, the channel-specific status flags of both channels are updated. The message buffer status is updated only when the slot counter reached the configured frame ID and when the cycle counter filter matched. When the Host updates a message buffer via Input Buffer, all MBS flags are reset to zero independent of which FLXnIBCM bits are set or not. For details about receive / transmit filtering see Sections Filtering and Masking, Transmit Process, and Receive Process. Whenever the Message Handler changes one of the flags FLXnMBSVFRA, FLXnMBSVFRB, FLXnMBSSEOA, FLXnMBSSEOB, FLXnMBSCEOA, FLXnMBSCEOB, FLXnMBSSVOA, FLXnMBSSVOB, FLXnMBSTCIA, FLXnMBSTCIB, FLXnMBSESA, FLXnMBSESB, FLXnMBSMLST, FLXnMBSFTA, FLXnMBSFTB the respective message buffer's MBC flag in registers FLXnMBSC1/2/3/4 is set.

Access This register can be read in 32-bit units.

Address <FLXn_base> + $070C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	FLXnMBSRES	FLXnMBSPPI	FLXnMBSNFI	FLXnMBSSYN	FLXnMBSSFI	FLXnMBSRCI	0	0	FLXnMBSCCS[5:0]					
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXnMBSFTB	FLXnMBSFTA	0	FLXnMBSMLST	FLXnMBSESB	FLXnMBSESA	FLXnMBSTCIB	FLXnMBSTCIA	FLXnMBSSVOB	FLXnMBSSVOA	FLXnMBSCEOB	FLXnMBSCEOA	FLXnMBSSEOB	FLXnMBSSEOA	FLXnMBSVFRB	FLXnMBSVFRA
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Note For receive buffers (FLXnRHCFG = '0') the following status bits are updated from both valid data and null frames. If no valid frame was received, the previous value is maintained. For transmit buffers the flags have no meaning and should be ignored.

Table 30-93 FLXnMBS register contents - receive buffers

Bit position	Bit name	Function
29	FLXnMBS RESS	Reserved Bit Status (vRF!Header!Reserved) Reflects the state of the received reserved bit. The reserved bit is transmitted as '0'.
28	FLXnMBS PPIS	Payload Preamble Indicator Status (vRF!Header!PPIndicator) The payload preamble indicator defines whether a network management vector or message ID is contained within the payload segment of the received frame. 1 = Static segment: Network management vector at the beginning of the payload Dynamic segment: Message ID at the beginning of the payload 0 = The payload segment of the received frame does not contain a network management vector or a message ID
27	FLXnMBS NFIS	Null Frame Indicator Status (vRF!Header!NFIndicator) If set to '0' the payload segment of the received frame contains no usable data. 1 = Received frame is *not* a null frame 0 = Received frame is a null frame
26	FLXnMBS SYNS	Sync Frame Indicator Status (vRF!Header!SyFIndicator) A sync frame is marked by the sync frame indicator. 1 = The received frame is a sync frame 0 = No sync frame received
25	FLXnMBS SFIS	Startup Frame Indicator Status (vRF!Header!SuFIndicator) A startup frame is marked by the startup frame indicator. 1 = The received frame is a startup frame 0 = No startup frame received
24	FLXnMBS RCIS	Received on Channel Indicator Status (vSS!Channel) Indicates the channel on which the frame was received. 1 = Frame received on channel A 0 = Frame received on channel B

Note The FlexRay protocol specification requires that FLXnMBSFTA, and FLXnMBSFTB can only be reset by the Host. Therefore the Cycle Count Status FLXnMBSCCS[5:0] for these bits is only valid for the cycle where the bits are set to '1'.

Table 30-94 FLXnMBS register contents - cycle count status

Bit position	Bit name	Function
21 to 16	FLXnMBS CCS[5:0]	Cycle Count Status Actual cycle count when status was updated.

Table 30-95 FLXnMBS register contents - transmit & receive buffers (1/2)

Bit position	Bit name	Function
15	FLXnMBS FTB	Frame Transmitted on Channel B Indicates that this node has transmitted a data frame in the configured slot on channel B. 1 = Data frame transmitted on channel B 0 = No data frame transmitted on channel B
14	FLXnMBS FTA	Frame Transmitted on Channel A Indicates that this node has transmitted a data frame in the configured slot on channel A. 1 = Data frame transmitted on channel A 0 = No data frame transmitted on channel A
12	FLXnMBS MLST	Message Lost The flag is set in case the Host did not read the message before the message buffer was updated from a received data frame. Not affected by reception of null frames except for message buffers belonging to the receive FIFO. The flag is reset by a Host write to the message buffer via IBF or when a new message is stored into the message buffer **after** the message buffers ND flag was reset by reading out the message buffer via OBF. 1 = Unprocessed message was overwritten 0 = No message lost
11	FLXnMBS ESB	Empty Slot Channel B In an empty slot there is no activity on the bus i.e. no frame transmission detected. 1 = No bus activity detected in the configured slot on channel B 0 = Bus activity detected in the configured slot on channel B
10	FLXnMBS ESA	Empty Slot Channel A In an empty slot there is no activity on the bus i.e. no frame transmission detected. 1 = No bus activity detected in the configured slot on channel A 0 = Bus activity detected in the configured slot on channel A
9	FLXnMBS TCIB	Transmission Conflict Indication Channel B (*vSS!TxConflictB*) A transmission conflict indication is set if a transmission conflict has occurred on channel B. 1 = Transmission conflict occurred on Channel B 0 = No transmission conflict occurred on Channel B
8	FLXnMBS TCIA	Transmission Conflict Indication Channel A (*vSS!TxConflictA*) A transmission conflict indication is set if a transmission conflict has occurred on channel A. 1 = Transmission conflict occurred on channel A 0 = No transmission conflict occurred on channel A
7	FLXnMBS SVOB	Slot Boundary Violation Observed on Channel B (*vSS!BViolationB*) A slot boundary violation was observed on channel B i.e. channel active at the start or at the end of the configured slot. 1 = Slot boundary violation observed on channel B 0 = No slot boundary violation observed on channel B
6	FLXnMBS SVOA	Slot Boundary Violation Observed on Channel A (*vSS!BViolationA*) A slot boundary violation was observed on channel A i.e. channel active at the start or at the end of the configured slot. 1 = Slot boundary violation observed on channel A 0 = No slot boundary violation observed on channel A
5	FLXnMBS CEOB	Content Error Observed on Channel B (*vSS!ContentErrorB*) A content error was observed in the configured slot on channel B. 1 = Content error observed on channel B 0 = No content error observed on channel B

Table 30-95 FLXnMBS register contents - transmit & receive buffers (2/2)

Bit position	Bit name	Function
4	FLXnMBS CEOA	Content Error Observed on Channel A (*vSS!ContentErrorA*) A content error was observed in the configured slot on channel A. 1 = Content error observed on channel A 0 = No content error observed on channel A
3	FLXnMBS SEOB	Syntax Error Observed on Channel B (*vSS!SyntaxErrorB*) A syntax error was observed in the configured slot on channel B. 1 = Syntax error observed on channel B 0 = No syntax error observed on channel B
2	FLXnMBS SEOA	Syntax Error Observed on Channel A (*vSS!SyntaxErrorA*) A syntax error was observed in the configured slot on channel A. 1 = Syntax error observed on channel A 0 = No syntax error observed on channel A
1	FLXnMBS VFRB	Valid Frame Received on Channel B A valid frame indication is set if a valid frame was received on channel B. 1 = Valid frame received on Channel B 0 = No valid frame received on Channel B
0	FLXnMBS VFRA	Valid Frame Received on Channel A A valid frame indication is set if a valid frame was received on channel A. 1 = Valid frame received on channel A 0 = No valid frame received on channel A

(6) FLXnOBCM - Output buffer command mask

Configures how the Output Buffer is updated from the message buffer in the Message RAM selected by FLXnOBCR.FLXnOBRS[6:0]. Mask bits FLXnRDSS and FLXnRHSS are copied to the register internal storage when a Message RAM transfer is requested by FLXnOBCR.FLXnREG. When OBF Host and OBF Shadow are swapped, mask bits FLXnRDSH and FLXnRHSH are swapped with the register internal storage to keep them attached to the respective Output Buffer transfer. The data transfer between Output Buffer and Message RAM is described in detail in Section 21.5.11.2 Data Transfer from Message RAM to Output Buffer.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0710_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn RDSH	FLXn RHSH
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	FLXn RDSS	FLXn RHSS
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W

Table 30-96 FLXnOBCM register contents

Bit position	Bit name	Function
17	FLXn RDSH	Read Data Section Host 1 = Data section selected for transfer from Message RAM to Output Buffer 0 = Data section is not read
16	FLXn RHSH	Read Header Section Host 1 = Header section selected for transfer from Message RAM to Output Buffer 0 = Header section is not read
1	FLXn RDSS	Read Data Section Shadow 1 = Data section selected for transfer from Message RAM to Output Buffer 0 = Data section is not read
0	FLXn RHSS	Read Header Section Shadow 1 = Header section selected for transfer from Message RAM to Output Buffer 0 = Header section is not read

Note After the transfer of the header section from the Message RAM to OBF Shadow has completed, the message buffer status changed flag FLXnMBC of the selected message buffer in the FLXnMBSC1/2/3/4 registers is cleared. After the transfer of the data section from the Message RAM to OBF Shadow has completed, the new data flag ND of the selected message buffer in the FLXnNDAT1/2/3/4 registers is cleared.

(7) FLXnOBCR - Output buffer command request

After setting bit FLXnREG to ’1’ while FLXnOBSYS is ’0’, FLXnOBSYS is automatically set to ’1’, FLXnOBRS[6:0] is copied to the register internal storage, mask bits FLXnOBCM.FLXnRDSS and FLXnOBCM.FLXnRHSS are copied to register FLXnOBCM internal storage, and the transfer of the message buffer selected by FLXnOBRS[6:0] from the Message RAM to OBF Shadow is started. When the transfer between the Message RAM and OBF Shadow has completed, this is signalled by setting FLXnOBSYS back to ’0’.

By setting bit FLXnVIEW to ’1’ while FLXnOBSYS is ’0’, OBF Host and OBF Shadow are swapped. Additionally mask bits FLXnOBCM.FLXnRDSH and FLXnOBCM.FLXnRHSH are swapped with the register FLXnOBCM internal storage to keep them attached to the respective Output Buffer transfer. FLXnOBRH[6:0] signals the number of the message buffer currently accessible by the Host.

If bits FLXnREG and FLXnVIEW are set to ’1’ with the same write access while FLXnOBSYS is ’0’, FLXnOBSYS is automatically set to ’1’ and OBF Shadow and OBF Host are swapped. Additionally mask bits FLXnOBCM.FLXnRDSH and FLXnOBCM.FLXnRHSH are swapped with the registers internal storage to keep them attached to the respective Output Buffer transfer. Afterwards FLXnOBRS[6:0] is copied to the register internal storage, and the transfer of the selected message buffer from the Message RAM to OBF Shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF Host. When the current transfer between Message RAM and OBF Shadow has completed, this is signalled by setting FLXnOBSYS back to ’0’.

Any write access to FLXnOBCR[15:8] while FLXnOBSYS is set will cause the error flag FLXnEIR.FLXnIOBAE to be set. In this case the Output Buffer will not be changed.

The data transfer between Output Buffer and Message RAM is described in detail in Section 21.5.11.2 Data Transfer from Message RAM to Output Buffer.

Access This register can be read/written in 32-bit units.

Address <FLXn_base> + 0714_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	FLXnOBRH[6:0]						
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLXn OBSYS	0	0	0	0	0	FLXn REQ	FLXn VIEW	0	FLXnOBRS[6:0]						
R	R	R	R	R	R	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 30-97 FLXnOBCR register contents

Bit position	Bit name	Function
22 to 16	FLXn OBRH[6:0]	Output Buffer Request Host Number of message buffer currently accessible by the Host via FLXnRDHS[1...3], MBS, and FLXnRDDS[1...64]. By writing FLXnVIEW to 1 OBF Shadow and OBF Host are swapped and the transferred message buffer is accessible by the Host. Valid values are 00_H to $7F_H$ (0...127).
15	FLXn OBSYS	Output Buffer Shadow Busy Set to '1' after setting bit FLXnVIEW. When the transfer between the Message RAM and OBF Shadow has completed, FLXnOBSYS is set back to '0'. 1 = Transfer between Message RAM and OBF Shadow in progress 0 = No transfer in progress
9	FLXn REQ	Request Message RAM Transfer Requests transfer of message buffer addressed by FLXnOBRS[6:0] from Message RAM to OBF Shadow. Only writeable while FLXnOBSYS = '0'. 1 = Transfer to OBF Shadow requested 0 = No request
8	FLXn VIEW	View Shadow Buffer Toggles between OBF Shadow and OBF Host. Only writeable while FLXnOBSYS = '0'. 1 = Swap OBF Shadow and OBF 0 = No action
6	FLXn OBRS[6:0]	Output Buffer Request Shadow Number of source message buffer to be transferred from the Message RAM to OBF Shadow. Valid values are 0000_H to $007F_H$ (0…127). If the number of the first message buffer of the receive FIFO is written to this register the Message Handler transfers the message buffer addressed by the GET Index Register (GIDX) to OBF Shadow.

30.4 Functional Description

This chapter describes the E-Ray implementation together with the related FlexRay protocol features. More information about the FlexRay protocol itself can be found in the FlexRay protocol specification v2.1.

Communication on FlexRay networks is based on frames and symbols. The wakeup symbol (WUS) and the collision avoidance symbol (CAS) are transmitted outside the communication cycle to setup the time schedule. Frames and media access test symbols (MTS) are transmitted inside the communication cycle.

30.4.1 Communication cycle

A communication cycle in FlexRay consists of the following elements:

- Static Segment
- Dynamic Segment (optional)
- Symbol Window (optional)
- Network Idle Time (NIT)

Static segment, dynamic segment, and symbol window form the Network Communication Time (NCT). For each communication channel the slot counter starts at 1 and counts up until the end of the dynamic segment is reached. Both channels share the same arbitration grid which means that they use the same synchronized macrotick.

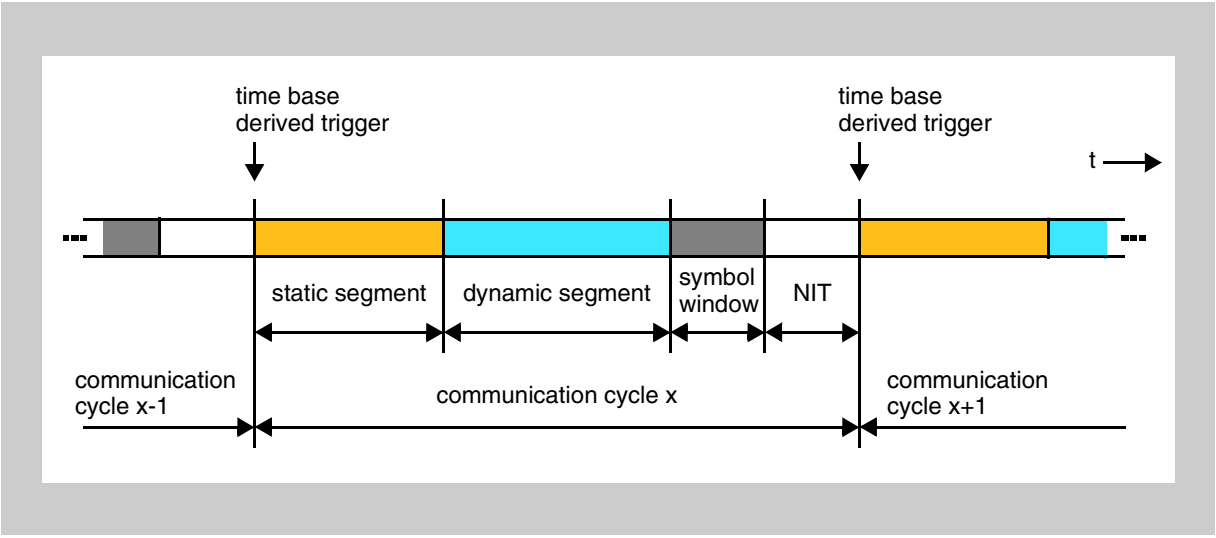


Figure 30-3 Structure of communication cycle

(1) Static Segment

The Static Segment is characterized by the following features:

• Time slots of fixed length (optionally protected by bus guardian)

• Start of frame transmission at action point of the respective static slot

• Payload length same for all frames on both channels

Parameters: Number of Static Slots FLXnGTUC07.NSS[9:0], Static Slot Length FLXnGTUC07.FLXnSSL[9:0], Payload Length Static FLXnMHDC.FLXnSFDL[6:0], Action Point Offset FLXnGTUC09.FLXnAPO[5:0]

(2) Dynamic segment

The Dynamic Segment is characterized by the following features:

- All controllers have bus access (no bus guardian protection possible)
- Variable payload length and duration of slots, different for both channels
- Start of transmission at minislot action point

Parameters: Number of Minislots FLXnGTUC08.FLXnNMS[12:00], Minislot Length FLXnGTUC08.FLXnMSL[5:0], Minislot Action Point Offset FLXnGTUC09.FLXnMAPO[4:0], Start of Latest Transmit (last minislot) FLXnMHDC.FLXnSLT[12:00]

(3) Symbol window

During the symbol window only one media access test symbol (MTS) may be transmitted per channel. MTS symbols are send in NORMAL_ACTIVE state to test the bus guardian.

The symbol window is characterized by the following features:

- Send single symbol
- Transmission of the MTS symbol starts at the symbol windows action point

Parameters: Symbol Window Action Point Offset FLXnGTUC09.FLXnAPO[4:0] (same as for static slots), Network Idle Time Start FLXnGTUC04.FLXnNIT[13:0]

(4) Network idle time (NIT)

During network idle time the CC has to perform the following tasks:

- Calculate clock correction terms (offset and rate)
- Distribute offset correction over multiple macroticks after offset correction start
- Perform cluster cycle related tasks

Parameters: Network Idle Time Start FLXnGTUC04.FLXnNIT[13:0], Offset Correction Start FLXnGTUC04.FLXnOCS[13:00]

(5) Configuration of NIT Start and Offset Correction Start

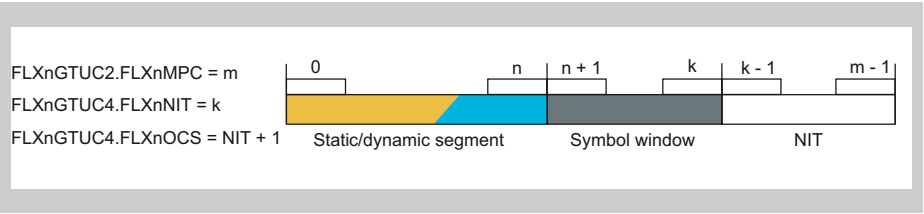


Figure 30-4 Configuration of NIT start and offset correction start

The number of macroticks per cycle gMacroPerCycle is assumed to be m. It is configured by programming FLXnGTUC02.FLXnMPC = m.

The static / dynamic segment starts with macrotick 0 and ends with macrotick n:

n = static segment length + dynamic segment offset
+ dynamic segment length - 1MT

n = gNumberOfStaticSlots • gdStaticSlot + dynamic segment offset
+ gNumberOfMinislots • gdMinislot - 1 MT

The static segment length is configured by FLXnGTUC07.FLXnSSL and FLXnGTUC07.NSS.
The dynamic segment length is configured by FLXnGTUC08.FLXnMSL and FLXnGTUC08.FLXnNMS.

The dynamic segment offset is:
If gdActionPointOffset gdMinislotActionPointOffset:
dynamic segment offset = 0 MT

Else if gdActionPointOffset > gdMinislotActionPointOffset:
dynamic segment offset =gdActionPointOffset
- gdMinislotActionPointOffset

The NIT starts with macrotick k+1 and ends with the last macrotick of cycle m-1. It has to be configured by setting FLXnGTUC04.NIT = k.

For the E-Ray the offset correction start is required to be FLXnGTUC04.FLXnOCS >= FLXnGTUC04.NIT + 1 = k+1.

The length of symbol window results from the number of macroticks between the end of the static / dynamic segment and the beginning of the NIT. It can be calculated by k - n.

30.4.2 Communication modes

The FlexRay Protocol Specification v2.1 defines the Time-Triggered Distributed (TT-D) mode.

(1) Time-triggered distributed (TT-D)

In TT-D mode the following configurations are possible:

- **Pure static**: minimum 2 static slots + symbol window (optional)
- **Mixed static/dynamic**: minimum 2 static slots + dynamic segment + symbol window (optional)

A minimum of two coldstart nodes need to be configured for distributed time-triggered operation. Two fault-free coldstart nodes are necessary for the cluster startup. Each startup frame must be a sync frame, therefore all coldstart nodes are sync nodes.

30.4.3 Clock synchronization

In TT-D mode a distributed clock synchronization is used. Each node individually synchronizes itself to the cluster by observing the timing of received sync frames from other nodes.

(1) Global time

Activities in a FlexRay node, including communication, are based on the concept of a global time, even though each individual node maintains its own view of it. It is the clock synchronization mechanism that differentiates the FlexRay cluster from other node collections with independent clock mechanisms. The global time is a vector of two values; the cycle (cycle counter) and the cycle time (macrotick counter).

Cluster specific:

- Macrotick (MT) = basic unit of time measurement in a FlexRay network, a macrotick consists of an integer number of microticks (µT)
- Cycle length = duration of a communication cycle in units of macroticks (MT)

(2) Local time

Internally, nodes time their behaviour with microtick resolution. Microticks are time units derived from the oscillator clock tick of the specific node. Therefore microticks are controller-specific units. They may have different duration in different controllers. The precision of a node's local time difference measurements is a microtick (µT).

Node specific:

- Oscillator clock -> prescaler -> microtick (µT)
- µT = basic unit of time measurement in a CC, clock correction is done in units of µTs
- Cycle counter + macrotick counter = nodes local view of the global time

(3) Synchronization process

Clock synchronization is performed by means of sync frames. Only preconfigured nodes (sync nodes) are allowed to send sync frames. In a two-channel cluster a sync node has to send its sync frame on both channels.

For synchronization in FlexRay the following constraints have to be considered:

- Max. one sync frame per node in one communication cycle
- Max. 15 sync frames per cluster in one communication cycle
- Every node has to use a preconfigured number of sync frames (FLXnGTUC02.FLXnSNM[3:0]) for clock synchronization
- Minimum of two sync nodes required for clock synchronization and startup

For clock synchronization the time difference between expected and observed arrival time of sync frames received during the static segment, valid on both channels (two-channel cluster), is measured. The calculation of correction terms is done during NIT (offset: every cycle, rate: odd cycle) by using a FTA / FTM algorithm.

Offset (phase) Correction

- Only deviation values measured and stored in the current cycle used
- For a two channel node the smaller value will be taken
- Calculation during NIT of **every** communication cycle
- Offset correction value calculated in even cycles used for error checking only
- Checked against limit values
- Correction value is a signed integer number of Ts
- Correction done in **odd** numbered cycles, distributed over the macroticks beginning at offset correction start up to cycle end (end of NIT) to shift nodes next start of cycle (MTs lengthened / shortened)

Rate (frequency) Correction

- Pairs of deviation values measured and stored in even / odd cycle pair used
- For a two channel node the average of the differences from the two channels is used
- Calculated during NIT of **odd** numbered cycles
- Cluster drift damping is performed using global damping value
- Checked against limit values
- Correction value is a signed integer number of Ts
- Distributed over macroticks comprising the next **even / odd** cycle pair (MTs lengthened / shortened)

Sync Frame Transmission

Sync frame transmission is only possible from buffer 0 and 1. Message buffer 1 may be used for sync frame transmission in case that sync frames should have different payloads on the two channels. In this case bit FLXnMRC.FLXnSPLM has to be programmed to ’1’.

Message buffers used for sync frame transmission have to be configured with the key slot ID and can be (re)configured in DEFAULT_CONFIG or CONFIG state only. For nodes transmitting sync frames FLXnSUCC1.FLXnTXSY must be set to ’1’.

(4) External Clock Synchronization

During normal operation, independent clusters can drift significantly. If synchronous operation across independent clusters is desired, external synchronization is necessary; even though the nodes within each cluster are synchronized. This can be accomplished with synchronous application of host-deduced rate and offset correction terms to the clusters.

- External offset / rate correction value is a signed integer
- External offset / rate correction value is added to calculated offset / rate correction value
- Aggregated offset / rate correction term (external + internal) is **not** checked against configured limits

30.4.4 Error handling

The implemented error handling concept is intended to ensure that, in case of a lower layer protocol error in one single node, communication between non-affected nodes can be maintained. In some cases, higher layer program activity is required for the CC to resume normal operation. A change of the error handling state will set FLXnEIR.FLXnPEMCE and may trigger an interrupt to the Host if enabled. The actual error mode is signalled by FLXnCCEV.FLXnERRM[1:0].

Table 30-98 Error modes of the POC (degradation model)

Error Mode	Activity
ACTIVE (green)	**Full operation**, State: NORMAL_ACTIVE The CC is fully synchronized and supports the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers FLXnEIR and FLXnSIR.
PASSIVE (yellow)	**Reduced operation**, State: NORMAL_PASSIVE, CC self rescue allowed The CC stops transmitting frames and symbols, but received frames are still processed. Clock synchronization mechanisms are continued based on received frames. No active contribution to the cluster wide clock synchronization. The host is informed of any error condition(s) or status change by interrupt (if enabled) or by reading the error and status interrupt flags from registers FLXnEIR and FLXnSIR.
COMM_HALT (red)	**Operation halted**, State: HALT, CC self rescue not allowed The CC stops frame and symbol processing, clock synchronization processing, and the macrotick generation. The host has still access to error and status information by reading the error and status interrupt flags from registers FLXnEIR and FLXnSIR. The bus drivers are disabled.

(1) Clock correction failed counter

When the Clock Correction Failed Counter reaches the "maximum without clock correction passive" limit defined by FLXnSUCC3.FLXnWCP[3:0], the POC transits from NORMAL_ACTIVE to NORMAL_PASSIVE state. When it reaches the "maximum without clock correction fatal" limit defined by FLXnSUCC3.FLXnWCF[3:0], it transits from NORMAL_ACTIVE or NORMAL_PASSIVE to HALT state.

The Clock Correction Failed Counter FLXnCCEV.FLXnCCFC[3:0] allows the Host to monitor the duration of the inability of a node to compute clock correction terms after the CC passed protocol startup phase. It will be incremented by one at the end of any **odd** communication cycle during which either the missing offset correction FLXnSFS.FLXnMOCS or the missing rate correction FLXnSFS.FLXnMRCS flag is set.

The Clock Correction Failed Counter is reset to zero at the end of an **odd** communication cycle if neither the missing offset correction FLXnSFS.FLXnMOCS nor the missing rate correction FLXnSFS.FLXnMRCS flag is set.

The Clock Correction Failed Counter stops incrementing when the "maximum without clock correction fatal" value FLXnSUCC3.FLXnWCF[3:0] is reached (i.e. incrementing the counter at its maximum value will not cause it to wrap around back to zero). The Clock Correction Failed Counter is initialized to zero when the CC enters READY state or when NORMAL_ACTIVE state is entered.

Note The transition to HALT state is prevented if FLXnSUCC1.FLXnHCSE is not set.

(2) Passive to active counter

The passive to active counter controls the transition of the POC from NORMAL_PASSIVE to NORMAL_ACTIVE state. FLXnSUCC1.FLXnPTA[4:0] defines the number of consecutive even / odd cycle pairs that must have valid clock correction terms before the CC is allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state. If FLXnSUCC1.FLXnPTA[4:0] is set to zero the CC is not allowed to transit from NORMAL_PASSIVE to NORMAL_ACTIVE state.

(3) HALT command

In case the Host wants to stop FlexRay communication of the local node it can bring the CC into HALT state by asserting the HALT command. This can be done by writing FLXnSUCC1.FLXnCMD[3:0] = "0110". In order to shut down communication on an entire FlexRay network, a higher layer protocol is required to assure that all nodes apply the HALT command at the same time.

The POC state from which the transition to HALT state took place can be read from FLXnCCSV.FLXnPSL[5:0].

When called in NORMAL_ACTIVE or NORMAL_PASSIVE state the POC transits to HALT state at the end of the current cycle. When called in any other state FLXnSUCC1.FLXnCMD[3:0] will be reset to "0000" = command_not_accepted and bit FLXnEIR.FLXnCNAE is set to ’1’. If enabled an interrupt to the Host is generated.

(4) FREEZE command

In case the Host detects a severe error condition it can bring the CC into HALT state by asserting the FREEZE command. This can be done by writing FLXnSUCC1.FLXnCMD[3:0] = "0111". The FREEZE command triggers the entry of the HALT state immediately regardless of the actual POC state.

The POC state from which the transition to HALT state took place can be read from FLXnCCSV.FLXnPSL[5:0].

30.4.5 Communication controller states

(1) Communication controller state diagram

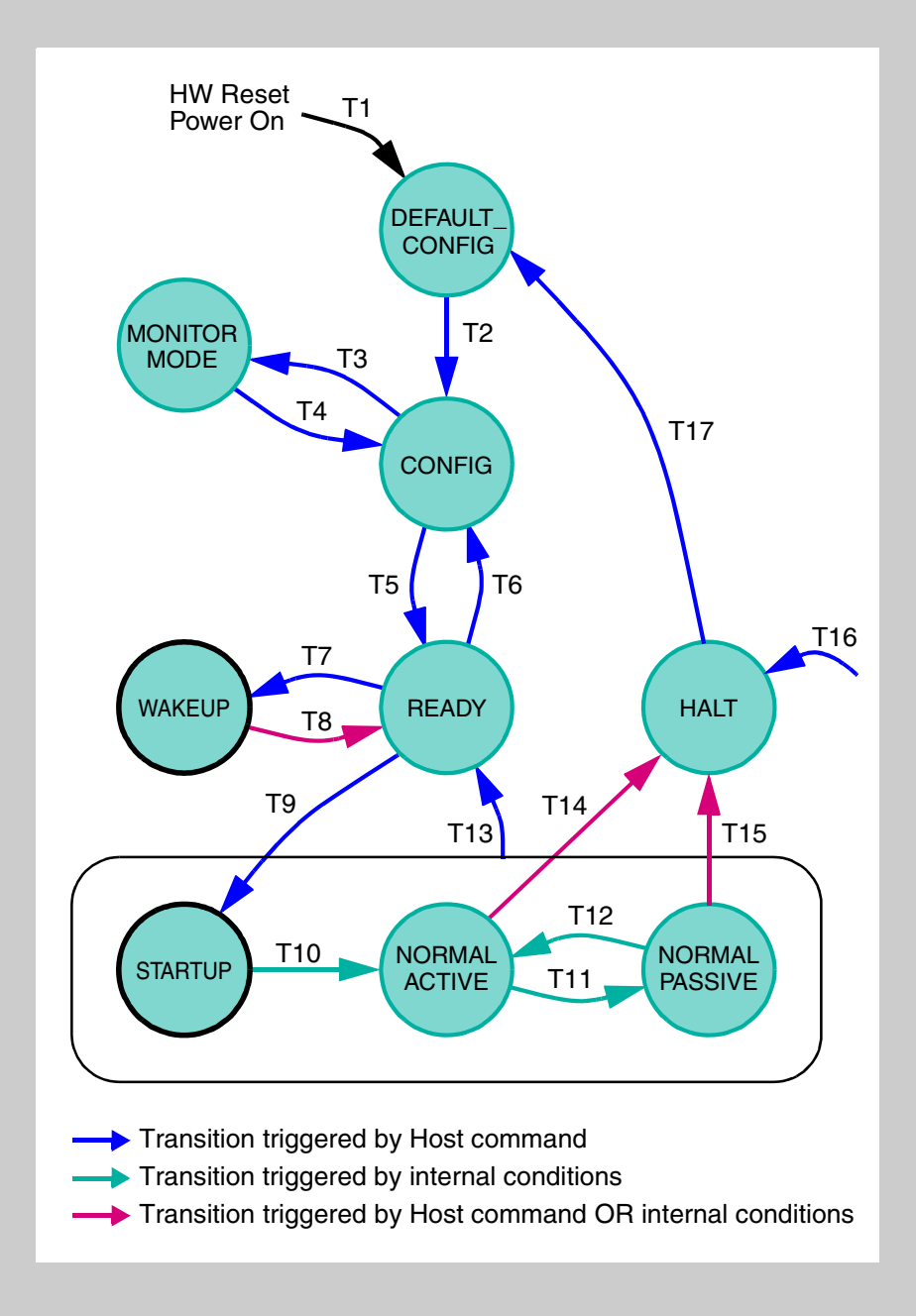


Figure 30-5 Overall state diagram of E-Ray communication controller

State transitions are controlled by externals pins eray_reset and eray_rxd1,2, by the POC state machine, and by the CHI Command Vector FLXnSUCC1.FLXnCMD[3:0].

The CC exits from **all** states to HALT state after application of the FREEZE command (FLXnSUCC1.FLXnCMD[3:0] = "0111").

Table 30-99 State transitions of E-Ray overall state machine

T#	Condition	From	To
1	Hard reset	All States	DEFAULT_CONFIG
2	Command CONFIG, FLXnSUCC1.FLXnCMD[3:0] = "0001"	DEFAULT_CONFIG	CONFIG
3	Unlock sequence followed by command MONITOR_MODE, FLXnSUCC1.FLXnCMD[3:0] = "1011"	CONFIG	MONITOR_MODE
4	Command CONFIG, FLXnSUCC1.FLXnCMD[3:0] = "0001"	MONITOR_MODE	CONFIG
5	Unlock sequence followed by command READY, FLXnSUCC1.FLXnCMD[3:0] = "0010"	CONFIG	READY
6	Command CONFIG, FLXnSUCC1.FLXnCMD[3:0] = "0001"	READY	CONFIG
7	Command WAKEUP, FLXnSUCC1.FLXnCMD[3:0] = "0011"	READY	WAKEUP
8	Complete, non-aborted transmission of wakeup pattern OR received WUP OR received frame header OR command READY, FLXnSUCC1.FLXnCMD[3:0] = "0010"	WAKEUP	READY
9	Command RUN, FLXnSUCC1.FLXnCMD[3:0] = "0100"	READY	STARTUP
10	Successful startup	STARTUP	NORMAL_ACTIVE
11	Clock Correction Failed counter reached Maximum Without Clock Correction Passive limit configured by FLXnSUCC3.FLXnWCP[3:0]	NORMAL_ACTIVE	NORMAL_PASSIVE
12	Number of valid correction terms reached the Passive to Active limit configured by FLXnSUCC1.FLXnPTA[4:0]	NORMAL_PASSIVE	NORMAL_ACTIVE
13	Command READY, FLXnSUCC1.FLXnCMD[3:0] = "0010"	STARTUP, NORMAL_ACTIVE, NORMAL_PASSIVE	READY
14	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by FLXnSUCC3.FLXnWCF[3:0] AND bit FLXnSUCC1.FLXnHCSE set to ’1’ OR command HALT, FLXnSUCC1.FLXnCMD[3:0] = "0110"	NORMAL_ACTIVE	HALT
15	Clock Correction Failed counter reached Maximum Without Clock Correction Fatal limit configured by FLXnSUCC3.FLXnWCF[3:0] AND bit FLXnSUCC1.FLXnHCSE set to ’1’ OR command HALT, FLXnSUCC1.FLXnCMD[3:0] = "0110"	NORMAL_PASSIVE	HALT
16	Command FREEZE, FLXnSUCC1.FLXnCMD[3:0] = "0111"	All States	HALT
17	Command CONFIG, FLXnSUCC1.FLXnCMD[3:0] = "0001"	HALT	DEFAULT_CONFIG

(2) DEFAULT_CONFIG state

In DEFAULT_CONFIG state, the CC is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state.

The CC enters this state

- When leaving hardware reset (external reset signal eray_reset is disasserted)
- When exiting from HALT state

To leave DEFAULT_CONFIG state the Host has to write FLXnCMD[3:0] = "0001" in the SUC Configuration Register 1. The CC then transits to CONFIG state.

(3) CONFIG state

In CONFIG state, the CC is stopped. All configuration registers are accessible and the pins to the physical layer are in their inactive state. This state is used to initialize the CC configuration.

The CC enters this state

- When exiting from DEFAULT_CONFIG state
- When exiting from MONITOR_MODE or READY state

When the state has been entered via HALT and DEFAULT_CONFIG state, the Host can analyse status information and configuration. Before leaving CONFIG state the Host has to assure that the configuration is fault-free.

To leave CONFIG state, the Host has to perform the unlock sequence as described in chapter Lock Register (FLXnLCK). Directly after unlocking the CONFIG state the Host has to write FLXnSUCC1.FLXnCMD[3:0] to enter the next state.

Note IStatus bits FLXnMHDS[14:0], registers FLXnTXRQ1/2/3/4, and status data stored in the Message RAM are not affected by the transition of the POC from CONFIG to READY state.

When the CC is in CONFIG state it is also possible to bring the CC into a power saving mode by halting the module clocks (eray_sclk, eray_bclk). To do this the Host has to assure that all Message RAM transfers have finished before turning off the clocks.

(4) MONITOR_MODE

After unlocking CONFIG state and writing FLXnSUCC1.FLXnCMD[3:0] = "1011" the CC enters MONITOR_MODE. In this mode the CC is able to receive FlexRay frames and to detect wakeup pattern. The temporal integrity of received frames is not checked, and therefore cycle counter filtering is not supported. This mode can be used for debugging purposes in case e.g. that startup of a FlexRay network fails. After writing FLXnSUCC1.FLXnCMD[3:0] = "0001" the CC transits back to CONFIG state.

In MONITOR_MODE the pick first valid mechanism is disabled. This means that a receive message buffer may only be configured to receive on one channel. Received frames are stored into message buffers according to frame ID and receive channel. Null frames are handled like data frames. After frame reception only status bits FLXnMBS.FLXnMBSVFRA, FLXnMBS.FLXnMBSVFRB, FLXnMBS.FLXnMBSMLST, FLXnMBS.FLXnMBSRCI, FLXnMBS.FLXnMBSSFI, FLXnMBS.FLXnMBSSYN, FLXnMBS.FLXnMBSNFI, FLXnMBS.FLXnMBSPPI, FLXnMBS.FLXnMBSRES have valid values. In MONITOR_MODE the receive FIFO is not available.

(5) READY state

After unlocking CONFIG state and writing FLXnSUCC1.FLXnCMD[3:0] = "0010" the CC enters READY state. From this state the CC can transit to WAKEUP state and perform a cluster wakeup or to STARTUP state to perform a coldstart or to integrate into a running cluster.

The CC enters this state

- When exiting from CONFIG, WAKEUP, STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state by writing FLXnSUCC1.FLXnCMD[3:0] = "0010" (READY command).

The CC exits from this state

- To CONFIG state by writing FLXnSUCC1.FLXnCMD[3:0] = "0001" (CONFIG command)
- To WAKEUP state by writing FLXnSUCC1.FLXnCMD[3:0] = "0011" (WAKEUP command)
- To STARTUP state by writing FLXnSUCC1.FLXnCMD[3:0] = "0100" (RUN command)

Internal counters and the CC status flags are reset when the CC enters STARTUP state.

Note Status bits FLXnMHDS[14:0], registers FLXnTXRQ1/2/3/4, and status data stored in the Message RAM are not affected by the transition of the POC from READY to STARTUP state.

(6) WAKEUP state

The description below is intended to help configuring wakeup for the E-Ray IP-module. A detailed description of the wakeup procedure together with the respective SDL diagrams can be found in the FlexRay protocol specification v2.1, section 7.1.

The CC enters this state

- When exiting from READY state by writing FLXnSUCC1.FLXnCMD[3:0] = "0011" (WAKEUP command).

The CC exits from this state to READY state

- After complete non-aborted transmission of wakeup pattern
- After WUP reception
- After detecting a WUP collision
- After reception of a frame header
- By writing FLXnSUCC1.FLXnCMD[3:0] = "0010" (READY command)

The cluster wakeup must precede the communication startup in order to ensure that all nodes in a cluster are awake. The minimum requirement for a cluster wakeup is that all bus drivers are supplied with power. A bus driver has the ability to wake up the other components of its node when it receives a wakeup pattern on its channel. At least one node in the cluster needs an **external** wakeup source.

The Host completely controls the wakeup procedure. It is informed about the state of the cluster by the bus driver and the CC and configures bus guardian (if available) and CC to perform the cluster wakeup. The CC provides to the Host the ability to transmit a special wakeup pattern on each of its available channels separately. The CC needs to recognize the wakeup pattern only during WAKEUP state.

Wakeup may be performed on only one channel at a time. The Host has to configure the wakeup channel while the CC is in CONFIG state by writing FLXnSUCC1.FLXnWUCS. The CC ensures that ongoing communication on this channel is not disturbed. The CC cannot guarantee that all nodes connected to the configured channel awake upon the transmission of the wakeup pattern, since these nodes cannot give feedback until the startup phase. The wakeup procedure enables single-channel devices in a two-channel system to trigger the wakeup, by only transmitting the wakeup pattern on the single channel to which they are connected. Any coldstart node that deems a system startup necessary will then wake the remaining channel before initiating communication startup.

The wakeup procedure tolerates any number of nodes simultaneously trying to wakeup a single channel and resolves this situation such that only one node transmits the pattern. Additionally the wakeup pattern is collision resilient, so even in the presence of a fault causing two nodes to simultaneously transmit a wakeup pattern, the resulting collided signal can still wake the other nodes.

After wakeup the CC returns to READY state and signals the change of the wakeup status to the Host by setting flag FLXnSIR.FLXnWSTF. The wakeup status vector can be read from FLXnCCSV.FLXnWSV[2:0]. If a valid wakeup pattern was received also either flag FLXnSIR.FLXnWUPAF or flag FLXnSIR.FLXnWUPBF is set.

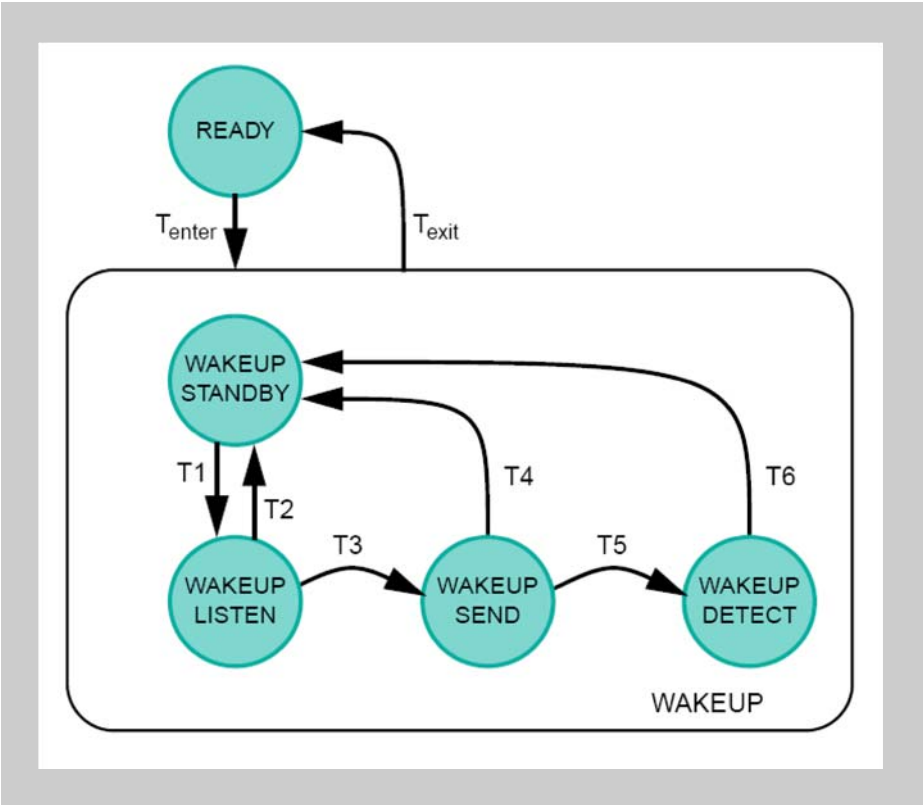


Figure 30-6 Structure of POC state WAKEUP

Table 30-100 State transitions WAKEUP (1/2)

T#	Condition	From	To
enter	Host commands change to POC WAKEUP state by writing FLXnSUCC1.FLXnCMD[3:0] = "0011" (WAKEUP command)	READY	WAKEUP
1	CHI command WAKEUP triggers wakeup FSM to transit to WAKEUP_LISTEN state	WAKEUP_STANDBY	WAKEUP_LISTEN
2	Received WUP on wakeup channel selected by bit FLXnWUCS in the SUC Configuration Register 1 OR frame header on either available channel	WAKEUP_LISTEN	WAKEUP_STANDBY
3	Timer event	WAKEUP_LISTEN	WAKEUP_SEND
4	Complete, non-aborted transmission of wakeup pattern	WAKEUP_SEND	WAKEUP_STANDBY
5	Collision detected	WAKEUP_SEND	WAKEUP_DETECT

Table 30-100 State transitions WAKEUP (2/2)

T#	Condition	From	To
6	Wakeup timer expired OR WUP detected on wakeup channel selected by bit FLXnSUCC1.FLXnWUCS OR frame header received on either available channel	WAKEUP_DETECT	WAKEUP_STANDBY
exit	Wakeup completed (after T2 or T4 or T6) OR Host commands change to READY state by writing FLXnSUCC1.FLXnCMD[3:0] = "0010" (READY command). This command also resets the wakeup FSM to WAKEUP_STANDBY state	WAKEUP	READY

The WAKEUP_LISTEN state is controlled by the wakeup timer and the wakeup noise timer. The two timers are controlled by the parameters listen timeout FLXnSUCC2.LT[20:0] and listen timeout noise FLXnSUCC2.FLXnLTN[3:0]. Listen timeout enables a fast cluster wakeup in case of a noise free environment, while listen timeout noise enables wakeup under more difficult conditions regarding noise interference.

In WAKEUP_SEND state the CC transmits the wakeup pattern on the configured channel and checks for collisions. After return from wakeup the Host has to bring the CC into STARTUP state by CHI command RUN.

In WAKEUP_DETECT state the CC attempts to identify the reason for the wakeup collision detected in WAKEUP_SEND state. The monitoring is bounded by the expiration of listen timeout as configured by FLXnSUCC2.LT[20:0]. Either the detection of a wakeup pattern indicating a wakeup attempt by another node or the reception of a frame header indicating ongoing communication, causes the direct transition to READY state. Otherwise WAKEUP_DETECT is left after expiration of listen timeout; in this case the reason for wakeup collision is unknown.

The Host has to be aware of possible failures of the wakeup and act accordingly. It is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal time it takes another coldstart node to become awake and to be configured.

The FlexRay Protocol Specification v2.1 recommends that two different CCs shall awake the two channels.

Host activities

The host must coordinate the wakeup of the two channels and must decide whether, or not, to wake a specific channel. The sending of the wakeup pattern is initiated by the Host. The wakeup pattern is detected by the remote BDs and signalled to their local Host.

Wakeup procedure controlled by Host (single-channel wakeup):

- Configure the CC in CONFIG state
  - Select wakeup channel by programming bit FLXnSUCC1.FLXnWUCS
- Check local BDs whether a WUP was received
- Activate BD of selected wakeup channel
- Command CC to enter READY state
- Command CC to start wakeup on the configured channel by writing FLXnSUCC1.FLXnCMD[3:0] = "0011"
  - CC enters WAKEUP
  - CC returns to READY state and signals status of wakeup attempt to the Host
- Wait predefined time to allow the other nodes to wakeup and configure

themselves

- Coldstart node:
  - In a dual channel cluster wait for WUP on the other channel
  - Reset coldstart inhibit flag FLXnCCSV.FLXnCSI by writing FLXnSUCC1.FLXnCMD[3:0] = "1001" (ALLOW_COLDSTART command)
- Command CC to enter startup by writing FLXnSUCC1.FLXnCMD[3:0] = "0100" (RUN command)

Wakeup procedure triggered by BD:

- Wakeup recognized by BD
- BD triggers power-up of Host (if required)
- BD signals wakeup event to Host
- Host configures its local CC
- If necessary, Host commands wakeup of second channel and waits predefined time to allow the other nodes to wakeup and configure themselves
- Host commands CC to enter STARTUP state by writing FLXnSUCC1.FLXnCMD[3:0] = "0100" (RUN command)

Wakeup pattern (WUP)

The wakeup pattern (WUP) is composed of at least two wakeup symbols (WUS). Wakeup symbol and wakeup pattern are configured by registers FLXnPRTC1 and FLXnPRTC2.

- Single channel wakeup, wakeup symbol may not be sent on both channels at the same time
- Wakeup symbol collision resilient for at least two sending nodes (two overlapping wakeup symbols always recognizable)
- Wakeup symbol must be configured identical in all nodes of a cluster
- Wakeup symbol transmit low time configured by FLXnPRTC2.FLXnTXL[5:0]
- Wakeup symbol idle time used to listen for activity on the bus, configured by FLXnPRTC2.FLXnTXI[7:0]
- A wakeup pattern composed of at least two Tx-wakeup symbols needed for wakeup
- Number of repetitions configurable by FLXnPRTC1.FLXnRWPFR[5:0] (2 to 63 repetitions)
- Wakeup symbol receive window length configured by FLXnPRTC1.FLXnRXW[8:0]
- Wakeup symbol receive low time configured by FLXnPRTC2.FLXnRXL[5:0]
- Wakeup symbol receive idle time configured by FLXnPRTC2.FLXnRXI[5:0]

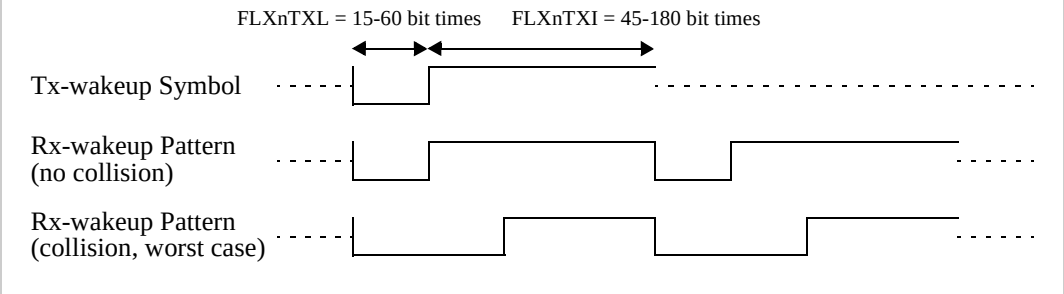


Figure 30-7 Timing of wakeup pattern

(7) STARTUP state

The description below is intended to help configuring startup for the E-Ray IP-module. A detailed description of the startup procedure together with the respective SDL diagrams can be found in the FlexRay protocol specification v2.1, section 7.2.

Any node entering STARTUP state that has coldstart capability should assure that both channels attached have been awakened before initiating coldstart.

It cannot be assumed that all nodes and stars need the same amount of time to become completely awake and to be configured. Since at least two nodes are necessary to start up the cluster communication, it is advisable to delay any potential startup attempt of the node having instigated the wakeup by the minimal amount of time it takes another coldstart node to become awake, to be configured and to enter startup. It may require several hundred milliseconds (depending on the hardware used) before all nodes and stars are completely awakened and configured.

Startup is performed on all channels synchronously. During startup, a node only transmits startup frames. Startup frames are both sync frames and null frames during startup.

A fault-tolerant, distributed startup strategy is specified for initial synchronization of all nodes. In general, a node may enter NORMAL_ACTIVE state via:

- Coldstart path initiating the schedule synchronization (leading coldstart node)
- Coldstart path joining other coldstart nodes (following coldstart node)
- Integration path integrating into an existing communication schedule (all other nodes)

A coldstart attempt begins with the transmission of a collision avoidance symbol (CAS). Only a coldstart node that had transmitted the CAS transmits frames in the first four cycles after the CAS, it is then joined firstly by the other coldstart nodes and afterwards by all other nodes.

A coldstart node has bits FLXnSUCC1.FLXnTXST and FLXnSUCC1.FLXnTXSY set to ’1’. Message buffer 0 holds the key slot ID which defines the slot number where the startup frame is send. In the frame header of the startup frame the startup frame indicator bit is set.

In clusters consisting of three or more nodes, at least three nodes shall be configured to be coldstart nodes. In clusters consisting of two nodes, both

nodes must be coldstart nodes. At least two fault-free coldstart nodes are necessary for the cluster to startup.

Each startup frame must also be a sync frame; therefore each coldstart node will also be a sync node. The number of coldstart attempts is configured by FLXnSUCC1.FLXnCSA[4:0].

A non-coldstart node requires at least two startup frames from distinct nodes for integration. It may start integration before the coldstart nodes have finished their startup. It will not finish its startup until at least two coldstart nodes have finished their startup.

Both non-coldstart nodes and coldstart nodes start passive integration via the integration path as soon as they receive sync frames from which to derive the TDMA schedule information. During integration, the node has to adapt its own clock to the global clock (rate and offset) and has to make its cycle time consistent with the global schedule observable at the network. Afterwards, these settings are checked for consistency with all available network nodes. The node can only leave the integration phase and actively participate in communication when these checks are passed.

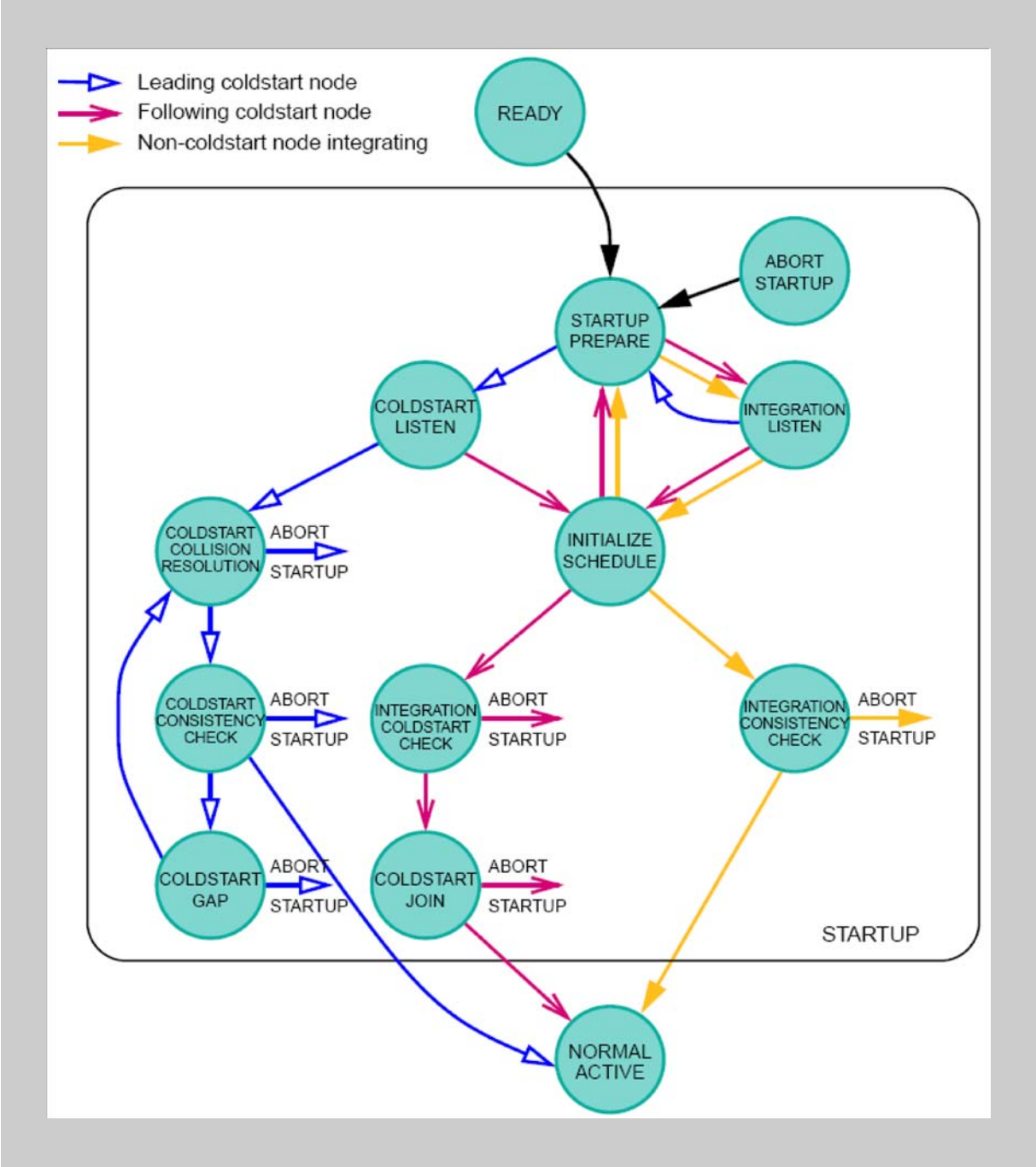


Figure 30-8 State diagram time-triggered startup

Coldstart Inhibit Mode

In coldstart inhibit mode the node is prevented from initializing the TDMA communication schedule. If bit FLXnCCSV.FLXnCSI is set, the node is not allowed to initialize the cluster communication, i.e. entering the coldstart path is prohibited. The node is allowed to integrate to a running cluster or to transmit startup frames after another coldstart node started the initialization of the cluster communication.

The coldstart inhibit bit FLXnCCSV.FLXnCSI is set whenever the POC enters READY state. The bit has to be cleared under control of the Host by CHI command ALLOW_COLDSTART (FLXnSUCC1.FLXnCMD[3:0] = "1001")

Startup Timeouts

The CC supplies two different µT timers supporting two timeout values, startup timeout and startup noise timeout. The two timers are reset when the CC enters the COLDSTART_LISTEN state. The expiration of either of these timers causes the node to leave the initial sensing phase (COLDSTART_LISTEN state) with the intention of starting up communication.

Note The startup and startup noise timers are identical with the wakeup and wakeup noise timers and use the same configuration values FLXnSUCC2.LT[20:0] and FLXnSUCC2.FLXnLTN[3:0].

Startup timeout

The startup timeout limits the listen time used by a node to determine if there is already communication between other nodes or at least one coldstart node actively requesting the integration of others. The startup timer is configured by programming FLXnSUCC2.LT[20:0] (see chapter SUC Configuration Register 2 (FLXnSUCC2)).

The startup timeout is: pdListenTimeout = FLXnSUCC2.LT[20:0]

The startup timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Both channels reaching idle state while in COLDSTART_LISTEN state

The startup timer is stopped:

- If communication channel activity is detected on one of the configured channels while the node is in the COLDSTART_LISTEN state
- When the COLDSTART_LISTEN state is left

Once the startup timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The timer status is kept for further processing by the startup state machine.

Startup noise timer

At the same time the startup timer is started for the first time (transition from STARTUP_PREPARE state to COLDSTART_LISTEN state), the startup noise timer is started. This additional timeout is used to improve reliability of the startup procedure in the presence of noise. The startup noise timeout is configured by programming FLXnSUCC2.FLXnLTN[3:0] (see chapter SUC Configuration Register 2 (FLXnSUCC2)).

The startup noise timeout is:
pdListenTimeout • gListenNoise =
FLXnSUCC2.FLXnLT[20:00] • (FLXnSUCC2.FLXnLTN[3:0] + 1)

The startup noise timer is restarted upon:

- Entering the COLDSTART_LISTEN state
- Reception of correctly decoded headers or CAS symbols while the node is in COLDSTART_LISTEN state

The startup noise timer is stopped when the COLDSTART_LISTEN state is left.

Once the startup noise timeout expires, neither an overflow nor a cyclic restart of the timer is performed. The status is kept for further processing by the startup state machine. Since the startup noise timer won't be restarted when random channel activity is sensed, this timeout defines the fall-back solution that guarantees that a node will try to start up the communication cluster even in the presence of noise.

Path of leading Coldstart Node (initiating coldstart)

When a coldstart node enters COLDSTART_LISTEN, it listens to its attached channels.

If no communication is detected, the node enters the COLDSTART_COLLISION_RESOLUTION state and commences a coldstart attempt. The initial transmission of a CAS symbol is succeeded by the first regular cycle. This cycle has the number zero.

From cycle zero on, the node transmits its startup frame. Since each coldstart node may perform a coldstart attempt, it may occur that several nodes simultaneously transmit the CAS symbol and enter the coldstart path. This situation is resolved during the first four cycles after CAS transmission.

As soon as a node that initiates a coldstart attempt receives a CAS symbol or a frame header during these four cycles, it re-enters the COLDSTART_LISTEN state. Thereby, only one node remains in this path. In cycle four, other coldstart nodes begin to transmit their startup frames.

After four cycles in COLDSTART_COLLISION_RESOLUTION state, the node that initiated the coldstart enters the COLDSTART_CONSISTENCY_CHECK state. It collects all startup frames from cycle four and five and performs the clock correction. If the clock correction does not deliver any errors and it has received at least one valid startup frame pair, the node leaves COLDSTART_CONSISTENCY_CHECK and enters NORMAL_ACTIVE state.

The number of coldstart attempts that a node is allowed to perform is configured by FLXnSUCC1.FLXnCSA[4:0]. The number of remaining coldstarts attempts can be read from FLXnCCSV.FLXnRCA[4:0]. The number of remaining coldstart attempts is reduced by one for each attempted coldstart. A node may enter the COLDSTART_LISTEN state only if this value is larger than one and it may enter the COLDSTART_COLLISION_RESOLUTION state only if this value is larger than zero. If the number of coldstart attempts is one, coldstart is inhibited but integration is still possible.

Path of following Coldstart Node (responding to leading Coldstart Node)

When a coldstart node enters the COLDSTART_LISTEN state, it tries to receive a valid pair of startup frames to derive its schedule and clock correction from the leading coldstart node.

As soon as a valid startup frame has been received the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_COLDSTART_CHECK state is entered.

In INTEGRATION_COLDSTART_CHECK state it is assured that the clock correction can be performed correctly and that the coldstart node from which this node has initialized its schedule is still available. The node collects all sync frames and performs clock correction in the following double-cycle. If clock correction does not signal any errors and if the node continues to receive sufficient frames from the same node it has integrated on, the COLDSTART_JOIN state is entered.

In COLDSTART_JOIN state integrating coldstart nodes begin to transmit their own startup frames. Thereby the node that initiated the coldstart and the nodes joining it can check if their schedules agree to each other. If for the following three cycles the clock correction does not signal errors and at least one other coldstart node is visible, the node leaves COLDSTART_JOIN state and enters NORMAL_ACTIVE state. Thereby it leaves STARTUP at least one cycle after the node that initiated the coldstart.

Path of Non-coldstart Node

When a non-coldstart node enters the INTEGRATION_LISTEN state, it listens to its attached channels.

As soon as a valid startup frame has been received, the INITIALIZE_SCHEDULE state is entered. If the clock synchronization can successfully receive a matching second valid startup frame and derive a schedule from this, the INTEGRATION_CONSISTENCY_CHECK state is entered.

In INTEGRATION_CONSISTENCY_CHECK state the node verifies that the clock correction can be performed correctly and that enough coldstart nodes (at least 2) are sending startup frames that agree with the node's own schedule. Clock correction is activated, and if any errors are signalled, the integration attempt is aborted.

During the first even cycle in this state, either two valid startup frames or the startup frame of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

During the first double-cycle in this state, either two valid startup frame pairs or the startup frame pair of the node that this node has integrated on must be received; otherwise the node aborts the integration attempt.

If after the first double-cycle less than two valid startup frames are received within an even cycle, or less than two valid startup frame pairs are received within a double-cycle, the startup attempt is aborted.

Nodes in this state need to see two valid startup frame pairs for two consecutive double-cycles each to be allowed to leave STARTUP and enter NORMAL_OPERATION. Consequently, they leave startup at least one double-cycle after the node that initiated the coldstart and only at the end of a cycle with an odd cycle number.

(8) NORMAL_ACTIVE state

As soon as the node that transmitted the first CAS symbol (resolving the potential access conflict and entering STARTUP via coldstart path) and one additional node have entered the NORMAL_ACTIVE state, the startup phase for the cluster has finished. In the NORMAL_ACTIVE state, all configured messages are scheduled for transmission. This includes all data frames as well as the sync frames. Rate and offset measurement is started in all even cycles (even / odd cycle pairs required).

In NORMAL_ACTIVE state the CC supports regular communication functions

- The CC performs transmissions and reception on the FlexRay bus as configured
- Clock synchronization is running
- The Host interface is operational

The CC exits from that state to

- HALT state by writing FLXnSUCC1.FLXnCMD[3:0] = "0110" (HALT command, at the end of the current cycle)
- HALT state by writing FLXnSUCC1.FLXnCMD[3:0] = "0111" (FREEZE command, immediately)
- HALT state due to change of the error state from ACTIVE to COMM_HALT
- NORMAL_PASSIVE state due to change of the error state from ACTIVE to PASSIVE
- READY state by writing FLXnSUCC1.FLXnCMD[3:0] = "0010" (READY command)

(9) NORMAL_PASSIVE state

NORMAL_PASSIVE state is entered from NORMAL_ACTIVE state when the error state changes from ACTIVE to PASSIVE.

In NORMAL_PASSIVE state, the node is able to receive all frames (node is fully synchronized and performs clock synchronization). Contrary to the NORMAL_ACTIVE state, the node does not actively participate in communication, i.e. neither symbols nor frames are transmitted.

In NORMAL_PASSIVE state

- The CC performs reception on the FlexRay bus
- The CC does not transmit any frames or symbols on the FlexRay bus
- Clock synchronization is running
- The Host interface is operational

The CC exits from this state to

- HALT state by writing FLXnSUCC1.FLXnCMD[3:0] = "0110" (HALT command, at the end of the current cycle)
- HALT state by writing FLXnSUCC1.FLXnCMD[3:0] = "0111" (FREEZE command, immediately)
- HALT state due to change of the error state from PASSIVE to COMM_HALT
- NORMAL_ACTIVE state due to change of the error state from PASSIVE to ACTIVE. The transition takes place when FLXnCCEV.FLXnPTAC[4:0] equals FLXnSUCC1.FLXnPTA[4:0] - 1
- To READY state by writing FLXnSUCC1.FLXnCMD[3:0] = "0010" (READY command)

(10) HALT state

In this state all communication (reception and transmission) is stopped.

The CC enters this state

- By writing FLXnSUCC1.FLXnCMD[3:0] = "0110" (HALT command) while the CC is in NORMAL_ACTIVE or NORMAL_PASSIVE state
- By writing FLXnSUCC1.FLXnCMD[3:0] = "0111" (FREEZE command) from all states
- When exiting from NORMAL_ACTIVE state because the clock correction failed counter reached the "maximum without clock correction fatal" limit and FLXnSUCC1.FLXnHCSE is set
- When exiting from NORMAL_PASSIVE state because the clock correction failed counter reached the "maximum without clock correction fatal" limit and FLXnSUCC1.FLXnHCSE is set

The CC exits from this state to DEFAULT_CONFIG state

- By writing FLXnSUCC1.FLXnCMD[3:0] = "0001" (CONFIG command)

When the CC enters HALT state, all configuration and status data is maintained for analysing purposes.

When the Host writes FLXnSUCC1.FLXnCMD[3:0] = "0110" (HALT command), the CC sets bit FLXnCCSV.FLXnHRQ and enters HALT state at the next end of cycle.

When the Host writes FLXnSUCC1.FLXnCMD[3:0] = "0111" (FREEZE command), the CC enters HALT state immediately and sets bit FLXnCCSV.FLXnFSI.

The POC state from which the transition to HALT state took place can be read from FLXnCCSV.FLXnPSL[5:0].

30.4.6 Network management

The accrued Network Management (NM) vector can be read from registers NMV13. The CC performs a bit-wise OR operation over all NM vectors out of all received valid NM frames with the Payload Preamble Indicator (FLXnRHPPI) bit set. Only static frames may be configured to hold NM information. The CC updates the NM vector at the end of each cycle.

The length of the NM vector can be configured from 0 to 12 bytes by FLXnNEMC.FLXnNML[3:0]. The NM vector length must be configured identically in all nodes of a cluster.

To configure a transmit buffer to send FlexRay frames with the FLXnRHPPI bit set, bit FLXnWHPPIT in the header section of the respective transmit buffer has to be set via FLXnWRHS1.FLXnWHPPIT. In addition the Host has to write the NM information to the data section of the respective transmit buffer.

The evaluation of the NM vector has to be done by the application running on the Host.

Note In case a message buffer is configured for transmission / reception of network management frames, the payload length configured in header 2 of that message buffer should be equal or greater than the length of the NM vector configured by FLXnNEMC.FLXnNML[3:0].
When the CC transits to HALT state, the cycle count is not incremented and therefore the NM vector is not updated. In this case NMV1…3 holds the value from the cycle before.

30.4.7 Filtering and masking

Filtering is done by comparison of the configuration of assigned message buffers against actual slot and cycle counter values and channel ID (channel A, B). A message buffer is only updated / transmitted if the required matches occur.

Filtering is done on the following fields:

- Channel ID
- Frame ID
- Cycle Counter

The following filter combinations for acceptance / transmit filtering are allowed:

- Frame ID + Channel ID
- Frame ID + Channel ID + Cycle Counter

All configured filters must match in order to store a received message in a message buffer.

Note For the FIFO the acceptance filter is configured by the FIFO Rejection Filter and the FIFO Rejection Filter Mask.

A message will be transmitted in the time slot corresponding to the configured frame ID on the configured channel(s). If cycle counter filtering is enabled the configured cycle filter value must also match.

(1) Slot counter filtering

Every transmit and receive buffer contains a frame ID stored in the header section. This frame ID is compared against the actual slot counter value in order to assign receive and transmit buffers to the corresponding slot.

If two or more message buffers are configured with the same frame ID, and if they have a matching cycle counter filter value for the same slot, then the message buffer with the lowest message buffer number is used.

(2) Cycle counter filtering

Cycle counter filtering is based on the notion of a cycle set. For filtering purposes, a match is detected if any one of the elements of the cycle set is matched. The cycle set is defined by the cycle code field in header section 1 of each message buffer.

If message buffer 0 resp. 1 is configured to hold the startup / sync frame or the single slot frame by bits FLXnSUCC1.FLXnTXST, FLXnSUCC1.FLXnTXSY, and FLXnSUCC1.FLXnTSM, cycle counter filtering for message buffer 0 resp. 1 must be disabled.

Note Sharing of a static time slot via cycle counter filtering between different nodes of a FlexRay network is **not** allowed.

The set of cycle numbers belonging to a cycle set is determined as described in *Table 30-101 "Definition of cycle set"*.

Table 30-101 Definition of cycle set

Cycle Code	Matching Cycle Counter Values		
0b000000x	all Cycles		
0b000001c	every second Cycle	at (Cycle Count)mod2	= c
0b00001cc	every fourth Cycle	at (Cycle Count)mod4	= cc
0b0001ccc	every eighth Cycle	at (Cycle Count)mod8	= ccc
0b001cccc	every sixteenth Cycle	at (Cycle Count)mod16	= cccc
0b01ccccc	every thirty-second Cycle	at (Cycle Count)mod32	= ccccc
0b1cccccc	every sixty-fourth Cycle	at (Cycle Count)mod64	= cccccc

Table 30-102 "Examples for valid cycle sets" below gives some examples for valid cycle sets to be used for cycle counter filtering:

Table 30-102 Examples for valid cycle sets

Cycle Code	Matching Cycle Counter Values
0b0000011	1-3-5-7- .... -63 ↵
0b0000100	0-4-8-12- .... -60 ↵
0b0001110	6-14-22-30- .... -62 ↵
0b0011000	8-24-40-56 ↵
0b0100011	3-35 ↵
0b1001001	9 ↵

The received message is stored only if the cycle counter value of the cycle during which the message is received matches an element of the receive buffer's cycle set. Other filter criteria must also be met.

The content of a transmit buffer is transmitted on the configured channel(s) when an element of the cycle set matches the current cycle counter value. Other filter criteria must also be met.

(3) Channel ID filtering

There is a 2-bit channel filtering field (FLXnWHCHA, FLXnWHCHB, FLXnRHCHA, FLXnRHCHB) located in the header section of each message buffer in the Message RAM. It serves as a filter for receive buffers, and as a control field for transmit buffers (see *Table 30-103 "Channel filtering configuration"*).

Table 30-103 Channel filtering configuration

FLXnWHCHA/ FLXnRHCHA	FLXnWHCHB/ FLXnRHCHB	Transmit Buffer transmit frame	Receive Buffer store valid receive frame
1	1	on both channels (static segment only)	received on channel A or B (store first semantically valid frame, static segment only)
1	0	on channel A	received on channel A
0	1	on channel B	received on channel B
0	0	no transmission	ignore frame

The contents of a transmit buffer is transmitted on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a transmit buffer may be set up for transmission on both channels (FLXnWHCHA and FLXnWHCHB set).

Valid received frames are stored if they are received on the channels specified in the channel filtering field when the slot counter filtering and cycle counter filtering criteria are also met. Only in static segment a receive buffer may be setup for reception on both channels (FLXnWHCHA and FLXnRHCHB set).

Note If a message buffer is configured for the dynamic segment and both bits of the channel filtering field are set to '1', no frames are transmitted resp. received frames are ignored (same function as FLXnWHCHA = FLXnWHCHB = '0' respectively FLXnRHCHA = FLXnRHCHB = '0').

(4) FIFO filtering

For FIFO filtering there is one rejection filter and one rejection filter mask available. The FIFO filter consists of channel filter FLXnFRF.CH[1:0], frame ID filter FLXnFRF.FLXnFID[10:00], and cycle counter filter FLXnFRF.FLXnCYF[6:0]. Registers FLXnFRF and FLXnFRFM can be configured in DEFAULT_CONFIG or CONFIG state only. The filter configuration in the header section of message buffers belonging to the FIFO is ignored.

The 7-bit cycle counter filter determines the cycle set to which frame ID and channel rejection filter are applied. In cycles not belonging to the cycle set specified by FLXnFRF.FLXnCYF[6:0], all frames are rejected.

A valid received frame is stored in the FIFO if channel ID, frame ID, and cycle counter are not rejected by the configured rejection filter and rejection filter mask, and if there is no matching dedicated receive buffer.

30.4.8 Transmit process

(1) Static segment

For the static segment, if there are several messages pending for transmission, the message with the frame ID corresponding to the next sending slot is selected for transmission.

The data section of transmit buffers assigned to the static segment can be updated until the end of the preceding time slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

(2) Dynamic segment

In the dynamic segment, if several messages are pending, the message with the highest priority (lowest frame ID) is selected next. In the dynamic segment different slot counter sequences on channel A and channel B are possible (concurrent sending of different frame IDs on both channels).

The data section of transmit buffers assigned to the dynamic segment can be updated until the end of the preceding slot. This means that a transfer from the Input Buffer has to be started by writing to the Input Buffer Command Request register latest at this time.

The start of latest transmit configured by FLXnMHDC.FLXnSLT[12:00] defines the maximum minislot value allowed before inhibiting new frame transmission in the dynamic segment of the current cycle.

(3) Transmit buffers

E-Ray message buffers can be configured as transmit buffers by programming bit FLXnWHCFG in the header section of the respective message buffer to ’1’ via FLXnWRHS1.

There exist the following possibilities to assign a transmit buffer to the CC channels:

- Static segment: channel A or channel B, channel A and channel B
- Dynamic segment: channel A or channel B

Message buffer 0 resp. 1 is dedicated to hold the startup frame, the sync frame, or the designated single slot frame as configured by FLXnSUCC1.FLXnTXST, FLXnSUCC1.FLXnTXSY, and FLXnSUCC1.FLXnTSM. In this case, it can be reconfigured in DEFAULT_CONFIG or CONFIG state only. This ensures that any node transmits at most one startup / sync frame per communication cycle. Transmission of startup / sync frames from other message buffers is not possible.

All other message buffers configured for transmission in static or dynamic segment are reconfigurable during runtime depending on the configuration of FLXnMRC.FLXnSEC[1:0] (see 5.11.1 Reconfiguration of Message Buffers). Due to the organization of the data partition in the Message RAM (reference by data pointer), reconfiguration of the configured payload length and the data pointer in the header section of a message buffer may lead to erroneous configurations.

If a message buffer is reconfigured (header section updated) during runtime, it may happen that this message buffer is not send out in the respective communication cycle.

The CC does not have the capability to calculate the header CRC. The Host is supposed to provide the header CRCs for all transmit buffers. If network

management is required, the Host has to set the FLXnWHPPIT bit in the header section of the respective message buffer to ’1’ and write the network management information to the data section of the message buffer.

The payload length field configures the payload length in 2-byte words. If the configured payload length of a static transmit buffer is shorter than the payload length configured for the static segment by FLXnMHDC.FLXnSFDL[6:0], the CC generates padding bytes to ensure that frames have proper physical length. The padding pattern is logical zero.

Note In case of an odd payload length (FLXnPLC = 1,3,5,...) the application has to write zero to the last 16 bit of the message buffers data section to ensure that the padding pattern is all zero.

Each transmit buffer provides a transmission mode flag FLXnWHTXM that allows the Host to configure the transmission mode for the transmit buffer. If this bit is set, the transmitter operates in the single-shot mode. If this bit is cleared, the transmitter operates in the continuous mode.

In single-shot mode the CC resets the respective FLXnTXR flag after transmission has completed. Now the Host may update the transmit buffer.

In continuous mode, the CC does not reset the respective transmission request flag FLXnTXR after successful transmission. In this case a frame is sent out each time the filter criteria match. The FLXnTXR flag can be reset by the Host by writing the respective message buffer number to the FLXnIBCR register while bit FLXnIBCM.FLXnSTXRH is set to ’0’.

If two or more transmit buffers meet the filter criteria simultaneously, the transmit buffer with the lowest message buffer number will be transmitted in the respective slot.

(4) Frame transmission

The following steps are required to prepare a message buffer for transmission:

- Configure the transmit buffer in the Message RAM via FLXnWRHS1, FLXnWRHS2, and FLXnWRHS3
- Write the data section of the transmit buffer via FLXnWRDSm
- Transfer the configuration and message data from Input Buffer to the Message RAM by writing the number of the target message buffer to register FLXnIBCR
- If configured in register FLXnIBCM, the transmission request flag FLXnTXR for the respective message buffer will be set as soon as the transfer has completed, and the message buffer is ready for transmission.
- Check whether the message buffer has been transmitted by checking the respective FLXnTXR bit (FLXnTXR = ’0’) in the FLXnTRXQ1/2/3/4 registers (single-shot mode only).

After transmission has completed, the respective FLXnTXR flag in the FLXnTXRQ1/2/3/4 register is reset (single-shot mode), and, if bit FLXnWHMBI in the header section of the message buffer is set, flag FLXnSIR.FLXnTXIF is set to ’1’. If enabled, an interrupt is generated.

(5) Null frame transmission

If in static segment the Host does not set the transmission request flag before transmit time, and if there is no other transmit buffer with matching filter criteria, the CC transmits a null frame with the null frame indication bit **set to ’0’** and the payload data **set to zero**.

In the following cases the CC transmits a null frame:

- If the message buffer with the lowest message buffer number matching the filter criteria does not have its transmission request flag set (FLXnTXR = ’0’).
- No transmit buffer configured for the slot has a cycle counter filter that matches the current cycle. In this case, no message buffer status MBS is updated.

Null frames are not transmitted in the dynamic segment.

30.4.9 Receive process

(1) Dedicated Receive buffers

A portion of the E-Ray message buffers can be configured as dedicated receive buffers by programming bit FLXnRHCFG in the header section of the respective message buffer to ’0’ via FLXnWRHS1.

The following possibilities exist to assign a receive buffer to the CC channels:

- Static segment: channel A or channel B, channel A and channel B (the CC stores the first semantically valid frame)
- Dynamic segment: channel A or channel B

The CC transfers the payload data of valid received messages from the shift register of the FlexRay channel protocol controller (channel A or B) to the receive buffer with the matching filter configuration. A receive buffer stores all frame elements except the frame CRC.

All message buffers configured for reception in static or dynamic segment are reconfigurable during runtime depending on the configuration of FLXnMRC.FLXnSEC[1:0] Buffers). If a message buffer is reconfigured (header section updated) during runtime it may happen that in the respective communication cycle a received message is lost.

If two or more receive buffers meet the filter criteria simultaneously, the receive buffer with the lowest message buffer number is updated with the received message.

(2) Frame reception

The following steps are required to prepare a dedicated message buffer for reception:

- Configure the receive buffer in the Message RAM via FLXnWRHS1, FLXnWRHS2, and FLXnWRHS3
- Transfer the configuration from Input Buffer to the Message RAM by writing the number of the target message buffer to register FLXnIBCR

Once these steps are performed, the message buffer functions as an active receive buffer and participates in the internal acceptance filtering process which takes place every time the CC receives a message. The first matching receive buffer is updated from the received message.

If a valid payload segment was stored in the data section of a message buffer, the respective FLXnND flag in the FLXnNDAT1/2/3/4 registers is set, and, if bit FLXnRHMBI in the header section of that message buffer is set, flag FLXnSIR.FLXnRXIF is set to ’1’. If enabled, an interrupt is generated.

In case that bit FLXnND was already set when the Message Handler updates the message buffer, bit FLXnMBS.FLXnMBSMLST of the respective message buffer is set and the unprocessed message data is lost.

If no frame, a null frame, or a corrupted frame was received in a slot, the data section of the message buffer configured for this slot is not updated. In this case only the respective message buffer status FLXnMBS is updated.

When the Message Handler changed the message buffer status FLXnMBS in the header section of a message buffer, the respective FLXnMBC flag in the FLXnMBSC1/2/3/4 registers is set, and if bit FLXnRHMBI in the header section of that message buffer is set, flag FLXnSIR.FLXnMBSIF is set to ’1’. If enabled an interrupt is generated.

If the payload length of a received frame FLXnRHPLR[6:0] is longer than the value programmed by FLXnRHPLC[6:0] in the header section of the respective message buffer, the data field stored in the message buffer is truncated to that length.

To read a receive buffer from the Message RAM via the Output Buffer, proceed as described in chapter Data Transfer from Message RAM to Output Buffer.

Note The FLXnND and FLXnMBC flags are automatically cleared by the Message Handler when the payload data and the header of a received message have been transferred to the Output Buffer, respectively.

(3) Null frame reception

The payload segment of a received null frame is not copied into the matching dedicated receive buffer. If a null frame has been received, only the message buffer status FLXnMBS of the matching message buffer is updated from the received null frame. All bits in header 2 and 3 of the matching message buffer remain unchanged. They are updated from received data frames only.

When the Message Handler changed the message buffer status FLXnMBS in the header section of a message buffer, the respective FLXnMBC flag in the FLXnMBSC1/2/3/4 register is set, and if bit FLXnRHMBI in the header section of that message buffer is set, flag FLXnSIR.FLXnMBSIF is set to ’1’. If enabled, an interrupt is generated.

30.4.10 FIFO function

(1) Description

A group of the message buffers can be configured as a cyclic First-In-First-Out (FIFO) buffer. The group of message buffers belonging to the FIFO is contiguous in the register map starting with the message buffer referenced by FLXnMRC.FLXnFFB[7:0] and ending with the message buffer referenced by FLXnMRC.FLXnLCB[7:0]. Up to 127 message buffers can be assigned to the FIFO.

Every valid incoming message not matching with any dedicated receive buffer but passing the programmable FIFO filter is stored into the FIFO. In this case frame ID, payload length, receive cycle count, and the message buffer status FLXnMBS of the addressed FIFO message buffer are overwritten with frame ID, payload length, receive cycle count, and the status from the received frame. Bit FLXnSIR.FLXnRFNEF shows that the FIFO is not empty, bit FLXnSIR.FLXnRFCLF is set when the receive FIFO fill level FLXnFSR.FLXnRFFL[7:0] is equal or greater than the critical level as configured by FLXnFCL.CL[7:0], bit FLXnEIR.FLXnRFOE shows that a FIFO overrun has been detected. If enabled, interrupts are generated.

If null frames are not rejected by the FIFO rejection filter, the null frames will be treated like data frames when they are stored into the FIFO.

There are two index registers associated with the FIFO. The PUT Index Register (PIDX) is an index to the next available location in the FIFO. When a new message has been received it is written into the message buffer addressed by the PIDX register. The PIDX register is then incremented and addresses the next available message buffer. If the PIDX register is incremented past the highest numbered message buffer of the FIFO, the PIDX register is loaded with the number of the first (lowest numbered) message buffer in the FIFO chain. The GET Index Register (GIDX) is used to address the next message buffer of the FIFO to be read. The GIDX register is incremented after transfer of the contents of a message buffer belonging to the FIFO to the Output Buffer. The PUT Index Register and the GET Index Register are not accessible by the Host.

The FIFO is completely filled when the PUT index (PIDX) reaches the value of the GET index (GIDX). When the next message is written to the FIFO before the oldest message has been read, both PUT index and GET index are incremented and the new message overwrites the oldest message in the FIFO. This will set FIFO overrun flag FLXnEIR.FLXnRFOE.

A FIFO non empty status is detected when the PUT index (PIDX) differs from the GET index (GIDX). In this case flag FLXnSIR.FLXnRFNEF is set. This indicates that there is at least one received message in the FIFO. The FIFO empty, FIFO not empty, and the FIFO overrun states are explained in *Figure 30-9 “FIFO status: empty, not empty, overrun”* for a three message buffer FIFO.

The programmable FIFO Rejection Filter (FRF) defines a filter pattern for messages to be rejected. The FIFO filter consists of channel filter, frame ID filter, and cycle counter filter. If bit FLXnFRF.FLXnRSS is set to ’1’ (default), all messages received in the static segment are rejected by the FIFO. If bit FLXnFRF.FLXnRSS is set to ’1’ (default), received null frames are not stored in the FIFO.

The FIFO Rejection Filter Mask (FRFM) specifies which bits of the frame ID filter in the FIFO Rejection Filter register are marked ’don’t care’ for rejection filtering.

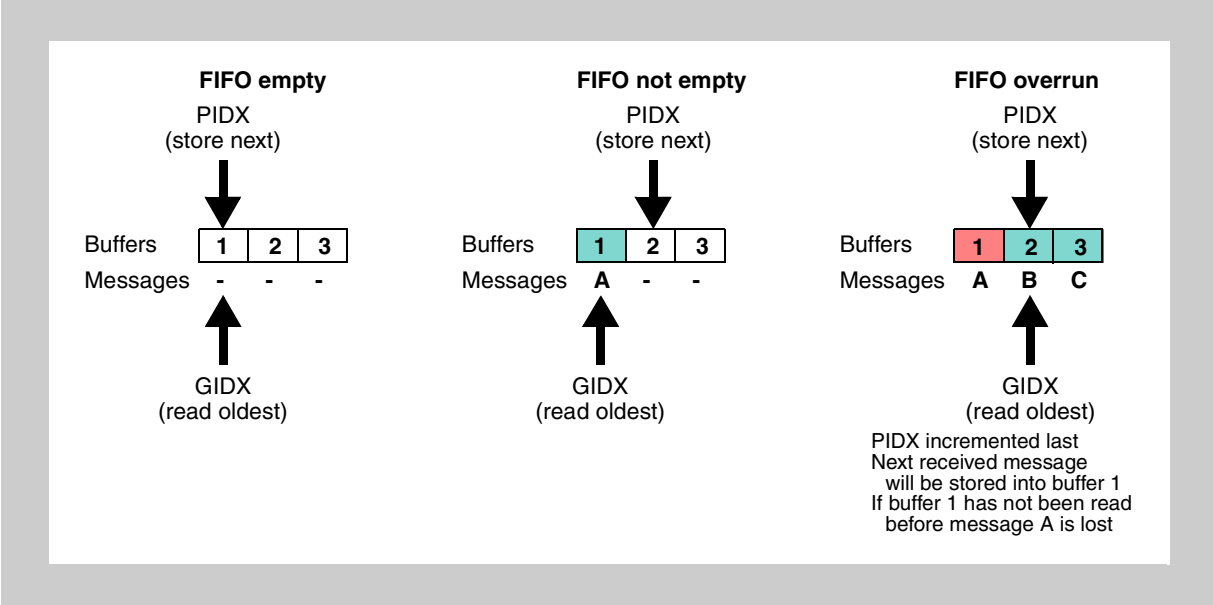


Figure 30-9 FIFO status: empty, not empty, overrun

(2) Configuration of the FIFO

(Re)configuration of message buffers belonging to the FIFO is only possible when the CC is in DEFAULT_CONFIG or CONFIG state. While the CC is in DEFAULT_CONFIG or CONFIG state, the FIFO function is not available.

For all message buffers belonging to the FIFO the payload length configured should be programmed to the same value via FLXnWRHS2.FLXnWHPLC[6:0]. The data pointer to the first 32-bit word of the data section of the respective message buffer in the Message RAM has to be configured via FLXnWRHS3.FLXnWHDP[10:00].

All information required for acceptance filtering is taken from the FIFO rejection filter and the FIFO rejection filter mask. The values configured in the header sections of the message buffers belonging to the FIFO are, with exception of FLXnWHDP and FLXnWHPLC, irrelevant.

Note It is recommended to program the FLXnWHMBI bits of the message buffers belonging to the FIFO to '0' via FLXnWRHS1.FLXnWHMBI to avoid generation of RX interrupts.
If the payload length of a received frame is longer than the value programmed by FLXnWRHS2.FLXnWHPLC[6:0] in the header section of the respective message buffer, the data field stored in a message buffer of the FIFO is truncated to that length.

(3) Access to the FIFO

For FIFO access outside DEFAULT_CONFIG and CONFIG state, the Host has to trigger a transfer from the Message RAM to the Output Buffer by writing the number of the first message buffer of the FIFO (referenced by FLXnMRC.FLXnFFB[7:0]) to the register FLXnOBCR. The Message Handler then transfers the message buffer addressed by the GET Index Register (GIDX) to the Output Buffer. After this transfer the GET Index Register (GIDX) is incremented.

30.4.11 Message handling

The Message Handler controls data transfers between the Input / Output Buffer and the Message RAM and between the Message RAM and the two Transient Buffer RAMs. All accesses to the internal RAMs are 32+7 bit accesses. The additional 7 bits are for the ECC code.

Access to the message buffers stored in the Message RAM is done under control of the Message Handler state machine. This avoids conflicts between accesses of the two FlexRay channel protocol controllers and the Host to the Message RAM.

Frame IDs of message buffers assigned to the static segment have to be in the range from 1 to FLXnGTUC07.NSS[9:0]. Frame IDs of message buffers assigned to the dynamic segment have to be in the range from FLXnGTUC07.NSS[9:0] + 1 to 2047.

Received messages with no matching dedicated receive buffer (static or dynamic segment) are stored in the receive FIFO (if configured) if they pass the FIFO rejection filter.

(1) Reconfiguration of Message Buffers

In case that an application needs to operate with more than 128 different messages, static and dynamic message buffers may be reconfigured during FlexRay operation. This is done by updating the header section of the respective message buffer via Input Buffer registers FLXnWRHS1...3.

Reconfiguration has to be enabled via control bits FLXnMRC.FLXnSEC[1:0] in the Message RAM Configuration register.

If a message buffer has not been transmitted / updated from a received frame before reconfiguration starts, the respective message is lost.

The point in time when a reconfigured message buffer is ready for transmission / reception according to the reconfigured frame ID depends on the actual state of the slot counter when the update of the header section has completed. Therefore it may happen that a reconfigured message buffer is not transmitted / updated from a received frame in the cycle where it was reconfigured.

The Message RAM is scanned according to Table 16 below:

Start of Scan in Slot	Scan for Slots
1	2...15, 1 (next cycle)
8	6...23, 1 (next cycle)
16	24...31, 1 (next cycle)
24	32...39, 1 (next cycle)
...	...

Table 30-104 Scan of Message RAM

A Message RAM scan is terminated with the start of NIT regardless whether it has completed or not. The scan of the Message RAM for slots 2 to 15 starts at the beginning of slot 1 of the actual cycle. The scan of the Message RAM for slot 1 is done in the cycle before by checking in parallel to each scan of the Message RAM whether there is a message buffer configured for slot 1 of the next cycle.

The number of the first dynamic message buffer is configured by FLXnMRC.FLXnFDB[7:0]. In case aMessage RAM scan starts while the CC is

in dynamic segment, the scan starts with the message buffer number configured by FLXnMRC.FLXnFDB[7:0].

In case a message buffer should be reconfigured to be used in slot 1 of the next cycle, the following has to be considered:

- If the message buffer to be reconfigured for slot 1 is part of the "Static Buffers", it will only be found if it is reconfigured before the last Message RAM scan in the static segment of the actual cycle evaluates this message buffer.
- If the message buffer to be reconfigured for slot 1 is part of the "Static + Dynamic Buffers", it will be found if it is reconfigured before the last Message RAM scan in the actual cycle evaluates this message buffer.
- The start of NIT terminates the Message RAM scan. In case the Message RAM scan has not evaluated the reconfigured message buffer until this point in time, the message buffer will not be considered for the next cycle.

Note Reconfiguration of message buffers may lead to the loss of messages and therefore has to be used very carefully. In worst case (reconfiguration in consecutive cycles) it may happen that a message buffer is never transmitted / updated from a received frame.

(2) Host access to Message RAM

The message transfer between Input Buffer and Message RAM as well as between Message RAM and Output Buffer is triggered by the Host by writing the number of the target / source message buffer to be accessed to FLXnIBCR or FLXnOBCR register.

The FLXnIBCM and FLXnOBCM registers can be used to write / read header and data section of the selected message buffer separately.

If bit FLXnIBCM.STXR is set to = ’1’, the transmission request flag FLXnTXR of the selected message buffer is automatically set after the message buffer has been updated. If bit FLXnIBCM.STXR is reset to ’0’, the transmission request flag FLXnTXR of the selected message buffer is reset. This can be used to stop transmission from message buffers operated in continuous mode.

Input Buffer (IBF) and Output Buffer (OBF) are build up as a double buffer structure. One half of this double buffer structure is accessible by the Host (IBF Host / OBF Host), while the other half (IBF Shadow / OBF Shadow) is accessed by the Message Handler for data transfers between IBF / OBF and Message RAM.

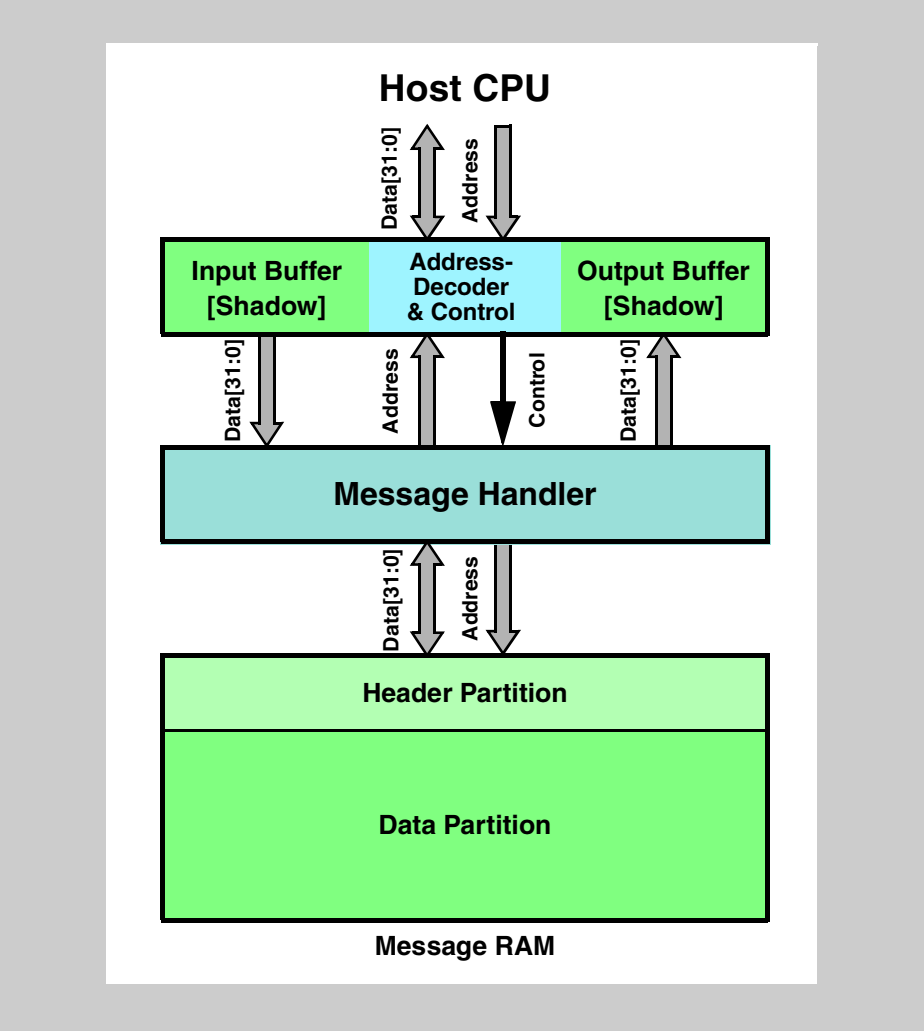


Figure 30-10 Host access to Message RAM

Data Transfer from Input Buffer to Message RAM

To configure / update a message buffer in the Message RAM, the Host has to write the data to FLXnWRDSm and the header to FLXnWRHS1…3. The specific action is selected by configuring the Input Buffer Command Mask FLXnIBCM.

When the Host writes the number of the target message buffer in the Message RAM to FLXnIBCR.FLXnIBRH[6:0], IBF Host and IBF Shadow are swapped (see *Figure 30-11 “Double buffer structure Input Buffer”*).

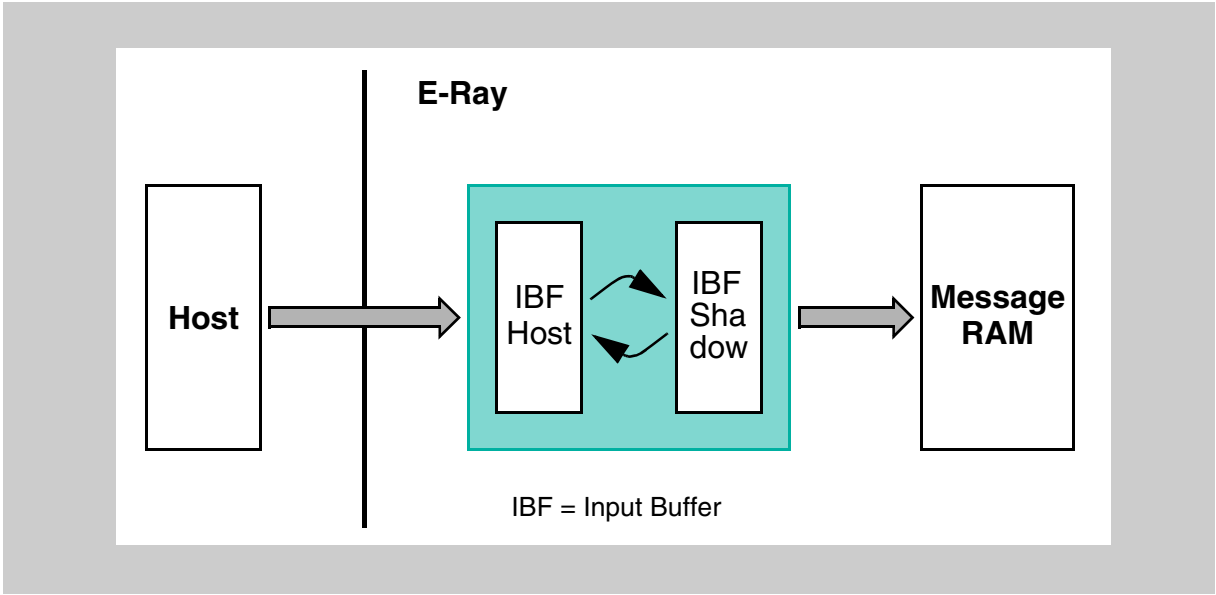


Figure 30-11 Double buffer structure Input Buffer

In addition the bits in the FLXnIBCM and FLXnIBCR registers are also swapped to keep them attached to the respective IBF section.

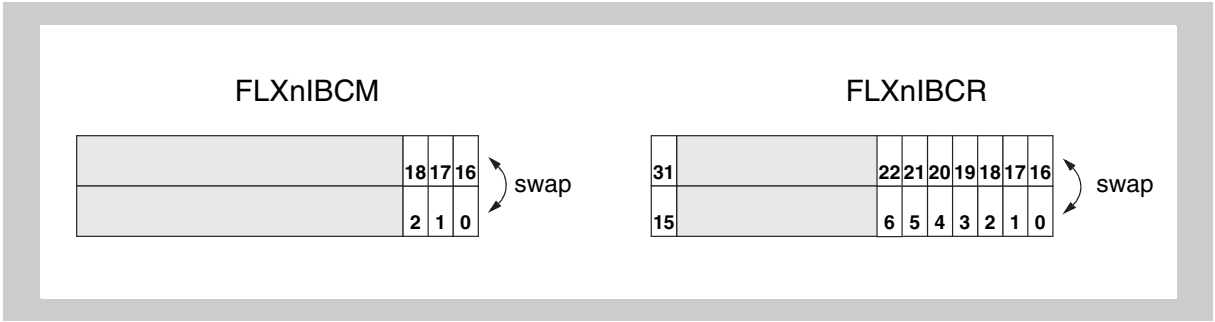


Figure 30-12 Swapping of FLXnIBCM and FLXnIBCR bits

With this write operation bit FLXnIBCR.FLXnIBSYS is set to ’1’. The Message Handler then starts to transfer the contents of IBF Shadow to the message buffer in the MessageRAMselected by FLXnIBCR.FLXnIBRS[6:0].

While the Message Handler transfers the data from IBF Shadow to the target message buffer in the Message RAM, the Host may write the next message to IBF Host. After the transfer between IBF Shadow and the Message RAM has completed, bit FLXnIBCR.FLXnIBSYS is set back to ’0’ and the next transfer to the Message RAM may be started by the Host by writing the respective target message buffer number to FLXnIBCR.FLXnIBRH[6:0].

If a write access to FLXnIBCR.FLXnIBRH[6:0] occurs while FLXnIBCR.FLXnIBSYS is ’1’, FLXnIBCR.FLXnIBSYH is set to ’1’. After completion of the ongoing data transfer from IBF Shadow to the Message RAM, IBF Host and IBF Shadow are swapped, FLXnIBCR.FLXnIBSYH is reset

to ’0’, FLXnIBCR.FLXnIBSYS remains set to ’1’, and the next transfer to the Message RAM is started. In addition the message buffer numbers stored under FLXnIBCR.FLXnIBRH[6:0] and FLXnIBCR.FLXnIBRS[6:0] and the command mask flags are also swapped.

Example of a 32-bit Host access sequence:

Configure / update n-th message buffer via IBF

- Wait until FLXnIBCR.FLXnIBSYH is reset
- Write data section to FLXnWRDSm
- Write header section to FLXnWRHS1...3
- Write Command Mask: write FLXnIBCM.FLXnSTXRH, FLXnIBCM.FLXnLDSH, FLXnIBCM.FLXnLHSH
- Demand data transfer to target message buffer: write FLXnIBCR.FLXnIBRH[6:0]

Configure / update (n+1)th message buffer via IBF

- Wait until FLXnIBCR.FLXnIBSYH is reset
- Write data section to FLXnWRDSm
- Write header section to FLXnWRHS1...3
- Write Command Mask: write FLXnIBCM.FLXnSTXRH, FLXnIBCM.FLXnIBCM, FLXnIBCM.FLXnLHSH
- Demand data transfer to target message buffer: write FLXnIBCR.FLXnIBRH[6:0]

Note Any write access to IBF while FLXnIBCR.FLXnIBSYH is ’1’ will set error flag FLXnEIR.FLXnIIBAE to ’1’. In this case the write access has no effect.

Table 30-105 Assignment of FLXnIBCM bits

Pos.	Access	Bit	Function
18	r	FLXn STXRS	Set Transmission Request Shadow ongoing or finished
17	r	FLXn LDSS	Load Data Section Shadow ongoing or finished
16	r	FLXn LHSS	Load Header Section Shadow ongoing or finished
2	r/w	FLXn STXRH	Set Transmission Request Host
1	r/w	FLXn LDSH	Load Data Section Host
0	r/w	FLXn LHSH	Load Header Section Host

Table 30-106 Assignment of FLXnIBCR bits

Pos.	Access	Bit	Function
31	r	FLXn IBSYS	IBF Busy Shadow, signals ongoing transfer from IBF Shadow to Message RAM
22...16	r	FLXn IBRS[6:0]	IBF Request Shadow, number of message buffer currently / last updated

Table 30-106 Assignment of FLXnIBCR bits

Pos.	Access	Bit	Function
15	r	FLXn IBSYH	IBF Busy Host, transfer request pending for message buffer referenced by IBRH[6:0]
6...0	r/w	FLXn IBRH[6:0]	IBF Request Host, number of message buffer to be updated next

Data Transfer from Message RAM to Output Buffer

To read a message buffer from the Message RAM, the Host has to write to register FLXnOBCR to trigger the data transfer as configured in FLXnOBCM. After the transfer has completed, the Host can read the transferred data from FLXnRDDSm, FLXnRDHS1...3, and FLXnMBS.

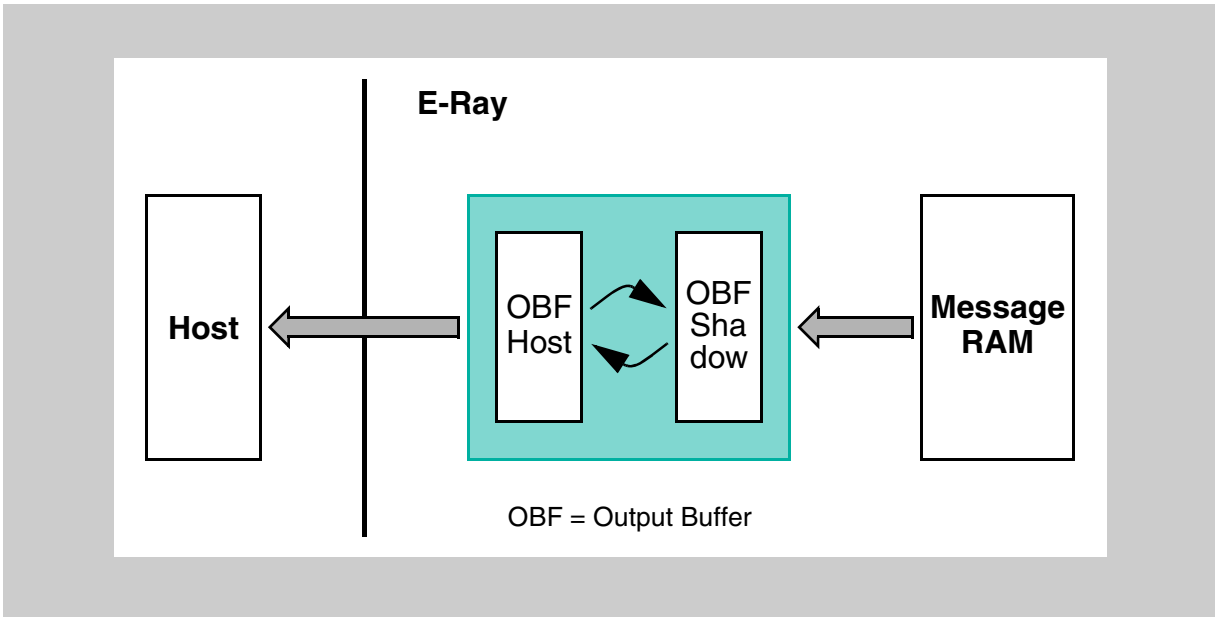


Figure 30-13 Double buffer structure Output Buffer

OBF Host and OBF Shadow as well as bits FLXnOBCM.FLXnRHSS, FLXnOBCM.FLXnRDSS, FLXnOBCM.FLXnRHSH, FLXnOBCM.FLXnRDSH and bits FLXnOBCR.FLXnOBRS[6:0], FLXnOBCR.FLXnOBRH[6:0] are swapped under control of bits FLXnOBCR.FLXnVIEW and FLXnOBCR.FLXnREG.

Writing bit FLXnOBCR.FLXnREG to '1' copies bits FLXnOBCM.FLXnRHSS, FLXnOBCM.FLXnRDSS and bits FLXnOBCR.FLXnOBRS[6:0] to an internal storage.

After setting FLXnOBCR.FLXnREG to '1', FLXnOBCR.FLXnOBSYS is set to '1', and the transfer of the message buffer selected by FLXnOBCR.FLXnOBRS[6:0] from the Message RAM to OBF Shadow is started. After the transfer between the Message RAM and OBF Shadow has completed, the FLXnOBCR.FLXnOBSYS bit is set back to '0'. Bits FLXnOBCR.FLXnREG and FLXnOBCR.FLXnVIEW can only be set to '1' while FLXnOBCR.FLXnOBSYS is '0'.

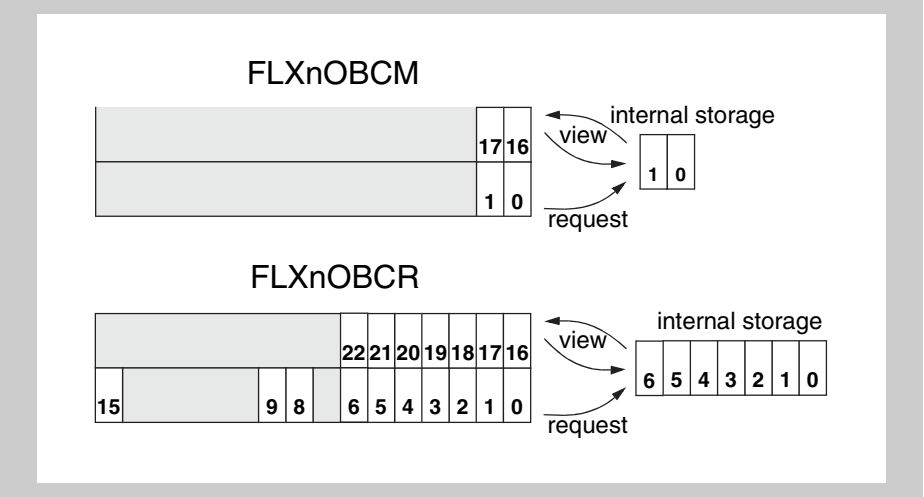


Figure 30-14 Swapping of FLXnOBCM and FLXnOBCR bits

OBF Host and OBF Shadow are swapped by setting bit FLXnOBCR.FLXnVIEW to ’1’ while bit FLXnOBCR.FLXnOBSYS is ’0’.

In addition bits FLXnOBCR.FLXnOBRH[6:0] and bits FLXnOBCM.FLXnRHSH, FLXnOBCM.FLXnRDSH are swapped with the registers internal storage thus assuring that the message buffer number stored in FLXnOBCR.FLXnOBRH[6:0] and the mask configuration stored in FLXnOBCM.FLXnRHSH, FLXnOBCM.FLXnRDSH matches the transferred data stored in OBF Host.

Now the Host can read the transferred message buffer from OBF Host while the Message Handler may transfer the next message from the Message RAM to OBF Shadow.

If bits FLXnREG and FLXnVIEW are set to ’1’ with the same write access while FLXnOBSYS is ’0’, FLXnOBSYS is automatically set to ’1’ and OBF Shadow and OBF Host are swapped. Additionally mask bits FLXnOBCM. FLXnRDSH and FLXnOBCM.FLXnRHSH are swapped with the registers internal storage to keep them attached to the respective Output Buffer transfer. Afterwards FLXnOBRS[6:0] is copied to the register internal storage, mask bits FLXnOBCM.FLXnRDSS and FLXnOBCM.FLXnRHSS are copied to register FLXnOBCM internal storage, and the transfer of the selected message buffer from the Message RAM to OBF Shadow is started. While the transfer is ongoing the Host can read the message buffer transferred by the previous transfer from OBF Host. When the current transfer between Message RAM and OBF Shadow has completed, this is signalled by setting FLXnOBSYS back to ’0’.

Example of an 32-bit Host access to a single message buffer:

If a single message buffer has to be read out, two separate write accesses to FLXnOBCR.FLXnREG and FLXnOBCR.FLXnVIEW are necessary:

- Wait until FLXnOBCR.FLXnOBSYS is reset
- Write Output Buffer Command Mask FLXnOBCM.FLXnRHSS, FLXnOBCM.FLXnRDSS
- Request transfer of message buffer to OBF Shadow by writing FLXnOBCR.FLXnOBRS[6:0] and FLXnOBCR.FLXnREG (in case of and 8-bit Host interface, FLXnOBCR.FLXnOBRS[6:0] has to be written before FLXnOBCR.FLXnREG).
- Wait until FLXnOBCR.FLXnOBSYS is reset

- Toggle OBF Shadow and OBF Host by writing FLXnOBCR.FLXnVIEW = '1'
- Read out transferred message buffer by reading FLXnRDDSn, FLXnRDHS1...3, and MBS

Example of an 32-bit Host access sequence:

Request transfer of 1st message buffer to OBF Shadow

- Wait until FLXnOBCR.FLXnOBSYS is reset
- Write Output Buffer Command Mask FLXnOBCM.FLXnRHSS, FLXnOBCM.FLXnRDSS for 1st message buffer
- Request transfer of 1st message buffer to OBF Shadow by writing FLXnOBCR.FLXnOBRS[6:0] and FLXnOBCR.FLXnREG (in case of an 8-bit Host interface, FLXnOBCR.FLXnOBRS[6:0] has to be written before FLXnOBCR.FLXnREG).

Toggle OBF Shadow and OBF Host to read out 1st transferred message buffer and request transfer of 2nd message buffer:

- Wait until FLXnOBCR.FLXnOBSYS is reset
- Write Output Buffer Command Mask FLXnOBCM.FLXnRHSS, FLXnOBCM.FLXnRDSS for 2nd message buffer
- Toggle OBF Shadow and OBF Host and start transfer of 2nd message buffer to OBF Shadow simultaneously by writing FLXnOBCR.FLXnOBRS[6:0] of 2nd message buffer, FLXnOBCR.FLXnREG, and FLXnOBCR.FLXnVIEW (in case of and 8-bit Host interface, FLXnOBCR.FLXnOBRS[6:0] has to be written before FLXnOBCR.FLXnREG and FLXnOBCR.FLXnVIEW).
- Read out 1st transferred message buffer by reading FLXnRDDSm, FLXnRDHS1...3, and FLXnMBS

Demand access to last requested message buffer without request of another message buffer:

- Wait until FLXnOBCR.FLXnOBSYS is reset
- Demand access to last transferred message buffer by writing FLXnOBCR.FLXnVIEW
- Read out last transferred message buffer by reading FLXnRDDSm, FLXnRDHS1...3, and FLXnMBS

Table 30-107 Assignment of FLXnOBCM bits

Pos.	Access	Bit	Function
17	r	FLXn RDSH	Data Section available for Host access
16	r	FLXnR HSH	Header Section available for Host access
1	r/w	FLXnR DSS	Read Data Section Shadow
0	r/w	FLXn RHSS	Read Header Section Shadow

Table 30-108 Assignment of FLXnOBCR bits

Pos.	Access	Bit	Function
22...16	r	FLXn OBRH[6:0]	OBF Request Host, number of message buffer available for Host access
15	r	FLXn OBSYS	OBF Busy Shadow, signals ongoing transfer from Message RAM to OBF Shadow
9	r/w	FLXn REG	Request Transfer from Message RAM to OBF Shadow
8	r/w	FLXn VIEW	View OBF Shadow, swap OBF Shadow and OBF Host
6...0	r/w	FLXn OBRS[6:0]	OBF Request Shadow, number of message buffer for next request

(3) FlexRay protocol controller access to Message RAM

The two Transient Buffer RAMs (TBF A,B) are used to buffer the data for transfer between the two FlexRay Protocol Controllers and the Message RAM.

Each Transient Buffer RAM is build up as a double buffer, able to store two complete FlexRay messages. There is always one buffer assigned to the corresponding Protocol Controller while the other one is accessible by the Message Handler.

If e.g. the Message Handler writes the next message to be send to Transient Buffer Tx, the FlexRay Channel Protocol Controller can access Transient Buffer Rx to store the message it is actually receiving. During transmission of the message stored in Transient Buffer Tx, the Message Handler transfers the last received message stored in Transient Buffer Rx to the Message RAM (if it passes acceptance filtering) and updates the respective message buffer.

Data transfers between the Transient Buffer RAMs and the shift registers of the FlexRay Channel Protocol Controllers are done in words of 32 bit. This enables the use of a 32 bit shift register independent of the length of the FlexRay messages.

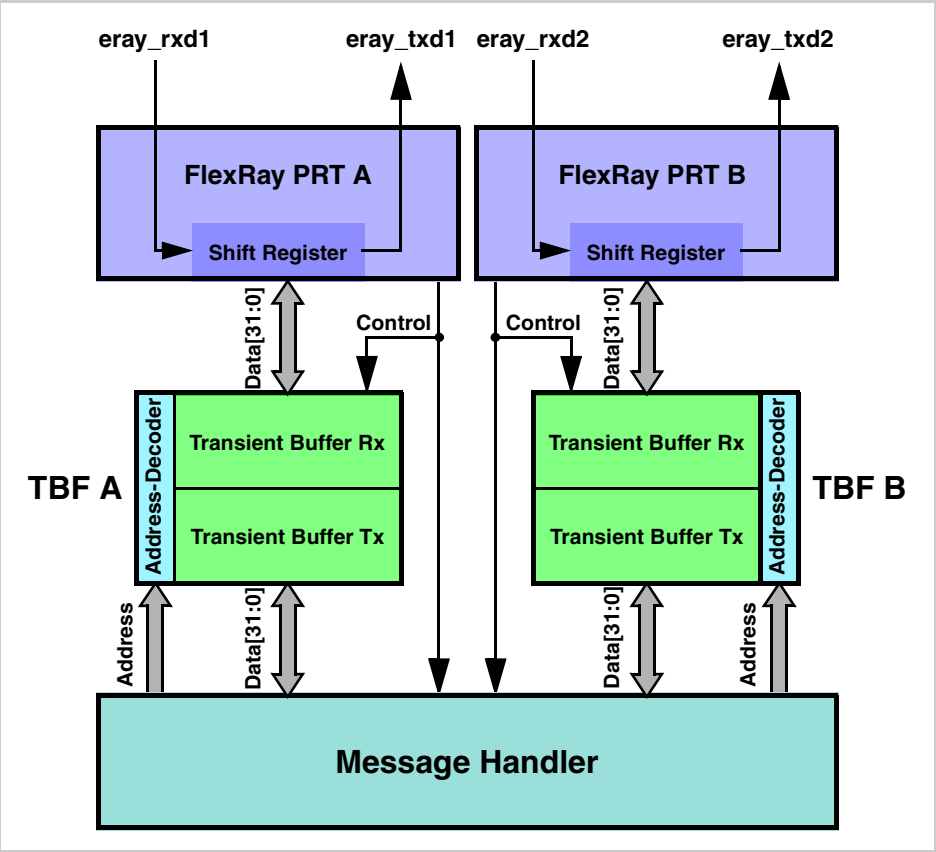


Figure 30-15 Access to transient buffer RAMs

30.4.12 Message RAM

To avoid conflicts between Host access to the Message RAM and FlexRay message reception / transmission, the Host cannot directly access the message buffers in the Message RAM. These accesses are handled via the Input and Output Buffers. The Message RAM is able to store up to 128 message buffers depending on the configured payload length.

The Message RAM is organized 2048 x 39 = 79872 bit. Each 32-bit word is protected by a 7-bit ECC-code. To achieve the required flexibility with respect to different numbers of data bytes per FlexRay frame (0 to 254), the Message RAM has a structure as shown in *Figure 30-16 "Structure of Message RAM"*.

The data partition is allowed to start at Message RAM word number:
(FLXnMRC.FLXnLCB + 1) • 4

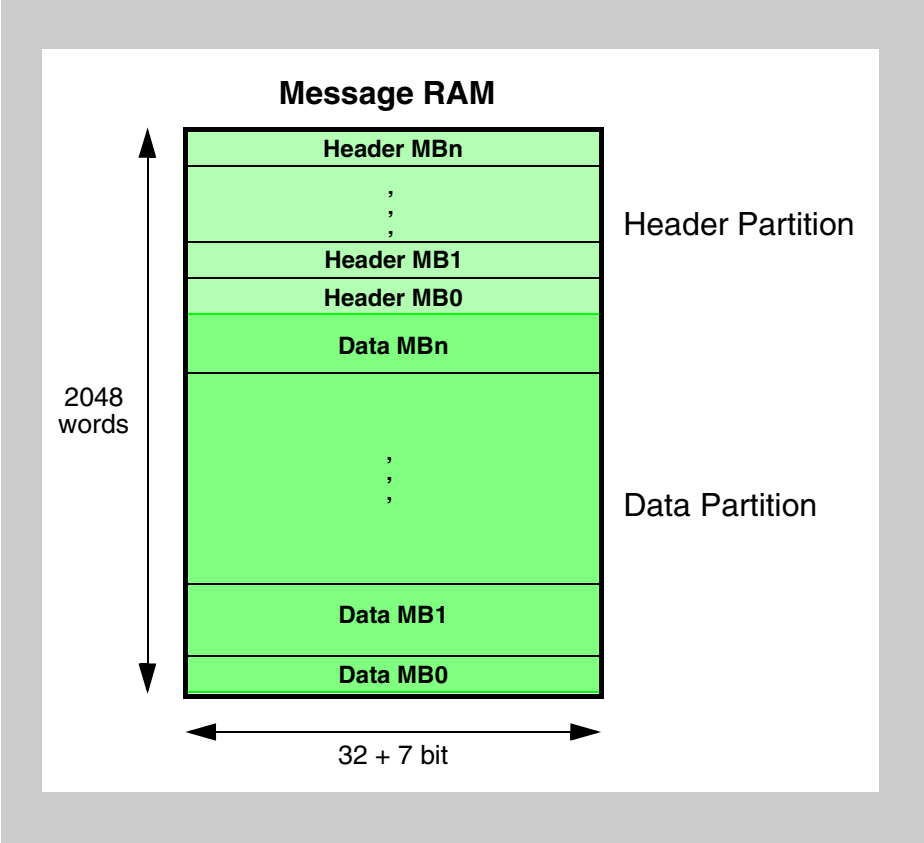


Figure 30-16 Structure of Message RAM

Header partition

Stores header sections of the configured message buffers:

- Supports a maximum of 128 message buffers
- Each message buffer has a header section of four 32+1 bit words
- Header 3 of each message buffer holds the 11-bit data pointer to the respective data section in the data partition

Data partition

Flexible storage of data sections with different length. Some maximum values are:

- 30 message buffers with 254 byte data section each
- Or 56 message buffers with 128 byte data section each
- Or 128 message buffers with 48 byte data section each

Restriction: header partition + data partition may not occupy more than 2048 39-bit words.

(1) Header partition

The elements used for configuration of a message buffer as well as the actual message buffer status are stored in the header partition of the Message RAM as listed in Table below. Configuration of the header sections of the message buffers is done via IBF (FLXnWRHS13). Read access to the header sections is done via OBF (FLXnRDHS13 + FLXnMBS). The data pointer has to be calculated by the programmer to define the starting point of the data section for the respective message buffer in the data partition of the Message RAM. The data pointer should not be modified during runtime. For message buffers belonging to the receive FIFO (re)configuration is possible in DEFAULT_CONFIG or CONFIG state only.

The header section of each message buffer occupies four 39-bit words in the header partition of the Message RAM. The header of message buffer 0 starts with the first word in the Message RAM.

For transmit buffers the Header CRC has to be calculated by the Host CPU.

Payload Length Received PLR[6:0], Receive Cycle Count FLXnRHRCC[5:0], Received on Channel Indicator FLXnRHRCI, Startup Frame Indicator FLXnRHSFI, Sync Frame Indicator FLXnRHSYN, Null Frame Indicator FLXnRHNFI, Payload Preamble Indicator FLXnRHPPI, and Reserved Bit FLXnRHRES are updated from received valid data frames only.

Header word 4 of each configured message buffer holds the respective Message Buffer Status information.

Table 30-109 Header section of a message buffer in the Message RAM

Bit Word	32 - 38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
1	E			MBI	TXM	PPIT	CFG	CHB	CHA		Cycle Code						
2	E		Payload Length Received								Payload Length Configured						
3	E			RES	PPI	NFI	SYN	SFI	RCI			Receive Cycle Count					
4	E			RESS	PPIS	NFIS	FLXn SYNS	SFIS	RCIS			Cycle Count Status					
…	E																
…	E																

Bit Word	32 - 38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	E						Frame ID										
2	E						Tx Buffer: Header CRC Configured Rx Buffer: Header CRC Received										
3	E						Data Pointer										
4	E	FTB	FTA		MLST	ESB	ESA	TCIB	TCIA	SVOB	SVOA	CEOB	CEOA	SEOB	SEOA	VFRB	VFRA
…	E																
…	E																

- Frame Configuration
- Filter Configuration
- Message Buffer Control
- Message RAM Configuration
- Updated from received Data Frame
- Message Buffer Status MBS
- ECC code
- unused

Header 1

Write access via FLXnWRHS1, read access via FLXnRDHS1:

- Frame ID - Slot counter filtering configuration
- Cycle Code - Cycle counter filtering configuration
- FLXnWHCHA, FLXnWHCHB - Channel filtering configuration
- FLXnRHCFG/FLXnWHCFG - Message buffer direction configuration: receive / transmit
- FLXnWHPPIT - Payload Preamble Indicator Transmit
- FLXnWHTXM - Transmit mode configuration: single-shot / continuous
- FLXnWHMBI/FLXnRHMBI - Message buffer receive / transmit interrupt enable

Header 2

Write access via FLXnWRHS2, read access via FLXnRDHS2:

- Header CRC
  - Transmit Buffer: Configured by the Host (calculated from frame header)
  - Receive Buffer: Updated from received frame
- Payload Length Configured
  - Length of data section (2-byte words) as configured by the Host
- Payload Length Received
  - Length of payload segment (2-byte words) stored from received frame

Header 3

Write access via FLXnWRHS3, read access via FLXnRDHS3:

- Data Pointer - Pointer to the beginning of the corresponding data section in the data partition

Read access via FLXnRDHS3, valid for receive buffers only, updated from received frames:

- Receive Cycle Count - Cycle count from received frame
- FLXnRHRCI - Received on Channel Indicator
- FLXnRHSFI - Startup Frame Indicator
- FLXnRHSYN - Sync Frame Indicator
- FLXnRHNFI - Null Frame Indicator
- FLXnRHPPI - Payload Preamble Indicator
- FLXnRHRES - Reserved bit

Header 4

Read access via FLXnMBS, updated by the CC at the end of the configured slot.

- FLXnMBSVFRA - Valid Frame Received on channel A
- FLXnMBSVFRB - Valid Frame Received on channel B
- FLXnMBSSEOA - Syntax Error Observed on channel A

- FLXnMBSSEOB - Syntax Error Observed on channel B
- FLXnMBSCEOA - Content Error Observed on channel A
- FLXnMBSCEOB - Content Error Observed on channel B
- FLXnMBSSVOA - Slot boundary Violation Observed on channel A
- FLXnMBSSVOB - Slot boundary Violation Observed on channel B
- FLXnMBSTCIA - Transmission Conflict Indication channel A
- FLXnMBSTCIB - Transmission Conflict Indication channel B
- FLXnMBSESA - Empty Slot Channel A
- FLXnMBSESB - Empty Slot Channel B
- FLXnMBSMLST - Message LoST
- FLXnMBSFTA - Frame Transmitted on Channel A
- FLXnMBSFTB - Frame Transmitted on Channel B
- Cycle Count Status- Actual cycle count when status was updated
- FLXnMBSRCI - Received on Channel Indicator Status
- FLXnMBSSFI - Startup Frame Indicator Status
- FLXnMBSSYN - Sync Frame Indicator Status
- FLXnMBSNFI - Null Frame Indicator Status
- FLXnMBSPPI - Payload Preamble Indicator Status
- FLXnMBSRES - Reserved bit Status

(2) Data partition

The data partition of the Message RAM stores the data sections of the message buffers configured for reception / transmission as defined in the header partition. The number of data bytes for each message buffer can vary from 0 to 254. To optimize the data transfer between the shift registers of the two FlexRay Protocol Controllers and the Message RAM as well as between the Host interface and the Message RAM, the physical width of the Message RAM is set to 4 bytes plus 7-bits ECC code.

The data partition starts after the last word of the header partition. When configuring the message buffers in the Message RAM the programmer has to assure that the data pointers point to addresses within the data partition. Table 27 below shows an example how the data sections of the configured message buffers can be stored in the data partition of the Message RAM.

The beginning and the end of a message buffer's data section is determined by the data pointer and the payload length configured in the message buffer's header section, respectively. This enables a flexible usage of the available RAM space for storage of message buffers with different data length.

If the size of the data section is an odd number of 2-byte words, the remaining 16 bits in the last 32-bit word are unused.

Table 30-110 Example for structure of the data partition in the Message RAM

Bit Word	32 - 38	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
…	E	MBn Data(m)								MBn Data(m-1)							
…	E	…								…							
…	E	…								…							
…	E	MBn Data3								MBn Data2							
…	E	…								…							
…	E	…								…							
…	E	…								…							
…	E	MB1 Data(k)								MB1 Data(k-1)							
…	E	…								…							
…	E	MB1 Data3								MB1 Data2							
…	E	unused								unused							
2048	E	MB0 Data3								MB0 Data2							

Bit Word	32 - 38	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
…	E	MBn Data(m-2)								MBn Data(m-3)							
…	E	…								…							
…	E	…								…							
…	E	MBn Data1								MBn Data0							
…	E	…								…							
…	E	…								…							
…	E	…								…							
…	E	MB1 Data(k-2)								MB1 Data(k-3)							
…	E	…								…							
…	E	MB1 Data1								MB1 Data0							
…	E	MB0 Data5								MB0 Data4							
2048	E	MB0 Data1								MB0 Data0							

(3) ECC

There is ECC implemented in the E-Ray module to assure the integrity of the data stored in all RAM blocks as shown in *Figure 30-17 “ECC generation and check”*.

The FlexRay RAMs have the ECC decoding/encoding module. When data is written to the RAMs, the local ECC encoder generates the ECC code. The ECC code is stored together with the respective data word. The ECC code is decoded each time a data word is read from the RAMs and corrects single-bit error per word and detects double-bit error per word.

If a ECC DED error is detected in the RAMs, the respective error flag is set. The ECC DED error flags FLXnMHDS.FLXnDIBFE, FLXnMHDS.FLXnDOBFE, FLX0DMRE, FLXnMHDS.FLXnDTBFAE, FLXnMHDS.FLXnDTBFBE, and the faulty message buffer indicators FLXnMHDS.FLXnFMBDE, FLXnMHDS.FLXnMFMBE, FLXnMHDS.FLXnFMB[6:0] are located in the

Message Handler Status register. These ECC DED error flags control the error interrupt flag.

The single-bit error is silently corrected with ECC unit. In case of that, no error information is displayed in any interrupt registers or status registers.

In case of double-bit error at reading Message RAM, FLXnMHDS.FLXnDMRE is set and FLXnMHDS.FLXnFMBDE, FLXnMHDS.FLXnMFMBE, FLXnMHDS.FLXnFMB[6:0] indicates the faulty message buffer.

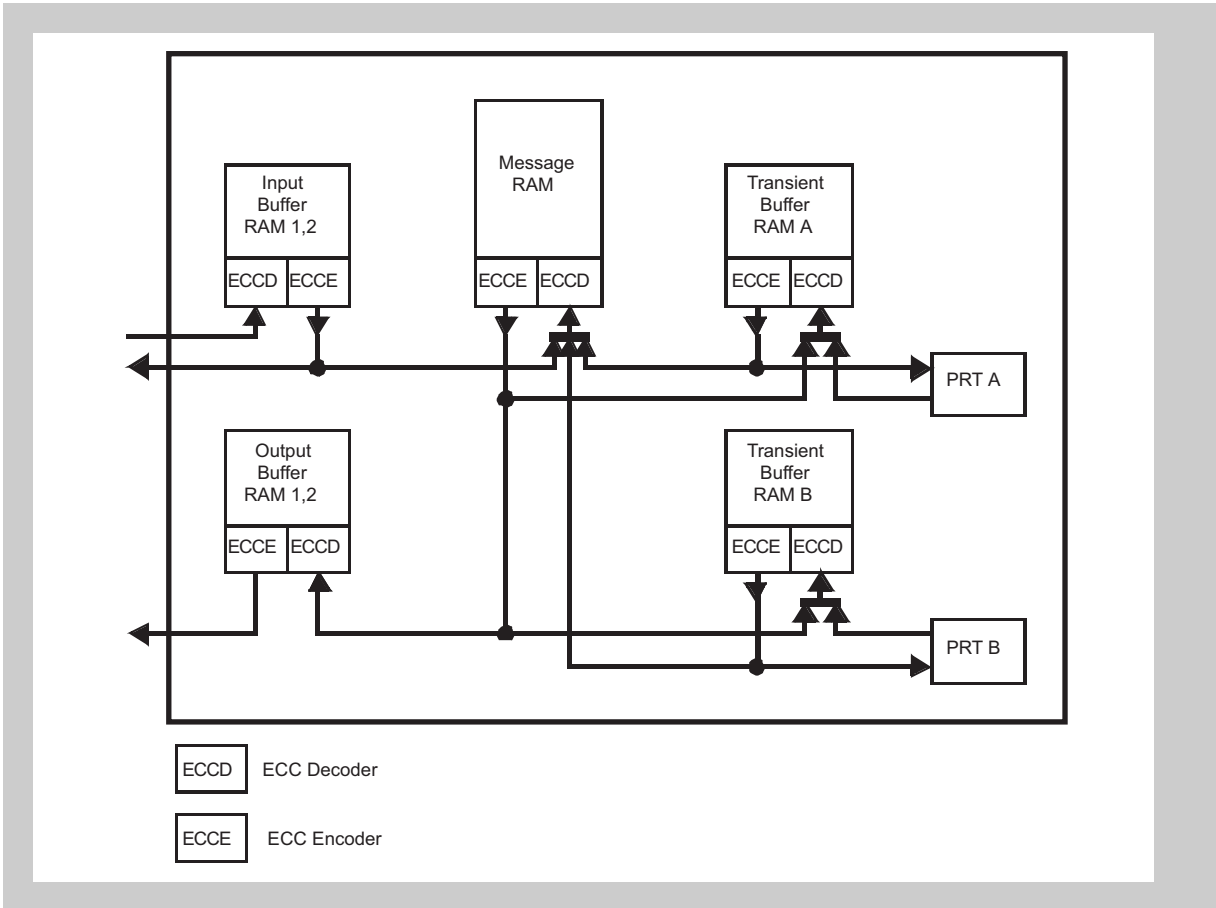


Figure 30-17 ECC generation and check

In case a ECC double-bit error has been detected, the following actions will be performed.

In all cases:

- The respective flag in the Message Handler Status register FLXnMHDS is set
- The ECC DED error interrupt is asserted.

Additionally in specific cases

1. ECC double-bit error during data transfer from Input Buffer RAM 1,2 ⇒ Message RAM
   a) Transfer of header and/or data section:

– FLXnMHDS.FLXnDIBFE bit is set
– FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
– FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
– Transmit buffer: Transmission request for the respective message buffer is not set

b) Transfer of data section only:
ECC double-bit error when reading header section of respective message buffer from Message RAM.

– FLXnMHDS.FLXnDMRE bit is set
– FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
– FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
– The data section of the respective message buffer is not updated
– Transmit buffer: Transmission request for the respective message buffer is not set

2. ECC double-bit error during Host reading Input Buffer RAM 1,2
   – FLXnMHDS.FLXnDIBFE bit is set

3. ECC double-bit error during scan of header sections in Message RAM
   – FLXnMHDS.FLXnDMRE bit is set
   – FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
   – FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
   – Ignore message buffer (message buffer is skipped)

4. ECC double-bit error during data transfer from Message RAM ⇒ Transient Buffer RAM 1,2
   – FLXnMHDS.FLXnDMRE bit is set
   – FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
   – FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
   – Frame not transmitted, frames already in transmission are invalidated by setting the frame CRC to zero

5. ECC double-bit error during data transfer from Transient Buffer RAM 1, 2 ⇒ Protocol Controller 1, 2
   – FLXnMHDS.FLXnDTBFAE,2 bit is set
   – Frames already in transmission are invalidated by setting the frame CRC to zero

6. ECC double-bit error during data transfer from Transient Buffer RAM 1, 2 ⇒ Message RAM

a) ECC double-bit error when reading header section of respective message buffer from Message RAM:

– FLXnMHDS.FLXnDMRE bit is set
– FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
– FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
– The data section of the respective message buffer is not updated

b) ECC double-bit error when reading Transient Buffer RAM 1, 2:

– FLXnMHDS.FLXnDTBFAE,2 bit is set
– FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
– FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer

7. ECC double-bit error during data transfer from Message RAM ⇒ Output Buffer RAM
   – FLXnMHDS.FLXnDMRE bit is set
   – FLXnMHDS.FLXnFMBDE bit is set to indicate that FLXnMHDS.FLXnFMB[6:0] points to a faulty message buffer
   – FLXnMHDS.FLXnFMB[6:0] indicates the number of the faulty message buffer
8. ECC double-bit error during Host reading Output Buffer RAM 1,2
   – FLXnMHDS.FLXnDOBFE bit is set
9. ECC double-bit error during data read of Transient Buffer RAM 1, 2
   When a ECC double-bit error occurs when the Message Handler reads a frame with network management information (FLXnRHPPI = ’1’) from the Transient Buffer RAM 1, 2 the corresponding network management vector NMV1 3 is not updated from that frame.

30.4.13 Module interrupt

In general, interrupts provide a close link to the protocol timing as they are triggered almost immediately when an error or status change is detected by the controller, a frame is received or transmitted, a configured timer interrupt is activated, or a stop watch event occurred. This enables the Host CPU to react very quickly on specific error conditions, status changes, or timer events. On the other hand too many interrupts can cause the Host to miss deadlines required for the application. Therefore the CC supports disable / enable controls for each individual interrupt source separately.

An interrupt may be triggered when

- An error was detected
- A status flag is set
- A timer reaches a preconfigured value
- A message transfer from Input Buffer to Message RAM or from Message RAM to Output Buffer has completed
- A stop watch event occurred

Tracking status and generating interrupts when a status change or an error occurs are two independent tasks. Regardless of whether an interrupt is enabled or not, the corresponding status is tracked and indicated by the CC. The Host has access to the actual status and error information by reading registers FLXnEIR and FLXnSIR.

Table 30-111 Module interrupt flags and interrupt line enable (1/2)

Register	Bit	Function
FLXnEIR	FLXnPEMCE	Protocol Error Mode Changed
	FLXnCNAE	Command Not Valid
	FLXnSFBME	Sync Frames Below Minimum
	FLXnSFOE	Sync Frame Overflow
	FLXnCCFE	Clock Correction Failure
	FLXnCCLE	CHI Command Locked
	FLXnRFOE	Receive FIFO Overrun
	FLXnEFAE	Empty FIFO Access
	FLXnIIBAE	Illegal Input Buffer Access
	FLXnIOBAE	Illegal Output Buffer Access
	FLXnMHFE	Message Handler Constraints Flag
	FLXnEDAE	Error Detected on Channel A
	FLXnLTVAE	Latest Transmit Violation Channel A
	FLXnTABAE	Transmission Across Boundary Channel A
	FLXnEDBE	Error Detected on Channel B
	FLXnLTVBE	Latest Transmit Violation Channel B
	FLXnTABBE	Transmission Across Boundary Channel B

Table 30-111 Module interrupt flags and interrupt line enable (2/2)

Register	Bit	Function
FLXnSIR	FLXnWSTF	Wakeup Status
	FLXnCASF	Collision Avoidance Symbol
	119 SF	Cycle Start Interrupt
	FLXnTXIF	Transmit Interrupt
	FLXnRXIF	Receive Interrupt
	FLXnRFNEF	Receive FIFO not Empty
	FLXnRFCLF	Receive FIFO Critical Level
	FLXnNMVCF	Network Management Vector Changed
	FLXnTI0F	Timer Interrupt 0
	FLXnTI1F	Timer Interrupt 1
	FLXnTIBCF	Transfer Input Buffer Completed
	FLXnTOBCF	Transfer Output Buffer Completed
	FLXnSWEF	Stop Watch Event
	FLXnSUCSF	Startup Completed Successfully
	FLXnMBSIF	Message Buffer Status Interrupt
	FLXnSDSF	Start of Dynamic Segment
	FLXnWUPAF	Wakeup Pattern Channel A
	FLXnMTSAF	MTS Received on Channel A
	FLXnWUPBF	Wakeup Pattern Channel B
	FLXnMTSBF	MTS Received on Channel B
FLXnILE	FLXnEINTL0	Enable Interrupt Line 0
	FLXnEINTL1	Enable Interrupt Line 1

The interrupt lines to the Host, eray_int0 and eray_int1, are controlled by the enabled interrupts. In addition each of the two interrupt lines can be enabled / disabled separately by programming bit FLXnILE.FLXnEINTL0 and FLXnILE.FLXnEINTL1.

The two timer interrupts generated by interrupt timer 0 and 1 are available on pins eray_tint0 and eray_tint1. They can be configured via registers FLXnT0C and FLXnT1C.

A stop watch event may be triggered via input pin eray_stpwt.

The status of the data transfer between IBF / OBF and the Message RAM is signalled on pins eray_ibusy and eray_obusy. When a transfer has completed bit FLXnSIR.FLXnTIBCF or FLXnSIR.FLXnTOBCF is set.

30.5 Appendix

30.5.1 Assignment of FlexRay Configuration Parameters

Table 30-112 FlexRay configuration parameters (1/2)

Parameter	Bit(field)
pKeySlotusedForStartup	FLXnSUCC1.FLXnTXST
pKeySlotUsedForSync	FLXnSUCC1.FLXnTXSY
gColdStartAttempts	FLXnSUCC1.FLXnCSA[4:0]
pAllowPassiveToActive	FLXnSUCC1.FLXnPTA[4:0]
pWakeupChannel	FLXnSUCC1.FLXnWUCS
pSingleSlotEnabled	FLXnSUCC1.FLXnTSM
pAllowHaltDueToClock	FLXnSUCC1.FLXnHCSE
pChannels	FLXnSUCC1.FLXnCCHA FLXnSUCC1.FLXnCCHB
pdListenTimeOut	FLXnSUCC2.FLXnLT[20:00]
gListenNoise	FLXnSUCC2.FLXnLTN[3:0]
gMaxWithoutClockCorrectionPassive	FLXnSUCC3.FLXnWCP[3:0]
gMaxWithoutClockCorrectionFatal	FLXnSUCC3.FLXnWCF[3:0]
gNetworkManagementVectorLength	FLXnNEMC.FLXnNML[3:0]
gdFLXnTSSTransmitter	FLXnPRTC1.FLXnTSST[3:0]
gdCASRxLowMax	FLXnPRTC1.FLXnCASM[6:0]
gdSampleClockPeriod	FLXnPRTC1.FLXnBRP[1:0]
pSamplesPerMicrotick	FLXnPRTC1.FLXnBRP[1:0]
gdWakeupSymbolRxWindow	FLXnPRTC1.FLXnRXW[8:0]
pWakeupPattern	FLXnPRTC1.FLXnRWP[5:0]
gdWakeupSymbolRxIdle	FLXnPRTC2.FLXnRXI[5:0]
gdWakeupSymbolRxLow	FLXnPRTC2.FLXnRXL[5:0]
gdWakeupSymbolTxIdle	FLXnPRTC2.FLXnTXI[7:0]
gdWakeupSymbolTxLow	FLXnPRTC2.FLXnTXL[5:0]
gPayloadLengthStatic	FLXnMHDC.FLXnSFDL[6:0]
pLatestTx	FLXnMHDC.FLXnSLT[12:00]
pMicroPerCycle	FLXnGTUC01.FLXnUT[19:00]
gMacroPerCycle	FLXnGTUC02.FLXnMPC[13:00]
gSyncNodeMax	FLXnGTUC02.FLXnSNM[3:0]
pMicroInitialOffset[A]	FLXnGTUC3.FLXnUIOA[7:0]
pMicroInitialOffset[B]	FLXnGTUC3.FLXnUIOB[7:0]
pMacroInitialOffset[A]	FLXnGTUC3.FLXnMIOA[6:0]
pMacroInitialOffset[B]	FLXnGTUC3.FLXnMIOB[6:0]
gdNIT	FLXnGTUC04.FLXnNIT[13:00]
gOffsetCorrectionStart	FLXnGTUC04.FLXnOCS[13:00]
pDelayCompensation[A]	FLXnGTUC5.FLXnDCA[7:0]
pDelayCompensation[B]	FLXnGTUC5.FLXnDCB[7:0]

Table 30-112 FlexRay configuration parameters (2/2)

Parameter	Bit(field)
pClusterDriftDamping	FLXnGTUC5.FLXnCDD[4:0]
pDecodingCorrection	FLXnGTUC5.FLXnDEC[7:0]
pdAcceptedStartupRange	FLXnGTUC6.FLXnASR[10:00]
pdMaxDrift	FLXnGTUC6.FLXnMOD[10:00]
gdStaticSlot	FLXnGTUC7.FLXnSSL[9:0]
gNumberOfStaticSlots	FLXnGTUC7.FLXnNSS[9:0]
gdMinislot	FLXnGTUC8.FLXnMSL[5:0]
gNumberOfMinislots	FLXnGTUC8.FLXnNMS[12:00]
gdActionPointOffset	FLXnGTUC9.FLXnAPO[5:0]
gdMinislotActionPointOffset	FLXnGTUC9.FLXnMAPO[4:0]
gdDynamicSlotIdlePhase	FLXnGTUC9.FLXnDSI[1:0]
pOffsetCorrectionOut	FLXnGTUC10.FLXnMOC[13:00]
pRateCorrectionOut	FLXnGTUC10.FLXnMRC[10:00]
pExternOffsetCorrection	FLXnGTUC11.FLXnEOC[2:0]
pExternRateCorrection	FLXnGTUC11.FLXnERC[2:0]

30.6 Cautions

30.6.1 Loop back mode operates only at 10 MBit/s.

Description: The looped back data is falsified at the two lower baud rates of 5 MBit/s and 2.5 MBit/s.

Scope: The erratum is limited to test cases where loop back is used with the baud rate prescaler (FLXnPRTC1.FLXnBRP[1:0]) configured to 5 or 2.5 MBit/s.

Effects: The loop back self test is only possible at the highest baud rate.

Workaround: Run loop back tests with 10 MBit/s (FLXnPRTC1.FLXnBRP[1:0] = 00_B).

30.6.2 Noise following a dynamic frame that delays idle detection may fail to stop slot counting for the remainder of the dynamic segment

Description If (in case of noise) the time between 'potential idle start on X' and 'CHIRP on X' (see Protocol Spec. v2.1, Figure 5-21) is greater than gdDynamicSlotIdlePhase, the E-Ray will not remain for the remainder of the current dynamic segment in the state 'wait for the end of dynamic slot rx'. Instead, the E-Ray continues slot counting. This may enable the node to further transmissions in the current dynamic segment.

Scope This behaviour is only of concern for noise that is seen only locally and that is detected in the time window between the end of a dynamic frame's DTS and idle detection ('CHIRP on X').

Effects In the described case the faulty node may not stop slot counting and may continue to transmit dynamic frames. This may lead to a frame collision in the current dynamic segment.

Workaround None.

30.6.3 Register FLXnRCV displays wrong value.

Description If the calculated rate correction value is in the range of [-pClusterDriftDamping .. +pClusterDriftDamping], vRateCorrection of the CSP process is set to *zero*. In this case register FLXnRCV should be updated with this value. Erroneously FLXnRCV.FLXnRCV[11:00] holds the calculated value in the range [-pClusterDriftDamping .. +pClusterDriftDamping] instead of *zero*.

Scope The erratum is limited to the case where the calculated rate correction value is in the range of [-pClusterDriftDamping .. +pClusterDriftDamping].

Effects The displayed rate correction value FLXnRCV.FLXnRCV[11:00] is in the range of [-pClusterDrift-Damping .. +pClusterDriftDamping] instead of *zero*. The error of the displayed value is limited to the range of [-pClusterDriftDamping .. +pClusterDriftDamping]. For rate correction in the next double cycle always the correct value of *zero* is used.

Workaround A value of FLXnRCV.FLXnRCV[11:00] in the range of [-pClusterDriftDamping .. +pClusterDriftDamping] has to be interpreted as *zero*.

30.6.4 After reception of a valid sync frame followed by a valid non-sync frame in the same static slot the received sync frame may be ignored.

Description If in a static slot of an even cycle a valid sync frame followed by a valid non-sync frame is received, and the frame valid detection (prt_frame_decoded_on_X) of the DEC process occurs one sclk after valid frame detection of FSP process (fsp_val_syncfr_chx), the sync frame is not taken into account by the CSP process (devte_xxs_reg).

Scope The erratum is limited to the case where more than one valid frame is received in a static slot of an even cycle.

Effects In the described case the sync frame is not considered by the CSP process. This may lead to a SyncCalcResult of MISSIMG_TERM (error flag FLXnSFS.FLXnMRCS set). As a result the POC state may switch to NORMAL_PASSIVE or HALT or the Startup procedure is aborted.

Workaround Avoid static slot configurations long enough to receive two valid frames.

30.6.5 Sync frame overflow flag FLXnEIR.FLXnSFO may be set if slot counter is greater than 1024.

Description If in the static segment the number of transmitted and received sync frames reaches gSyncNodeMax and the slot counter in the dynamic segment reaches the value cStaticSlotIDMax + gSyncNodeMax = 1023 + gSyncNodeMax, the sync frame overflow flag FLXnEIR.FLXnSFO is set erroneously.

Scope The erratum is limited to configurations where the number of transmitted and received sync frames equals to gSyncNodeMax and the number of static slots plus the number of dynamic slots is greater or equal than 1023 + gSyncNodeMax.

Effects In the described case the sync frame overflow flag FLXnEIR.FLXnSFO is set erroneously. This has no effect to the POC state.

Workaround Configure gSyncNodeMax to number of transmitted and received sync frames plus one or avoid configurations where the total of static and dynamic slots is greater than cStaticSlotIDMax.

30.6.6 Acceptance of startup frames received after reception of more than gSyncNodeMax sync frames.

Description If a node receives in an even cycle a startup frame after it has received more than gSyncNodeMax sync frames, this startup frame is added erroneously by process CSP to the number of valid startup frames (zStartupNodes). The faulty number of startup frames is delivered to the process POC. As a consequence this node may integrate erroneously to the running cluster because it assumes that it has received the required number of startup frames.

Scope The erratum is limited to the case of more than gSyncNodeMax sync frames.

Effects In the described case a node may erroneously integrate successfully into a running cluster.

Workaround Use frame schedules where all startup frames are placed in the first static slots. gSyncNodeMax should be configured to be greater than or equal to the number of sync frames in the cluster.

30.6.7 Initial rate correction value of an integrating node is *zero* if pMicroInitialOffsetA,B = 00_H.

Description The initial rate correction value as calculated in figure 8-8 of protocol spec v2.1 is zero if parameter pMicroInitialOffsetA,B was configured to be *zero*.

Scope The erratum is limited to the case where pMicroInitialOffsetA,B is configured to *zero*.

Effects Starting with an initial rate correction value of zero leads to an adjustment of the rate correction earliest 3 cycles later (see figure 7-10 of protocol spec v2.1). In a worst case scenario, if the whole cluster is drifting away too fast, the integrating node would not be able to follow and therefore abort integration.

Workaround Avoid configurations with pMicroInitialOffsetA,B equal to zero. If the related configuration constraint of the protocol specification results in pMicroInitialOffsetA,B equal to zero, configure it to one instead. This will lead to a correct initial rate correction value, it will delay the startup of the node by only one microtick.

30.6.8 Incorrect rate and/or offset correction value if second Secondary Time Reference Point (STRP) coincides with the action point after detection of a valid frame.

Description If a valid sync frame is received before the action point and additionally noise or a second frame leads to a STRP coinciding with the action point, an incorrect deviation value of *zero* is used for further calculations of rate and/or offset correction values.

Scope The erratum is limited to configurations with an action point offset greater than static frame length.

Effects In the described case a deviation value of *zero* is used for further calculations of rate and/or offset correction values. This may lead to an incorrect rate and/or offset correction of the node.

Workaround Configure action point offset smaller than static frame length.

30.6.9 Flag SFS.MRCS is set erroneously although at least one valid sync frame pair is received.

Description If in an odd cycle 2c+1 after reception of a sync frame in slot n the total number of different sync frames per double cycle has exceeded gSyncNodeMax and the node receives in slot n+1 a sync frame that matches with a sync frame received in the even cycle 2c, the sync frame pair is not taken into account by CSP process. This may cause the flags FLXnSFS.FLXnMRCS and FLXnEIR.FLXnCCF to be set erroneously.

Scope The erratum is limited to the case of a faulty cluster configuration where different sets of sync frames are transmitted in even and odd cycles and the total number of different sync frames is greater than gSyncNodeMax.

Effects In the described case the error interrupt flag FLXnEIR.FLXnCCF is set and the node may enter either the POC state NORMAL_PASSIVE or HALT.

Workaround Correct configuration of gSyncNodeMax.

30.6.10 Rate correction set to zero in case of SyncCalcResult=MISSING_TERM.

Description In case a node receives too few sync frames for rate correction calculation and signals a SyncCalcResult of MISSING_TERM, the rate correction value is set to zero instead to the last calculated value.

Scope The erratum is limited to the case of receiving too few sync frames for rate correction calculation (SyncCalcResult=MISSING_TERM in an odd cycle).

Effects In the described case a rate correction value of zero is applied in NORMAL_ACTIVE / NORMAL_PASSIVE state instead of the last rate correction value calculated in NORMAL_ACTIVE state. This may lead to a desynchronisation of the node although it may stay in NORMAL_ACTIVE state (depending on gMaxWithoutClockCorrectionPassive) and decreases the probability to re-enter NORMAL_ACTIVE state if it has switched to NORMAL_PASSIVE (pAllowHaltDueToclock=false).

Workaround Avoid configurations with pMicroInitialOffsetA,B equal to zero. If the related configuration constraint of the protocol specification results in pMicroInitialOffsetA,B equal to zero, configure it to one instead. This will lead to a correct initial rate correction value, it will delay the startup of the node by only one microtick.

30.6.11 A sequence of received WUS may generate redundant FLXnSIR.FLXnWUPA/B events.

Description If a sequence of wakeup symbols (WUS) is received, all separated by appropriate idle phases, a valid wakeup pattern (WUP) should be detected after every second WUS. The E-Ray detects a valid wakeup pattern after the second WUS and then after each following WUS.

Scope The erratum is limited to the case where the application program frequently resets the appropriate FLXnSIR.FLXnWUPA/B bits.

Effects In the described case there are more SIR.WUPA/B events seen than expected.

Workaround Ignore redundant FLXnSIR.FLXnWUPA/B events.

30.6.12 Erroneous cycle offset during startup after abort of startup or normal operation by a READY or FREEZE command.

Description An abort of startup or normal operation by a READY or FREEZE command near the macrotick border may lead to the effect that the state INITIALIZE_SCHEDULE is one macrotick too short during the first following integration attempt. This leads to an early cycle start in state INTEGRATION_COLDSTART_CHECK or INTEGRATION_CONSISTENCY_CHECK.

As a result the integrating node calculates a cycle offset of one macrotick at the end of the first even/odd cycle pair in the states INTEGRATION_COLDSTART_CHECK or INTEGRATION_CONSISTENCY_CHECK and tries to correct this offset.

If the node is able to correct the offset of one macrotick (pOffsetCorrectionOut >> gdMacrotick), the node enters NORMAL_ACTIVE with the first startup attempt.

If the node is not able to correct the offset error because pOffsetCorrectionOut is too small (pOffsetCorrectionOut . gdMacrotick), the node enters ABORT_STARTUP and is ready to try startup again. The next (second) startup attempt is not affected by this erratum.

Scope The erratum is limited to applications where READY or FREEZE command is used to leave STARTUP, NORMAL_ACTIVE, or NORMAL_PASSIVE state.

Effects In the described case the integrating node tries to correct an erroneous cycle offset of one macrotick during startup.

Workaround With a configuration of
pOffsetCorrectionOut >> gdMacrotick • (1+cClockDeviationMax)
the node will be able to correct the offset and therefore also be able to successfully integrate.

30.6.13 First WUS following received valid WUP may be ignored.

Description When the protocol engine is in state WAKEUP_LISTEN and receives a valid wakeup pattern (WUP), it transfers into state READY and updates the wakeup status vector CCSV.WSV[2:0] as well as the status interrupt flags SIR.WST and SIR.WUPA/B. If the received wakeup pattern continues, the protocol engine may ignore the first wakeup symbol (WUS) following the state transition and signals the next SIR.WUPA/B at the third instead of the second WUS.

Scope The erratum is limited to the reception of redundant wakeup patterns.

Effects Delayed setting of status interrupt flags SIR.WUPA/B for redundant wakeup patterns.

Workaround None.

30.6.14 READY command accepted in READY state.

Description The E-Ray module does not ignore a READY command while in READY state.

Scope The erratum is limited to the READY state.

Effects Flag CCSV.CSI is set. Cold starting needs to be enabled by POC command ALLOW_COLDSTART (SUCC1.CMD = "1001").

Workaround None.

30.6.15 Slot Status vPOC!SlotMode is reset immediately when entering HALT state (CCSV.SLM[1:0] = "00").

Description When the protocol engine is in the states NORMAL_ACTIVE or NORMAL_PASSIVE, a HALT or FREEZE command issued by the Host resets vPOC!SlotMode immediately to SINGLE slot mode (CCSV.SLM[1:0] = "00"). According to the FlexRay protocol specification, the slot mode should not be reset to SINGLE slot mode before the following state transition from HALT to DEFAULT_CONFIG state.

Scope The erratum is limited to the HALT state.

Effects The slot status vPOC!SlotMode is reset to SINGLE when entering HALT state.

Workaround None.

30.6.16 Received messages not stored in Message RAM when in Loop Back Mode.

Description After a FREEZE or HALT command has been asserted in NORMAL_ACTIVE state, and if state LOOP_BACK is then entered by transition from HALT state via DEF_CONFIG and CONFIG it may happen, that acceptance filtering for received messages is not started, and therefore these messages are not stored in the respective receive buffer in the Message RAM.

Scope The erratum is limited to the case where Loop Back Mode is entered after NORMAL_ACTIVE state was left by FREEZE or HALT command.

Effects Received messages are not stored in Message RAM because acceptance filtering is not started.

Workaround Leave HALT state by hardware reset.

Chapter 31 Ethernet Controller (ETHA)

This chapter contains a generic description of the Ethernet Controller.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

31.1 V850E2/Fx4-H ETHAn Features

Instances

This microcontroller has following number of instances of the Ethernet Controller ETHAn.

Table 31-1 Instances of ETHA

Ethernet Controller	
Instance	1
Name	ETHA0

Instances index n

Throughout this chapter, the instance of the Ethernet Controller is identified by the index "n" (n = 0).

Register addresses

All Ethernet Controller register addresses are given as address offsets to the individual base address <ETHAn_base>.
The <ETHAn_base> address of ETHAn are listed in the following table:

Table 31-2 Register base addresses <ETHAn_base>

ETHAn instance	<ETHAn_base> address
ETHA0	F093 2000_H

Clock supply

All Ethernet Controllers provide two clock inputs.

Table 31-3 ETHAn clock supply

ETHAn instance	ETHAn clock	Connected to
ETHA0	HCLK	Clock Generator CKSCLK_000 / 2

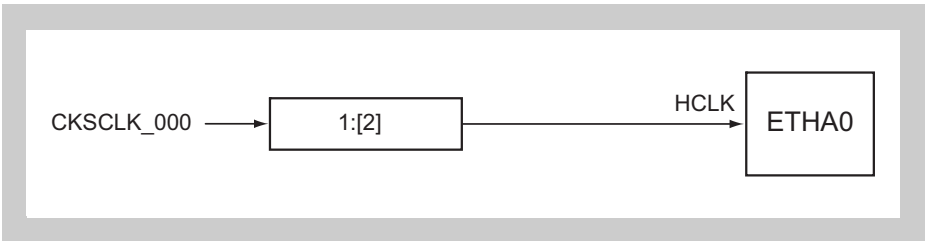


Figure 31-1 ETHA clock supply

Interrupts and DMA/DTS

The Ethernet Controller can generate following interrupt and DMA/DTS requests:

Table 31-4 ETHAn interrupt and DMA/DTS requests

ETHAn signals	Function	Connected to
ETHA0:		
INTETMRQ	Receive data and ready	Interrupt Controller INTETHA0SRX DMA Controller trigger 101 DTS Controller trigger 26
INTETMRX	Packet reception	Interrupt Controller INTETHA0SCRX DMA Controller trigger 102 DTS Controller trigger 27
INTETMTX	Packet transmission	Interrupt Controller INTETHA0SCTX DMA Controller trigger 103 DTS Controller trigger 28
INTIRQSC TX_TCH	Data calculation completion	Interrupt Controller INTETHA0SCTXTCH
INTIRQSC RX_TCH	Checksum data transmission	Interrupt Controller INTETHA0SCRXTCH
INTETMRS	Reception status	Interrupt Controller INTETHA0RS DMA Controller trigger 104 DTS Controller trigger 29
INTETMTS	Transmission status	Interrupt Controller INTETHA0TS DMA Controller trigger 105 DTS Controller trigger 30
INTETMFS	FIFO status	Interrupt Controller INTETHA0FS DMA Controller trigger 106 DTS Controller trigger 31
INTETMOV	MAC core	Interrupt Controller INTETHA0MAC

ETHA H/W reset

The Ethernet Controllers and their registers are initialized by the following reset signal:

Table 31-5 ETHAn reset signals

ETHAn	Reset signal
ETHA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals

The I/O signals of the Ethernet Controller are listed in the table below.
All signals are connected to ports.

Table 31-6 ETHAn I/O signals

ETHA0 signals	Connected to
Transmission clock	Port ETHA0TXCLK
Transmission data	Port ETHA0TXD[3:0]
Transmission data enable	Port ETHA0TXEN
Transmission error	Port ETHA0TXER
Collision detection	Port ETHA0COL
Carrier detection	Port ETHA0CRSDV
Reception clock	Port ETHA0RXCLK
Reception data	Port ETHA0RXD[3:0]
Reception data valid	Port ETHA0RXDV
Reception error	Port ETHA0RXER

31.2 General

The Ethernet Controller includes a 10/100 Mbps Ethernet Media Access Controller (MAC) conforming to IEEE802.3, a FIFO controller for flow control, and a checksum calculation unit (only for received packets) conforming to RFC1071.

31.2.1 Functions

(1) MAC

- 10/100 Mbps full-duplex communication, half-duplex communication, and flow control conforming to IEEE802.3 supported
- MII supported as physical layer device (PHY) interface
- Accessing PHY registers via serial management interface supported
- Statistics counter to support RMON/SNMP (RFC2665, RFC2819)
- Packet filtering based on address types
- VLAN frame detection

(2) FIFO

- Transmit/receive FIFO size: Transmit FIFO = 2 KB, receive FIFO = 2 KB
- FIFO status register
- Interrupts generated according to transmission/reception status and FIFO status

(3) DMAC for Ethernet Controller

- Data transfer (DMA)
- Reception status DMA transfer
- Reading (in pointer chain format), analyzing, and writing back buffer descriptors
- Controlling interrupts in packet transfers

(4) Checksum calculation

- Transmit checksum calculation function conforming to RFC1071

  DMAC for transmit checksum can calculate multiple checksums successively and save the result to any address.
- Receive checksum calculation conforming to RFC1071

  The MAC header and FCS of a received packet are automatically identified and the checksum for verifying the received packet (excluding the dummy header) is generated.

31.3 Configuration

31.3.1 System configuration

The Ethernet Controller transmits and receives data by using a dedicated direct memory access controller (DMAC). The Ethernet Controller supports the MII (Media Independent Interface) of IEEE802.3 and can create a 10 Mbps or 100 Mbps Ethernet environment when it is connected to a PHY device conforming to MII. In addition, data can be communicated in full-duplex or half-duplex mode, which can be selected.

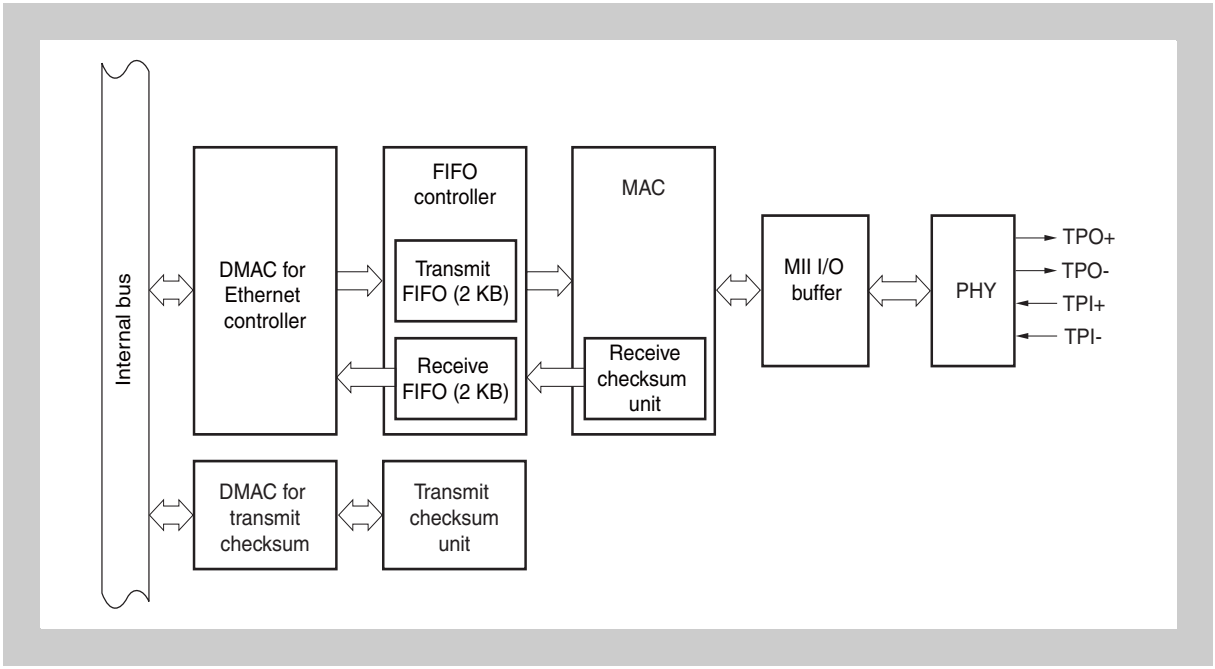


Figure 31-2 Configuration of Ethernet Controller

(1) MAC

This unit has MAC functions and supports MII-based interfacing with an external PHY device.

- Receive checksum unit
  This unit calculates the receive checksum.

(2) FIFO controller

This unit controls the transmit/receive FIFO buffers.

2 KB FIFO buffers are separately available for transmission and reception.

(3) DMAC for Ethernet Controller

This DMA controller controls data transmission and reception with the internal bus.

Caution The DMAC for the Ethernet Controller processes all the data the Ethernet Controller transmits and receives. Data cannot be transmitted or received in packet units by reading or writing a register.

(4) Transmit Checksum DMAC

This DMA controller that interfaces with the internal bus is dedicated to the transmit checksum function.

(5) Transmit checksum unit

This unit can only calculate the transmit checksum. Independent of the data transmission/reception interface, the transmit checksum unit calculates and transfers transmit data by using the transmit checksum DMA function and descriptor.

31.3.2 Interrupt requests and sources

The Ethernet Controller interrupt requests and their sources are listed below.

Table 31-7 Interrupt requests

Interrupt request		Interrupt source
INTETMRQ	Ethernet receive data ready interrupt	Received packet read request
INTETMRX	Ethernet packet reception interrupt	Packet reception (DMA) completion interrupt (RXI)
		Reception (DMA) end of chain interrupt (RECI)
		Receive data buffer access error interrupt (RBEI)
		Pause interrupt (RUPI) triggered by the “U” (used) bit in the receive descriptor
INTETMTX	Ethernet packet transmission interrupt	Packet transmission (DMA) completion interrupt (TXI)
		Transmission (DMA) end of chain interrupt (TECI)
		Transmit data buffer access error interrupt (TBEI)
		Pause interrupt (TUPI) triggered by the “U” (used) bit in the transmit descriptor
INTETMFS	Ethernet FIFO status interrupt	FIFO status (FSTATUS) interrupt
INTETMTS	Ethernet transmission status interrupt	Transmission status (TXSTATUS) interrupt
INTETMRS	Ethernet reception status interrupt	Reception status (RXSTATUS) interrupt
INTETMOV	Ethernet MAC interrupt	Statistics counter overflow (CARRY status)
IRQSCTX_TCH	Ethernet transmit data calculation completion interrupt	1 transmit checksum calculation completion interrupt (TCH_TXI)
		All transmit checksums calculation completion interrupt (TCH_TECI)
		Transmit checksum buffer access error interrupt (TCH_TBEI)
		Transmit checksum calculation pause interrupt (TCH_RUPI)
IRQSCRX_TCH	Ethernet transmit checksum interrupt	1 transmit checksum writing completion interrupt (TCH_RXI)
		All transmit checksums calculation writing completion interrupt (TCH_RECI)
		Data write error interrupt (TCH_RBEI)

- Each interrupt source can be masked. If an interrupt source is generated while the interrupt is masked, the corresponding bit in the status register is set but the interrupt request is not generated.
- It is recommended to read the interrupt register if multiple sources have been generated concurrently.

31.4 Registers for Controlling the Ethernet Controller

(1) Register setting procedure

To update the values of the control registers, make sure that transmission and reception of frames and DMA are stopped.

If these registers are updated while frames are being transmitted or received, or while DMA is in progress, the operation is not guaranteed.

Caution When using the Ethernet Controller, the Ethernet control register (MIICTL) must be set up first. For details, refer to *1 "ETHAnMFFCONT - FIFO controller control register" in 31.4.4 "FIFO controller control registers"*.

31.4.1 Ethernet Controller register overview

Table 31-8 ETHA registers overview (MAC) (1/3)

Register name	Shortcut	Address
MAC setting register 1	ETHAnMACC1	<ETHAn_base> + 0000_H
MAC setting register 2	ETHAnMACC2	<ETHAn_base> + 0004_H
Back-to-back IPG register	ETHAnIPGT	<ETHAn_base> + 0008_H
Non back-to-back IPG register	ETHAnIPGR	<ETHAn_base> + $000C_H$
Collision register	ETHAnCLRT	<ETHAn_base> + 0010_H
Maximum packet length register	ETHAnLMAX	<ETHAn_base> + 0014_H
Station address register 1	ETHAnLSA1	<ETHAn_base> + 0054_H
Station address register 2	ETHAnLSA2	<ETHAn_base> + 0058_H
Pause timer value read register	ETHAnPTVR	<ETHAn_base> + $005C_H$
VLAN type register	ETHAnVLTP	<ETHAn_base> + 0064_H
MII configuration register	ETHAnMIIC	<ETHAn_base> + 0080_H
MII command register	ETHAnMCMD	<ETHAn_base> + 0094_H
MII address register	ETHAnMADR	<ETHAn_base> + 0098_H
MII write data register	ETHAnMWTD	<ETHAn_base> + $009C_H$
MII read data register	ETHAnMRDD	<ETHAn_base> + $00A0_H$
MII indicator register	ETHAnMIND	<ETHAn_base> + $00A4_H$
Address filter register	ETHAnAFR	<ETHAn_base> + $00C8_H$
Hash table register 1	ETHAnHT1	<ETHAn_base> + $00CC_H$
Hash table register 2	ETHAnHT2	<ETHAn_base> + $00D0_H$
Carry register 1	ETHAnCAR1	<ETHAn_base> + $00DC_H$
Carry register 2	ETHAnCAR2	<ETHAn_base> + $00E0_H$
Carry mask register 1	ETHAnCAM1	<ETHAn_base> + 0130_H
Carry mask register 2	ETHAnCAM2	<ETHAn_base> + 0134_H
Reception byte counter	ETHAnRBYT	<ETHAn_base> + 0140_H

Table 31-8 ETHA registers overview (MAC) (2/3)

Register name	Shortcut	Address
Reception packet counter	ETHAnRPKT	<ETHAn_base> + 0144_H
Reception FCS error frame counter	ETHAnRFCS	<ETHAn_base> + 0148_H
Reception multicast packet counter	ETHAnRMCA	<ETHAn_base> + $014C_H$
Reception broadcast packet counter	ETHAnRBCA	<ETHAn_base> + 0150_H
Reception control frame packet counter	ETHAnRXCF	<ETHAn_base> + 0154_H
Reception pause frame packet counter	ETHAnRXPF	<ETHAn_base> + 0158_H
Reception undefined control packet counter	ETHAnRXUO	<ETHAn_base> + $015C_H$
Reception alignment error counter	ETHAnRALN	<ETHAn_base> + 0160_H
Reception frame length error counter	ETHAnRFLR	<ETHAn_base> + 0164_H
Reception code error counter	ETHAnRCDE	<ETHAn_base> + 0168_H
Reception false carrier counter	ETHAnRFCR	<ETHAn_base> + $016C_H$
Reception undersize packet counter	ETHAnRUND	<ETHAn_base> + 0170_H
Reception oversize packet counter	ETHAnROVR	<ETHAn_base> + 0174_H
Reception fragment counter	ETHAnRFRG	<ETHAn_base> + 0178_H
Reception jabber counter	ETHAnRJBR	<ETHAn_base> + $017C_H$
Receive 64-byte frame counter	ETHAnR64	<ETHAn_base> + 0180_H
Receive 65- to 127-byte frame counter	ETHAnR127	<ETHAn_base> + 0184_H
Receive 128- to 255-byte frame counter	ETHAnR255	<ETHAn_base> + 0188_H
Receive 256- to 511-byte frame counter	ETHAnR511	<ETHAn_base> + $018C_H$
Receive 512- to 1023-byte frame counter	ETHAnR1K	<ETHAn_base> + 0190_H
Receive 1024- to RMAX-byte frame counter	ETHAnRMAX	<ETHAn_base> + 0194_H
Receive valid byte counter	ETHAnRVBT	<ETHAn_base> + 0198_H
Transmission byte counter	ETHAnTBYT	<ETHAn_base> + $01C0_H$
Transmission packet counter	ETHAnTPKT	<ETHAn_base> + $01C4_H$
Transmission FCS error frame counter	ETHAnTFCS	<ETHAn_base> + $01C8_H$
Transmission multicast packet counter	ETHAnTMCA	<ETHAn_base> + $01CC_H$
Transmission broadcast packet counter	ETHAnTBCA	<ETHAn_base> + $01D0_H$
Transmission unicast packet counter	ETHAnTUCA	<ETHAn_base> + $01D4_H$

Table 31-8 ETHA registers overview (MAC) (3/3)

Register name	Shortcut	Address
Transmission pause control frame counter	ETHAnTXPF	<ETHAn_base> + $01D8_H$
Transmission delay packet counter	ETHAnTDFR	<ETHAn_base> + $01DC_H$
Transmission excessive delay packet counter	ETHAnTXDF	<ETHAn_base> + $01E0_H$
Transmission single collision packet counter	ETHAnTSCL	<ETHAn_base> + $01E4_H$
Transmission multiple collision packet counter	ETHAnTMCL	<ETHAn_base> + $01E8_H$
Transmission late collision packet counter	ETHAnTLCL	<ETHAn_base> + $01EC_H$
Transmission excessive collision packet counter	ETHAnTXCL	<ETHAn_base> + $01F0_H$
Transmission total collision counter	ETHAnTNCL	<ETHAn_base> + $01F4_H$
Transmission carrier sense error counter	ETHAnTCSE	<ETHAn_base> + $01F8_H$
MAC internal error counter	ETHAnTIME	<ETHAn_base> + $01FC_H$

Table 31-9 ETHA FIFO controller registers overview (1/2)

Register name	Shortcut	Address
FIFO controller control register	ETHAnMFFCONT	<ETHAn_base> + 0200_H
Software reset control register	ETHAnRSTCNT	<ETHAn_base> + 0204_H
Flow control threshold value register	ETHAnFLOWTHRESH	<ETHAn_base> + 0218_H
Pause timer value register	ETHAnPAUSETM	<ETHAn_base> + $021C_H$
Receive error selection register	ETHAnRXERSEL	<ETHAn_base> + 0220_H
Transmission status monitor 1 register	ETHAnTXSTMONI1	<ETHAn_base> + 0230_H
Transmission status monitor 2 register	ETHAnTXSTMONI2	<ETHAn_base> + 0234_H
Transmission status 1 register	ETHAnTXFINF1	<ETHAn_base> + 0238_H
Transmission status 2 register	ETHAnTXFINF2	<ETHAn_base> + $023C_H$
Reception status monitor register	ETHAnRXSTMONI	<ETHAn_base> + 0240_H
Reception status 1 register	ETHAnRXFINF1	<ETHAn_base> + 0244_H
Reception status 2 register	ETHAnRXFINF2	<ETHAn_base> + 0248_H
Reception status 3 register	ETHAnRXFINF3	<ETHAn_base> + $024C_H$
FIFO status interrupt register	ETHAnFSTATUS	<ETHAn_base> + 0250_H
FIFO status interrupt mask register	ETHAnFSTATUSM	<ETHAn_base> + 0254_H
Transmission status interrupt register	ETHAnTXSTATUS	<ETHAn_base> + 0258_H
Transmission status interrupt mask register	ETHAnTXSTATUSM	<ETHAn_base> + $025C_H$

Table 31-9 ETHA FIFO controller registers overview (2/2)

Register name	Shortcut	Address
Reception status interrupt register	ETHAnRXSTATUS	<ETHAn_base> + 0260_H
Reception status interrupt mask register	ETHAnRXSTATUSM	<ETHAn_base> + 0264_H
TX abort counter	ETHAnTXABTCNT	<ETHAn_base> + 0270_H
RX abort counter	ETHAnRXABTCNT	<ETHAn_base> + 0274_H

Table 31-10 ETHA DMAC registers overview

Register name	Shortcut	Address
Core function setting register	ETHAnMODE	<ETHAn_base> + 0300_H
Interrupt register	ETHAnINTMS	<ETHAn_base> + 0304_H
Transmission control register	ETHAnTRANSCTL	<ETHAn_base> + 0308_H
Software reset register	ETHAnSFTRST	<ETHAn_base> + $030C_H$
DMA controller mode setting register	ETHAnDMACM	<ETHAn_base> + 0310_H
Receive descriptor pointer register	ETHAnRXDP	<ETHAn_base> + 0320_H
Last receive descriptor pointer register	ETHAnLSTRXDP	<ETHAn_base> + 0324_H
Transmit descriptor pointer register	ETHAnTXDP	<ETHAn_base> + 0328_H
Last transmit descriptor pointer register	ETHAnLSTTXDP	<ETHAn_base> + $032C_H$
Transmit checksum unit function setting register	ETHAnTCHMODE	<ETHAn_base> + 1300_H
Transmit checksum interrupt register	ETHAnTCHINTMS	<ETHAn_base> + 1304_H
Transmit checksum transfer control register	ETHAnTCHTRANSCTL	<ETHAn_base> + 1308_H
Transmit checksum software reset register	ETHAnTCHSFTRST	<ETHAn_base> + $130C_H$
Transmit checksum DMA control mode setting register	ETHAnTCHDMACM	<ETHAn_base> + 1310_H
Transmit checksum receive descriptor pointer	ETHAnTCHRXDP	<ETHAn_base> + 1314_H
Transmit checksum last receive descriptor pointer	ETHAnTCHLSTRXDP	<ETHAn_base> + 1318_H
Transmit checksum transmit descriptor pointer	ETHAnTCHTXDP	<ETHAn_base> + $131C_H$
Transmit checksum last transmit descriptor pointer	ETHAnTCHLSTTXDP	<ETHAn_base> + 1320_H

31.4.2 MAC control registers

(1) ETHAnMACC1 - MAC setting register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0000_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to execute a software reset after setting the operation mode. To execute a software reset, set the ETHAnMCRST, ETHAnRFRST, and ETHAnTFRST bits of the ETHAnMACC2 register at the same time. Cancel the software reset by clearing these bits at the same time.
2. Be sure to set bits 31 to 15, 13, 12, and 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	ETHAn MACLB	0	0	ETHAn TXFC	ETHAn RXFC	ETHAn SRXEN	ETHAn PARF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn PUREP	ETHAn FLCHT	ETHAn NOBO	0	ETHAn CRCEN	ETHAn PADEN	ETHAn FULLD	ETHAn HUGEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-11 ETHAnMACC1 register contents (1/2)

Bit position	Bit name	Function
14	ETHAn MACLB	MAC loopback 0: Disables loopback operation. 1: Operation loops back from transmission block to reception block in MAC. To execute the loopback operation, set the ETHAnFULLD bit to enable full-duplex operation.
11	ETHAnn TXFC	Transmission flow control enable 0: Disables transmission of a pause control frame executed by inputting the TPCF signal. 1: Enables transmission of a pause control frame executed by inputting the TPCF signal.
10	ETHAnn RXFC	Reception flow control enable 0: A pause operation is not executed. 1: A pause operation is executed for the pause period set to the pause timer. The value of the pause timer is updated, regardless of the setting of this bit, when a valid pause control frame is received.

Table 31-11 ETHAnMACC1 register contents (2/2)

Bit position	Bit name	Function
9	ETHAn SRXEN	Reception enable 0: Reception is disabled. 1: The function of the reception data interface is enabled. If the setting of this bit is changed while the CRS signal is asserted, the new setting becomes valid after the CRS signal has been deasserted, regardless of the setting of the ETHAnFULLD bit.
8	ETHAn PARF	Control packet pass 0: A control frame is judged as a control frame. 1: No received packet, including a control frame, is judged as a control frame. The value of the pause timer is not updated even if a valid pause control frame is received, regardless of the setting of the ETHAnRXFC bit.
7	ETHAn PUREP	Pure preamble 0: The data of a preamble is not checked. 1: A reception status interrupt is generated if an illegal preamble is detected.
6	ETHAn FLCHT	Length field check 0: The length field is not checked. 1: The value of the length field and data field length are checked, and a status interrupt is generated.
5	ETHAn NOBO	No backoff 0: Packets are transmitted by using the backoff algorithm. 1: Packets are always transmitted without using the backoff algorithm.
3	ETHAn CRCEN	CRC appending 0: CRC is not appended. The end of the transmitted packet must be a valid frame check sequence (FCS). The MAC checks the FCS, and, if the FCS value is not correct, the MAC generates a transmission status interrupt (INTCTS) to report an error. 1: CRC is automatically appended to the end of a packet. An internally generated frame check sequence (FCS) is appended to the end of the transmit packet.
2	ETHAn PADEN	PAD appending 0: PAD is not appended. 1: PAD is appended to packets that are less than 64 bytes long. At this time, CRC is automatically appended to the end of the packet regardless of the setting of the ETHAnCRCEN bit.
1	ETHAn FULLD	Full-duplex enable 0: Half-duplex operation 1: Full-duplex operation
0	ETHAn HUGEN	Huge packet enable 0: Transmission/reception of a packet that exceeds the value of the maximum packet length register (ETHAnLMAX) is stopped. 1: Transmission/reception of a packet that exceeds the value of the maximum packet length register (ETHAnLMAX) is not stopped.

(2) ETHAnMACC2 - MAC setting register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0004_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to execute a software reset after setting the operation mode. To execute a software reset, set the ETHAnMCRST, ETHAnRFRST, and ETHAnTFRST bits of the ETHAnMACC2 register at the same time. Cancel the software reset by clearing these bits at the same time.

   Manipulate these reset bits at an interval of five or more RXCLK or TXCLK cycles.

2. Be sure to set bits 31 to 11, 7, and 3 to 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	ETHAn MCRST	ETHAn RFRST	ETHAn TFRST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	ETHAn BPNB	ETHAn APD	ETHAn VPD	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-12 ETHAnMACC2 register contents

Bit position	Bit name	Function
10	ETHAn MCRST	MAC control block software reset 0: Cancels a software reset for the MAC control block. 1: Executes a software reset for the MAC control block.
9	ETHAn RFRST	Reception block software reset 0: Cancels a software reset for the reception block. 1: Executes a software reset for the reception block.
8	ETHAn TFRST	Transmission block software reset 0: Cancels a software reset for the transmission block. 1: Executes a software reset for the transmission block.
6	ETHAn BPNB	No backoff after back pressure When this bit is set, backoff is not performed for a transmission after back pressure.
5	ETHAn APD	Auto VLAN PAD If a packet that matches the VLAN type registered to the ETHAnVLTP register is transmitted, it is treated as a VLAN packet and PAD is appended.
4	ETHAn VPD	VLAN pad mode The packet to be transmitted is always treated as a VLAN packet and PAD is appended.

(3) ETHAnIPGT - Back-to-back IPG register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0008_H

Initial value $0000\ 0013_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 7 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	ETHAnIPGT[6:0]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-13 ETHAnIPGT register contents

Bit position	Bit name	Function
6 to 0	ETHAn IPGT[6:0]	IPG in back-to-back transmission These bits specify the gap between packets (inter-packet gap (IPG)) in back-to-back transmission. The expression used to calculate the IPG is as follows. • IPG = (5 + ETHAnIPGT) x time required to transmit 4 bits (Time required to transmit 1 bit = 100 ns when the data rate is 10 Mbps or 10 ns when the data rate is 100 Mbps) Set the IPG to the time required to transmit at least 96 bits to satisfy the specification of IEEE802.3 (refer to *5 “Inter-packet gap (IPG)” in 31.5.2 “Frame transmission”*).

(4) ETHAnIPGR - Non back-to-back IPG register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $000C_H$

Initial value $0000\ 0E13_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 15 and 7 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	ETHAnIPGR1[6:0]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	ETHAnIPGR2[6:0]						
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-14 ETHAnIPGR register contents

Bit position	Bit name	Function
14 to 8	ETHAn IPGR1[6:0]	Carrier sense period These bits specify the carrier sense period of the first half of the IPG in transmission other than back-to-back transmission. The calculation expression used to calculate the carrier sense period is as follows. • Carrier sense period = (2 + ETHAnIPGR1) x time required to transmit 4 bits Set the carrier sense period to 2/3IPG to satisfy the specification of IEEE802.3 (refer to *5 “Inter-packet gap (IPG)” in 31.5.2 “Frame transmission”*).
6 to 0	ETHAn IPGR2[6:0]	IPG in transmission other than back-to-back transmission These bits specify the IPG in transmission other than back-to-back transmission. The expression used to calculate the IPG is as follows. • IPG = (5 + ETHAnIPGR2) x time required to transmit 4 bits The carrier sense period specified by the ETHAnIPGR1 bits is included in the IPG specified by the ETHAnIPGR2 bits. Set the IPG to the time required to transmit at least 96 bits to satisfy the specification of IEEE802.3 (refer to *5 “Inter-packet gap (IPG)” in 31.5.2 “Frame transmission”*).

(5) ETHAnCLRT - Collision register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0010_H

Initial value $0000\ 380F_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 14 and 7 to 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	ETHAnLCOL[5:0]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	ETHAnRETRY[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-15 ETHAnCLRT register contents

Bit position	Bit name	Function
13 to 8	ETHAn LCOL[5:0]	Collision window These bits specify the collision window width. The width of the collision window to be set is calculated by the following expression. • Collision window width = (ETHAnLCOL + 8) x time required to transmit 8 bits The IEEE802.3 defines the collision window width as the time required to transmit 512 bits.
3 to 0	ETHAn RETRY[3:0]	Maximum number of times of retransmission in case of collision These bits specify the maximum number of times to attempt retransmission when a collision occurs. If retransmission does not finish within the value specified by these bits, transmission is aborted. This value indicates the maximum number of collisions. The IEEE802.3 defines the maximum number of collisions as 15.

(6) ETHAnLMAX - Maximum packet length register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0014_H

Initial value $0000\ 0600_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnMAXF[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnMAXF[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-16 ETHAnLMAX register contents

Bit position	Bit name	Function
15 to 0	ETHAn MAXF[15:0]	Maximum packet length (bytes) When the ETHAnMACC1.ETHAnHUGEN bit is 0, the transmit and receive packet length is limited by the value specified by these bits. During reception:Reception is terminated immediately when the receive frame length exceeds the value specified by the ETHAnMAXF[15:0] bits. During transmission:Transmission is aborted immediately when the transmit frame length exceeds the value specified by the ETHAnMAXF[15:0] bits.

(7) ETHAnLSA1 - Station address register 1

This register is used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used. This register is used in combination with the ETHAnLSA2 register as a 48-bit register.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0054_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnLSA1[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnLSA1[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-17 ETHAnLSA1 register contents

Bit position	Bit name	Function
15 to 0	ETHAn LSA1[15:0]	Station address (SA) (47:32) The SA bits (47:0) are used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used (refer to *a “Filtering of unicast addresses” in 1 “Overview of address filtering” of 31.5.6 “Address filtering”*).

(8) ETHAnLSA2 - Station address register 2

This register is used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used. This register is used in combination with the ETHAnLSA1 register as a 48-bit register.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0058_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnLSA2[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnLSA2[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnLSA2[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnLSA2[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-18 ETHAnLSA2 register contents

Bit position	Bit name	Function
31 to 0	ETHAn LSA2[31:0]	Station address (SA) (31:0) The SA bits (47:0) are used to compare a source address when a pause control frame is assembled and a destination address when address filtering is used (refer to *a “Filtering of unicast addresses” in 1 “Overview of address filtering” of 31.5.6 “Address filtering”*).

(9) ETHAnPTVR - Pause timer value read register

This register is used to read the value of the pause timer counter.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $005C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnPTCT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnPTCT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-19 ETHAnPTVR register contents

Bit position	Bit name	Function
15 to 0	ETHAn PTCT[15:0]	Pause timer counter These bits indicate the value currently set in the pause timer. The value of this register is valid only when reception flow control is enabled (the ETHAnMACC1.ETHAnRXFC bit is 1) (refer to *1 “Flow control” in 31.5.4 “MAC control function”*).

(10) ETHAnVLTP - VLAN type register

This register is used to specify the operation to be performed on a VLAN frame.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0064_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnVLTP[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnVLTP[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-20 ETHAnVLTP register contents

Bit position	Bit name	Function
15 to 0	ETHAn VLTP[15:0]	VLAN frame operation These bits specify the operation to be performed or a VLAN frame (refer to *3 “Operations related to VLAN frame” in 31.5.4 “MAC control function”*). During reception: The value of ETHAnVLTP is compared with the value of the TPID field (2 bytes following the source address) of a frame to detect a VLAN frame. During transmission: If the value of the VLAN field matches the value of ETHAnVLTP when the ETHAnMACC2.ETHAnAPD bit is 1, PAD is appended to the VLAN frame.

(11) ETHAnMIIC - Serial management interface configuration register

This register is used to set the operation mode of the serial management interface block.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0080_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Manipulate the ETHAnMIRST bit at an interval of five or more Ethernet Controller clock cycles (f_{EC}).
2. Be sure to set bits 31 to 16, 14 to 5, and 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn MIRST	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	ETHAn CLKS[2:0]			ETHAn PHYSEL	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-21 ETHAnMIIC register contents

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>ETHAn
MIRST</td><td>Serial management interface block software reset
0: Cancels a software reset for the serial management interface block.
1: Executes a software reset for the serial management interface block.</td></tr>
<tr><td>4 to 2</td><td>ETHAn
CLKS[2:0]</td><td>MDC division ratio
These bits select a division ratio according to an Ethernet Controller clock (f_{EC}) to be used (refer to a “MDC clock” in 1 “Overview of serial management interface” of 31.5.5 “Serial management interface”).
To satisfy the specification of IEEE802.3, set a division ratio so that the MDC frequency is 2.5 MHz or less.
<table>
<tr><th>ETHAnCLKS[2:0]</th><th>Input frequency of f_{EC}</th></tr>
<tr><td>001_B</td><td>33 MHz or less (division ratio: 14)</td></tr>
<tr><td>010_B</td><td>50 MHz or less (division ratio: 20)</td></tr>
<tr><td>011_B</td><td>66 MHz or less (division ratio: 28)</td></tr>
<tr><td>100_B</td><td>100 MHz or less (division ratio: 40)</td></tr>
<tr><td>Other than above</td><td>Setting prohibited</td></tr>
</table>
</td></tr>
<tr><td>1</td><td>ETHAn
PHYSEL</td><td>MDC output setting
Set this bit if data is not correctly transferred during communication with PHY when the MDC is stopped.
1: The MDC is always output for any frames other than the management frame.
0: The MDC is stopped for frames other than the management frame.</td></tr>
</table>

(12) ETHAnMCMD - MII command register

This register is used to read an external PHY device by using the SCAN command and MII management interface.

Access This register can be read/written in 32-bit units.

The value written to the ETHAnMCMD register must be read from the ETHAnMIND register.

Address <ETHAn_base> + 0094_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 2 to “0”. Bits 1 and 0 can only be written.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	ETHAn SCANC	ETHAn RSTAT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-22 ETHAnMCMD register contents

Bit position	Bit name	Function
1	ETHAn SCANC	SCAN command When this bit is set, the SCAN command is executed.
0	ETHAn RSTAT	MII management read When this bit is set, the MII management interface reads the external PHY device.

(13) ETHAnMADR - MII address register

This register is used to set a PHY address and a PHY register address.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0098_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 13 and 7 to 5 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	ETHAnFIAD[4:0]				
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	ETHAnRGAD[4:0]				
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-23 ETHAnMADR register contents

Bit position	Bit name	Function
1	ETHAn FIAD[4:0]	PHY address These bits specify a PHY address. One Ethernet Controller can control up to 31 PHY devices.
0	ETHAn RGAD[4:0]	PHY register address These bits specify the address of the register to be accessed. The Ethernet Controller can access 32 16-bit registers in one PHY device.

(14) ETHAnMWTD - MII write data register

This register is used to set the data to be written to an external PHY device when the MII management interface writes a PHY device.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $009C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnCTLD[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnCTLD[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-24 ETHAnMWTD register contents

Bit position	Bit name	Function
15 to 0	ETHAn CTLD[15:0]	MII write data This is a write data field when the MII management interface writes an external PHY device.

(15) ETHAnMRDD - MII read data register

This register is used to read data that has been read from an external PHY device by the MII management interface.

Access This register can be read in 32-bit units.

Address <ETHAn_base> + $00A0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnPRSD[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAnPRSD[7:0]							
R	R	R	R	R	R	R	R

Table 31-25 ETHAnMRDD register contents

Bit position	Bit name	Function
15 to 0	ETHAn PRSD[15:0]	MII read data This is a read data field when the MII management interface reads an external PHY device.

(16) ETHAnMIND - MII indicator register

This register indicates the statuses of SCAN command execution and MII management interface access.

Access This register can be read in 32-bit units.

Address <ETHAn_base> + $00A4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 3 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	ETHAn NVALID	ETHAn SCANA	ETHAn BUSY
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-26 ETHAnMIND register contents

Bit position	Bit name	Function
2	ETHAn NVALID	SCAN command start status 1: The SCAN command is under execution and the first read access has not finished. 0: Normal status
1	ETHAn SCANA	SCAN command active 1: The SCAN command is under execution. 0: Normal status
0	ETHAn BUSY	BUSY 1: The MII management interface is accessing an external PHY device. 0: The MII management interface is not accessing an external PHY device.

(17) ETHAnAFR - Address filter register

This register is used to set the conditions under which a receive packet is received.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $00C8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 4 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	ETHAn PRO	ETHAn PRM	ETHAn AMC	ETHAn ABC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-27 ETHAnAFR register contents

Bit position	Bit name	Function
3	ETHAn PRO	Promiscuous mode All packets are valid in this mode.
2	ETHAn PRM	Multicast reception In this mode, all multicast packets are valid and other packets are discarded.
1	ETHAn AMC	Conditional multicast reception In this mode, multicast packets that satisfy the conditions are valid and other packets are discarded. Only multicast packets whose multicast address matches the values of the hash table specified by the ETHAnHT1 and ETHAnHT2 registers are received. The hash table is specified by the ETHAnHT1 and ETHAnHT2 registers.
0	ETHAn ABC	Broadcast reception In this mode, broadcast packets are valid and other packets are discarded.

For details of the settings of the ETHAnAFR register and the packets to be filtered, refer to *Table 31-28 “Settings of ETHAnAFR register and packets to be filtered”*.

Table 31-28 Settings of ETHAnAFR register and packets to be filtered

ETHAnAFR.ETHAn				Receive packet				
PRO	PRM	AMC	ABC	LSA mismatch, unicast	LSA match, unicast	HT mismatch, multicast	HT match, multicast	Broadcast packet
1	–	–	–	Received	Received	Received	Received	Received
0	1	–	–	Discarded				
	0	1	1			Discarded		
		1	0					Discarded[a]
		0	1				Discarded	Received
		0	0					Discarded

a) A broadcast address can be received if the corresponding bit in the hash table is set because a broadcast address is included in a multicast address.

Note –: Any

(18) ETHAnHT1 - Hash table register 1

This register is used to specify the hash table used for conditional multicast packet detection.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $00CC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnHT1[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnHT1[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnHT1[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnHT1[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-29 ETHAnHT1 register contents

Bit position	Bit name	Function
31 to 0	ETHAn HT1[31:0]	Hash table 1 The hash table is used for conditional multicast packet detection. These bits indicate the higher 32 bits of the hash table. HT (63:32)

(19) ETHAnHT2 - Hash table register 2

This register is used to specify the hash table used for conditional multicast packet detection.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $00D0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnHT2[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnHT2[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnHT2[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnHT2[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-30 ETHAnHT2 register contents

Bit position	Bit name	Function
31 to 0	ETHAn HT2[31:0]	Hash table 2 The hash table is used for conditional multicast packet detection. These bits indicate the lower 32 bits of the hash table. HT (31:0)

In conditional multicast packet detection mode, a multicast packet is received if the corresponding HT bit in *Table 31-31 "Correspondence between HT bits and hash values"* that is determined by a hash value (CRC[28:23]) is set (refer to *Figure 31-17 "Image of filtering by referencing hash table"*).

Table 31-31 Correspondence between HT bits and hash values

	CRC[25:23]							
CRC[28:26]	**111_B (7)**	**110_B (6)**	**101_B (5)**	**100_B (4)**	**011_B (3)**	**010_B (2)**	**001_B (1)**	**000_B (0)**
111_B (7)	HT1[31]	HT1[30]	HT1[29]	HT1[28]	HT1[27]	HT1[26]	HT1[25]	HT1[24]
110_B (6)	HT1[23]	HT1[22]	HT1[21]	HT1[20]	HT1[19]	HT1[18]	HT1[17]	HT1[16]
101_B (5)	HT1[15]	HT1[14]	HT1[13]	HT1[12]	HT1[11]	HT1[10]	HT1[9]	HT1[8]
100_B (4)	HT1[7]	HT1[6]	HT1[5]	HT1[4]	HT1[3]	HT1[2]	HT1[1]	HT1[0]
011_B (3)	HT2[31]	HT2[30]	HT2[29]	HT2[28]	HT2[27]	HT2[26]	HT2[25]	HT2[24]
010_B (2)	HT2[23]	HT2[22]	HT2[21]	HT2[20]	HT2[19]	HT2[18]	HT2[17]	HT2[16]
001_B (1)	HT2[15]	HT2[14]	HT2[13]	HT2[12]	HT2[11]	HT2[10]	HT2[9]	HT2[8]
000_B (0)	HT2[7]	HT2[6]	HT2[5]	HT2[4]	HT2[3]	HT2[2]	HT2[1]	HT2[0]

Note The bit name's prefix "ETHAn" is omitted in the above table, thus HT1[31:0] = ETHnHT1[31:0] and HT2[31:0] = ETHnHT2[31:0].

(20) ETHAnCAR1 - Carry register 1

This register indicates that a statistics counter has overflowed. Each bit of this register corresponds to a statistics counter. When a statistics counter overflows, the corresponding bit in this register is set.

Each bit is cleared when it is read.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $00DC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn C1VT	ETHAn C1UT	ETHAn C1BT	ETHAn C1MT	ETHAn C1PT	ETHAn C1TB	ETHAn C1MX	ETHAn C11K
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn C1FE	ETHAn C1TF	ETHAn C1OT	ETHAn C1SF	ETHAn C1BR	ETHAn C1MR	ETHAn C1PR	ETHAn C1RB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-32 ETHAnCAR1 register contents

Bit position	Bit name	Function
15	ETHAn C1VT	Overflow of ETHAnRVBT counter 0: Counter did not overflow. 1: Counter overflowed.
14	ETHAn C1UT	Overflow of ETHAnTUCA counter 0: Counter did not overflow. 1: Counter overflowed.
13	ETHAn C1BT	Overflow of ETHAnTBCA counter 0: Counter did not overflow. 1: Counter overflowed.
12	ETHAn C1MT	Overflow of ETHAnTMCA counter 0: Counter did not overflow. 1: Counter overflowed.
11	ETHAn C1PT	Overflow of ETHAnTPKT counter 0: Counter did not overflow. 1: Counter overflowed.
10	ETHAn C1TB	Overflow of ETHAnTBYT counter 0: Counter did not overflow. 1: Counter overflowed.
9	ETHAn C1MX	Overflow of ETHAnRMAX counter 0: Counter did not overflow. 1: Counter overflowed.
8	ETHAn C11K	Overflow of ETHAnR1K counter 0: Counter did not overflow. 1: Counter overflowed.
7	ETHAn C1FE	Overflow of ETHAnR511 counter 0: Counter did not overflow. 1: Counter overflowed.
6	ETHAn C1TF	Overflow of ETHAnR255 counter 0: Counter did not overflow. 1: Counter overflowed.
5	ETHAn C1OT	Overflow of ETHAnR127 counter 0: Counter did not overflow. 1: Counter overflowed.
4	ETHAn C1SF	Overflow of ETHAnR64 counter 0: Counter did not overflow. 1: Counter overflowed.
3	ETHAn C1BR	Overflow of ETHAnRBCA counter 0: Counter did not overflow. 1: Counter overflowed.
2	ETHAn C1MR	Overflow of ETHAnRMCA counter 0: Counter did not overflow. 1: Counter overflowed.
1	ETHAn C1PR	Overflow of ETHAnRPKT counter 0: Counter did not overflow. 1: Counter overflowed.
0	ETHAn C1RB	Overflow of ETHAnRBYT counter 0: Counter did not overflow. 1: Counter overflowed.

(21) ETHAnCAR2 - Carry register 2

This register indicates that a statistics counter has overflowed. Each bit of this register corresponds to a statistics counter. When a statistics counter overflows, the corresponding bit in this register is set.

Each bit is cleared when it is read.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $00E0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 30 to 23 to “0”.

31	30	29	28	27	26	25	24
ETHAnC2DV	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	ETHAnC2IM	ETHAnC2CS	ETHAnC2NC	ETHAnC2XC	ETHAnC2LC	ETHAnC2MC	ETHAnC2SC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnC2XD	ETHAnC2DF	ETHAnC2XF	ETHAnC2TE	ETHAnC2JB	ETHAnC2FG	ETHAnC2OV	ETHAnC2UN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnC2FC	ETHAnC2CD	ETHAnC2FO	ETHAnC2AL	ETHAnC2UO	ETHAnC2PF	ETHAnC2CF	ETHAnC2RE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-33 ETHAnCAR2 register contents (1/2)

Bit position	Bit name	Function
31	ETHAn C2DV	Overrunning of status vector 0: Status vector does not overrun. 1: Status vector overruns.
22	ETHAn C2IM	Overflow of ETHAnTIME counter 0: Counter did not overflow. 1: Counter overflowed.
21	ETHAn C2CS	Overflow of ETHAnTCSE counter 0: Counter did not overflow. 1: Counter overflowed.
20	ETHAn C2NC	Overflow of ETHAnTNCL counter 0: Counter did not overflow. 1: Counter overflowed.
19	ETHAn C2XC	Overflow of ETHAnTXCL counter 0: Counter did not overflow. 1: Counter overflowed.
18	ETHAn C2LC	Overflow of ETHAnTLCL counter 0: Counter did not overflow. 1: Counter overflowed.
17	ETHAn C2MC	Overflow of ETHAnTMCL counter 0: Counter did not overflow. 1: Counter overflowed.
16	ETHAn C2SC	Overflow of ETHAnTSCL counter 0: Counter did not overflow. 1: Counter overflowed.
15	ETHAn C2XD	Overflow of ETHAnTXDF counter 0: Counter did not overflow. 1: Counter overflowed.
14	ETHAn C2DF	Overflow of ETHAnTDFR counter 0: Counter did not overflow. 1: Counter overflowed.
13	ETHAn C2XF	Overflow of ETHAnTXPF counter 0: Counter did not overflow. 1: Counter overflowed.
12	ETHAn C2TE	Overflow of ETHAnTFCS counter 0: Counter did not overflow. 1: Counter overflowed.
11	ETHAn C2JB	Overflow of RBJR counter 0: Counter did not overflow. 1: Counter overflowed.
10	ETHAn C2FG	Overflow of ETHAnRFRG counter 0: Counter did not overflow. 1: Counter overflowed.
9	ETHAn C2OV	Overflow of ETHAnROVR counter 0: Counter did not overflow. 1: Counter overflowed.
8	ETHAn C2UN	Overflow of ETHAnRUND counter 0: Counter did not overflow. 1: Counter overflowed.
7	ETHAn C2FC	Overflow of ETHAnRFCR counter 0: Counter did not overflow. 1: Counter overflowed.

Table 31-33 ETHAnCAR2 register contents (2/2)

Bit position	Bit name	Function
6	ETHAn C2CD	Overflow of ETHAnRCDE counter 0: Counter did not overflow. 1: Counter overflowed.
5	ETHAn C2FO	Overflow of ETHAnRFLR counter 0: Counter did not overflow. 1: Counter overflowed.
4	ETHAn C2AL	Overflow of ETHAnRALN counter 0: Counter did not overflow. 1: Counter overflowed.
3	ETHAn C2UO	Overflow of ETHAnRXUO counter 0: Counter did not overflow. 1: Counter overflowed.
2	ETHAn C2PF	Overflow of ETHAnRXPF counter 0: Counter did not overflow. 1: Counter overflowed.
1	ETHAn C2CF	Overflow of ETHAnRXCF counter 0: Counter did not overflow. 1: Counter overflowed.
0	ETHAn C2RE	Overflow of ETHAnRFCS counter 0: Counter did not overflow. 1: Counter overflowed.

(22) ETHAnCAM1 - Carry mask register 1

This register is used to mask the CAINT signal that is generated when a statistics counter has overflowed and the corresponding bit in the ETHAnCAR1 register has been set.

Each bit of this register can be masked.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0130_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn M1VT	ETHAn M1UT	ETHAn M1BT	ETHAn M1MT	ETHAn M1PT	ETHAn M1TB	ETHAn M1MX	ETHAn M11K
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn M1FE	ETHAn M1TF	ETHAn M1OT	ETHAn M1SF	ETHAn M1BR	ETHAn M1MR	ETHAn M1PR	ETHAn M1RB
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-34 ETHAnCAM1 register contents

Bit position	Bit name	Function
15	ETHAn M1VT	ETHAnRVBT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
14	ETHAn M1UT	ETHAnTUCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
13	ETHAn M1BT	ETHAnTBCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
12	ETHAn M1MT	ETHAnTMCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
11	ETHAn M1PT	ETHAnTPKT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
10	ETHAn M1TB	ETHAnTBYT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
9	ETHAn M1MX	ETHAnRMAX counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
8	ETHAn M11K	ETHAnR1K counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
7	ETHAn M1FE	ETHAnR511 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
6	ETHAn M1TF	ETHAnR255 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
5	ETHAn M1OT	ETHAnR127 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
4	ETHAn M1SF	ETHAnR64 counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
3	ETHAn M1BR	ETHAnRBCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
2	ETHAn M1MR	ETHAnRMCA counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
1	ETHAn M1PR	ETHAnRPKT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
0	ETHAn M1RB	ETHAnRBYT counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

(23) ETHAnCAM2 - Carry mask register 2

This register is used to mask the CAINT signal that is generated when a statistics counter has overflowed and the corresponding bit in the ETHAnCAR2 register has been set.

Each bit of this register can be masked.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0134_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 30 to 23 to “0”.

31	30	29	28	27	26	25	24
ETHAn M2DV	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	ETHAn M2IM	ETHAn M2CS	ETHAn M2NC	ETHAn M2XC	ETHAn M2LC	ETHAn M2MC	ETHAn M2SC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn M2XD	ETHAn M2DF	ETHAn M2XF	ETHAn M2TE	ETHAn M2JB	ETHAn M2FG	ETHAn M2OV	ETHAn M2UN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn M2FC	ETHAn M2CD	ETHAn M2FO	ETHAn M2AL	ETHAn M2UO	ETHAn M2PF	ETHAn M2CF	ETHAn M2RE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-35 ETHAnCAM2 register contents (1/2)

Bit position	Bit name	Function
31	ETHAn M2DV	Status vector overrun interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
22	ETHAn M2IM	ETHAnTIME counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
21	ETHAn M2CS	ETHAnTCSE counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
20	ETHAn M2NC	ETHAnTNCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
19	ETHAn M2XC	ETHAnTXCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
18	ETHAn M2LC	ETHAnTLCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
17	ETHAn M2MC	ETHAnTMCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
16	ETHAn M2SC	ETHAnTSCL counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
15	ETHAn M2XD	ETHAnTXDF counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
14	ETHAn M2DF	ETHAnTDFR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
13	ETHAn M2XF	ETHAnTXPF counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
12	ETHAn M2TE	ETHAnTFCS counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
11	ETHAn M2JB	RBJR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
10	ETHAn M2FG	ETHAnRFRG counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
9	ETHAn M2OV	ETHAnROVR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
8	ETHAn M2UN	ETHAnRUND counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
7	ETHAn M2FC	ETHAnRFCR counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

Table 31-35 ETHAnCAM2 register contents (2/2)

Bit position	Bit name	Function
6	ETHAn M2CD	ETHAnRCDE counter overflow interrupt mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
5	ETHAn M2FO	ETHAnRFLR counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
4	ETHAn M2AL	ETHAnRALN counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
3	ETHAn M2UO	ETHAnRXUO counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
2	ETHAn M2PF	ETHAnRXPF counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
1	ETHAn M2CF	ETHAnRXCF counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).
0	ETHAn M2RE	ETHAnRFCS counter carry mask bit 0: Interrupt generation is enabled (not masked). 1: Interrupt generation is disabled (masked).

31.4.3 Statistics counters

(1) ETHAnRBYT - Reception byte counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0140_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRBYT[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRBYT[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRBYT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRBYT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-36 ETHAnRBYT register contents

Bit position	Bit name	Function
31 to 0	ETHAn RBYT[31:0]	This counter indicates the number of bytes in a received packet. It counts from the destination address byte to the FCS byte. It continues counting bytes even if an error occurs. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.ETHAnHUGEN bit is 0, this counter is incremented because the packet length is assumed to be that specified by the ETHAnLMAX register.

(2) ETHAnRPKT - Reception packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0144_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRPKT[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRPKT[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRPKT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRPKT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-37 ETHAnRPKT register contents

Bit position	Bit name	Function
31 to 0	ETHAn RPKT[31:0]	This counter is incremented when any packet, including packets in which an error has occurred, all unicast packets, all multicast packets, and broadcast packets, is received.

(3) ETHAnRFCS - Reception FCS error frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0148_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRFCS[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRFCS[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRFCS[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRFCS[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-38 ETHAnRFCS register contents

Bit position	Bit name	Function
31 to 0	ETHAn RFCS[31:0]	This counter is incremented if a CRC error occurs in a receive packet. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.ETHAnHUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register, so this counter might be incremented because the received packet is assumed to contain a CRC error.

(4) ETHAnRMCA - Reception multicast packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $014C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRMCA[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRMCA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRMCA{15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRMCA[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-39 ETHAnRMCA register contents

Bit position	Bit name	Function
31 to 0	ETHAn RMCA[31:0]	This counter is incremented when a multicast packet whose length is 64 to 1,518 bytes (64 to 1,522 bytes for a VLAN frame) is received. It is not incremented when a multicast packet in which a CRC error has occurred or a broadcast packet is received.

(5) ETHAnRBCA - Reception broadcast packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0150_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRBCA[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRBCA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRBCA[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRBCA[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-40 ETHAnRBCA register contents

Bit position	Bit name	Function
31 to 0	ETHAn RBCA[31:0]	This counter is incremented when a broadcast packet whose length is 64 to 1,518 bytes (64 to 1,522 bytes for a VLAN frame) is received. It is not incremented when a broadcast packet in which a CRC error has occurred or a multicast packet is received.

(6) ETHAnRXCF - Reception control frame packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0154_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRXCF[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRXCF[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRXCF[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRXCF[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-41 ETHAnRXCF register contents

Bit position	Bit name	Function
31 to 0	ETHAn RXCF[31:0]	This counter is incremented when any control frame, including pause control frames and unsupported control frames is received. It is not incremented when a control frame in which a CRC error has been detected is received.

(7) ETHAnRXPF - Reception pause frame packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0158_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRXPF[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRXPF[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRXPF[16:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRXPF[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-42 ETHAnRXPF register contents

Bit position	Bit name	Function
31 to 0	ETHA RXPF[31:0]	This counter is incremented when a valid pause control frame is received.

(8) ETHAnRXUO - Reception undefined control packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $015C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRXUO[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRXUO{23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRXUO[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRXUO[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-43 ETHAnRXUO register contents

Bit position	Bit name	Function
31 to 0	ETHAn RXUO[31:0]	This counter is incremented when a control frame that has an opcode other than PAUSE or a pause control frame that has an invalid destination address is received. It is not incremented when a pause control frame in which a CRC error has been detected is received.

(9) ETHAnRALNF - Reception alignment error counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0160_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRALN[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRALN[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRALN[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRALN[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-44 ETHAnRALNF register contents

Bit position	Bit name	Function
31 to 0	ETHAn RALN[31:0]	This counter is incremented if a CRC error and a dribble nibble occur in a received packet. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.ETHAnHUGEN bit is 0, an alignment error check is executed when the packet length reaches the length (in bytes) specified by the ETHAnLMAX register, so this counter is not incremented.

(10) ETHAnRFLR - Reception frame length error counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0164_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRFLR[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRFLR[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRFLR[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRFLR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-45 ETHAnRFLR register contents

Bit position	Bit name	Function
31 to 0	ETHAn RFLR[31:0]	This counter is incremented if the value of the length field of a receive packet does not match the data field length of a packet actually received. If the value of the length field is 1,501 or more (for example, when the bytes equivalent to the length field are used as the Ethernet type field), this counter is not incremented.

(11) ETHAnRCDE - Reception code error counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0164_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRCDE[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRCDE[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRCDE[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRCDE[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-46 ETHAnRCDE register contents

Bit position	Bit name	Function
31 to 0	ETHAn RCDE[31:0]	This counter is incremented if an illegal data symbol has been detected at least once while a carrier is being detected.

(12) ETHAnRFCR - Reception false carrier counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $016C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRFCR[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRFCR[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRFCR[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRFCR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-47 ETHAnRFCR register contents

Bit position	Bit name	Function
31 to 0	ETHAn RFCR[31:0]	If a false carrier is generated in the idle status, this counter is incremented after the next packet is received. It is assumed that a false carrier has occurred if 1110_B is input as nibble data from RXD while RXER is high. This counter is incremented only once even if multiple false carriers are generated in the idle status.

(13) ETHAnRUND - Reception undersize packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0170_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRUND[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRUND[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRUND[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRUND[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-48 ETHAnRUND register contents

Bit position	Bit name	Function
31 to 0	ETHAn RUND[31:0]	This counter is incremented if the receive packet length is less than 64 bytes and the packet contains a valid FCS field.

(14) ETHAnROVR - Reception oversize packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0174_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnROVR[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnROVR[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnROVR[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnROVR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-49 ETHAnROVR register contents

Bit position	Bit name	Function
31 to 0	ETHAn ROVR[31:0]	This counter is incremented if the receive packet length exceeds 1,518 bytes (1,522 bytes for a VLAN frame) and the packet contains a valid FCS field. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.ETHAnHUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register. Therefore, a CRC error might be detected and this counter might not be incremented.

(15) ETHAnRFRG - Reception fragment counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0178_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRFRG[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRFRG[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRFRG[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRFRG[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-50 ETHAnRFRG register contents

Bit position	Bit name	Function
31 to 0	ETHAn RFRG[31:0]	This counter is incremented if the receive packet length is less than 64 bytes and the packet contains a CRC error or an alignment error.

(16) ETHAnRJBR - Reception jabber counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $017C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRJBR[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRJBR[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRJBR[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRJBR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-51 ETHAnRJBR register contents

Bit position	Bit name	Function
31 to 0	ETHAn RJBR[31:0]	This counter is incremented if the receive packet length exceeds 1,518 bytes (1,522 bytes for a VLAN frame) and the packet contains a CRC error or an alignment error. If a packet longer than the value specified by the ETHAnLMAX register is received when the ETHAnMACC1.ETHAnHUGEN bit is 0, a CRC check is executed when the packet length reaches the value specified by the ETHAnLMAX register. Therefore, a CRC error might be detected and this counter might be incremented.

(17) ETHAnR64 - Receive 64-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0180_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnR64[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnR64[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnR64[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnR64[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-52 ETHAnR64 register contents

Bit position	Bit name	Function
31 to 0	ETHAn R64[31:0]	This counter is incremented if the receive packet length is 64 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(18) ETHAnR127 - Receive 65- to 127-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0184_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnR127[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnR127[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnR127[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnR127[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-53 ETHAnR127 register contents

Bit position	Bit name	Function
31 to 0	ETHAn R127[31:0]	This counter is incremented if the receive packet length is 64 to 127 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(19) ETHAnR255 - Receive 128- to 255-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0188_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnR255[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnR255[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnR255[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnR255[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-54 ETHAnR255 register contents

Bit position	Bit name	Function
31 to 0	ETHAn R255[31:0]	This counter is incremented if the receive packet length is 128 to 255 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(20) ETHAnR511 - Receive 256- to 511-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $018C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnR511[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnR511[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnR511[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnR511[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-55 ETHAnR511 register contents

Bit position	Bit name	Function
31 to 0	ETHAn R511[31:0]	This counter is incremented if the receive packet length is 256 to 511 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(21) ETHAnR1K - Receive 512- to 1023-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0190_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnR1K[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnR1K[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnR1K[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnR1K[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-56 ETHAnR1K register contents

Bit position	Bit name	Function
31 to 0	ETHAn R1K[31:0]	This counter is incremented if the receive packet length is 512 to 1,023 bytes. It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(22) ETHAnRMAX - Receive 1024- to RMAX-byte frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0194_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRMAX[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRMAX[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRMAX[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRMAX[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-57 ETHAnRMAX register contents

Bit position	Bit name	Function
31 to 0	ETHAn RMAX[31:0]	This counter is incremented if the receive packet length is 1,024 to 1,518 bytes (1,024 to 1,522 bytes for a VLAN frame). It is incremented even if the receive packet contains a CRC error, symbol error, or length/type error.

(23) ETHAnRVBT - Receive valid byte counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0198_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRVBT[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRVBT[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRVBT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRVBT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-58 ETHAnRVBT register contents

Bit position	Bit name	Function
31 to 0	ETHAn RVBT[31:0]	This counter indicates the byte count of a valid packet. It counts from the destination address byte to the FCS byte.

(24) ETHAnTBYT - Transmission byte counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01C0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTBYT[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTBYT[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTBYT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTBYT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-59 ETHAnTBYT register contents

Bit position	Bit name	Function
31 to 0	ETHAn TBYT[31:0]	This counter indicates the number of bytes in a transmit packet. When a collision occurs before transmission is finished or aborted, the transmission byte at which the collision occurred is also counted. The preamble and SFD are not included in the byte count indication.

(25) ETHAnTPKT - Transmission packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01C4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTPKT[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTPKT[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTPKT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTPKT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-60 ETHAnTPKT register contents

Bit position	Bit name	Function
31 to 0	ETHAn TPKT[31:0]	This counter is incremented when any packet is transmitted. This includes when a packet in which an error has occurred, unicast packet, multicast packet, or broadcast packet is transmitted.

(26) ETHAnTFCS - Transmission FCS error frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01C8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTFCS[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTFCS[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTFCS[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTFCS[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-61 ETHAnTFCS register contents

Bit position	Bit name	Function
31 to 0	ETHAn TFCS[31:0]	This counter is incremented if a CRC error is detected in the FCS field that is appended to a transmit packet. It is not incremented if transmission is aborted.

(27) ETHAnTMCA - Transmission multicast packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01CC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTMCA[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTMCA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTMCA[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTMCA[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-62 ETHAnTMCA register contents

Bit position	Bit name	Function
31 to 0	ETHAn TMCA[31:0]	This counter is incremented when a multicast packet is transmitted. It is not incremented when a broadcast packet is transmitted. Nor is it incremented if transmission is aborted or if a CRC error is detected.

(28) ETHAnTBCA - Transmission broadcast packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01D0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTBCA[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTBCA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTBCA[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTBCA[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-63 ETHAnTBCA register contents

Bit position	Bit name	Function
31 to 0	ETHAn TBCA[31:0]	This counter is incremented when a broadcast packet is transmitted. It is not incremented when a multicast packet is transmitted. Nor is it incremented if transmission is aborted or if a CRC error is detected.

(29) ETHAnTUCA - Transmit unicast packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01D4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTUCA[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTUCA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTUCA[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTUCA[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-64 ETHAnTUCA register contents

Bit position	Bit name	Function
31 to 0	ETHAn TUCA[31:0]	This counter is incremented when a unicast packet is transmitted. It is not incremented if transmission is aborted or if a CRC error is detected.

(30) ETHAnTXPF - Transmission pause control frame counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01D8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTXPF[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTXPF[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTXPF[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTXPF[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-65 ETHAnTXPF register contents

Bit position	Bit name	Function
31 to 0	ETHAn TXPF[31:0]	This counter is incremented each time a pause control frame is transmitted when the maximum amount of data has been stored in the receive FIFO.

(31) ETHAnTDFR - Transmission delay packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01DC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTDFR[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTDFR[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTDFR[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTDFR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-66 ETHAnTDFR register contents

Bit position	Bit name	Function
31 to 0	ETHAn TDFR[31:0]	This counter is incremented if a transmit delay occurs because of carrier detection when transmission is about to start. It is not incremented if a collision occurs during transmission that was started after the delay occurred.

(32) ETHAnTXDF - Transmission excessive delay packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01E0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTXDF[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTXDF[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTXDF[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTXDF[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-67 ETHAnTXDF register contents

Bit position	Bit name	Function
31 to 0	ETHAn TXDF[31:0]	This counter is incremented if transmission is aborted by an excessive delay.

(33) ETHAnTSCL - Transmission single collision packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01E4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTSCL[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTSCL[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTSCL[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTSCL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-68 ETHAnTSCL register contents

Bit position	Bit name	Function
31 to 0	ETHAn TSCL[31:0]	This counter is incremented if a transmission finishes successfully after a single collision occurred during the transmission.

(34) ETHAnTMCL - Transmission multiple collision packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01E8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTMCL[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTMCL[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTMCL[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTMCL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-69 ETHAnTMCL register contents

Bit position	Bit name	Function
31 to 0	ETHAn TMCL[31:0]	This counter is incremented if a transmission finishes successfully after a collision occurred multiple times (but less times than the value specified by ETHAnCLRT.ETHAnRETRY[3:0]) during the transmission.

(35) ETHAnTLCL - Transmission late collision packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01EC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTLCL[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTLCL[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTLCL[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTLCL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-70 ETHAnTLCL register contents

Bit position	Bit name	Function
31 to 0	ETHAn TLCL[31:0]	This counter is incremented if a late collision occurs during transmission.

(36) ETHAnTXCL - Transmission excessive collision packet counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01F0_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTXCL[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTXCL[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTXCL[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTXCL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-71 ETHAnTXCL register contents

Bit position	Bit name	Function
31 to 0	ETHAn TXCL[31:0]	This counter is incremented if collision occurs more than the times specified by ETHAnCLRT.ETHAnRETRY[3:0] in a single transmission.

(37) ETHAnTNCL - Transmission total collision counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01F4_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTNCL[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTNCL[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTNCL[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTNCL[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-72 ETHAnTNCL register contents

Bit position	Bit name	Function
31 to 0	ETHAn TNCL[31:0]	This counter counts the number of collisions after which transmission finishes successfully.

(38) ETHAnTCSE - Transmission carrier sense error counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01F8_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTCSE[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTCSE[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTCSE[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTCSE[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-73 ETHAnTCSE register contents

Bit position	Bit name	Function
31 to 0	ETHAn TCSE[31:0]	This counter is incremented if a carrier sense error occurs during transmission.

(39) ETHAnTIME - MAC internal error counter

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $01FC_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTIME[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTIME[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTIME[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTIME[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-74 ETHAnTIME register contents

Bit position	Bit name	Function
31 to 0	ETHAn TIME[31:0]	This counter is incremented if an error occurs in MAC during transmission or if a packet longer than the value specified by the ETHAnLMAX register is transmitted.

31.4.4 FIFO controller control registers

(1) ETHAnMFFCONT - FIFO controller control register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0200_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. Be sure to set the following bits to the values specified below (fixed values). If other values are set, the correct operation cannot be guaranteed.
   - ETHAnRXSDMA[1:0] = 10
   - ETHAnASOE = 0
   - ETHAnAPS = 1
   - ETHAnAPL = 1
   - ETHAnRXTHRC = 0
   - ETHAnTXTHRC = 0
2. Be sure to set bits 29, 28, 23 to 19, 13, and 7 to 3 to “0”.

31	30	29	28	27	26	25	24
ETHAn LOOPBACK	ETHAn RCSEL	0	0	ETHAn IMLP[3:0]			
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	ETHAn FLOWCNT	ETHAn IVPAUSE	ETHAn ZEROPAUSE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn RXSDMA[1:0]		0	ETHAn ASOE	ETHAn APS	ETHAn APL	ETHAn RXTHRC	ETHAn MFFRXEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	ETHAn TABT	ETHAn TXTHRC	ETHAn MFFTXEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-75 ETHAnMFFCONT register contents (1/2)

Bit position	Bit name	Function
31	ETHAn LOOPBACK	Loopback mode Loopback between the transmit FIFO and receive FIFO is performed. 0: Normal mode 1: Loopback mode
30	ETHAn RCSEL	RXCLK selection TXCLK is selected to be internally connected, instead of RXCLK. Specify a value for this bit if it is necessary to switch RXCLK to TXCLK when the MAC or the FIFO controller is in the loopback mode. 0: Normal mode 1: Clock switch mode (RXCLK switched to TXCLK)
27 to 24	ETHAn IMLP[3:0]	Set these bits to 0000.
18	ETHAn FLOWCNT	Flow control enable/disable 0: Flow control is disabled. 1: Flow control is enabled.
17	ETHAn IVPAUSE	Interval pause packet transmission control This bit specifies the method of retransmitting a pause packet. 0: Retransmission by threshold value of FIFO Internal pause timer (ETHAnPAUSETM.ETHAnIPTIME[15:0]) is not used. 1: Retransmission by internal pause timer Internal pause timer (ETHAnPAUSETM.ETHAnIPTIME[15:0]) is used.
16	ETHAn ZEROPAUSE	Zero pause packet output enable/disable 0: Zero pause packet output is disabled. 1: Zero pause packet output is enabled.
15, 14	ETHAn RXSDMA[1:0]	Fix these bits to 10.
12	ETHAn ASOE	Fix this bit to 0.
11	ETHAn APS	Fix this bit to 1.
10	ETHAn APL	Fix this bit to 1.
9	ETHAn RXTHRC	Fix this bit to 0.
8	ETHAn MFFRXEN	Reception enable 0: Disables reception. 1: Enables reception. [Timing for disabling reception] If this bit is cleared to disable reception while a packet is being written from the MAC to the receive FIFO, the receive FIFO write circuit is stopped after the packet has been written. The receive FIFO of the system stops after all the packets written in the receive FIFO have been read. The flow control circuit is not stopped by this bit.

Table 31-75 ETHAnMFFCONT register contents (2/2)

Bit position	Bit name	Function
2	ETHAn TABT	Transmission abort control This bit retransmits a packet that has been aborted by the MAC. 0: Discards the packet. 1: Retransmits the packet.
1	ETHAn TXTHRC	Fix this bit to 0.
0	ETHAn MFFTXEN	Transmission enable 0: Disables transmission. 1: Enables transmission. [Timing for disabling transmission] If this bit is cleared to disable transmission while a packet is being written to the transmit FIFO, the transmit FIFO write circuit is stopped after the packet has been written (after the END flag has been set), cancelling a request to write the next packet. Packet transfer to the MAC is stopped after all the packets in the transmit FIFO have been transferred (after the empty status is indicated).

(2) ETHAnRSTCNT - Software reset control register

This register is used to control software reset.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0204_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 17, 15 to 9, and 7 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	ETHAn RFFLSH
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	ETHAn TFFLSH
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn MFFSFTRST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-76 ETHAnRSTCNT register contents

Bit position	Bit name	Function
16	ETHAn RFFLSH	Receive FIFO clear (flush) This bit clears the receive FIFO, reception control circuit, flow control circuit, reception status register, and interrupt registers related to reception. Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared. This bit is always read as 0.
8	ETHAn TFFLSH	Transmit FIFO clear (flush) This bit clears the transmit FIFO, transmission control circuit, transmission status register, and interrupt registers related to transmission. Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared.
0	ETHAn MFFSFTRST	Software reset This bit resets all the circuits of the FIFO controller (MFF). Writing 1 to this bit starts the reset operation, and then this bit is automatically cleared.

(3) ETHAnFLOWTHRESH - Flow control threshold value register

This register is used to specify a threshold value for the receive FIFO to start flow control, and a threshold value to transmit a zero pause control frame.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0218_H

Initial value $0600\ 0200_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 27 and 15 to 11 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	ETHAnFLOWTHR[10:8]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnFLOWTHR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	ETHAnZPTHR[10:8]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnZPTHR[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-77 ETHAnFLOWTHRESH register contents

Bit position	Bit name	Function
26 to 16	ETHAn FLOWTHR[10:0]	These bits specify the threshold value in bytes for the receive FIFO to start flow control. Flow control starts when the amount of data in the receive FIFO reaches the value specified by these bits. Back pressure transmission is executed in the half-duplex mode and pause control packets are transmitted in the full-duplex mode. Writing bit 17 or 16 is ignored because the receive FIFO can only be written in 32-bit (4-byte) units. Bits 17 and 16 are always read as 0.
10 to 0	ETHAn ZPTHR[10:0]	These bits specify the threshold value in bytes for transmitting a zero pause control packet. When zero pause packet transmission is enabled by setting the MFFCNT.ZEROPAUSE bit to 1 (high level) and flow is controlled by pause control packets, a zero pause packet is transmitted if the amount of data in the receive FIFO falls below the threshold value specified by these bits. Writing bit 1 or 0 is ignored because the receive FIFO can only be written in 32-bit (4-byte) units. Bits 1 and 0 are always read as 0.

(4) ETHAnPAUSETM - Pause timer value register

This register is used to set the pause time.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $021C_H$

Initial value $7FFF\ FFFF_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnIPTIME[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnIPTIME[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnPAUSETMM[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnPAUSETMM[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-78 ETHAnPAUSETM register contents

Bit position	Bit name	Function
31 to 16	ETHAn IPTIME[15:0]	Interval pause packet timer value These bits specify the interval for transmitting a pause packet when interval pause packet transmission is enabled by setting the MFFCNT.IVPAUSE bit. Time required to transmit one packet = Time required to transmit 512 bits (circuit size) = 128 TXCLK cycles Default value: About 168 ms when the data rate is 100 Mbps, or about 1.68 seconds when the data rate is 10 Mbps
15 to 0	ETHAn PAUSETMM[15:0]	Pause control timer value of MAX pause packet These bits specify the value of TPTV[15:0] when a pause control request is issued to the MAC. Time required to transmit one packet = Time required to transmit 512 bits (circuit size) = 128 TXCLK cycles Default value: About 336 ms when the data rate is 100 Mbps, or about 3.36 seconds when the data rate is 10 Mbps

(5) ETHAnRXERSEL - Receive error selection register

This register is used to specify whether to accept or discard the packet if a reception error occurs.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0220_H

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Caution Be sure to set bits 25 to 23 and 15 to 2 to “0”.

31	30	29	28	27	26	25	24
ETHAnRLENE	ETHAnVLAN	ETHAnUSOP	ETHAnRPCF	ETHAnRCFR	ETHAnDBNB	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	ETHAnRLER	ETHAnRLER	ETHAnRXER	ETHAnRXER	ETHAnCEPS	ETHAnREPS	ETHAnPAIG
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	ETHAnTXRX	ETHAnDVCF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-79 ETHAnRXERSEL register contents (1/2)

Bit position	Bit name	Function
31	ETHAn RLENE	Specifies whether to accept a packet that contains a packet length error. 0: Accepts the packet. 1: Discards the packet.
30	ETHAn VLAN	Specifies whether to accept a VLAN packet. 0: Accepts the packet. 1: Discards the packet.
29	ETHAn USOP	Specifies whether to accept an undefined opcode control packet. 0: Accepts the packet. 1: Discards the packet.
28	ETHAn RPCF	Specifies whether to accept a pause control packet. 0: Accepts the packet. 1: Discards the packet.
27	ETHAn RCFR	Specifies whether to accept a control packet. 0: Accepts the packet. 1: Discards the packet.
26	ETHAn DBNB	Specifies whether to accept a packet that contains dribble nibble. 0: Accepts the packet. 1: Discards the packet.

Table 31-79 ETHAnRXERSEL register contents (2/2)

Bit position	Bit name	Function
22	ETHAn RLER	Specifies whether to accept a packet that has a length field exceeding 1,500 bytes. 0: Accepts the packet. 1: Discards the packet.
21	ETHAn RLER	Specifies whether to accept a packet whose length field does not match the data field length. 0: Accepts the packet. 1: Discards the packet.
20	ETHAn RXER	Specifies whether to accept a packet in which a CRC error has occurred. 0: Accepts the packet. 1: Discards the packet.
19	ETHAn RXER	Specifies whether to accept a packet in which RXER has been detected. 0: Accepts the packet. 1: Discards the packet.
18	ETHAn CEPS	Specifies whether to accept a packet in which a false carrier has been detected. 0: Accepts the packet. 1: Discards the packet.
17	ETHAn REPS	Specifies whether to accept a packet in which the total size of the preamble and SFD or the data field size is one nibble. 0: Accepts the packet. 1: Discards the packet.
16	ETHAn PAIG	Specifies whether to accept a packet for which one of the following events occurred after the previous packet was received. • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received. • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.ETHAnPUREP bit is set). 0: Accepts the packet. 1: Discards the packet.
1	ETHAn TXRX	Specifies whether to accept a packet in which a collision has been detected by the MAC. 0: Accepts the packet. 1: Discards the packet.
0	ETHAn DVCF	Specifies whether to accept a packet received by the MAC that is judged to be a valid control packet. 0: Accepts the packet. 1: Discards the packet.

(6) ETHAnTXSTMONI1 - Transmission status monitor 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0230_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 21 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	ETHAn CSE	ETHAn TBP	ETHAn TPP	ETHAn TPCF	ETHAn TCFR
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAn TTBC[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn TTBC[7:0]							
R	R	R	R	R	R	R	R

Table 31-80 ETHAnTXSTMONI1 register contents

Bit position	Bit name	Function
20	ETHAn CSE	Detection of carrier loss during transmission
19	ETHAn TBP	Occurrence of a collision due to the back pressure function after the previous transmission[a]
18	ETHAn TPP	End of transmission of a transmit packet requested during pausing[b]
17	ETHAn TPCF	Transmission of a pause control packet
16	ETHAn TCFR	Transmission of a control packet
15 to 0	ETHAn TTBC[15:0]	Number of bytes of transmitted data, including packets in which collisions occurred

a) This bit is set if a collision has occurred between when the transmission status was previously updated and when the status is updated this time.

b) This bit is not set if the packet requested during pausing is a control frame.

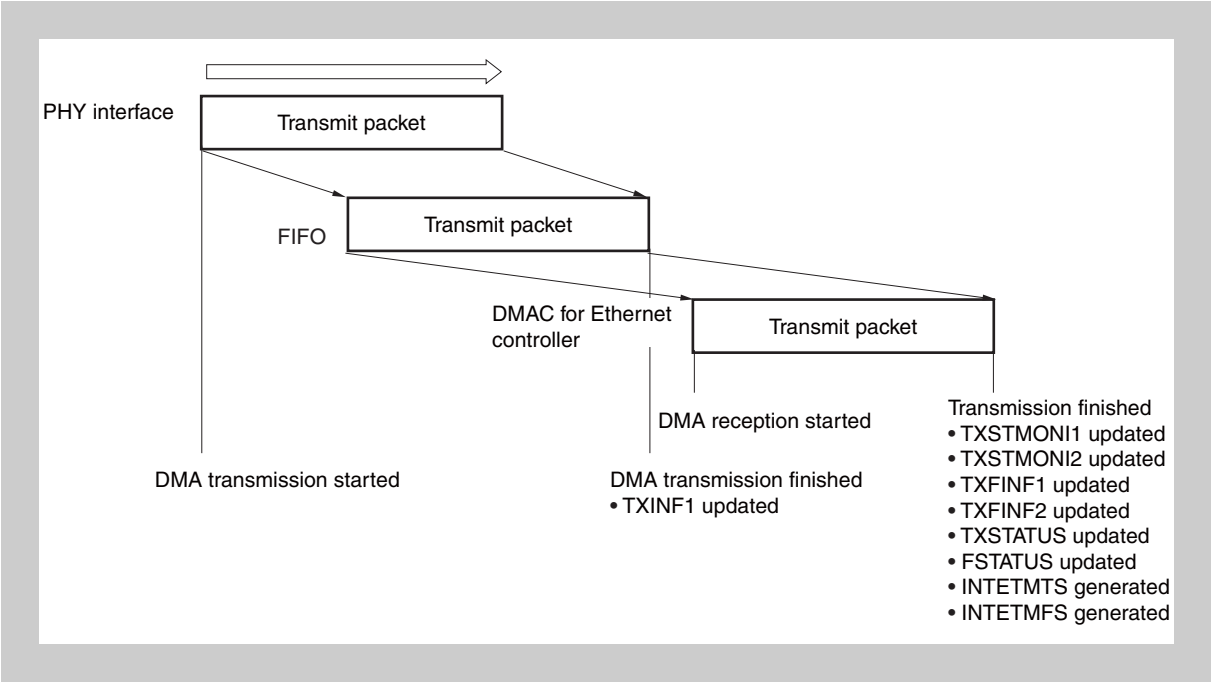


Figure 31-3 Timing at which transmission status is updated

(7) ETHAnTXSTMONI2 - Transmission status monitor 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0234_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAn TUDRF	ETHAn TGNTF	ETHAn LCOLF	ETHAn ECOLF	ETHAn TEDFRF	ETHAn TDFRF	ETHAn TBROF	ETHAn TMULF
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAn TDONEF	ETHAn TFLORF	ETHAn TFLERF	ETHAn TCRCEF	ETHAn TCBC[3:0]			
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAnTBYT[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAnTBYT[7:0]							
R	R	R	R	R	R	R	R

Table 31-81 ETHAnTXSTMONI2 register contents (1/2)

Bit position	Bit name	Function
31	ETHAn TUDRF	Detection of a transmit packet underrun[a]
30	ETHAn TGNTF	Transmission of a packet longer than the value specified by ETHAnLMAX[b]
29	ETHAn LCOLF	Occurrence of a late collision
28	ETHAn ECOLF	Occurrence of collisions exceeding the specified maximum number
27	ETHAn TEDFRF	Detection of an excessive transmission delay
26	ETHAn TDFRF	Occurrence of a transmission delay
25	ETHAn TBROF	Transmission of a broadcast packet
24	ETHAn TMULF	Transmission of a multicast packet
23	ETHAn TDONEF	End of transmission[c]
22	ETHAn TFLORF	Detection of a length field greater than 1,500 bytes[d]
21	ETHAn TFLERF	Detection of a packet whose length field does not match the data field length[d, e]

Table 31-81 ETHAnTXSTMONI2 register contents (2/2)

Bit position	Bit name	Function
20	ETHAn TCRCEF	Occurrence of a CRC error when CRC automatic appending mode was disabled
19 to 16	ETHAn TCBC[3:0]	Number of times retransmission occurred due to collisions[f]
15 to 0	ETHAn TBYT[15:0]	Transmit packet length (number of bytes) when transmission finished normally[f]

a) This bit is set only if no collision has occurred.
b) This bit is set only if ETHAnMACC1.ETHAnHUGEN is 0.
c) This bit is not set if transmission has been aborted.
d) This bit is not set if ETHAnMACC1.ETHAnFLCHT is 0.
e) If the length field exceeds 1,500 bytes, ETHAnTFLORF is set and ETHAnTFLERF is not.
f) This value is not correct if transmission has been aborted.

(8) ETHAnTXFINF1 - Transmission status 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0238_H

Initial value $0000\ 0800_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 25 and 15 to 12 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	ETHAn TPCNT[8]
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAn TPCNT[7:0]							
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	ETHAn TREMAIN[11:8]			
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn TREMAIN[7:0]							
R	R	R	R	R	R	R	R

Table 31-82 ETHAnTXFINF1 register contents

Bit position	Bit name	Function
24 to 16	ETHAn TPCNT[8:0]	Number of packets in the transmit FIFO These bits indicate the number of packets (start flag to end flag) that exist in the transmit FIFO. The value is incremented when one packet has been written by the system. The value is decremented when one packet has been read by the MAC (upon completion or halting of transmission).
11 to 0	ETHAn TREMAIN[11:0]	These bits indicate the remaining capacity of the transmit FIFO (in bytes). Bits 1 and 0 are always 00 because the data in the transmit FIFO are aligned in 32-bit (4-byte) units.

(9) ETHAnTXFINF2 - Transmission status 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $023C_H$

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Cautions

1. Before rewriting a mode register related to transmission, be sure to confirm that the ETHAnTXFINF2.ETHAnTXSTOP bit is 1.
2. Be sure to set bits 31 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn TXSTOP
R	R	R	R	R	R	R	R

Table 31-83 ETHAnTXFINF2 register contents

Bit position	Bit name	Function
0	ETHAn TXSTOP	This bit is set if no data is in the transmit FIFO while transmission is stopped (by clearing ETHAnMFFCONT.ETHAnMFFTXEN). Before rewriting a mode setting register related to transmission, be sure to confirm that this bit is 1. 0: The transmit FIFO is operating. 1: The transmit FIFO is stopped.

(10) ETHAnRXSTMONI - Reception status monitor register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0240_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAn RLENEF	ETHAn VLANF	ETHAn USOPF	ETHAn RPCFF	ETHAn RCFRF	ETHAn DBNBF	ETHAn RBROF	ETHAn RMULF
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAn RXOKF	ETHAn RLORF	ETHAn RLERF	ETHAn RCRCEF	ETHAn RXERF	ETHAn CEPSF	ETHAn REPSF	ETHAn PAIGF
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAn RBYT[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn RBYT[7:0]							
R	R	R	R	R	R	R	R

Table 31-84 ETHAnRXSTMONI register contents

Bit position	Bit name	Function
31	ETHAn RLENEF	Occurrence of a receive packet length error This bit is set if the received packet size is less than 64 bytes or greater than 1,518 bytes (or less than 64 bytes or greater than 1,522 bytes for a VLAN packet).
30	ETHAn VLANF	Reception of a VLAN packet This bit is set if a packet whose TPID field matches ETHAnVLTP is received.[a]
29	ETHAn USOPF	Reception of an undefined opcode control packet[b]
28	ETHAn RPCFF	Reception of a pause control packet[b]
27	ETHAn RCFRF	Reception of a control packet[b]
26	ETHAn DBNBF	Reception of a packet containing dribble nibble
25	ETHAn RBROF	Reception of a broadcast packet
24	ETHAn RMULF	Reception of a multicast packet
23	ETHAn RXOKF	End of reception[a]
22	ETHAn RLORF	Reception of a packet that has a length field exceeding 1,500 bytes[c]
21	ETHAn RLERF	Detection of a packet whose length field does not match the data field length[c, d]
20	ETHAn RCRCEF	Occurrence of a CRC error
19	ETHAn RXERF	Detection of RXER
18	ETHAn CEPSF	Detection of a false carrier [e]
17	ETHAn REPSF	Reception of a packet in which the total size of the preamble and SFD or the data field size is one nibble[e, f]
16	ETHAn PAIGF	This bit is set if one of the following events occurred after the previous packet was received.[e] • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received.[f] • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.ETHAnPUREP bit is set).[f]
15 to 0	ETHAn RBYT[15:0]	Number of received bytes

a) This bit is not set if a CRC error or RXERF has occurred.
b) This bit is not set if a CRC error has occurred.
c) This bit is not set if ETHAnMACC1.ETHAnFLCHT is 0.
d) If the length field exceeds 1,500 bytes, ETHAnRLORF is set and ETHAnRLERF is not.
e) The bit is set if a false carrier is detected between when the reception status was updated previously and when it is updated this time.

f) A packet in which this event has occurred is ignored and not transferred to the upstream system. The ETHAnRXSTMONI register is updated when a DMA transfer of the receive packet has finished. (This register indicates the status of the packet transferred by DMA.) The ETHAnRXFINF1 register is also updated at the same time.

The ETHAnRXSTATUS register is updated when a DMA transfer of the receive packet has finished. The reception status register 1 (ETHAnRXFINF1) is also updated at the same time.

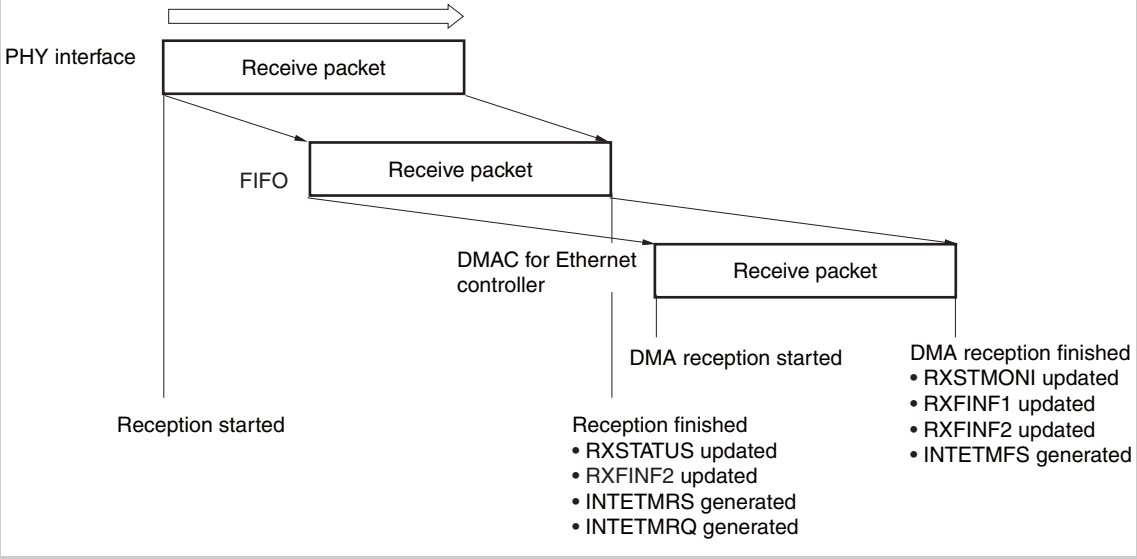


Figure 31-4 Timing at which receive data status is updated

(11) ETHAnRXFINF1 - Reception status 1 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0244_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAnRPLEN[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAnRPLEN[7:0]							
R	R	R	R	R	R	R	R

Table 31-85 ETHAnRXFINF1 register contents

Bit position	Bit name	Function
15 to 0	ETHAn RPLEN[15:0]	These bits indicate the receive packet length in bytes. The Ethernet Controller uses the value of RPLEN for the size field when writing back the receive descriptor.

(12) ETHAnRXFINF2 - Reception status 2 register

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0248_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 25 and 15 to 12 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	ETHAnRPCNT[8]
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAn RPCNT[7:0]							
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	ETHAn RREMAIN[11:8]			
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn RREMAIN[7:0]							
R	R	R	R	R	R	R	R

Table 31-86 ETHAnRXFINF2 register contents

Bit position	Bit name	Function
24 to 16	ETHAn RPCNT[8:0]	Number of packets in the receive FIFO These bits indicate the number of packets (start flag to end flag) that exist in the receive FIFO. The value is incremented when one packet has been written by the MAC. (A packet that is discarded upon being received does not increment this value.) If a packet has been read from the DMAC for the Ethernet Controller or if the packet is discarded, this value is decremented after the internal operation to discard the packet finishes.
11 to 0	ETHAn RREMAIN[11:0]	These bits indicates the remaining receive FIFO capacity (in bytes). Bits 1 and 0 are always 00 because the data in the receive FIFO is aligned in 32-bit (4-byte) units.

(13) ETHAnRXFINF3 - Reception status 3 register

This register indicates the status of the receive FIFO while reception is stopped.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + $024C_H$

Initial value $0000\ 0001_H$. This register is initialized by any reset.

Cautions

1. Before rewriting a mode setting register related to reception or flow control, be sure to confirm that the ETHAnRXFINF3.ETHAnRXSTOP bit is 1.
2. Be sure to set bits 31 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn RXSTOP
R	R	R	R	R	R	R	R

Table 31-87 ETHAnRXFINF3 register contents

Bit position	Bit name	Function
0	ETHAn RXSTOP	This bit is set if no data is in the receive FIFO while reception is stopped (by clearing ETHAnMFFCONT.ETHAnMFFRXEN). Before rewriting a mode setting register related to reception or flow control, confirm that the ETHAnRXSTOP bit is 1. 0: Receive FIFO is operating. 1: Receive FIFO is stopped.

(14) ETHAnFSTATUS - FIFO status interrupt register

If an interrupt source that is not masked by the ETHAnFSTATUSM register is generated, the INTETMFS interrupt (FIFO status interrupt) is generated. The INTETMFS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHAnFSTATUSM register is generated, the corresponding bit in this register is set. All the bits of the ETHAnFSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0250_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. The FIFO status interrupt status register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.
2. Be sure to set bits 31 to 25, 23 to 17, 15 to 13, 9, 8, 5, and 2 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	ETHAn TACOF
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	ETHAn RACOF
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	ETHAn TSUP	ETHAn TFNRTY	ETHAn TFWE	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn RFFE	ETHAn RSUP	0	ETHAn RFWE	ETHAn RFOF	0	ETHAn RFFLW	ETHAn RFZP
R	R	R	R	R	R	R	R

Table 31-88 ETHAnFSTATUS register contents

Bit position	Bit name	Function
24	ETHAn TACOF	This bit is set when the ETHAnTXABTCNT register (TX abort counter) overflows.
16	ETHAn RACOF	This bit is set when the RXABCNT register (RX abort counter) overflows.
12	ETHAn TSUP	TX status update This bit is set when the transmission status is updated in the ETHAnTXSTMONI1 or ETHAnTXSTMONI2 register.
11	ETHAn TFNRTY	Transmit FIFO abort (transmit FIFO no retry) This bit is set if transmission fails and the data in the transmit FIFO is discarded. In this case, ETHAnTXABTCNT is incremented.
10	ETHAn TFWE	This bit is set if a transmit FIFO write error has occurred.
7	ETHAn RFFE	Receive FIFO flag error This bit is set if handshaking is not correctly performed when receive data is written from the MAC to the receive FIFO. The receive packet and reception status are invalid but reception is not canceled. • If the reception status is updated before all receive data is stored in the receive FIFO, the end of the packet is assumed as soon as the reception status has been updated. • If the reception status is not updated after all receive data has been stored in the receive FIFO, the reception status is assumed to be all 0.
6	ETHAn RSUP	This bit is set when the reception status monitor register (ETHAnRXSTMONI) is updated. A valid value can be read from the ETHAnRXSTMONI or ETHAnRXFINF1 register when this bit is 1.
4	ETHAn RFWE	Receive FIFO write error This bit is set if a packet whose size is less than 32 bits (4 bytes) has been received and could not be written to the receive FIFO.
3	ETHAn RFOF	This bit is set if the receive FIFO overflows.
1	ETHAn RFFLW	This bit is set if the data in the receive FIFO reaches or exceeds the value specified by the ETHAnFLOWTHRESH.FLOWTHR bits.
0	ETHAn RFZP	This bit is set if the data in the receive FIFO reaches or exceeds the value specified by the ETHAnFLOWTHRESH.ETHAnZPTHR[10:0] bits.

(15) ETHAnFSTATUSM - FIFO status interrupt mask register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0254_H

Initial value 0101 1 FFF_H. This register is initialized by any reset.

Caution Be sure to set bits 31 to 25, 23 to 17, and 15 to 13 to “0”, and set bits 9, 8, 5, and 2 to “1”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	ETHAn TACOFM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	ETHAn RACOFM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	ETHAn TSUPM	ETHAn TFNRTYM	ETHAn TFWEM	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn RFFEM	ETHAn RSUPM	1	ETHAn RFWEM	ETHAn RFOFM	1	ETHAn RFFLWM	ETHAn RFZPM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-89 ETHAnFSTATUSM register contents (1/2)

Bit position	Bit name	Function
24	ETHAn TACOFM	0: Does not mask interrupt. 1: Masks interrupt.
16	ETHAn RACOFM	0: Does not mask interrupt. 1: Masks interrupt.
12	ETHAn TSUPM	0: Does not mask interrupt. 1: Masks interrupt.
11	ETHAn TFNRTYM	0: Does not mask interrupt. 1: Masks interrupt.
10	ETHAn TFWEM	0: Does not mask interrupt. 1: Masks interrupt.
7	ETHAn RFFEM	0: Does not mask interrupt. 1: Masks interrupt.
6	ETHAn RSUPM	0: Does not mask interrupt. 1: Masks interrupt.
4	ETHAn RFWEM	0: Does not mask interrupt. 1: Masks interrupt.

Table 31-89 ETHAnFSTATUSM register contents (2/2)

Bit position	Bit name	Function
3	ETHAn RFOFM	0: Does not mask interrupt. 1: Masks interrupt.
1	ETHAn RFFLWM	0: Does not mask interrupt. 1: Masks interrupt.
0	ETHAn RFZPM	0: Does not mask interrupt. 1: Masks interrupt.

(16) ETHAnTXSTATUS - Transmission status interrupt register

This register stores the cumulative result of the transmission status. If an interrupt that is not masked by the ETHAnTXSTATUSM register is generated, the INTETMTS interrupt (transmission status interrupt) is generated. The INTETMTS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHAnTXSTATUSM register is generated, the corresponding bit in this register is set. All the bits of the ETHAnTXSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0258_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. The transmission status interrupt register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.
2. Be sure to set bits 31 to 17 and 15 to 8 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	ETHAn TABIF
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn TGNTIF	ETHAn LCOLIF	ETHAn ECOLIF	ETHAn TEDFRIF	ETHAn TDFRIF	ETHAn TFLORIF	ETHAn TFLERIF	ETHAn TCRCEIF
R	R	R	R	R	R	R	R

Table 31-90 ETHAnTXSTATUS register contents

Bit position	Bit name	Function
16	ETHAn TABIF	This bit is set if transmission has been aborted.
7	ETHAn TGNTIF	This bit is set if a packet longer than the value specified by ETHAnLMAX has been transmitted (TAB source). This bit is not set if MACC.ETHAnHUGEN is 1.
6	ETHAn LCOLIF	This bit is set if a late collision has been detected (TAB source).
5	ETHAn ECOLIF	This bit is set if collisions have occurred exceeding the specified maximum number (TAB source).
4	ETHAn TEDFRIF	This bit is set if an excessive transmission delay has been detected (TAB source).
3	ETHAn TDFRIF	This bit is set if a transmission delay has occurred.
2	ETHAn TFLORIF	This bit is set if a packet whose length field is greater than 1,500 bytes is detected. This bit is also set when a VLAN packet pause control frame is transmitted. This bit is not set if ETHAnMACC1.ETHAnFLCHT is 0.
1	ETHAn TFLERIF	This bit is set if a packet whose length field does not match the data field length is detected. This bit is not set if ETHAnMACC1.ETHAnFLCHT is 0. If the length field exceeds 1,500 bytes, ETHAnTFLORIF is set and ETHAnTFLERIF is not.
0	ETHAn TCRCEIF	This bit is set if a CRC error occurred. This bit is set if transmission is performed with the CRC automatic appending mode disabled (the ETHAnMACC1.ETHAnPADEN and ETHAnMACC1.ETHAnCRCEN bits are 0).

(17) ETHAnTXSTATUSM - Transmission status interrupt mask register

This register is used to mask the transmission status interrupt (INTCTS).

If an interrupt source that is not masked by this register is generated, the INTCTS interrupt is generated. The INTCTS interrupt signal is asserted until all the related interrupt sources are cleared. If an interrupt source masked by the ETHAnTXSTATUSM register is generated, the corresponding bit in the ETHAnTXSTATUS register is set.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $025C_H$

Initial value $0001\ 01FF_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 17 and 15 to 8 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	ETHAn TABIM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn TGNTIM	ETHAn LCOLIM	ETHAn ECOLIM	ETHAn TEDFRIM	ETHAn TDFRIM	ETHAn TFLORIM	ETHAn TFLERIM	ETHAn TCRCEIM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-91 ETHAnTXSTATUSM register contents (1/2)

Bit position	Bit name	Function
16	ETHAn TABIM	0: Does not mask interrupt. 1: Masks interrupt.
7	ETHAn TGNTIM	0: Does not mask interrupt. 1: Masks interrupt.
6	ETHAn LCOLIM	0: Does not mask interrupt. 1: Masks interrupt.
5	ETHAn ECOLIM	0: Does not mask interrupt. 1: Masks interrupt.
4	ETHAn TEDFRIM	0: Does not mask interrupt. 1: Masks interrupt.
3	ETHAn TDFRIM	0: Does not mask interrupt. 1: Masks interrupt.

Table 31-91 ETHAnTXSTATUSM register contents (2/2)

Bit position	Bit name	Function
2	ETHAn TFLORM	0: Does not mask interrupt. 1: Masks interrupt.
1	ETHAn TFLERIM	0: Does not mask interrupt. 1: Masks interrupt.
0	ETHAn TCRCEIM	0: Does not mask interrupt. 1: Masks interrupt.

(18) ETHAnRXSTATUS - Reception status interrupt register

This register stores the cumulative result of the reception status. If an interrupt source that is not masked by the ETHAnRXSTATUSM register is generated, the INTETMRS interrupt is generated. The INTETMRS interrupt signal is asserted until all bits in this register are cleared.

If an interrupt source masked by the ETHAnRXSTATUSM register is generated, the corresponding bit in this register is set.

This register is not affected by the setting of ETHAnRXERSEL (receive error selection register).

All the bits of the ETHAnRXSTATUS register are cleared when the register is read.

Access This register is read-only, in 32-bit units.

Address <ETHAn_base> + 0260_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Cautions

1. The reception status interrupt status register is cleared when it is read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.
2. Be sure to set bits 31 to 15 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
0	ETHAn RLENEIF	ETHAn VLANIF	ETHAn USOPIF	ETHAn RPCFIF	ETHAn RCFRIF	ETHAn DBNBIF	ETHAn RLORIF
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAn RLERIF	ETHAn RCRCEIF	ETHAn RXERIF	ETHAn CEPSIF	ETHAn REPSIF	ETHAn PAIGIF	ETHAn TXRXIF	ETHAn DVCFIF
R	R	R	R	R	R	R	R

Table 31-92 ETHAnRXSTATUS register contents

Bit position	Bit name	Function
14	ETHAn RLENEIF	Occurrence of a receive packet length error This bit is set if the received packet size is less than 64 bytes or greater than 1,518 bytes (or less than 64 bytes or greater than 1,522 bytes for a VLAN packet).
13	ETHAn VLANIF	Reception of a VLAN packet This bit is set if a packet whose TPID field matches ETHAnVLTP.[a]
12	ETHAn USOPIF	Reception of an undefined opcode control packet[b]
11	ETHAn RPCFIF	Reception of a pause control packet[b]
10	ETHAn RCFRIF	Reception of a control packet[b]
9	ETHAn DBNBIF	Reception of a packet containing dribble nibble
8	ETHAn RLORIF	Reception of a packet that has a length field exceeding 1,500 bytes[c]
7	ETHAn RLERIF	Detection of a packet whose length field does not match the data field length[c, d]
6	ETHAn RCRCEIF	Occurrence of a receive CRC error
5	ETHAn RXERIF	Detection of RXER
4	ETHAn CEPSIF	Detection of a false carrier[e]
3	ETHAn REPSIF	Reception of a packet in which the total size of the preamble and SFD or the data field size is one nibble[e, f]
2	ETHAn PAIGIF	This bit is set if one of the following events occurred after the previous packet was received.[e] • A carrier longer than 6,072 nibbles (3,036 bytes) has been detected. • The next packet with IFG + preamble + SFD has been received before the time required to transmit 80 bits has elapsed after a packet has been received. • An illegal preamble or SFD has been received while the pure preamble data check is enabled (the ETHAnMACC1.ETHAnPUREP bit is set).
1	ETHAn TXRXIF	This bit is set if transmission starts (a collision occurs) during half-duplex reception (immediately after starting reception).
0	ETHAn DVCFIF	This bit is set if the received packet is a valid control packet (that does not contain an error).

a) This bit is not set if a CRC error or RXER has occurred.
b) This bit is not set if a CRC error has occurred.
c) This bit is not set if ETHAnMACC1.ETHAnFLCHT is 0.
d) If the length field exceeds 1,500 bytes, ETHAnRLORIF is set and ETHAnRLERIF is not.
e) This bit indicates that this event occurred between when the reception status was updated previously and when it is updated this time.
f) A packet in which this event has occurred is ignored and not transferred to the upstream system.

(19) ETHAnRXSTATUSM - Reception status interrupt mask register

This register is used to mask the reception status interrupt (INTETMRS).

If an interrupt source that is not masked by this register is generated, INTETMRS is generated. The INTETMRS interrupt signal is asserted until all the related interrupt sources are cleared. If an interrupt source masked by this register is generated, the corresponding bit in the ETHAnRXSTATUS register is set.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0264_H

Initial value $0000\ 07FF_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 15 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	ETHAn RLENEIM	ETHAn VLANIM	ETHAn USOPIM	ETHAn RPCFIM	ETHAn RCFRIM	ETHAn DBNBIM	ETHAn RLORIM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn RLERIM	ETHAn RCRCEIM	ETHAn RXERIM	ETHAn CEPSIM	ETHAn REPSIM	ETHAn PAIGIM	ETHAn TXRXIM	ETHAn DVCFIM
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-93 ETHAnRXSTATUSM register contents

Bit position	Bit name	Function
14	ETHAn RLENEIM	0: Does not mask interrupt. 1: Masks interrupt.
13	ETHAn VLANIM	0: Does not mask interrupt. 1: Masks interrupt.
12	ETHAn USOPIM	0: Does not mask interrupt. 1: Masks interrupt.
11	ETHAn RPCFIM	0: Does not mask interrupt. 1: Masks interrupt.
10	ETHAn RCFRIM	0: Does not mask interrupt. 1: Masks interrupt.
9	ETHAn DBNBIM	0: Does not mask interrupt. 1: Masks interrupt.
8	ETHAn RLORIM	0: Does not mask interrupt. 1: Masks interrupt.
7	ETHAn RLERIM	0: Does not mask interrupt. 1: Masks interrupt.
6	ETHAn RCRCEIM	0: Does not mask interrupt. 1: Masks interrupt.
5	ETHAn RXERIM	0: Does not mask interrupt. 1: Masks interrupt.
4	ETHAn CEPSIM	0: Does not mask interrupt. 1: Masks interrupt.
3	ETHAn REPSIM	0: Does not mask interrupt. 1: Masks interrupt.
2	ETHAn PAIGIM	0: Does not mask interrupt. 1: Masks interrupt.
1	ETHAn TXRXIM	0: Does not mask interrupt. 1: Masks interrupt.
0	ETHAn DVCFIM	0: Does not mask interrupt. 1: Masks interrupt.

(20) ETHAnTXABTCNT - TX abort counter

This is a transmission abort counter. It counts the number of packets that have resulted in a MAC transmission error (including an underrun).

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0270_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTABCNT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTABCNT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-94 ETHAnTXABTCNT register contents

Bit position	Bit name	Function
15 to 0	ETHAn TABCNT[15:0]	Transmission abort count These bits count the number of packets that have resulted in a MAC transmission error (including an underrun). This counter is not incremented when ETHAnMFFCONT.ETHAnTABT is set to retransmit an aborted packet. If ETHAnMFFCONT.ETHAnTXTHRC is set, retransmission is not executed when transmission has been aborted. In this case, the counter is incremented. Packets are counted after 68 bytes have been transferred because they are not retransmitted by a retry request (usually, the MAC does not issue a retry request after 64 bytes have been transferred). If the count value overflows, the value of these bits returns to 0 and the ETHAnFSTATUS.ETHAnTACOF bit is set. These bits are not cleared by resetting the transmission circuit (by setting ETHAnTFRST and ETHAnTFFLSH).

(21) ETHAnRXABTCNT - RX abort counter

This is a reception abort counter. These bits count the number of receive packets that have been discarded because of the status of the receive packet, the status of the receive FIFO, address filtering by the MAC, and reception of a control packet.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0274_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 16 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRABCNT[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRABCNT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-95 ETHAnRXABTCNT register contents

Bit position	Bit name	Function
15 to 0	ETHAn RABCNT[15:0]	Reception abort count These bits count the number of receive packets that have been discarded because of the status of the receive packet, the status of the receive FIFO, address filtering by the MAC, and reception of a control packet. If the count value overflows, the value of these bits returns to 0 and the ETHAnFSTATUS.ETHAnRACOF bit is set. These bits are not cleared by resetting the reception circuit (by setting ETHAnRFRST and ETHAnRFFLSH).

31.4.5 DMAC control registers in Ethernet Controller

(1) ETHAnMODE - Core function setting register

This register is used to analyze the reception start descriptor and transmission start descriptor.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0300_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 19 and 16 to 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	ETHAn RXS	ETHAn TXS	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-96 ETHAnMODE register contents

Bit position	Bit name	Function
18	ETHAn RXS	The ETHAnRXS bit is used to start analyzing the reception start descriptor. This bit is automatically cleared after 1 is written to it.
17	ETHAn TXS	The ETHAnTXS bit is used to start analyzing the transmission start descriptor. This bit is automatically cleared after 1 is written to it.

(2) ETHAnINTMS - Interrupt register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0304_H

Initial value $0F00\ 0F00_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 28, 23 to 20, 15 to12, and 7 to 4 to “0”.

Note The ETHAnRBEI, ETHAnRECI, ETHAnRXI, ETHAnTBEI, ETHAnTECI, and ETHAnTXI bits of the ETHAnINTMS register are cleared when they are read.

31	30	29	28	27	26	25	24
0	0	0	0	ETHAn RUSMSK	ETHAn RBEMSK	ETHAn RECMSK	ETHAn RXMSK
R	R	R	R	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	ETHAn RUPI	ETHAn RBEI	RECI	ETHAn RXI
R	R	R	R	R/W	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	ETHAn TUSMSK	ETHAn TBEMSK	ETHAn TECMSK	ETHAn TXMSK
R	R	R	R	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	ETHAn TUPI	ETHAn TBEI	ETHAn TECI	ETHAn TXI
R	R	R	R	R/W	R	R	R

Table 31-97 ETHAnINTMS register contents (1/2)

Bit position	Bit name	Function
27	ETHAn RUSMSK	This bit specifies whether to mask the RUPI interrupt indicated by bit 19. 0: Does not mask interrupt. 1: Masks interrupt.
26	ETHAn RBEMSK	This bit specifies whether to mask the RBEI interrupt indicated by bit 18. 0: Does not mask interrupt. 1: Masks interrupt.
25	ETHAn RECMSK	This bit specifies whether to mask the RECI interrupt indicated by bit 17. 0: Does not mask interrupt. 1: Masks interrupt.
24	ETHAn RXMSK	This bit specifies whether to mask the RXI interrupt indicated by bit 16. 0: Does not mask interrupt. 1: Masks interrupt.
19	ETHAn RUPI	This bit indicates whether a pause interrupt specified by the “U” (used) bit of a receive descriptor has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
18	ETHAn RBEI	This bit indicates whether a receive data buffer access error interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
17	ETHAn RECI	This bit indicates whether a reception (DMA) end of chain interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
16	ETHAn RXI	This bit indicates whether a packet reception (DMA) completion interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
11	ETHAn TUSMSK	This bit specifies whether to mask the TUPI interrupt indicated by bit 3. 0: Does not mask interrupt. 1: Masks interrupt.
10	ETHAn TBEMSK	This bit specifies whether to mask the TBEI interrupt indicated by bit 2. 0: Does not mask interrupt. 1: Masks interrupt.
9	ETHAn TECMSK	This bit specifies whether to mask the TECI interrupt indicated by bit 1. 0: Does not mask interrupt. 1: Masks interrupt.
8	ETHAn TXMSK	This bit specifies whether to mask the TXI interrupt indicated by bit 0. 0: Does not mask interrupt. 1: Masks interrupt.
3	ETHAn TUPI	This bit indicates whether a pause interrupt specified by the “U” (used) bit of a transmit descriptor has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.

Table 31-97 ETHAnINTMS register contents (2/2)

Bit position	Bit name	Function
2	ETHAn TBEI	This bit indicates whether a transmit data buffer access error interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
1	ETHAn TECI	This bit indicates whether a transmission (DMA) end of chain interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.
0	ETHAn TXI	This bit indicates whether a packet transmission (DMA) completion interrupt has been generated. It is cleared when read. 0: No interrupt generated. 1: Interrupt generated.

(3) ETHAnTRANSCTL - Transmission control register

This register is used to control transfer by the DMAC for the Ethernet Controller.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0308_H

Initial value $0003\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 26, 23 to 18, and 15 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	ETHAn RXEN_STA	ETHA TEN_STA
R/W	R/W	R/W	R/W	R/W	R/W	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	ETHAn MADRXEN	ETHAn MADTXEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn RXCHKSMEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-98 ETHAnTRANSCTL register contents

Bit position	Bit name	Function
25	ETHAn RXEN_STA	This bit indicates the reception status. 0: Reception is not in progress (IDLE status). 1: Reception is in progress. Reception stops if an RBEI or RECI interrupt, which is controlled by the interrupt register (ETHAnINTMS), is generated. At this time, this bit is cleared.
24	ETHAn TXEN_STA	This bit indicates the transmission status. 0: Transmission is not in progress (IDLE status). 1: Transmission is in progress. Transmission stops if a TBEI or TECI interrupt, which is controlled by the interrupt register (ETHAnINTMS), is generated. At this time, this bit is cleared.
17	ETHAn MADRXEN	This bit specifies whether to enable reception. 0: Disables reception (DMA reception stops.) 1: Enables reception.
16	ETHAn MADTXEN	This bit specifies whether to enable transmission. 0: Disables transmission (DMA transmission stops). 1: Enables transmission.
0	ETHAn RXCHKSMEN	This bit enables or disables the receive checksum appending function. 0: Disables the receive check sum appending function. 1: Enables the receive check sum appending function.

(4) ETHAnSFTRST - Software reset setting register

This register is used to execute a software reset for the DMAC for the Ethernet Controller.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $030C_H$

Initial value $0000\ 0000_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 1 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn MADSFTRST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-99 ETHAnSFTRST register contents

Bit position	Bit name	Function
0	ETHA MADSFTRST	Software reset When this bit is set, the DMAC for the Ethernet Controller is reset. The receive checksum unit is also reset. This bit is automatically cleared after 1 is written to it.

(5) ETHAnDMACM - DMA controller mode setting register

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0310_H

Initial value $0000\ 0010_H$. This register is initialized by any reset.

Caution Be sure to set bits 31 to 11 and 7 to 0 to “0”.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	ETHAnBURST[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-100 ETHAnDMACM register contents

Bit position	Bit name	Function
10 to 8	ETHAn BURST[2:0]	These bits specify the type of burst transfer.

ETHAnBURST[2:0]	Type	Operation
000_B	SINGLE	Single transfer mode
011_B	INCR4	4-beat incremental burst transfer mode
101_B	INCR8	8-beat incremental burst transfer mode
111_B	INCR16	16-beat incremental burst transfer mode
Other than above	Setting prohibited	

(6) ETHAnRXDP - Receive descriptor pointer register

This register is used to specify the pointer position of the receive descriptor of the DMAC for the Ethernet Controller.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 0320_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnRXDP[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnRXDP[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnRXDP[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnRXDP[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-101 ETHAnRXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn RXDP[31:0]	These bits specify the pointer position of the receive descriptor. Specify the first address of the receive descriptor chain. Bits 1 and 0 are fixed to 00.

(7) ETHAnLSTRXDP - Last receive descriptor pointer register

This register indicates the last receive descriptor address of the DMAC for the Ethernet Controller.

Access This register can be read in 32-bit units.

Address <ETHAn_base> + 0324_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnLSTRXDP[31:24]							
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAnLSTRXDP[23:16]							
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAnLSTRXDP[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAnLSTRXDP[7:0]							
R	R	R	R	R	R	R	R

Table 31-102 ETHAnLSTRXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn LSTRXDP[31:0]	These bits indicate the last receive descriptor pointer address. They hold the address of the last accessed receive descriptor. Bits 1 and 0 are fixed to 00.

(8) ETHAnTXDP - Transmit descriptor pointer register

This register is used to specify the pointer position of the transmit descriptor of the DMAC for the Ethernet Controller.

Access This register can be read in 32-bit units.

Address <ETHAn_base> + $032C_H$

Initial value $FFFF\ FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTXDP[31:24]							
R	R	R	R	R	R	R	R
23	22	21	20	19	18	17	16
ETHAnTXDP[23:16]							
R	R	R	R	R	R	R	R
15	14	13	12	11	10	9	8
ETHAnTXDP[15:8]							
R	R	R	R	R	R	R	R
7	6	5	4	3	2	1	0
ETHAnTXDP[7:0]							
R	R	R	R	R	R	R	R

Table 31-103 ETHAnTXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn TXDP[31:0]	These bits specify the pointer position of the transmit descriptor. Specify the first address of the transmit descriptor chain. Bits 1 and 0 are fixed to 00.

(9) ETHAnLSTTXDP - Last transmit descriptor pointer register

This register indicates the last transmit descriptor address of the DMAC for the Ethernet Controller.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $032C_H$

Initial value $FFFF\ FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnLSTTXDP[31:24]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnLSTTXDP[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnLSTTXDP[15:8]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnLSTTXDP[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-104 ETHAnLSTTXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn LSTTXDP[31:0]	These bits indicate the last transmit descriptor pointer address. They hold the address of the last accessed transmit descriptor. Bits 1 and 0 are fixed to 00.

31.4.6 Control registers of DMAC for transmit checksum

(1) ETHAnTCHMODE - Transmit checksum unit function setting register

This register is used to analyze the reception start descriptor and transmission start descriptor.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 1300_H

Initial value $0000\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	ETHAn TCHRXS	ETHAn TCHTXS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-105 ETHAnTCHMODE register contents

Bit position	Bit name	Function
18	ETHAn TCHRXS	The ETHAnTCHRXS bit is used to start analyzing the reception start descriptor. This bit is cleared immediately after 1 is written to it.
17	ETHAn TCHTXS	The ETHAnTCHTXS bit is used to start analyzing the transmission start descriptor. This bit is cleared immediately after 1 is written to it.

(2) ETHAnTCHINTMS - Transmit checksum interrupt register

This register indicates the status of and masks the INTSCRXTCH and INTSCTXTCH interrupts of the DMAC for the transmit checksum.

Access This register can be read/written in 32-bit units. Bits 18 to 16 and 2 to 0 can only be read, however.

Address <ETHAn_base> + 1304_H

Initial value 0700 0700_H. This register is initialized by any reset.

Caution The TCH_RBEI, TCH_RECI, TCH_RXI, TCH_TEBI, TCH_TECI, and TCH_TXI bits of the transmit checksum interrupt register are cleared when they are read. It is recommended to copy interrupt sources to variables so that multiple interrupt sources generated concurrently can be detected.

31	30	29	28	27	26	25	24
0	0	0	0	0	ETHAn TCH RBEMSK	ETHAn TCH RECMSK	ETHAn TCH RXMSK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	ETHAn TCHRBEI	ETHAn TCHRECI	ETHAn TCHRXI
R/W	R/W	R/W	R/W	R/W	R	R	R
15	14	13	12	11	10	9	8
0	0	0	0	0	ETHAn TCH TBEMSK	ETHAn TCH TECMSK	ETHAn TCH TXMSK
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	ETHAn TCHTBEI	ETHAn TCHTECI	ETHAn TCHTXI
R/W	R/W	R/W	R/W	R/W	R	R	R

Table 31-106 ETHAnTCH_INTMS register contents

Bit position	Bit name	Function
26	ETHAn TCHRBEMSK	This bit specifies whether to mask the ETHAnTCHRBEI interrupt indicated by bit 18. 0: Does not mask interrupt. 1: Masks interrupt.
25	ETHAn TCHRECMSK	This bit specifies whether to mask the ETHAnTCHRECI interrupt indicated by bit 17. 0: Does not mask interrupt. 1: Masks interrupt.
24	ETHAn TCHRXMSK	This bit specifies whether to mask the ETHAnTCHRXI interrupt indicated by bit 16. 0: Does not mask interrupt. 1: Masks interrupt.
18	ETHAn TCHRBEI	This bit indicates whether a receive data buffer access error interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
17	ETHAn TCHRECI	This bit indicates whether a receive DMA end of chain interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
16	ETHAn TCHRXI	This bit indicates whether a packet reception DMA transfer completion interrupt of INTSCRXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
10	ETHAn TCHTBEMSK	This bit specifies whether to mask the ETHAnTCHTBEI interrupt indicated by bit 2. 0: Does not mask interrupt. 1: Masks interrupt.
9	ETHAn TCHTECMSK	This bit specifies whether to mask the ETHAnTCHTECI interrupt indicated by bit 1. 0: Does not mask interrupt. 1: Masks interrupt.
8	ETHAn TCHTXMSK	This bit specifies whether to mask the ETHAnTCHTXI interrupt indicated by bit 0. 0: Does not mask interrupt. 1: Masks interrupt.
2	ETHAn TCHTBEI	This bit indicates whether a transmit data buffer access error interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
1	ETHAn TCHTECI	This bit indicates whether a transmit DMA end of chain interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.
0	ETHAn TCHTXI	This bit indicates whether a packet transmission DMA transfer completion interrupt of INTSCTXTCH has been generated. It is cleared (0) when read. 0: No interrupt generated. 1: Interrupt generated.

(3) ETHAnTCHTRANSCTL - Transmit checksum transfer control register

This register is used to control transmission/reception of the DMAC for the transmit checksum.

Access This register can be read/written in 32-bit units. Bits 25 and 24 can only be read, however.

Address <ETHAn_base> + 1308_H

Initial value $0003\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	ETHAnTCH RXEN_STA	ETHAnTCH TXEN_STA
R/W	R/W	R/W	R/W	R/W	R/W	R	R
23	22	21	20	19	18	17	16
0	0	0	0	0	0	ETHAnTCH RXEN	ETHAnTCH TXEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-107 ETHAnTCHTRANSCTL register contents

Bit position	Bit name	Function
25	ETHAn TCHRXEN_STA	This bit indicates the reception status. 0: Reception is not in progress (IDLE status). 1: Reception is in progress. Reception stops if a ETHAnTCHRBEI or ETHAnTCHRECI interrupt, which is controlled by the transmit checksum interrupt register (ETHAnTCHINTMS), is generated. At this time, this bit is cleared.
24	ETHAn TCHTXEN_STA	This bit indicates the transmission status. 0: Transmission is not in progress (IDLE status). 1: Transmission is in progress. Transmission stops if a ETHAnTCHTBEI or ETHAnTCHTECI interrupt, which is controlled by the transmit checksum interrupt register (ETHAnTCHINTMS), is generated. At this time, this bit is cleared.
17	ETHAn TCHRXEN	This bit specifies whether to enable reception. 0: Disables reception (DMA reception stops.) 1: Enables reception.
16	ETHAn TCHTXEN	This bit specifies whether to enable transmission. 0: Disables transmission (DMA transmission stops). 1: Enables transmission.

(4) ETHAnTCHSFTRST - Transmit checksum software reset register

This register is used to execute a software reset for the DMAC for the transmit checksum.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $130C_H$

Initial value $0003\ 0000_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ETHAn TCH SFTRST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-108 ETHAnTCHSFTRST register contents

Bit position	Bit name	Function
0	ETHAn TCHSFTRST	This is a software reset bit for the DMAC for the transmit checksum. When this bit is set, the DMAC for the transmit checksum and the transmit checksum unit are reset. This bit is automatically cleared after the reset is executed.

(5) ETHAnTCHDMACM - Transmit checksum DMA control mode setting register

This register is used to specify the burst transfer type of DMA by the DMAC for the transmit checksum.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 1310_H

Initial value $0000\ 0010_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
0	0	0	0	0	ETHAn BURST2	ETHAn BURST1	ETHAn BURST0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 31-109 ETHAnTCHDMACM register contents

Bit position	Bit name	Function
10 to 8	ETHAn BURST [2:0]	These bits specify the type of burst transfer for the internal system bus.

ETHAn BURST 2	ETHAn BURST 1	ETHAn BURST 0	Transfer type
0	0	0	Single transfer mode
0	1	1	4-beat incremental burst transfer mode
1	0	0	8-beat incremental burst transfer mode
1	1	1	16-beat incremental burst transfer mode
Other than above			Setting prohibited

(6) ETHAnTCHRXDP - Transmit checksum receive descriptor pointer

This register is used to specify the pointer position of the receive descriptor of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 1314_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAn TCH RXDP31	ETHAn TCH RXDP30	ETHAn TCH RXDP29	ETHAn TCH RXDP28	ETHAn TCH RXDP27	ETHAn TCH RXDP26	ETHAn TCH RXDP25	ETHAn TCH RXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAn TCH RXDP23	ETHAn TCH RXDP22	ETHAn TCH RXDP21	ETHAn TCH RXDP20	ETHAn TCH RXDP19	ETHAn TCH RXDP18	ETHAn TCH RXDP17	ETHAn TCH RXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn TCH RXDP15	ETHAn TCH RXDP14	ETHAn TCH RXDP13	ETHAn TCH RXDP12	ETHAn TCH RXDP11	ETHAn TCH RXDP10	ETHAn TCH RXDP9	ETHAn TCH RXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn TCH RXDP7	ETHAn TCH RXDP6	ETHAn TCH RXDP5	ETHAn TCH RXDP4	ETHAn TCH RXDP3	ETHAn TCH RXDP2	ETHAn TCH RXDP1	ETHAn TCH RXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-110 ETHAnTCHRXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn TCHRXDP[31:0]	These bits specify the pointer position of the receive descriptor. Specify the first address of the receive descriptor chain. Bits 1 and 0 are fixed to 00.

(7) ETHAnTCHLSTRXDP - Transmit checksum last receive descriptor pointer

This register indicates the last receive descriptor address of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 1318_H

Initial value FFFF $FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTCH LSTRXDP31	ETHAnTCH LSTRXDP30	ETHAnTCH LSTRXDP29	ETHAnTCH LSTRXDP28	ETHAnTCH LSTRXDP27	ETHAnTCH LSTRXDP26	ETHAnTCH LSTRXDP25	ETHAnTCH LSTRXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTCH LSTRXDP23	ETHAnTCH LSTRXDP22	ETHAnTCH LSTRXDP21	ETHAnTCH LSTRXDP20	ETHAnTCH LSTRXDP19	ETHAnTCH LSTRXDP18	ETHAnTCH LSTRXDP17	ETHAnTCH LSTRXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTCH LSTRXDP15	ETHAnTCH LSTRXDP14	ETHAnTCH LSTRXDP13	ETHAnTCH LSTRXDP12	ETHAnTCH LSTRXDP11	ETHAnTCH LSTRXDP10	ETHAnTCH LSTRXDP9	ETHAnTCH LSTRXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTCH LSTRXDP7	ETHAnTCH LSTRXDP6	ETHAnTCH LSTRXDP5	ETHAnTCH LSTRXDP4	ETHAnTCH LSTRXDP3	ETHAnTCH LSTRXDP2	ETHAnTCH LSTRXDP1	ETHAnTCH LSTRXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-111 ETHAnTCHLSTRXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn TCHLSTRXDP[31:0]	These bits indicate the last receive descriptor address. They hold the address of the last accessed receive descriptor. Bits 1 and 0 are fixed to 00.

(8) ETHAnTCHTXDP - Transmit checksum transmit descriptor pointer

This register is used to specify the pointer position of the transmit descriptor of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + $131C_H$

Initial value $FFFF\ FFFC_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAn TCH TXDP31	ETHAn TCH TXDP30	ETHAn TCH TXDP29	ETHAn TCH TXDP28	ETHAn TCH TXDP27	ETHAn TCH TXDP26	ETHAn TCH TXDP25	ETHAn TCH TXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAn TCH TXDP23	ETHAn TCH TXDP22	ETHAn TCH TXDP21	ETHAn TCH TXDP20	ETHAn TCH TXDP19	ETHAn TCH TXDP18	ETHAn TCH TXDP17	ETHAn TCH TXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAn TCH TXDP15	ETHAn TCH TXDP14	ETHAn TCH TXDP13	ETHAn TCH TXDP12	ETHAn TCH TXDP11	ETHAn TCH TXDP10	ETHAn TCH TXDP9	ETHAn TCH TXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAn TCH TXDP7	ETHAn TCH TXDP6	ETHAn TCH TXDP5	ETHAn TCH TXDP4	ETHAn TCH TXDP3	ETHAn TCH TXDP2	ETHAn TCH TXDP1	ETHAn TCH TXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-112 ETHAnTCHTXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn TCHTXDP[31:0]	These bits specify the pointer position of the transmit descriptor. Specify the first address of the transmit descriptor chain. Bits 1 and 0 are fixed to 00.

(9) ETHAnTCHLSTTXDP - Transmit checksum last transmit descriptor pointer

This register indicates the last transmit descriptor address of the DMAC for the transmit checksum. The lower 2 bits are fixed to 00_B.

Access This register can be read/written in 32-bit units.

Address <ETHAn_base> + 1320_H

Initial value FFFF FFFC$_H$. This register is initialized by any reset.

31	30	29	28	27	26	25	24
ETHAnTCH LSTTXDP31	ETHAnTCH LSTTXDP30	ETHAnTCH LSTTXDP29	ETHAnTCH LSTTXDP28	ETHAnTCH LSTTXDP27	ETHAnTCH LSTTXDP26	ETHAnTCH LSTTXDP25	ETHAnTCH LSTTXDP24
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
23	22	21	20	19	18	17	16
ETHAnTCH LSTTXDP23	ETHAnTCH LSTTXDP22	ETHAnTCH LSTTXDP21	ETHAnTCH LSTTXDP20	ETHAnTCH LSTTXDP19	ETHAnTCH LSTTXDP18	ETHAnTCH LSTTXDP17	ETHAnTCH LSTTXDP16
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8
ETHAnTCH LSTTXDP15	ETHAnTCH LSTTXDP14	ETHAnTCH LSTTXDP13	ETHAnTCH LSTTXDP12	ETHAnTCH LSTTXDP11	ETHAnTCH LSTTXDP10	ETHAnTCH LSTTXDP9	ETHAnTCH LSTTXDP8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
7	6	5	4	3	2	1	0
ETHAnTCH LSTTXDP7	ETHAnTCH LSTTXDP6	ETHAnTCH LSTTXDP5	ETHAnTCH LSTTXDP4	ETHAnTCH LSTTXDP3	ETHAnTCH LSTTXDP2	ETHAnTCH LSTTXDP1	ETHAnTCH LSTTXDP0
R/W	R/W	R/W	R/W	R/W	R/W	R	R

Table 31-113 ETHAnTCHLSTTXDP register contents

Bit position	Bit name	Function
31 to 0	ETHAn TCHLSTTXDP[31:0]	These bits indicate the last transmit descriptor pointer address. They hold the address of the last accessed transmit descriptor. Bits 1 and 0 are fixed to 00.

31.5 MAC/FIFO Function

31.5.1 Frame format

With Ethernet/IEEE802.3, data is transmitted and received as a packet or frame.

The Ethernet Controller supports the following three types of frames.

- Basic frame
- VLAN frame
- Pause control frame

(1) Basic frame

The basic frame used with Ethernet consists of a preamble (PA), frame start delimiter (SFD), destination address (DA), source address (SA), type/length field (TYPE/LEN), data field (DATA), and frame check sequence (FCS).

The packet size is defined to be 64 bytes minimum and 1,518 bytes maximum, excluding the preamble (PA) and frame start delimiter (SFD).

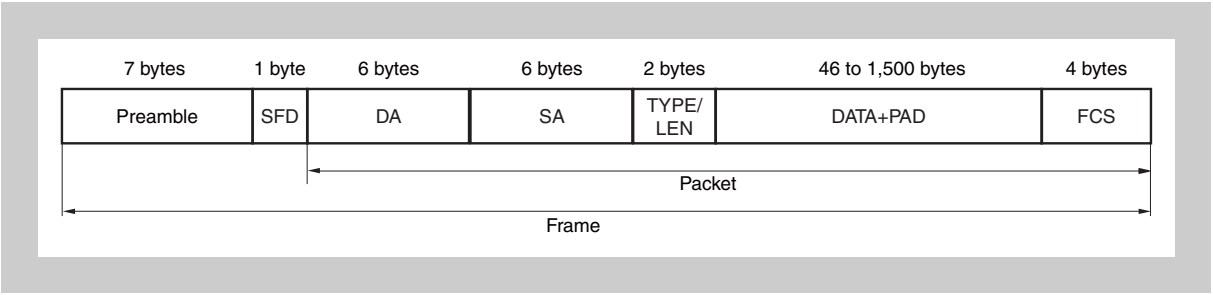


Figure 31-5 Basic frame structure

(a) Preamble and frame start delimiter (SFD)

The preamble and SFD consist of a repetition of 10 for 62 bits followed by 11 at the end, indicating the header of each frame.

(b) Destination address (DA)

The destination address field indicates the destination MAC address. A unicast address, multicast address, or broadcast address is written in this field.

(c) Source address (SA)

The source MAC address is written in this field.

(d) Type/length field

As an Ethernet frame, this field is a type field that indicates a protocol type. As an IEEE802.3 frame, this field is a length field that indicates the length of the data field.

(e) Data field

The data field size ranges from 46 bytes to 1,500 bytes.

Depending on the communication protocol, the data field might be divided to insert special header information. The Ethernet Controller uses the data in this field only for calculating the CRC (Cycle Redundancy Check) for the FCS and does not check its contents.

(f) Frame check sequence (FCS)

The frame check sequence field is used to write a 32-bit CRC code to check the transfer data.

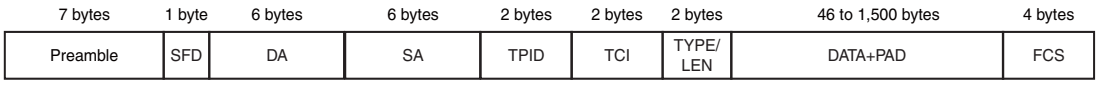
The Ethernet Controller can automatically append a CRC to a transmit frame.

(2) VLAN frame

The structure of a VLAN frame (Qtag frame) is slightly different from that of a basic frame.

A 4-byte VLAN header is inserted immediately after the source address field. As a result, the minimum packet length of a VLAN frame is 64 bytes and the maximum packet length is 1,522 bytes.

The Ethernet Controller has a VLAN frame detection function. If a transmit or receive packet is detected to be a VLAN frame, packet processing is performed based on the receive packet length.



Note TPID: Tag Protocol ID
TCI: Tag Control Information
TPID + TCI: VLAN header

Figure 31-6 VLAN frame structure

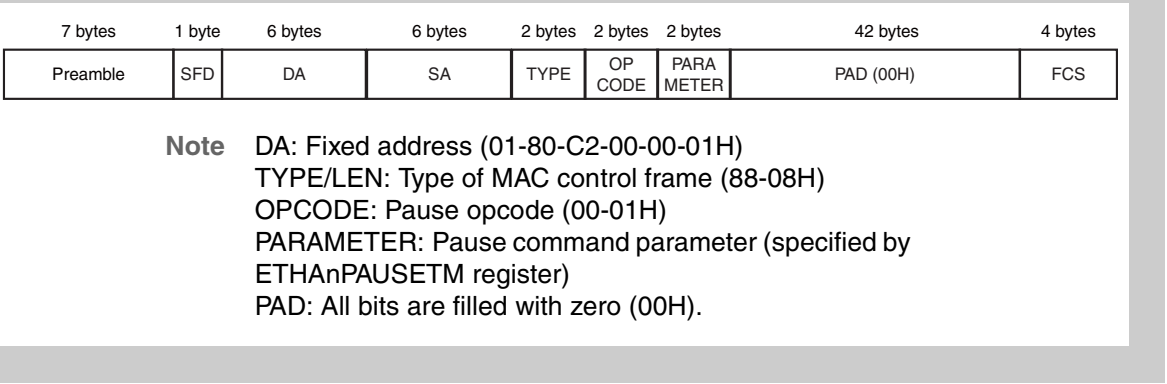
Caution The Ethernet Controller recognizes the value set to the ETHAnVLTP.ETHAnVLTP[15:0] bits as a VLAN frame (TPID). The default value is 0000_H. For details, refer to *10 “ETHAnVLTP - VLAN type register” in 31.4.2 “MAC control registers”*.

(3) Pause control frame

The pause control frame is a 64-byte packet that has a dedicated format.

The destination address field has a fixed value of 01-80-C2-00-00-01_H.

The type/length field has a value of 8808H, which indicates a control frame, and the opcode has a value of 0001H, which indicates a pause control frame. The parameter field stores the value specified by the ETHAnPAUSETM register. The unused area following the parameter field is filled with PAD data consisting of zeros.



Note DA: Fixed address (01-80-C2-00-00-01H)
TYPE/LEN: Type of MAC control frame (88-08H)
OPCODE: Pause opcode (00-01H)
PARAMETER: Pause command parameter (specified by ETHAnPAUSETM register)
PAD: All bits are filled with zero (00H).

Figure 31-7 Pause control frame structure

The Ethernet Controller can automatically transmit a pause control frame according to the amount of the remaining in the receive FIFO.

When a frame is received, the frame type is identified by the DA, TYPE, and OPCODE fields as shown in *Table 31-114 “Frame reception”*.

Table 31-114 Frame reception

DA	TYPE	OPCODE	Frame identified
01-80-C2-00-00-01	8808_H	0001_H	Pause frame
01-80-C2-00-00-01	8808_H	Other than 0001_H	Not supported
01-80-C2-00-00-01	Other than 8808_H	xxxx	Data frame
Unicast (station address)	8808_H	0001_H	Pause frame
Unicast (station address)	8808_H	Other than 0001_H	Not supported
Unicast (station address)	Other than 8808_H	xxxx	Data frame
Multicast	8808_H	xxxx	Not supported
Multicast	Other than 8808_H	xxxx	Data frame
Unicast (station address)	8808_H	xxxx	Not supported
Unicast (station address)	Other than 8808_H	xxxx	Data frame

(4) Pause control frame containing VLAN tag

The Ethernet Controller does not support a pause control frame containing a VLAN tag.

It receives such a frame as an ordinary VLAN packet, but the ETHAnRBROF and ETHAnRLORF flags of the reception status monitor register (ETHAnRXSTMONI) are set (the ETHAnRPCFF and ETHAnRFCR flags are not set).

(5) Envelope frame

The envelope frame format was added to IEEE802.3as (2005). Because it is a frame for 1,000 Mbps half-duplex communication, the Ethernet Controller does not support this type of frame. If an envelope frame containing an EXTENSION field is received, a CRC error or length field mismatch occurs, or the receive FIFO overflows. Check the reception status and discard such a frame.

The Ethernet Controller cannot transmit an envelope frame containing an EXTENSION field.

31.5.2 Frame transmission

The Ethernet Controller generates a transmit frame as defined in IEEE802.3 from the transmit packet data stored in the transmit FIFO by the DMAC for the Ethernet Controller via a DMA transfer, and then outputs that frame to a PHY device. If a collision is detected, the frame is retransmitted by using a random back-off algorithm. The status information of each transmit frame, such as excessive transmit delays and collisions exceeding the maximum number, is reflected in the ETHAnTXSTATUS register, and the number of times each event has occurred in all transmit frames is counted by statistics counters.

(1) Transmit frame

The transmit frame defined by IEEE802.3 consists of the following six fields (refer to *Figure 31-5 "Basic frame structure"*).

- Preamble (PA)
- Frame start delimiter (SFD)
- Destination address (DA)
- Source address (SA)
- Length field (LEN)
- Data and frame check sequence (FCS)

During transmission, the Ethernet Controller generates a preamble, frame start delimiter, and FCS data.

(2) Transmission clock

The Ethernet Controller operates in synchronization with the transmission clock (TXCLK) supplied by an external PHY device. Transmit packet data stored in the transmit FIFO via a DMA transfer is synchronized with TXCLK in the FIFO, and then output to the PHY device. IEEE802.3 defines the frequency of TXCLK as 25 MHz±100 ppm at a data transfer rate of 100 Mbps, or 2.5 MHz ± 100 ppm at 10 Mbps.

(3) Carrier sense signal (CRS)

During half-duplex communication, if a carrier is detected (CRS = 1) after the Ethernet Controller has stored transmit data in the FIFO and transmission is enabled, the Ethernet Controller postpones transmission until the end of the carrier (CRS = 0). After the carrier ends, transmission starts when the inter-packet gap (IPG) count specified by the ETHAnIPGT register has been reached.

If no carrier is detected (CRS = 0) when transmission is enabled and the IPG count is reached after the end of the previous carrier, transmission starts immediately.

When a frame is transmitted from the local side, the transmitted carrier sense signal is looped back from the external PHY device. If the transmitted carrier sense signal is masked by the external PHY device in the user-defined system,

the Ethernet Controller detects a carrier sense error, but this does not affect the transmission.

(4) Collision detection (COL) and retransmission

If the Ethernet Controller detects a collision during half-duplex communication, it transmits jam data (an error CRC) and then stops transmission.

If less than the maximum number of collisions (default value: 15) are detected in the collision window, transmission is kept waiting by a random back-off algorithm and data in the transmit FIFO is retransmitted. (In this case, no data is captured into the FIFO again by DMA.)

If collisions exceeding the maximum number are detected or if a late collision (a collision detected outside the collision window) occurs, transmission is aborted and the transmit data is discarded.

(5) Inter-packet gap (IPG)

The IPG for successive data transmission from the local side is specified by the ETHAnIPGT register, and that for other cases is specified by the ETHAnIPGR register.

When transmission from the local side or the communicating device is finished, the Ethernet Controller starts counting the IPG. If a request for the next transmission is issued from the FIFO after transmission from the local side is finished and before the IPG count reaches the value of the ETHAnIPGT register, it is assumed that the data is to be transmitted successively (back to back), and transmission starts as soon as counting finishes.

If a packet is to be transmitted after transmission from the communicating device is finished, the IPG count is controlled by the ETHAnIPGR register. In the ETHAnIPGR register, the entire period of the IPG is specified for the ETHAnIPGR2 bits and the interval time to detect a carrier in the first half of the IPG is specified for the ETHAnIPGR1 bits. If a carrier is detected during the period specified for the ETHAnIPGR1 bits, the Ethernet Controller waits for the end of the carrier and then starts IPG counting from the beginning. If no carrier is detected during the period specified for the ETHAnIPGR1 bits, transmission starts after the IPG period specified for the ETHAnIPGR2 bits elapses.

If transmission does not start within the time required to transmit 24,288 bits (2.43 ms at 10 Mbps or 243.88 μs at 100 Mbps) after the next transmission request is received from the FIFO, an excessive transmission delay is assumed, transmission is aborted, and the transmit data is discarded.

The set value of the ETHAnIPGT and ETHAnIPGR registers and the actual IPG period are calculated by the following expression.

[At 100 Mbps]

Back-to-back transmission:

IPG = (5 + ETHAnIPGT) × 40 ns (default value: 960 ns)

Non back-to-back transmission:

IPG = (5 + ETHAnIPGR2) × 40 ns (default value: 960 ns)

Carrier sense time:

(2 + ETHAnIPGR1) × 40 ns (default value: 640 ns)

[At 10 Mbps]

Back-to-back transmission:

IPG = (5 + ETHAnIPGT) × 400 ns (default value: 9.6 µs)

Non back-to-back transmission:

IPG = (5 + ETHAnIPGR2) × 400 ns (default value: 9.6 µs)

Carrier sense time:

(2 + ETHAnIPGR1) × 400 ns (default value: 6.4 µs)

Caution According to the specification of IEEE802.3, set the IPG to 960 ns or more at a data transfer rate of 100 Mbps, or 9.6 µs or more at 10 Mbps.

The default value of the ETHAnIPGT and ETHAnIPGR registers is the minimum rated value and may be used as is.

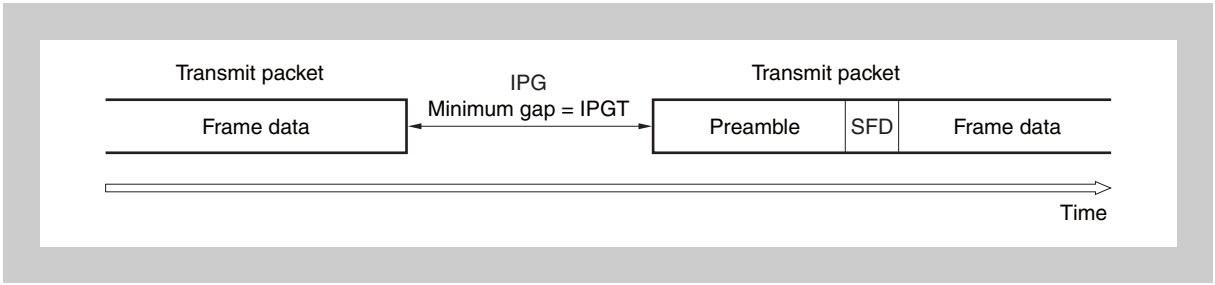


Figure 31-8 IPG during back-to-back transmission

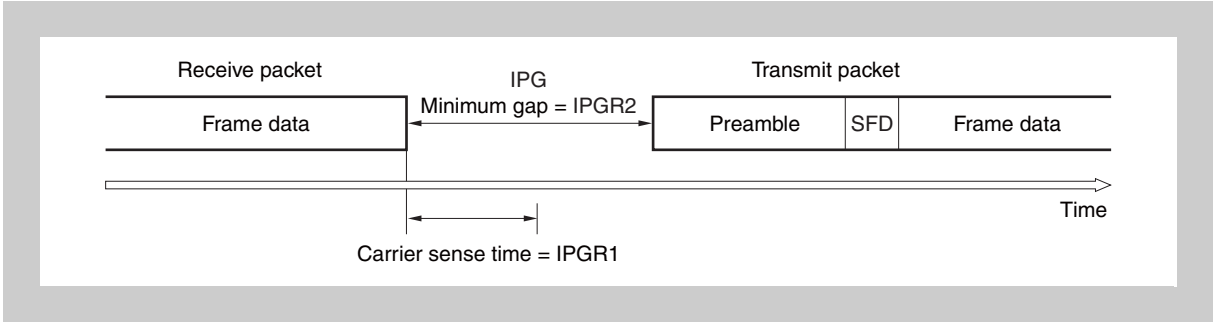


Figure 31-9 IPG during non back-to-back transmission

(6) Appending preamble, CRC, and PAD

A 7-byte preamble and a 1-byte frame start delimiter (SFD) are appended to the beginning of the transmit packet supplied from the FIFO.

ETHAnMACC1. ETHAnCRCEN	Operation
0	The transmit packet must end with a valid frame check sequence (FCS). The MAC checks the FCS, and if the FCS value is not correct, the MAC generates a transmission status interrupt (INTCTS) to report an error.
1	An internally generated frame check sequence (FCS) is appended to the end of the transmit packet.

If the ETHAnMACC1.ETHAnCRCEN bit is set, a frame check sequence (FCS) that has been internally generated is appended to the end of the transmit packet.

If the ETHAnMACC1.ETHAnCRCEN bit is cleared, the transmit packet must end with a valid FCS. The Ethernet Controller can check the FCS. If the value of the FCS is not correct, the Ethernet Controller generates an Ethernet transmission status interrupt.

If the ETHAnMACC1.ETHAnPADEN bit is set, zeros (PAD) are appended to a transmit packet shorter than 64 bytes (this is known as padding). In this case, the Ethernet Controller appends the correct FCS to the end of the frame, regardless of the setting of the ETHAnCRCEN bit.

If the ETHAnMACC2.ETHAnAPD or ETHAnMACC2.ETHAnVPD bit is set when the ETHAnMACC1.ETHAnPADEN bit is 1, PAD is appended to a VLAN frame. If the ETHAnAPD bit is set, only a packet that matches the VLAN type specified by the ETHAnVLTP register is regarded as a VLAN frame and is padded. If the ETHAnVPD bit is set, all packets are regarded as VLAN frames and padded. A packet regarded as a VLAN frame is padded to extend its length to 68 bytes. The data that is appended as a pad is all 0.

(7) Aborting transmission

The Ethernet Controller aborts transmission under the following conditions.

It does not abort transmission if the transmit FIFO underruns within the normal operating range.

- If more than the maximum number of collisions (MAX collision) occur
- If collision occurs outside the collision window (late collision)
- If there is an excessive transmission delay
- If a packet exceeding the frame length specified by the ETHAnLMAX register is to be transmitted.

  (If the ETHAnMACC1.ETHAnHUGEN bit is 1, however, the transmit frame length is not limited.)

(8) Full-duplex operation

A full-duplex operation is enabled when the ETHAnMACC1.ETHAnFULLD bit is set. The IPG is always the value specified by the ETHAnIPGT register. The FULLD signal is asserted if the ETHAnMACC1.ETHAnFULLD bit is set, reporting to the external circuit that full-duplex operation is specified.

(9) Flow control and back pressure functions

The Ethernet Controller has a flow control function (see *1 “Flow control” in 31.5.4 “MAC control function”*) and a back pressure function (see *2 “Back pressure” in 31.5.4 “MAC control function”*) associated with the receive FIFO. These functions are automatically activated to prevent the FIFO from overflowing when the vacant capacity of the receive FIFO runs short.

31.5.3 Frame reception

The Ethernet Controller generates a receive packet from a receive frame to be stored in the FIFO, detects the SFD, checks the length field and FCS, and identifies whether the frame is a VLAN frame.

The status information of each receive packet is set to the reception status monitor register (ETHAnRXSTMONI) and the number of times each event has occurred in all receive frames is counted by statistics counters.

(1) Receive clock

The Ethernet Controller receives data in synchronization with the reception clock (RXCLK) supplied by an external (PHY) device.

IEEE802.3 defines the frequency of RXCLK as 25 MHz ± 100 ppm at a data transfer rate of 100 Mbps, or 2.5 MHz ± 100 ppm at 10 Mbps.

(2) Reception of MII data

The Ethernet Controller recognizes the RXD signal data as receive frames while the RXDV signal is asserted, and recognizes the end of the frames when the RXDV signal is deasserted.

(3) Detecting preamble and SFD

The Ethernet Controller detects the preamble and SFD at the beginning of a receive frame and recognizes the data that follows as a receive packet.

(4) Checking length field

The Ethernet Controller counts the length of a receive packet, and checks the length of the data field assuming the 2 bytes following the source address to be a length field. The result of this check can be read as the reception status from the ETHAnRXSTMONI register. An interrupt signal can be output to report a mismatch between the counted receive packet length and the length field value.

(5) CRC

The Ethernet Controller calculates the 4-byte frame check sequence (FCS) from a receive packet and compares it with the FCS data appended to the end of the receive packet. The result of the comparison can be read from the ETHAnRXSTMONI register. An interrupt signal can be output to report a mismatch between the calculated 4-byte FCS and the FCS data at the end of the receive packet.

(6) Transmitting data to FIFO

The Ethernet Controller assumes that a packet of 6 bytes or more is valid and discards a packet of less than 6 bytes.

(7) Detection of huge packet

If the ETHAnMACC1.ETHAnHUGEN bit is cleared, the Ethernet Controller receives only packets shorter than the maximum frame length specified by the ETHAnLMAX register (default value: 1,536 bytes), and stops receiving longer packets midway.

For details about the receivable packet length, refer to *Table 31-123 "Restrictions on receive FIFO"*.

(8) Detecting VLAN frame

The Ethernet Controller checks whether received packets are VLAN frames.

If the value of the TPID field (the 2 bytes following the source address) of a received packet matches the value specified by the ETHAnVLTP register, the packet is recognized as a VLAN packet and the ETHAnRXSTMONI.VLAN flag is set. In a packet recognized as a VLAN frame, the 2 bytes immediately after the VLAN header (the 4 bytes following the source address) including the TPID field are regarded as the length field.

31.5.4 MAC control function

(1) Flow control

The Ethernet Controller controls the flow by using the pause control frame defined in “IEEE802.3 Annex 31”.

The purpose of flow control is to decrease the frequency of frame transmission executed by the communicating device (link partner) connected point-to-point during full-duplex operation. The amount of data a system can receive and process is limited. If frames are received too frequently, system processing cannot keep up, so the receive FIFO might overflow. Flow control is used to avoid this situation.

When the Ethernet Controller receives a pause control frame, it loads the value of the parameter field in the control frame to the pause timer in the MAC. If the value of the pause timer is not 0, the next transmission starts after the time set to the pause timer has elapsed.

If the value of the parameter field in the received pause control frame is 0 (a zero pause control frame), the value of the pause timer is cleared to 0 and transmission is resumed after the packet interval specified by the ETHAnIPGR register has elapsed.

To suppress data transmission from the link partner, the reserved multicast address (01-80-C2-00-00-01), pause opcode (00-01), and the pause timer value of the ETHAnPAUSETM register (ETHAnPAUSETMM[15:0]) are transmitted as a pause control frame.

Starting transmission of a pause frame takes precedence over starting transmission of a basic frame. If a condition for transmitting a pause frame is satisfied while a basic frame is being transmitted, however, the pause frame is transmitted after transmission of the basic frame has finished.

The Ethernet Controller controls the flow by setting ETHAnMFFCONT.ETHAnFLOWCNT.

The necessity of transmitting a pause control frame request is judged according to the amount of data in the receive FIFO.

If the ETHAnMFFCONT.ETHAnIVPAUSE bit is 0 in the full-duplex communication mode, the Ethernet Controller monitors the amount of data in the receive FIFO during reception (see *(a)* in *Figure 31-10 “Flow control”*). And if the amount of data in the receive FIFO exceeds the value specified by the ETHAnFLOWTHRESH.ETHAnFLOWTHR[10:0] bits, a pause control frame is transmitted (see *(b)* in *Figure 31-10 “Flow control”*).

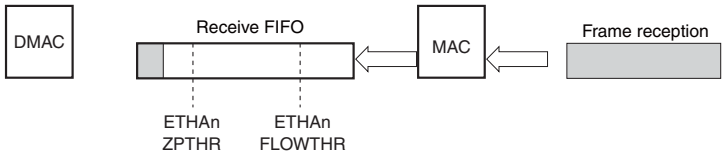
If the ETHAnMFFCONT.ETHAnIVPAUSE bit is 1, the pause control frame is continually transmitted at the interval specified by the ETHAnPAUSETM.ETHAnIPTIME[15:0] bits as long as the amount of data in the receive FIFO exceeds the value specified by the ETHAnFLOWTHRESH.ETHAnFLOWTHR[10:0] bits.

The Ethernet Controller monitors the quantity of data in the receive FIFO even while receive data is being transferred by DMA (see *(c)* in *Figure 31-10 “Flow control”*).

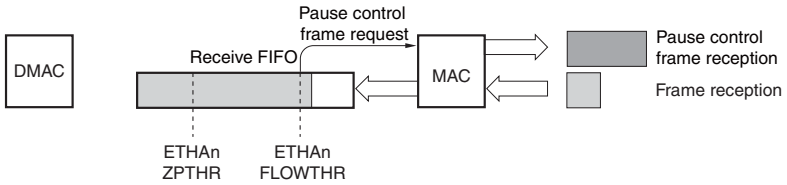
If the ETHAnMFFCONT.ETHAnZEROPAUSE bit is 1, a zero pause control frame is transmitted if the amount of data in the receive FIFO falls below the value specified by the ETHAnFLOWTHRESH.ETHAnZPTHR[10:0] bits (see *(d)* in *Figure 31-10 “Flow control”*).

If the ETHAnMFFCONT.ETHAnZEROPAUSE bit is 0, a zero pause control frame is not transmitted.

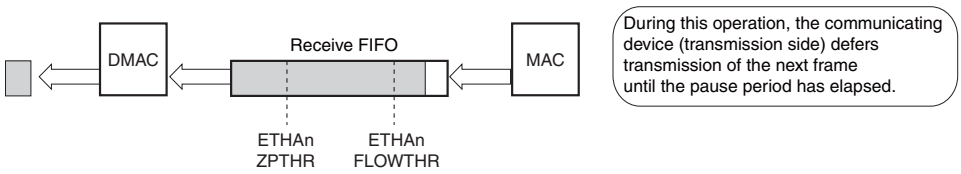
(a) When a frame is received, data is stored in the receive FIFO.



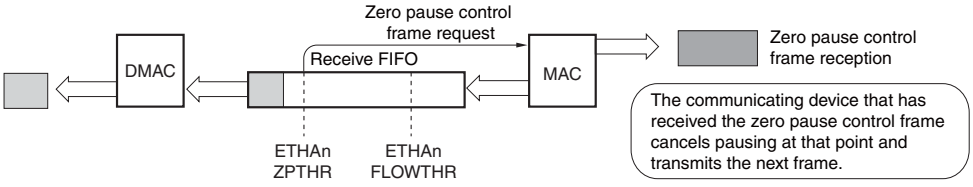
(b) When the amount of data in the receive FIFO exceeds the value specified by ETHAnFLOWTHR, a pause control frame is transmitted.



(c) The data in the receive FIFO is read by DMA.



(d) When the amount of data in the receive FIFO falls below the value specified by ETHAnZPTHR, a zero pause control frame is transmitted.



Note The shaded part ▭ of the receive FIFO in the above figures indicates the amount of data in the receive FIFO, not how data is actually stored.

Figure 31-10 Flow control

(2) Back pressure

This function is available only during half-duplex operation.

If the amount of data in the receive FIFO exceeds the value specified by the ETHAnFLOWTHRESH.ETHAnFLOWTHR[10:0] bits when the ETHAnMFFCONT.ETHAnFLOWCNT bit is 1 and the ETHAnMACC1.ETHAnFULLD bit is 0, the back pressure function is enabled (see *(b)* in *Figure 31-11 "Back pressure control"*).

If the next frame is received in this status, a collision is intentionally generated by transmitting a dummy packet, prompting the communicating device to retransmit the frame (see *(c)* in *Figure 31-11 "Back pressure control"*).

A collision that occurs in the back pressure status is not included in the number of collisions.

The back pressure status is released if the amount of data in the receive FIFO falls below the value specified by the ETHAnFLOWTHR[10:0] bits (see *(d)* in *Figure 31-11 "Back pressure control"*).

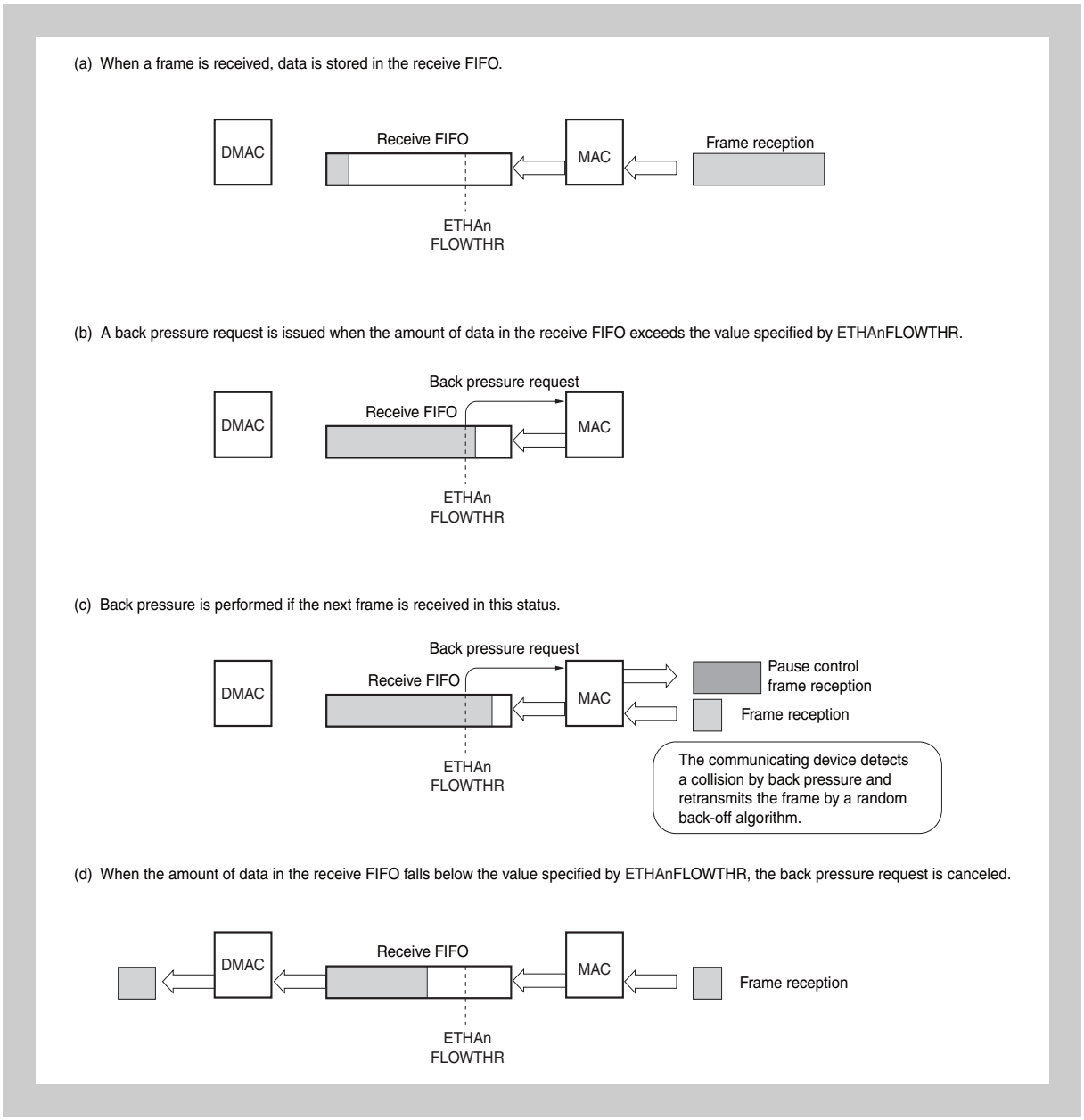


Figure 31-11 Back pressure control

(3) Operations related to VLAN frame

The Ethernet Controller detects a VLAN frame by comparing the TPID field in a receive/transmit packet with the value of the VLAN type register (ETHAnVLTP). Operations related to the VLAN frame are described below.

(a) Detection of VLAN frame

The Ethernet Controller constantly monitors the value of the 2-byte TPID field that follows the source address in a receive packet.

During transmission, the value of the TPID field is checked when the ETHAnMACC2.ETHAnAPD or ETHAnMACC2.ETHAnVPD bit is 1.

The Ethernet Controller recognizes a packet whose TPID field matches the value of the VLAN type register (ETHAnVLTP) as a VLAN frame.

(b) Reception of VLAN frame

If the value of the TPID field of a received packet matches the value of the VLAN type register (ETHAnVLTP), all judgments concerning the frame size are made based on the VLAN frame size (MAX: 1,522 bytes, MIN: 64 bytes).

(c) Transmission of VLAN frame

If a frame whose TPID field value matches the value of the VLAN type register (ETHAnVLTP) is transmitted from an upper layer when the ETHAnMACC2.ETHAnAPD bit is 1, the frame is recognized as a VLAN frame and is padded to extend its length to 68 bytes.

When the ETHAnMACC2.ETHAnVPD bit is 1, all frames are recognized as VLAN frames and are padded to extend their lengths to 68 bytes.

31.5.5 Serial management interface

The Ethernet Controller has a pair of serial management interfaces which can be used to set a PHY device, to read statuses, and for communicating with the PHY device when auto-negotiation is used.

Set the address of the PHY device to be connected by Ethernet Controller to the ETHAnMADR register before using the serial management interface.

(1) Overview of serial management interface

(a) MDC clock

The management data clock (MDC) is generated by dividing the Ethernet control clock (f_{EC}).

The division ratio is specified by the ETHAnMIIC.ETHAnCLKS[2:0] bits.

Table 31-115 ETHAnMIIC register: ETHAnCLKS[2:0] bits and f_{EC} frequency

ETHAnMIIC bits			Frequency range of f_{EC} input
CLKS[2]	CLKS[1]	CLKS[0]	
0	0	0	Setting prohibited
0	0	1	33 MHz or less
0	1	0	50 MHz or less
0	1	1	66 MHz or less
1	0	0	100 MHz or less
1	0	1	133 MHz or less
1	1	0	Setting prohibited
1	1	1	Setting prohibited

If the ETHAnMIIC.ETHAnPHYSEL bit is 0 (the default value), the MDC is output only when a management frame is transmitted or received.

The MDC is always output when the ETHAnMIIC.ETHAnPHYSEL bit is set.

If communication with the PHY device fails when the ETHAnMIIC.ETHAnPHYSEL bit is 0, set this bit.

(b) Serial Management Frame Structure

The Ethernet Controller generates a serial management frame shown below by writing a value to the ETHAnMCMD or ETHAnMWTD register.

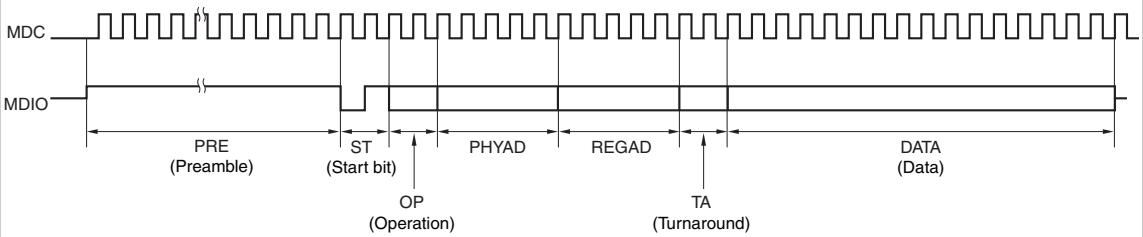
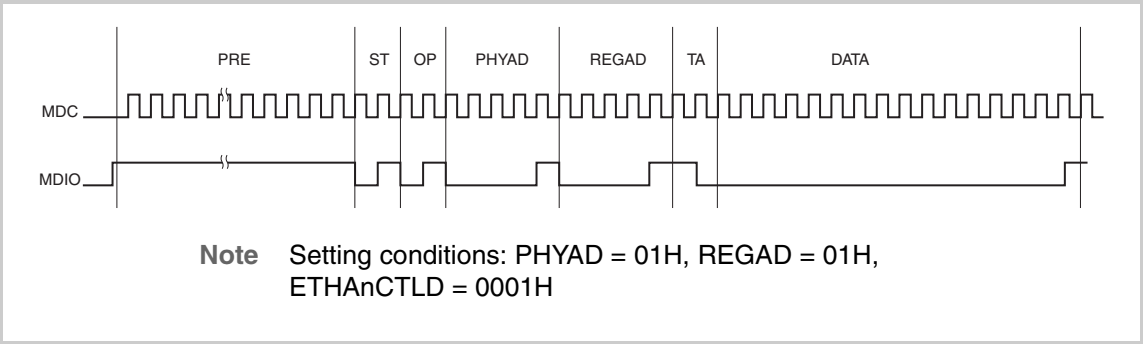


Figure 31-12 Serial management frame structure

A 32-bit preamble, 2-bit start bit field, and 2-bit opcode that indicates whether a register in the PHY device is read or written, are automatically appended to the serial management frame. PHYAD and REGAD indicate the address of the externally connected PHY device and the address of a register in that PHY device, respectively. The values set to the ETHAnMADR.ETHAnFIAD[4:0] and ETHAnMADR.ETHAnRGAD[4:0] bits are appended to PHYAD and REGAD, respectively.

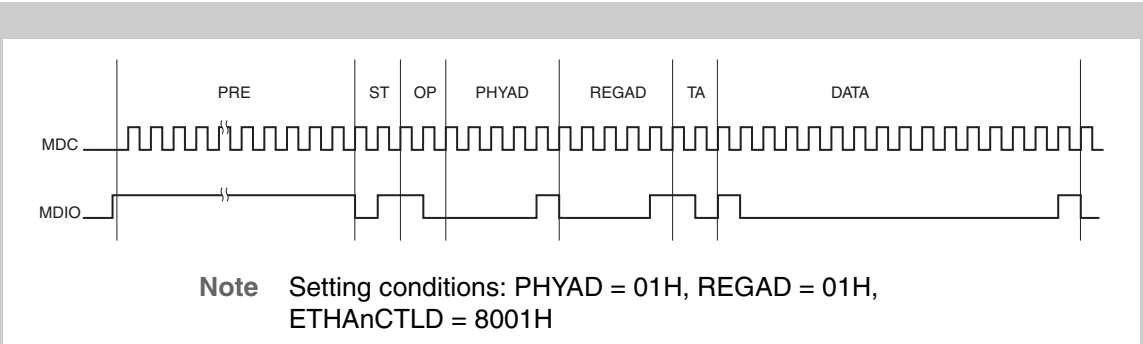
The Ethernet Controller serially outputs data from the preamble to REGAD, and after a 2-bit turnaround, the data set to the ETHAnMWTD.ETHAnCTLD[15:0] bits is output for a write access. For a read access, serial data is input by the MDI signal and written to the ETHAnMRDD.ETHAnPRSD[15:0] bits.

While the MDO signal is being output, the MDOEN signal is asserted to 1.



Note Setting conditions: PHYAD = 01H, REGAD = 01H, ETHAnCTLD = 0001H

Figure 31-13 Timing of MII management interface signals (write access)



Note Setting conditions: PHYAD = 01H, REGAD = 01H, ETHAnCTLD = 8001H

Figure 31-14 Timing of MII management interface signals (read access)

(c) SCAN command

The Ethernet Controller has a SCAN command to successively read a specific PHY register. When the ETHAnMCMD.ETHAnSCANC bit is set, read accesses are generated successively. By reading the ETHAnMRDD.ETHAnPRSD[15:0] bit, the specific PHY register can be polled.

(2) Procedure for transmitting or receiving a serial management frame

Serial management frames are transmitted and received as follows.

First, the ETHAnMIND.ETHAnSCANA bit is checked to see whether the SCAN command is under execution.

If not, the ETHAnMIND.ETHAnBUSY bit is checked to see whether the serial management frame is being accessed. If the ETHAnBUSY bit is 1, the Ethernet Controller waits until it is cleared. If the SCAN command is being executed, the ETHAnMCMD.ETHAnSCANC bit is cleared and then the Ethernet Controller waits until the ETHAnBUSY bit is cleared.

Next, the address of the external PHY device to which the frame is to be transmitted and the address of a register in the PHY device are set to the ETHAnMADR.ETHAnFIAD[4:0] and ETHAnMADR.ETHAnRGAD[4:0] bits, respectively.

Write access is started by writing to the ETHAnMWTD.ETHAnCTLD[15:0] bits.

The ETHAnBUSY bit is set when data has been written to the MWDT register and cleared when writing has finished.

Read access is started by writing 1 to the ETHAnMCMD.ETHAnRSTAT bit. When the ETHAnRSTAT bit is set, the ETHAnBUSY bit is set. The ETHAnBUSY bit is cleared after completion of reading. The host system can obtain the data of the PHY register by confirming that the ETHAnBUSY bit is 0 and then reading the ETHAnMRDD.ETHAnPRSD[15:0] bits.

To execute the SCAN command, set the ETHAnMCMD.ETHAnSCANC bit. While this bit is 1, reading is repeatedly executed. The ETHAnMIND.ETHAnSCANA bit holds 1 while the SCAN command is executed. The ETHAnMIND.ETHAnNVALID bit holds 1 until the first read access finishes after the SCAN command is executed. The ETHAnMIND.ETHAnBUSY bit is set when the SCAN command is executed. If the SCAN command is disabled (by clearing the ETHAnMCMD.ETHAnSCANC bit), the ETHAnMIND.ETHAnBUSY bit is cleared after the current read access finishes.

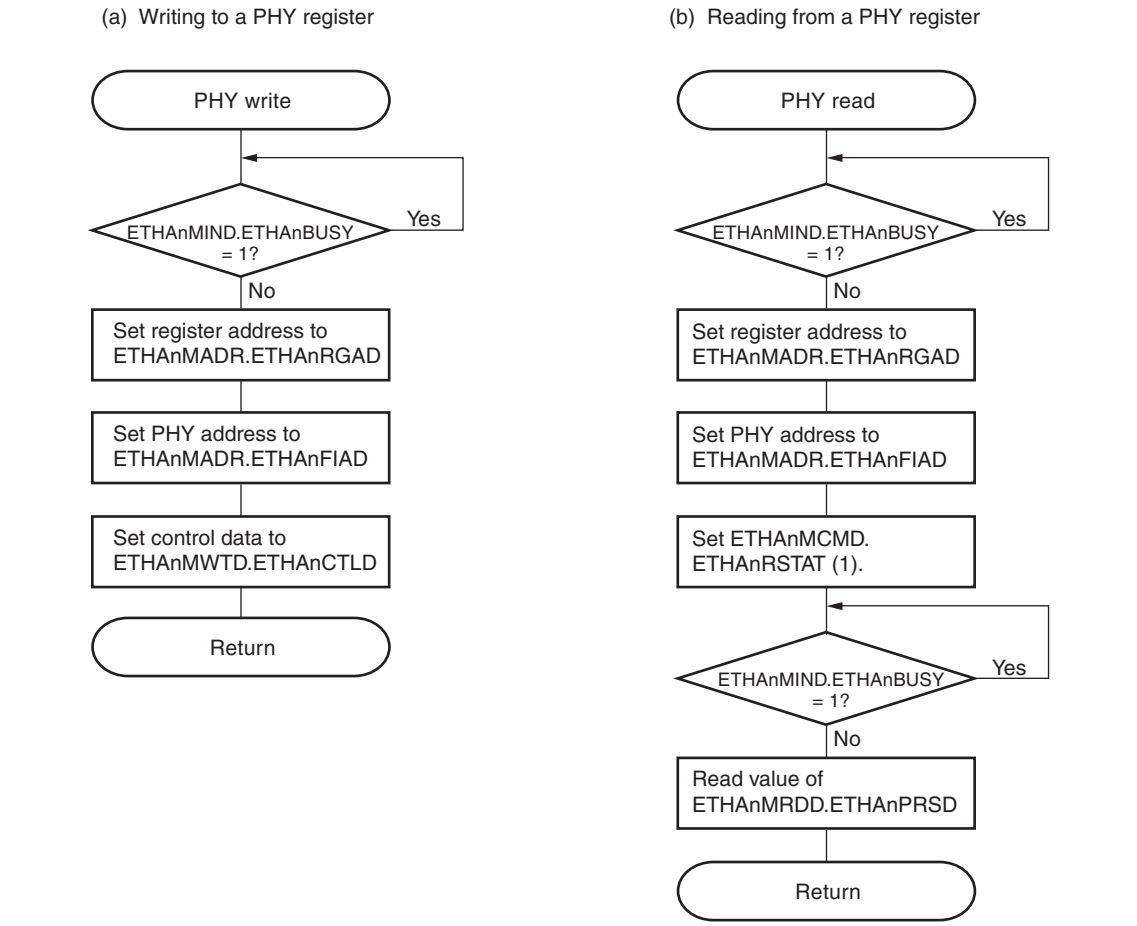


Figure 31-15 Accessing PHY register

31.5.6 Address filtering

(1) Overview of address filtering

The Ethernet Controller filters addresses by using the destination address of a received packet and, based on the filtering result, decides whether to accept or discard the received packet.

The filtering conditions can be specified by the ETHAnAFR register. Conditions can be individually specified for unicast addresses, multicast addresses, and broadcast addresses, or conditions can be combined.

(a) Filtering of unicast addresses

The address set to the ETHAnLSA1 and ETHAnLSA2 registers is compared with the destination address of a received packet as a unicast address. A received packet is accepted if its destination address matches the address specified for these registers, and is discarded if not. Each receive packet is checked to see if its destination address matches the set unicast address.

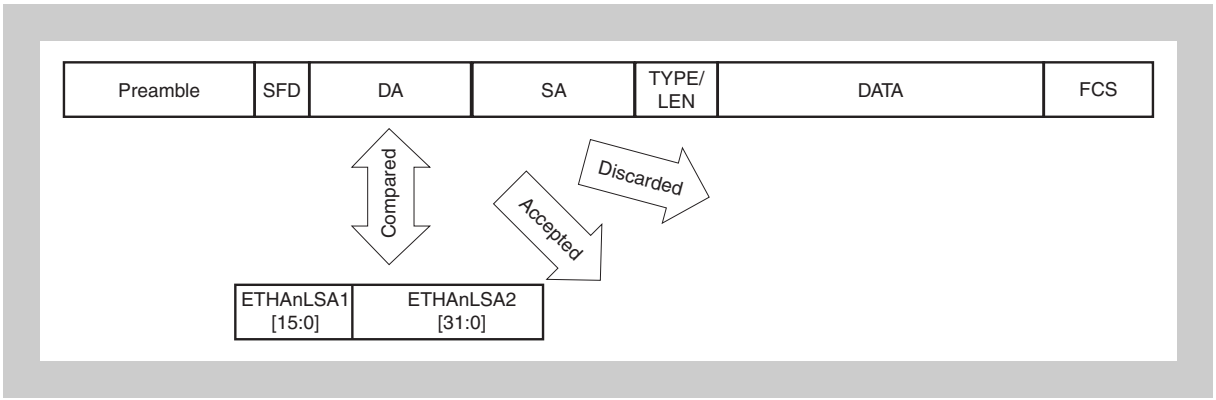


Figure 31-16 Image of filtering by unicast address during reception

(b) Filtering of multicast addresses

A multicast address is filtered in two ways. If the ETHAnAFR.ETHAnPRM bit is set, all received packets that have a multicast address as the DA are accepted.

If the ETHAnAFR.ETHAnAMC bit is set, only a received packet that has a multicast address that matches the hash table set up by the ETHAnHT1 and ETHAnHT2 registers is accepted, and a received packet whose multicast address does not match is discarded.

The hash table is used as follows for multicast address filtering.

The hash table is referenced by using bits [28:32] of the 32 bits of the CRC calculation result of the received multicast address. The following polynomial is used for calculating the CRC.

$$CRC\ (x) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$$

If 1 is specified at the bit position indicated by the value resulting from decoding the above 6 bits in the ETHAnHT1 and ETHAnHT2 registers (shown in *Table 31-116 "Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)"*), a receive packet that has that multicast address will be accepted. To set the hash table, it is necessary to execute a CRC calculation on a multicast address defined in advance, and set the corresponding bits.

Table 31-116 Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)

	CRC[25:23]							
CRC[28:26]	**111_B**	**110_B**	**101_B**	**100_B**	**011_B**	**010_B**	**001_B**	**000_B**
111_B	HT1[31]	HT1[30]	HT1[29]	HT1[28]	HT1[27]	HT1[26]	HT1[25]	HT1[24]
110_B	HT1[23]	HT1[22]	HT1[21]	HT1[20]	HT1[19]	HT1[18]	HT1[17]	HT1[16]
101_B	HT1[15]	HT1[14]	HT1[13]	HT1[12]	HT1[11]	HT1[10]	HT1[9]	HT1[8]
100_B	HT1[7]	HT1[6]	HT1[5]	HT1[4]	HT1[3]	HT1[2]	HT1[1]	HT1[0]
011_B	HT2[31]	HT2[30]	HT2[29]	HT2[28]	HT2[27]	HT2[26]	HT2[25]	HT2[24]
010_B	HT2[23]	HT2[22]	HT2[21]	HT2[20]	HT2[19]	HT2[18]	HT2[17]	HT2[16]
001_B	HT2[15]	HT2[14]	HT2[13]	HT2[12]	HT2[11]	HT2[10]	HT2[9]	HT2[8]
000_B	HT2[7]	HT2[6]	HT2[5]	HT2[4]	HT2[3]	HT2[2]	HT2[1]	HT2[0]

Note The bit name's prefix "ETHAn" is omitted in the above table, thus HT1[31:0] = ETHnHT1[31:0] and HT2[31:0] = ETHnHT2[31:0].

An example of a program that executes hash table calculations is shown below.

If DA = 123456789ABC, for example, CRC = D4E8 8056_H, CRC[28:26] = 5, and CRC[25:23] = 1. If ETHAnHT1[9] in *Table 31-116 "Correspondence between HT bits and hash values (ETHAnHT1 and ETHAnHT2)"* is set, a multicast packet with the target DA is received. If the value of both the ETHAnHT1 and ETHAnHT2 registers is 0000 0000_H, all packets are discarded.

// Calculate the set value of the hash table.

```
#include <stdio.h>

unsigned long crc32_for_ethernet( const unsigned char *data, int size );

// Address to be calculated
const unsigned char DA[] = { 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC };

int main( void ){
    unsigned long crc;

    printf("´nDA: ");
    crc = crc32_for_ethernet( DA, sizeof(DA) );
    printf("-----------------------------------------´n");
    printf("CRC = %02X,%02X,%02X,%02X´n", (crc>>24)&0xff, (crc>>16)&0xff, (crc>>8)&0xff,
    crc&0xff );
    printf("CRC[28:26] = %X, CRC[25:23] = %X ´n", (crc>>26)&0x07, (crc>>23)&0x07 );
    printf("´n");
    return(1);
    }

// Calculate the CRC.
unsigned long crc32_for_ethernet( const unsigned char *p, int size ){
    int i,j;
    const unsigned long poly = 0xEDB88320ul; // BigEndian
    unsigned long crc = 0xffffffff;
    unsigned long ans = 0x00000000;
    unsigned char c;

    for( j = 0; size-- != 0 ; j++ ) {
        c = *p++;
        printf("%02X " , c );
        if ( j == 15 ) {
            j = 0;
            printf("´n");
            }
        for ( i = 0; i < 8; i++ ) {
            crc = (crc>>1)^(((crc^c)&1)? poly : 0ul );
            c >>= 1;
            }
        }
    if ( j != 0 ) printf("´n");
    crc = ~crc;
    for( i = 0; i < 4; i++ ){
        ans = (ans << 8) | (crc & 0x000000ffful);
        crc >>= 8;
        }
    return( ans );
}
```

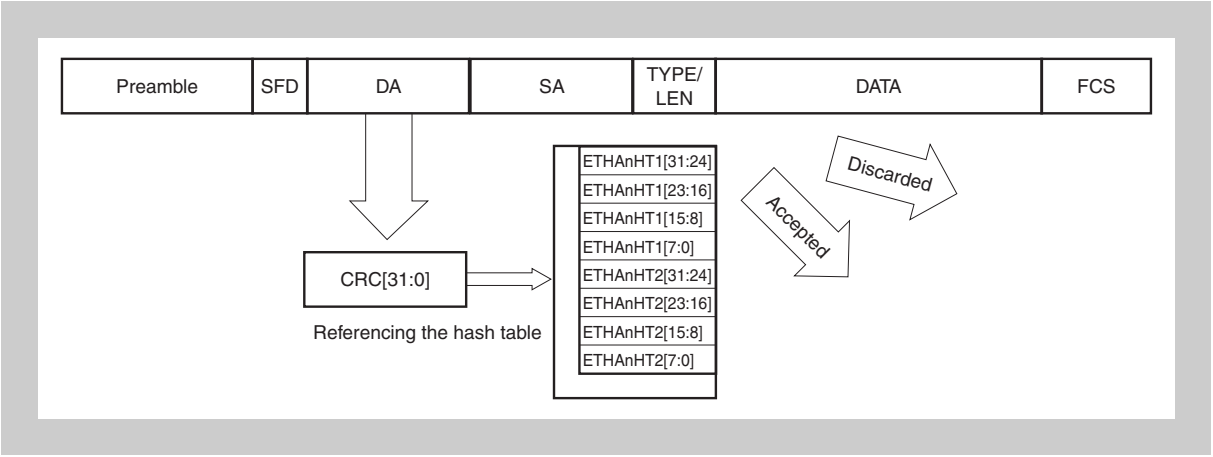


Figure 31-17 Image of filtering by referencing hash table

(c) Filtering of broadcast addresses

When the ETHAnAFR.ETHAnABC bit is set, a packet that has a broadcast address is received.

(d) Promiscuous mode

When the ETHAnAFR.ETHAnPRO bit is set, the promiscuous mode is set and all packets are received.

If none of reception conditions (a) to (d) above is satisfied, the received packet is discarded.

For the combinations of the above conditions, refer to *Table 31-117 "Settings of ETHAnAFR register and packets to be filtered"*.

Table 31-117 Settings of ETHAnAFR register and packets to be filtered

ETHAnAFR.ETHAn				Receive packet				
FRC	FRM	AMC	ABC	LSA mismatch, unicast	LSA match, unicast	HT mismatch, multicast	HT match, multicast	Broadcast packet
1	–	–	–	Accepted	Accepted	Accepted	Accepted	Accepted
0	1	–	–	Discarded				
0	0	1	1			Discarded		
0	0	1	0					Discarded[a]
0	0	0	1				Discarded	Accepted
0	0	0	0					Discarded

a) The broadcast packet can be received if the corresponding bit in the hash table is set because the broadcast address is included in a multicast address.

Note –: Any

(2) Setting address filtering conditions

Set address filtering as follows.

First, clear the ETHAnMACC1.ETHAnSRXEN bit. When the ETHAnSRXEN bit is 0, the receive data interface is disabled. Next, set a station address to the ETHAnLSA1 and ETHAnLSA2 registers. Set a combination of the necessary filtering conditions to the ETHAnAFR register. To conditionally receive multicast packets, a hash table must be set up by using the ETHAnHT1 and ETHAnHT2 registers. After making the above settings, enable packet reception by setting the ETHAnMACC1.ETHAnSRXEN bit.

31.5.7 Statistics counters

The Ethernet Controller has 39 statistics counters to check the communication quality and other line statuses.

Each time communication of one frame has been finished (or aborted), the communication status is checked and the corresponding statistics counter is updated. The statistics counters cannot be stopped. To avoid using a statistics counter, set the corresponding bit in the ETHAnCAM1 or ETHAnCAM2 register to mask the interrupt from that counter.

The statistics counters can be read at any time even during communication.

If a counter overflows, the corresponding bit in the ETHAnCAR1 or ETHAnCAR2 register is set, and, if the relevant interrupt is not masked by the ETHAnCAM1 and ETHAnCAM2 registers, an Ethernet MAC interrupt is generated. The ETHAnCAM1 and ETHAnCAM2 registers can be used to specify whether to mask the overflow interrupt of each counter.

To clear a statistics counter, write 0 to it. At this time, the current communication does not have to be stopped. If updating a statistics counter and writing data to it occur at the same time and conflict, updating takes precedence, and the counter is written after it has been updated.

Note that the statistics counters cannot be stopped. To avoid using a statistics counter, mask the counter by setting the corresponding bit in carry mask register 1 or 2 (ETHAnCAM1 and ETHAnCAM2) to prevent the INTCMAC interrupt from being generated.

The statistics counters can be read/written in 32-bit units.

Cautions

1. The Ethernet Controller updates the statistics counters based on the Ethernet Controller clock (f_{EC}). If the Ethernet Controller clock (f_{EC}) is considerably slower than the communication clock (TXCLK/RXCLK), the counters might miscount statistics information. If the statistics information is miscounted, a status vector overrun occurs, the ETHAnC2DV bit in carry register 2 (ETHAnCAR2) is set, and an INTCMAC interrupt is generated.
2. Carry registers 1 and 2 (ETHAnCAR1 and ETHAnCAR2) are cleared when they are read.

Note Aside from the statistics counters, a transmission abort counter (ETHAnTXABTCNT) and a reception abort counter (ETHAnRXABTCNT) are available for counting the number of times transmission/reception has been aborted.

31.6 Dedicated DMAC

31.6.1 DMAC overview

The dedicated DMA controller (DMAC) enables a DMA communication with the internal system bus of the Ethernet Controller.

Two dedicated DMACs are available: the DMAC for the Ethernet Controller used for transmission/reception and the DMAC for the transmit checksum used for transmit checksum.

All data to be transmitted and received is transferred by the DMAC for the Ethernet Controller. The DMAC for transmit checksum is used to store transmit data for the checksum calculation and output the result of the checksum calculation.

Note The functions of the DMAC for the Ethernet Controller and the DMAC for the transmit checksum are identical. This section describes the DMAC for the Ethernet Controller as representative. For the signals, registers, and bits of the DMAC for the transmit checksum, replace them as shown in the following table when reading this section.

Item	DMAC for Ethernet Controller	DMAC for transmit checksum
Interrupt signal name	INTSCTX, INTSCRX	INTSCTXTCH, INTSCRXTCH
Register and bit names	Example: ETHAnINTMS register RXMASK bit	Example: TCH_INTMS register TCH_RXMASK bit

(1) Register access

A register of the DMAC is accessed from the CPU as a slave of the system bus.

It can be accessed in units of bytes (8 bits), halfwords (16 bits), or words (32 bits). However, the MAC control registers (0F04 2000_H to 0F04 $21FCH_H$ can be accessed only in 32-bit units.

A writing to a reserved area of the registers is ignored. When a reserved area is read, the data is all read as zero.

(2) Bus master

The DMAC for the Ethernet Controller transfers data to/from the system bus as a bus master.

The burst size can be specified by setting up the transfer control registers (ETHAnTRANSCTL and TCH_TRANSCTL).

All data to be transmitted and received is transferred by the internal DMAC. In addition, the bus master function of the DMAC also calculates a transmit checksum, stores transmit data, and outputs the checksum calculation result.

31.6.2 DMA

(1) DMA transfer mode

The following settings can be specified by using the ETHAnDMACM register.

Transfer data size

– Word (32 bits) and byte (8 bits) data are supported.

Transfer type

– 2-cycle transfer

Transfer mode

– Single transfer mode
– 4-beat incremental burst transfer mode
– 8-beat incremental burst transfer mode
– 16-beat incremental burst transfer mode

After the transfer mode has been specified by the ETHAnDMACM register, it will be applied starting with the next DMA transfer.

Cautions

1. Undefined length burst transfer mode cannot be specified using the register.
   Undefined length burst transfer is automatically used by the Ethernet Controller to process fractional data.
   It is therefore not possible to transfer all transfer data in this mode intentionally.
2. The register that sets the transfer mode is not locked during a DMA transfer. If the setting of this register is changed during a DMA transfer, therefore, the current DMA cycle becomes illegal. Do not change the set value of the register during a DMA transfer (when ETHAnRXEN_STA or ETHAnTXEN_STA is 1).

(2) Areas accessible by DMA transfer

The areas subject to DMA transfers are shown below.

Table 31-118 Areas subject to transfers by DMAC for Ethernet Controller

Instruction RAM	Data RAM	Work RAM	External memory	SiP internally connected memory and user registers	Serial flash ROM	On-chip peripheral functions (with some exceptions)

(3) DMA address boundary

With the DMAC for the Ethernet Controller, the address boundary does not have to be considered when setting the start address of the data buffer and the number of transfer bytes.

If there is fractional data during a burst transfer, the fraction is automatically processed before the data is transferred.

However, because it is not possible to predict where the data to be received will end, the last transfer might be a dummy transfer when a burst transfer is used.

Note In the 4-, 8-, or 16-beat incremental burst transfer mode, if the last data is shorter than the fixed length, it is automatically transferred in undefined length burst transfer mode.

Byte access for byte alignment is always executed in the single transfer mode.

(4) DMA arbitration

Because the Ethernet Controller supports full-duplex transfer, DMA transmission and DMA reception might be executed together. If DMA requests for transmission and reception are issued at the same time, the reception request takes precedence.

31.6.3 Descriptor mechanism

The Ethernet Controller supports a descriptor mechanism to support a situation where the memory space that stores transmit data and receive data is not contiguous.

The Ethernet Controller uses the following three types of descriptors.

- Buffer descriptor
- Link pointer
- End of chain

Each descriptor loads two word-aligned data (64 bits) onto memory (refer to *Figure 31-26 "Example of descriptor chain configuration (during packet reception)"*).

The Ethernet Controller can consecutively process multiple descriptors in one DMA transfer (refer to *8 "Descriptor chain" in 31.6.3 "Descriptor mechanism"*).

A reception DMA transfer or transmission DMA transfer is started by setting the first address of a receive descriptor chain to ETHAnRXDP or the first address of a transmit descriptor chain to ETHAnTXDP, and then setting the ETHAnRXS or ETHAnTXS bit of the ETHAnMODE register.

A descriptor chain must end with the descriptor of an end of chain.

(1) Format of buffer descriptor

A buffer descriptor is configured of 2 words (64 bits). The lower word consists of control bits. The higher word indicates the start address of the data buffer indicated by the descriptor.

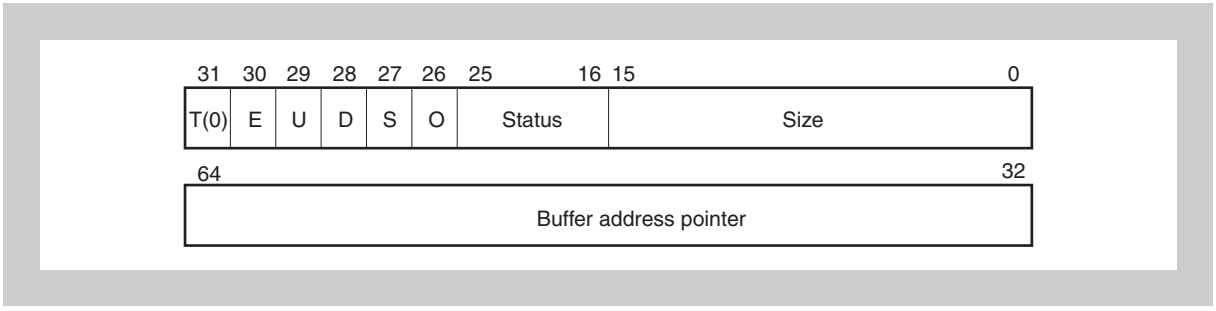


Figure 31-18 Buffer descriptor format

Table 31-119 Bit configuration of buffer descriptor (1/2)

Bit position	Bit name	Function
63 to 32	BAP	These bits specify an address pointer that indicates the start address of the data buffer. Byte alignment can be specified for BAP.
31	T	Descriptor type This bit indicates the type of the descriptor. Clear this bit for a buffer descriptor.
30	E	Last buffer flag This is a control bit that indicates the end of packet data. 0: Normal buffer data (not last data) 1: Last buffer data of the current packet If this bit is set for transmission, a TXI interrupt is generated when the data of the corresponding data buffer has been transferred, and then the next descriptor processing starts. Clear this bit for reception. When the last data of a frame has been written, this bit is set when the data is written back. Next, an RXI interrupt is generated, and then the next descriptor processing starts.
29	U	Used bit This bit indicates whether a DMA transfer has finished (including transfer in progress). 0: Transfer not finished (including transfer in progress) 1: Transfer finished The CPU clears this bit when it creates or obtains buffer data (a descriptor). When a DMA transfer to the buffer area indicated by this descriptor has finished, the Ethernet Controller sets this bit. The Ethernet Controller issues the TECI or RECI interrupt and stops DMA if it reads a descriptor whose U bit is set.
28	D	This bit indicates an access error in the data buffer. 0: No error 1: Access error in the data buffer The CPU clears this bit when it creates or obtains buffer data (a descriptor). If an access error occurs, the Ethernet Controller sets control bit D of the first descriptor that indicates the current packet, and control bit D of the descriptor that was responsible for the access error.
27	S[a]	This bit indicates that reception status information has been written to the Status field (only control bit S in the first descriptor of a received packet is valid). 0: Status information is not included. 1: Status information of the received packet is included. The CPU clears this bit when it creates or obtains buffer data (a descriptor). When a received packet is transferred via DMA, the Ethernet Controller writes a valid value to the Status field of the first descriptor of the current packet and sets control bit S each time one packet has been transferred.

Table 31-119 Bit configuration of buffer descriptor (2/2)

Bit position	Bit name	Function
26	O[a]	This bit indicates occurrence of an overflow error during reception. 0: No overflow 1: Overflow The CPU clears this bit when it creates or obtains buffer data (a descriptor). If an overflow error occurs during reception, the Ethernet Controller writes back 1 to the control bit O of the first descriptor of the packet, and sets control bit E of the descriptor in which the overflow error occurred. No interrupt is generated.
25 to 16	Status[a]	These bits form a Status field that indicates status information during reception. If control bit S is 1, the value of the Status field is valid. The CPU clears this bit when it creates or obtains buffer data (a descriptor). During a DMA transfer of a receive packet, the Ethernet Controller writes a valid value to the Status field of the first descriptor of the current packet and sets control bit S each time transfer of one packet has finished.
15 to 0	Size	These bits form a Size field that indicates the size (in bytes) of the buffer data indicated by this descriptor. During a DMA transfer of a receive packet, the Ethernet Controller writes the length of one transferred packet to the Size field of the last descriptor of the current packet each time transfer of one packet has finished.

a) This bit is not used during transmission. Clear this bit.

Note The Size field is 16 bits. Setting 0 to this field is prohibited. If 0 is set, an error interrupt is generated.
If $FFFF_H$ is set to this field, transfer of 64K – 1 bytes is executed.

(2) Format of link pointer

A link pointer consists of 2 words. The lower word consists of control bits. The higher word indicates the address of the next descriptor.

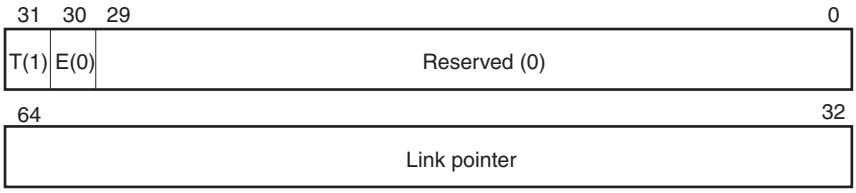


Figure 31-19 Link pointer format

Table 31-120 Bit configuration of link pointer

Bit position	Bit name	Function
63 to 32	Link Pointer	These bits indicate the address of the next descriptor. The lower 2 bits are ignored (word aligned).
31	T	Set this bit for a link pointer.
30	E	Clear this bit for a link pointer.
29 to 0	Reserved	These bits are reserved. Clear these bits.

(3) Format of end of chain

An end of chain consists of 2 words. The lower word consists of control bits. The higher word indicates 0.

When the Ethernet Controller detects an end of chain, it finishes the DMA transfer and generates an RECI or TECI interrupt.

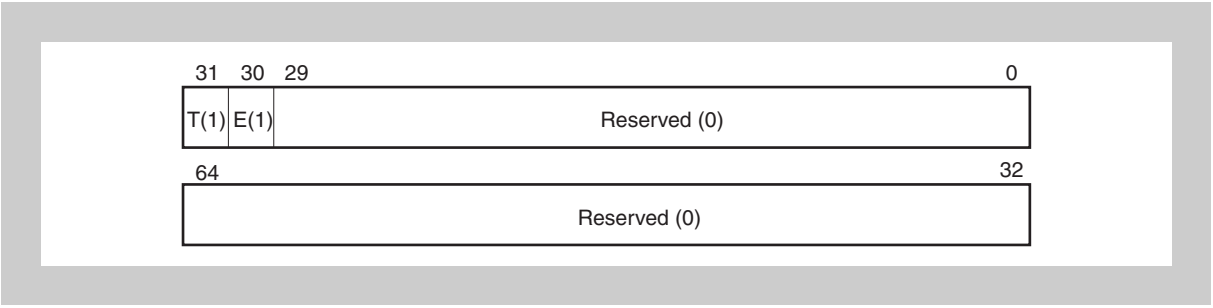


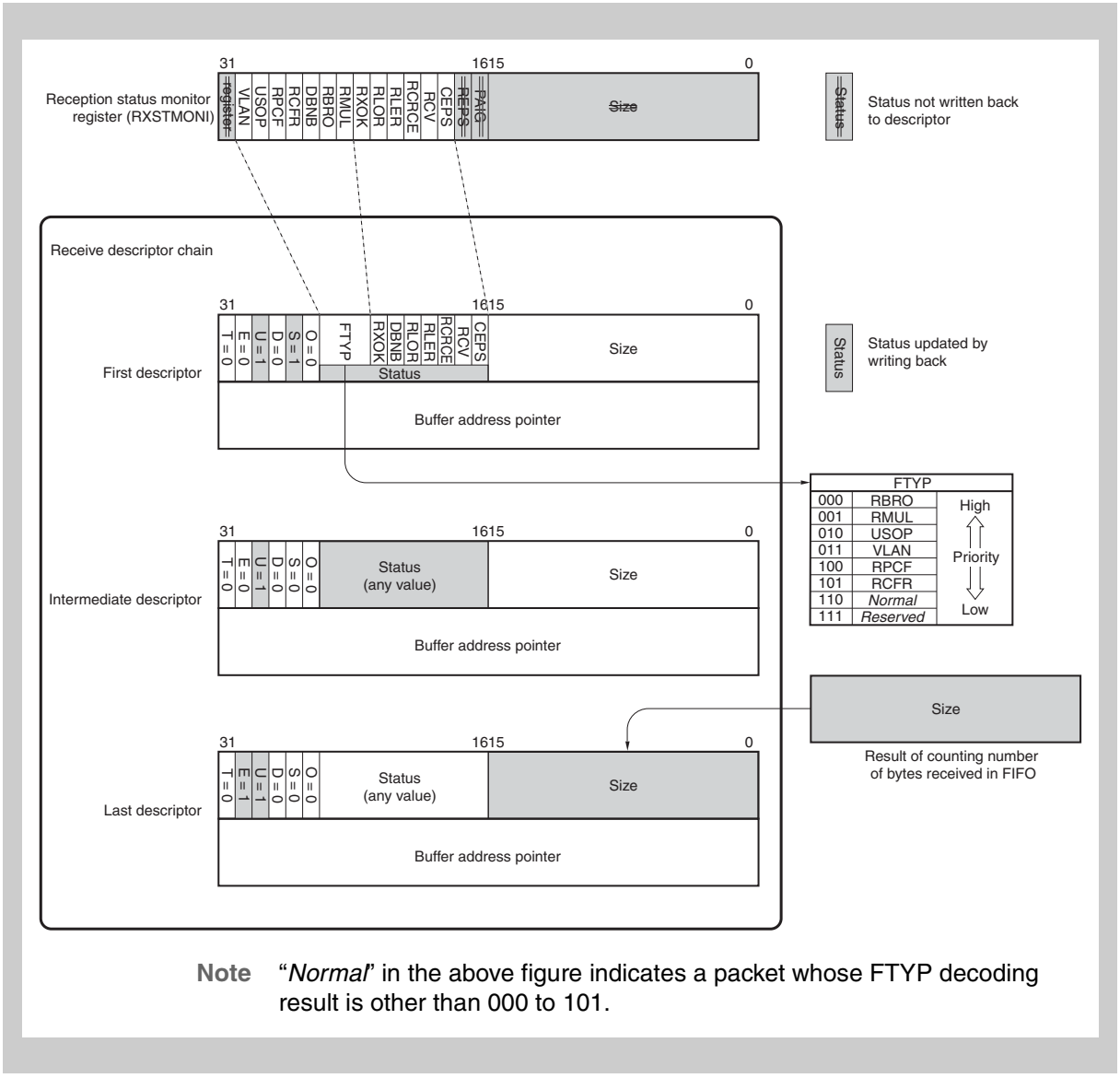
Figure 31-20 End of chain format

Table 31-121 Bit configuration of end of chain

Bit position	Bit name	Function
63 to 32	BAP	These bits are set to null (all zero) for an end of chain.
31	T	Set this bit for an end of chain.
30	E	Set this bit for an end of chain.
29 to 0	Reserved	These bits are reserved. Clear these bits.

(4) Writing back the status

When DMA reception is executed, the reception status is written back to the first descriptor of the packet, and the length of the packet transferred via DMA is written back to the last descriptor. The status is written back as shown below.



Note "*Normal*" in the above figure indicates a packet whose FTYP decoding result is other than 000 to 101.

Figure 31-21 Example of writing back status

(5) Error occurrence

If a bus error occurs when the data buffer is accessed during transmission or reception, an error interrupt is generated and DMA is stopped. In addition, the U and D bits of the first descriptor of the packet are set (the U bit might have already been set in some cases). The U, D, and E bits of the descriptor in which the error occurred are also set.

If an overflow occurs during reception, the U and O bits of the first descriptor of a packet are set (the U bit might have already been set in some cases). The U and E bits of the descriptor in which the overflow occurred are also set.

(6) Error interrupt

In addition to when an access error to a data buffer occurs, an error interrupt is also generated when an access error to a descriptor occurs. Whether an error interrupt has been generated can be checked using the status of the ETHAnINTMS.ETHAnRBEI and ETHAnINTMS.ETHAnTBEI bits.

If a data buffer or descriptor access error occurs, the descriptor chain containing the descriptor in which the error occurred must be reorganized.

Transmission: If a descriptor or data buffer access error occurs, the ETHAnTBEI bit is set and DMA is stopped.
Transmission is not performed until the ETHAnTXS bit is set next time.

Reception: If a descriptor or data buffer access error occurs, the ETHAnRBEI bit is set and DMA is stopped.
Reception is not performed until the ETHAnRXS bit is set next time. At this time, the packet being transferred from the FIFO is discarded.
No packet will be discarded if the packet transfer has not started. If a bus error occurs during a write back operation caused by a reception overflow, the same processing is performed as that performed when a descriptor access error occurs.

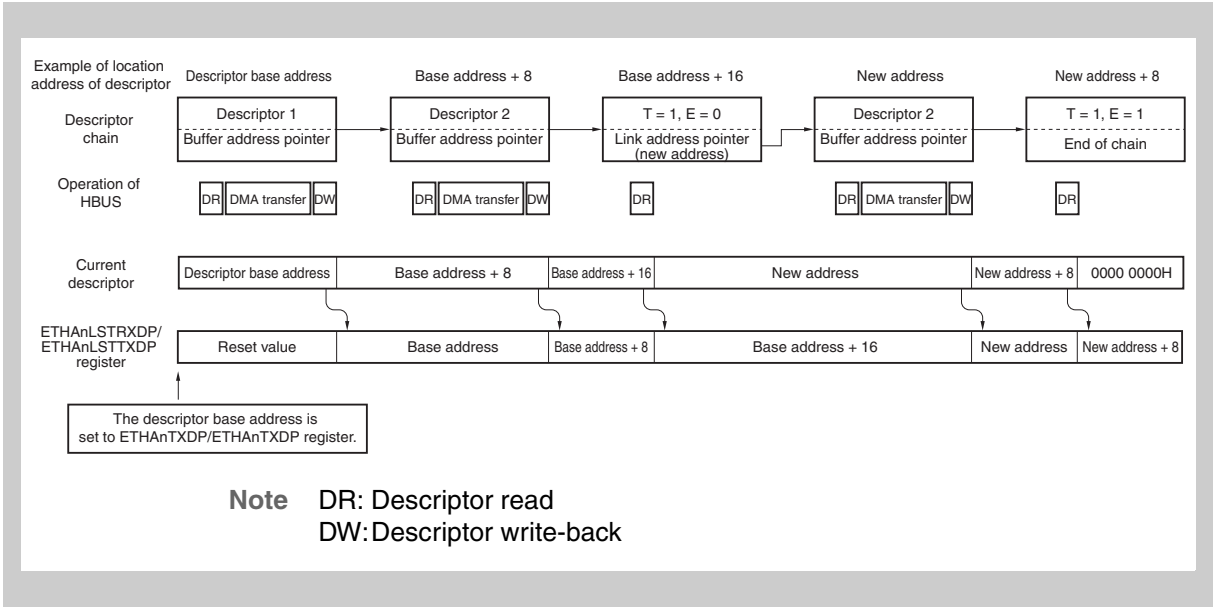
(7) Reporting last descriptor

The current descriptor can be reported. Two registers, ETHAnLSTRXDP and ETHAnLSTTXDP, hold the address of the descriptors processed by the Ethernet Controller. The address of the descriptor that was processed immediately before can be obtained by reading these two registers via software.

The timing to save the address of a descriptor to ETHAnLSTRXDP or ETHAnLSTTXDP is as follows.

After the data of a descriptor has been transferred and the descriptor has been written back, the address of the descriptor is copied to ETHAnLSTRXDP or ETHAnLSTTXDP.

When the link pointer is read, the address of the next descriptor can be read from the BAP bits. The address of the link pointer is then copied to ETHAnLSTRXDP or ETHAnLSTTXDP.



Note DR: Descriptor read
DW: Descriptor write-back

Figure 31-22 Timing to copy last descriptor

If a descriptor chain is configured in a ring buffer, the descriptor can be updated by reading ETHAnLSTRXDP or ETHAnLSTTXDP, using the TXI flag of the INTSCTX interrupt (the RXI flag of the INTSCRX interrupt) as a trigger.

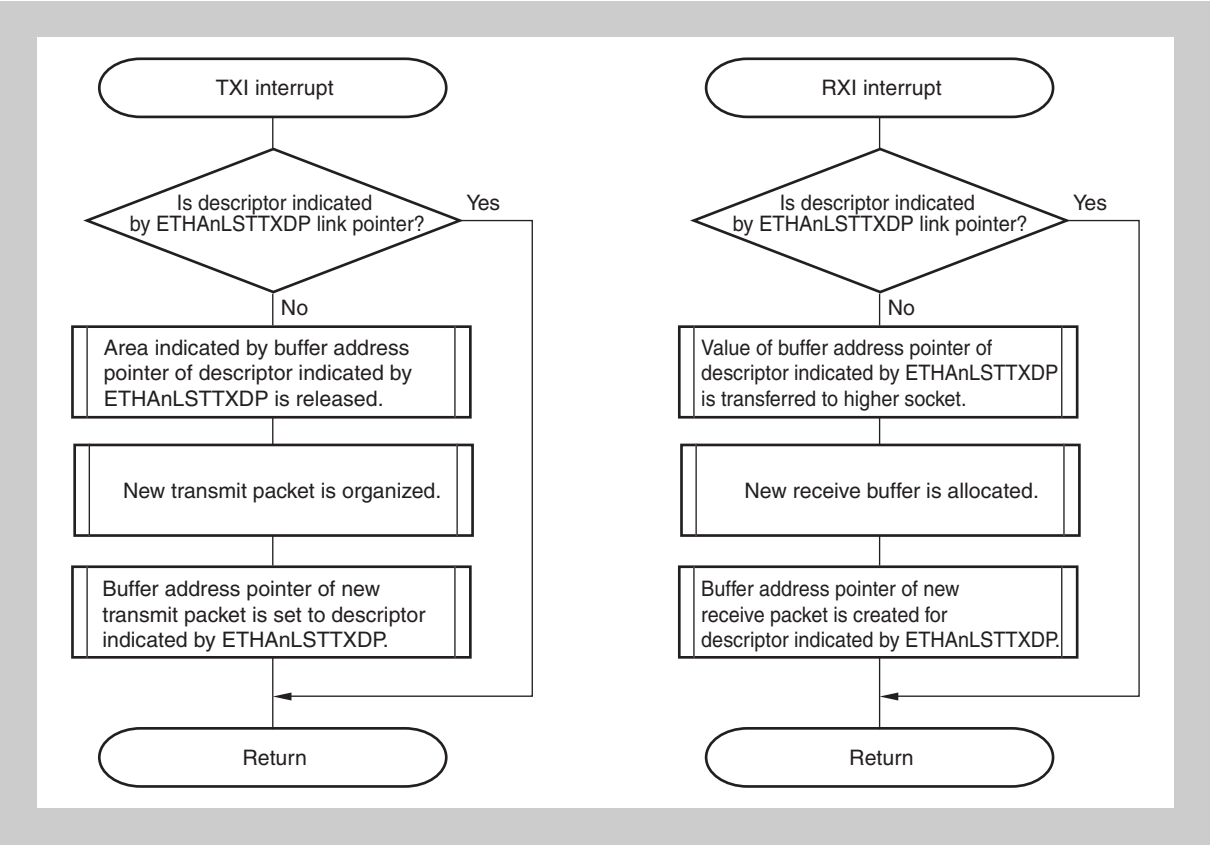


Figure 31-23 Updating descriptor chain by using ETHAnLSTRXDP or ETHAnLSTTXDP

(8) Descriptor chain

A descriptor uses a chain structure to indicate a data buffer (of an undefined length).

An image is as shown below.

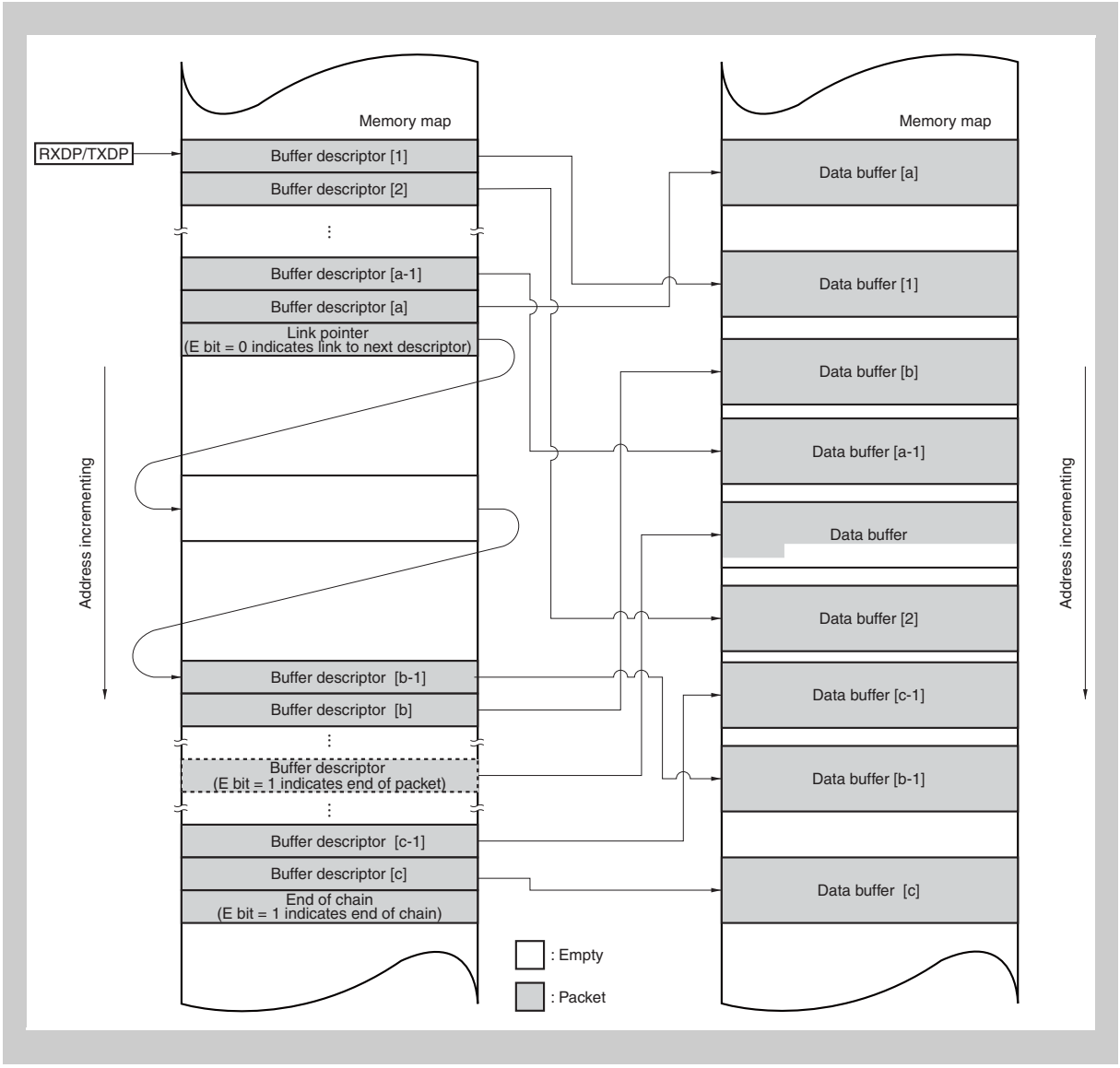


Figure 31-24 Overview of descriptor chain

Following a buffer descriptor, another buffer descriptor or a link pointer is allocated successively at the next memory address (+ 8). The link pointer indicates the address of the next buffer descriptor. This combination makes up a descriptor chain.

– The descriptor chain ends when an end of chain is set.

If an end of chain is detected before all receive packets are stored, the ETHAnTBEI (or ETHAnRBEI) bit is set to report that a buffer access error interrupt INTETMTX (or INTETMRX) has been generated.

Multiple packets can be stored in one descriptor chain.

A ring descriptor chain may be configured by using the last link pointer to specify the memory address of the first buffer descriptor.

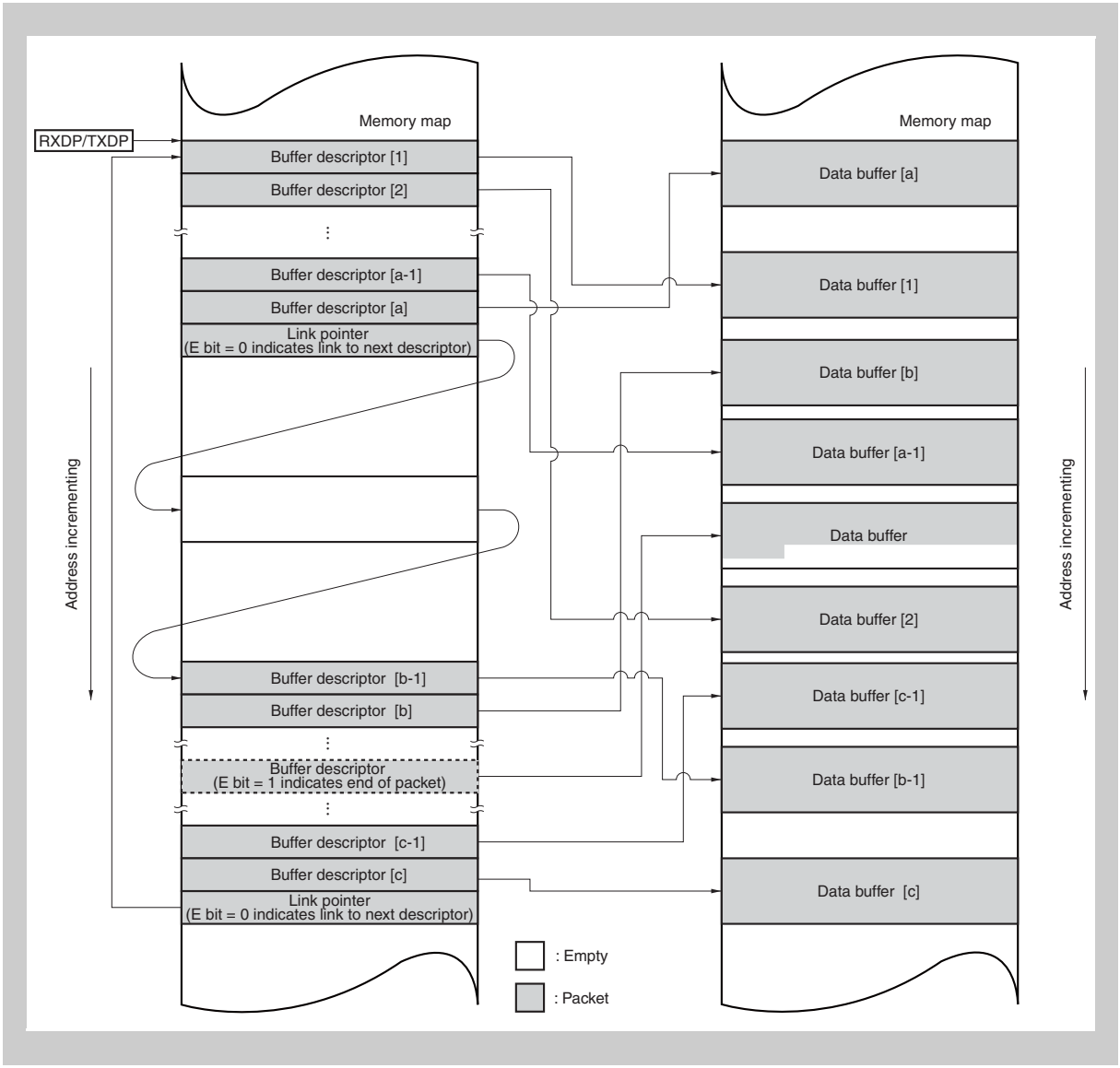


Figure 31-25 Overview of ring buffer formed by descriptor chain

A ring buffer is configured if the beginning of a descriptor chain is specified by a link pointer. In a ring buffer, if a descriptor whose U bit is set is read, the Ethernet Controller generates a RECI or TECI interrupt in the same manner as when detecting an end of chain, and then stops DMA.

Caution Handling of U bit

If the U bit of a transmit descriptor is set, it indicates that the processing for that descriptor has finished, so the CPU can specify a new descriptor by clearing the U bit.

For a receive descriptor that has the U bit set to 1, however, its content might be updated later due to status write back or error occurrence, so a new descriptor cannot be specified unless the completion of packet reception is confirmed. But if the E bit is set, it indicates that packet reception has finished, so the descriptor chain can be specified as a new descriptor.

Figure 31-26 "Example of descriptor chain configuration (during packet reception)" shows an example of using of a descriptor chain during reception.

If the ETHAnRXS bit of the core function setting register (ETHAnMODE) is set by software, the first descriptor is read from the address ($0F19\ 0400_H$) indicated by a receive descriptor pointer (ETHAnRXDP) and analyzing the receive descriptor starts. The first buffer address pointer ($0F18\ 4001_H$) is specified as the DMA transfer start address and the receive data in the FIFO is transferred to buffer A.

If buffer A becomes full after receiving the subsequent data, the next descriptor ($0F19\ 0408_H$) is read, the buffer address pointer ($0F18\ 4803_H$) of the descriptor indicated by the link address pointer (0F19 0500H) is specified as the DMA transfer start address, and then the receive data in the FIFO is transferred to buffer C.

The E and U bits of the last descriptor are set and the number of data that has been transferred to the Size field is written back.

After all packet data has been transferred, the U and S bits of the first descriptor are set and the receive status information is written back to the Status (A) field.

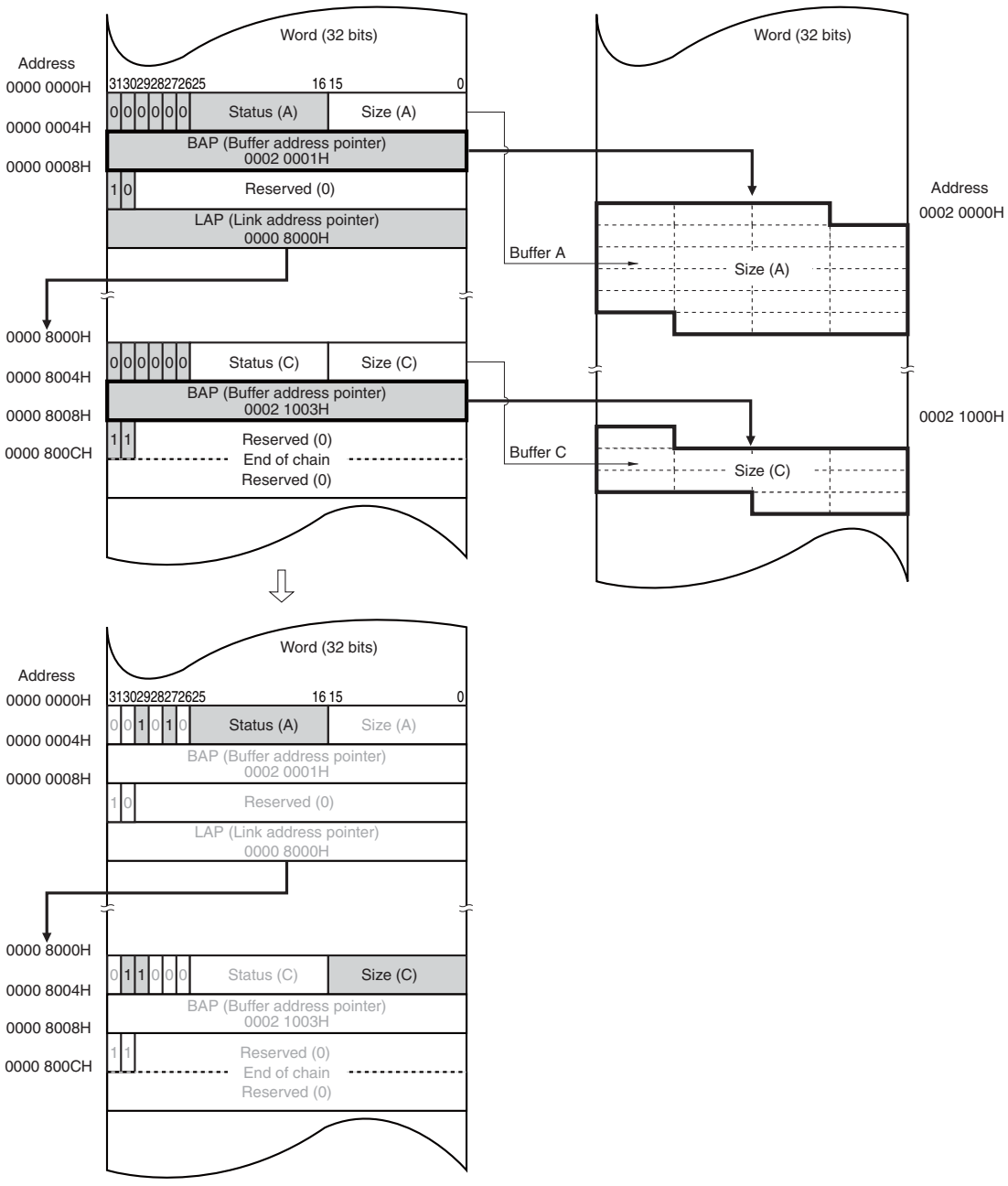


Figure 31-26 Example of descriptor chain configuration (during packet reception)

(9) Transfer by descriptor mechanism

Data can be transmitted or received easily by using the descriptor mechanism.

Data can be automatically transferred through descriptor chain control by the CPU, as shown below.

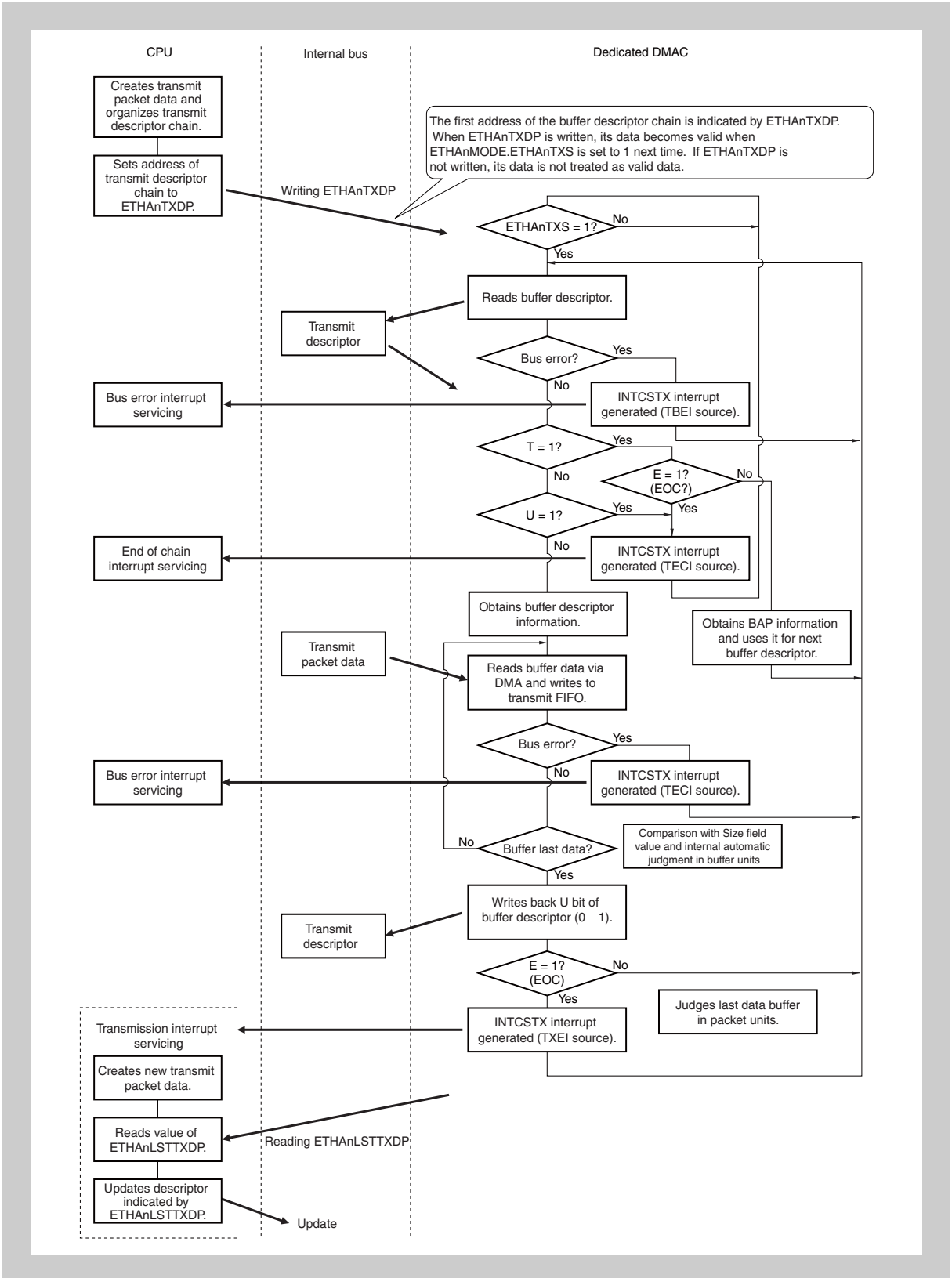


Figure 31-27 Data transfer flow (transmission)

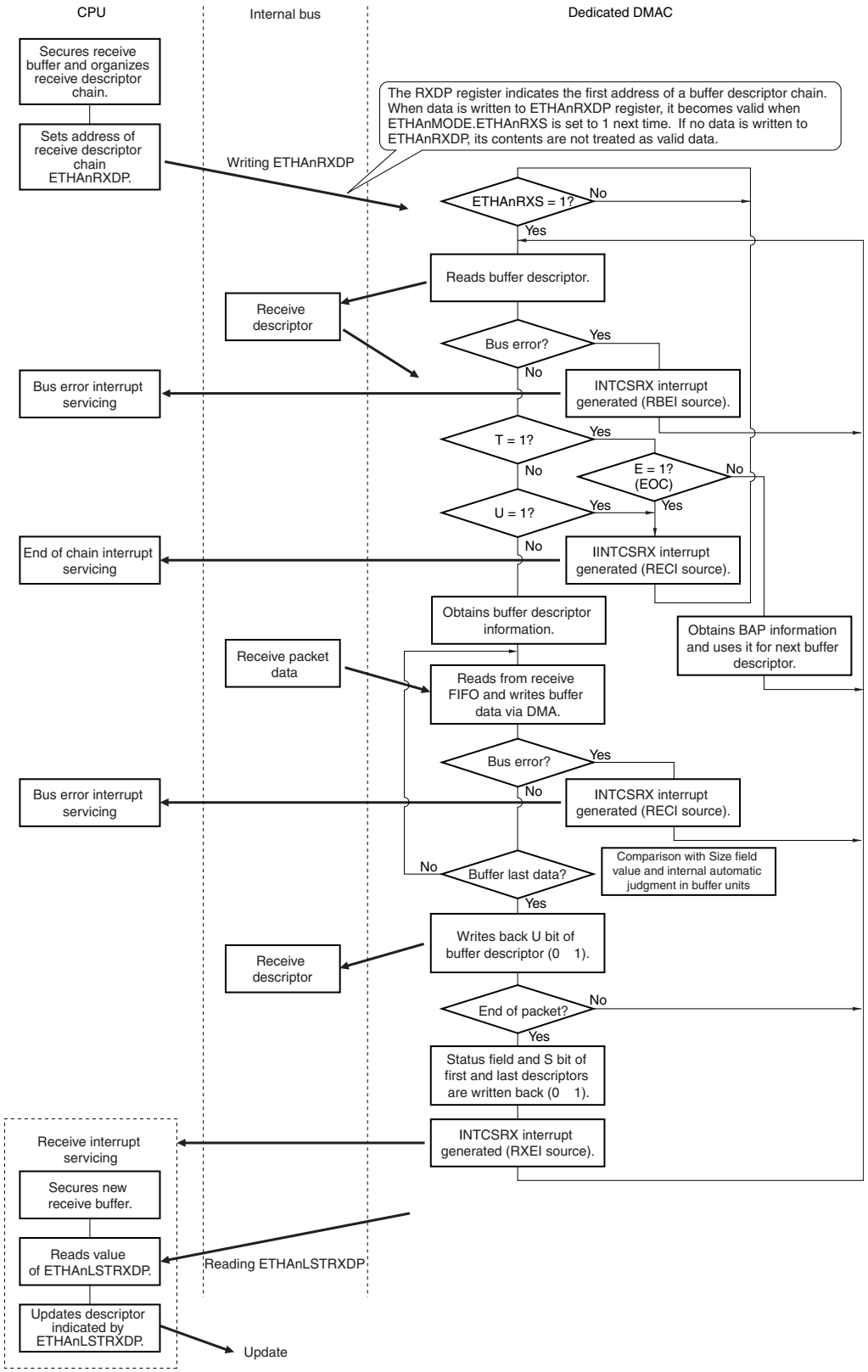


Figure 31-28 Data transfer flow (reception)

(10) Byte alignment and word boundary

Although a descriptor must be word-aligned, the data buffer can be set at a byte-aligned address.

The Ethernet Controller automatically identifies an address, executes a single transfer up to a word boundary and an undefined length word transfer up to the burst boundary, and then executes a burst transfer.

31.7 Receive Checksum

The Ethernet Controller has a receive checksum unit that calculates the receive checksum.

Receive checksum calculation is enabled and disabled by setting the ETHAnRXCHKSMEN bit of the ETHAnTRANSCTL register.

A checksum is appended to the end of receive data. Allocate enough memory for the amount of received data, which is the packet length plus 2 bytes.

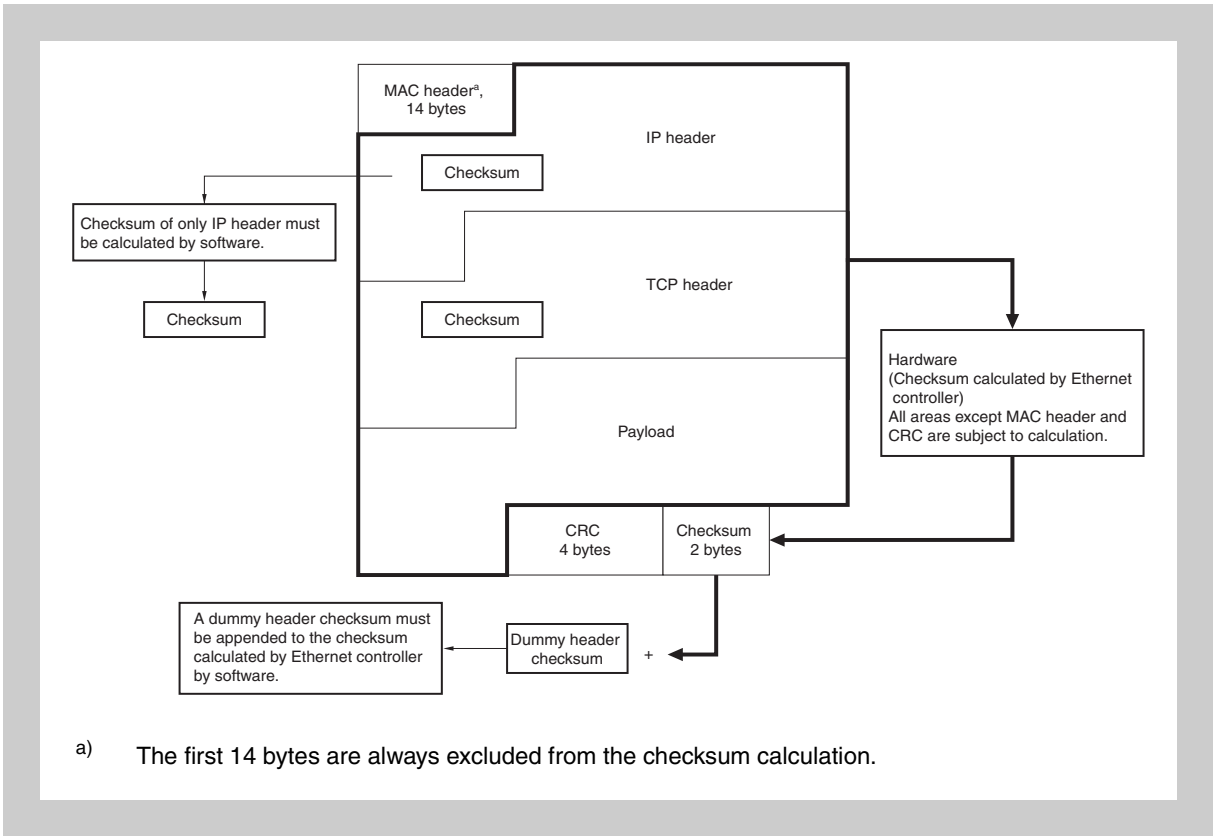
When checksum calculation is enabled, all parts (payload) of a receive frame, except the MAC header (first 14 bytes) and CRC (last 4 bytes), are subject to checksum calculation. If the number of bytes subject to calculation is odd, 00H is added to the last byte for checksum calculation.

The minimum receive packet length subject to checksum calculation is 19 bytes (payload = 1 byte). If the receive packet length is 18 bytes (payload = 0 bytes) or less, 0 is output as the checksum. However, the length information increases by 2 bytes.

Before changing the value of the ETHAnRXCHKSMEN bit, confirm that transfer of the receive frame is stopped.

31.7.1 Processing by software

The first 14 bytes of a packet are treated as the MAC header and are always excluded from checksum calculations. Therefore, calculation starts from the 15th byte. If the MAC header exceeds 14 bytes, such as in a VLAN and huge frame, correction by software is necessary.



a) The first 14 bytes are always excluded from the checksum calculation.

Figure 31-29 Checksum calculation

The minimum packet length is 64 bytes. When padding a short packet, many systems use 00_H, but some systems use a different code for padding. In this case, the checksum calculation is executed including the padding data, resulting in a mismatch between the calculation result and the checksum in the header. In this case, the checksum of the padding data must be corrected by software.

Caution According to RFC1071, the correct checksum comparison result can be obtained if the checksum result is compared using the same endian format (either big endian or little endian).

31.8 Transmit Checksum

The Ethernet Controller has a transmit checksum function. This function has a dedicated DMAC (transmit checksum DMAC), which accesses the internal system bus independently of the internal Ethernet Controller DMAC.

The algorithm for a transmit checksum complies with RFC1071, a payload is added in a 32-bit width, and a 2-byte checksum is output to the end of the data.

The minimum transmit payload subject to checksum calculations is 1 byte, and the maximum is 1,500 bytes as defined in IEEE802.3.

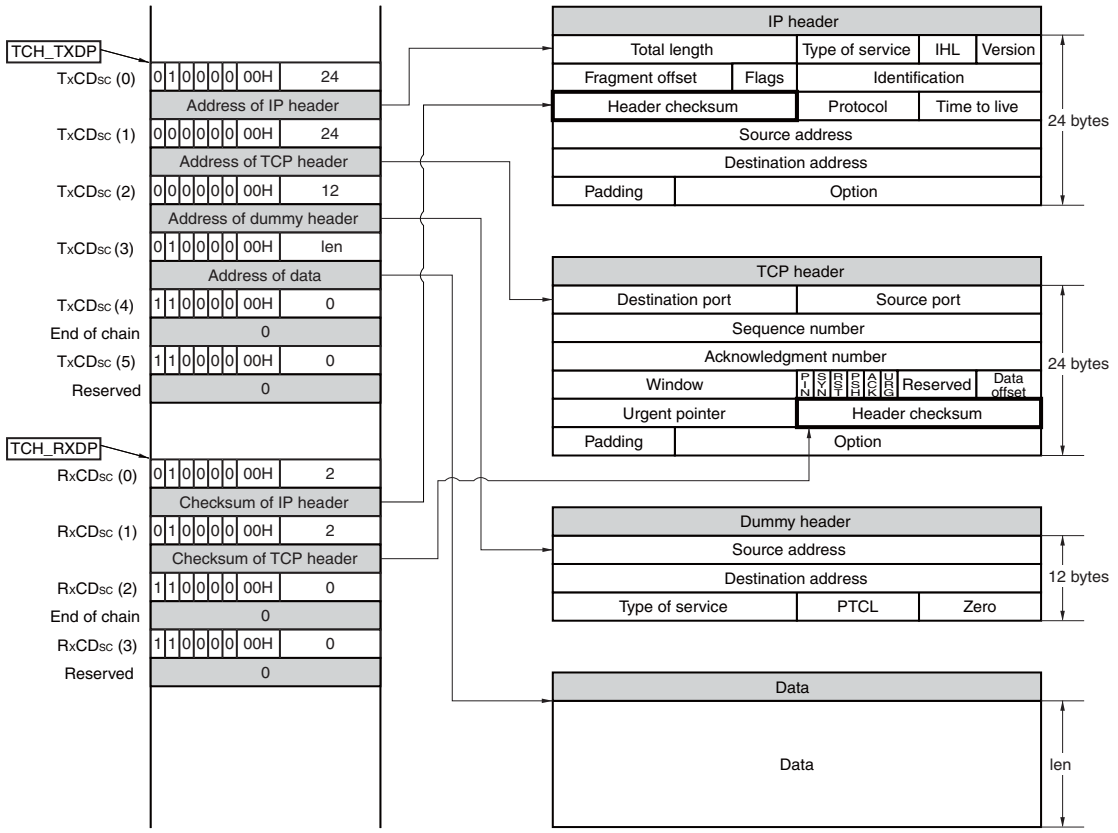
A checksum is calculated and the result is formatted in little endian format.

31.8.1 Configuration of transmit checksum descriptor

Calculation of a transmit checksum is controlled by a descriptor.

Depending on the configuration of the descriptor, the IP header and TCP header can be calculated separately or all at once.

The descriptor configuration for calculating the IP header and TCP header all at once is shown below.



The address to which the checksum of the IP header is to be written is obtained by adding 10 bytes to the IP header address, and the address to which the checksum of the TCP header is to be written is obtained by adding 16 bytes to the TCP header address.

Figure 31-30 Configuration of transmit checksum descriptor

31.9 Cautions

31.9.1 Cautions on FIFO

Note the following restrictions on the internal FIFO buffers of the Ethernet Controller.

Table 31-122 Restrictions on transmit FIFO

Maximum FIFO capacity	DMA transfer condition	Retry/abort	Condition to transmit data to PHY	Feature	Note
2,044 bytes or less	Data in transmit FIFO is less than 1,536 bytes[a].	Data is automatically retransmitted/ aborted when a collision is detected.	At least one packet is in the transmit FIFO.	Data can be retransmitted without underrun.	The transmit packet length must be 1,536 bytes or less.

a) Multiple packets may be stored in the transmit FIFO as long as its capacity is not exceeded. However, if the data in the transmit FIFO reaches 1,536 bytes, DMA transmission stops, preventing the transmit FIFO from overflowing. However, because the Ethernet Controller starts transmission after one packet of data has been stored in the transmit FIFO, the transmit FIFO is locked if the length of one packet is more than 1,536 bytes. Be sure to observe the rated size of one packet to use (1,518 bytes or less for non-VLAN frames or 1,522 bytes or less for VLAN frames).

Table 31-123 Restrictions on receive FIFO

Maximum FIFO capacity	DMA transfer condition	Condition to transmit pause control frame[a]	Feature	Note
2,036 bytes or less	At least one packet is in the receive FIFO.	• Transmission of pause control frame: Data in the receive FIFO is greater than the size specified by the ETHAnFLOWTHR[10:0] bits. • Transmission of 0 pause control frame: Data in the receive FIFO is less than the size specified by the ETHAnZPTHR[10:0] bits.	All error packets can be discarded.	The receive packet length must be 2,036 bytes or less.

a) Control by a pause control frame does not completely prevent the receive FIFO from overflowing. If the receive FIFO overflows, receive packets that are not accepted are discarded.

Chapter 32 Random Number Generator A (RNGA)

This chapter contains a generic description of the Random Number Generator A (RNGA).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

32.1 V850E2/Fx4-H RNGA Features

Instances

This microcontroller has the following number of instances of the Random Number Generator A .

Table 32-1 Instances of RNGA

Random Number Generator A	
Instance	1
Name	RNGA0

Instances index n

Throughout this chapter, the individual instances of a Random Number Generator A is identified by the index "n" (n = 0), for example, RNGAnRNGL for the RNGAn random number register.

Register addresses

All RNGAn register addresses are given as address offsets to the individual base address <RNGAn_base>.
The base address <RNGAn_base> of each RNGAn is listed in the following table:

Table 32-2 Register base addresses <RNGAn_base>

RNGAn instance	<RNGAn_base> address
RNGA0	FF82 $A000_H$

Clock supply All Random Number Generators A provide three clock inputs:

Table 32-3 RNGAn clock supply

RNGAn instance	RNGAn clock	Connected to
RNGA0	PCLK	Clock Controller CKSCLK_101
	OSCIN0	Clock Controller High Speed IntOsc f_{RH}
	OSCIN1	Clock Controller Low Speed IntOsc f_{RL}

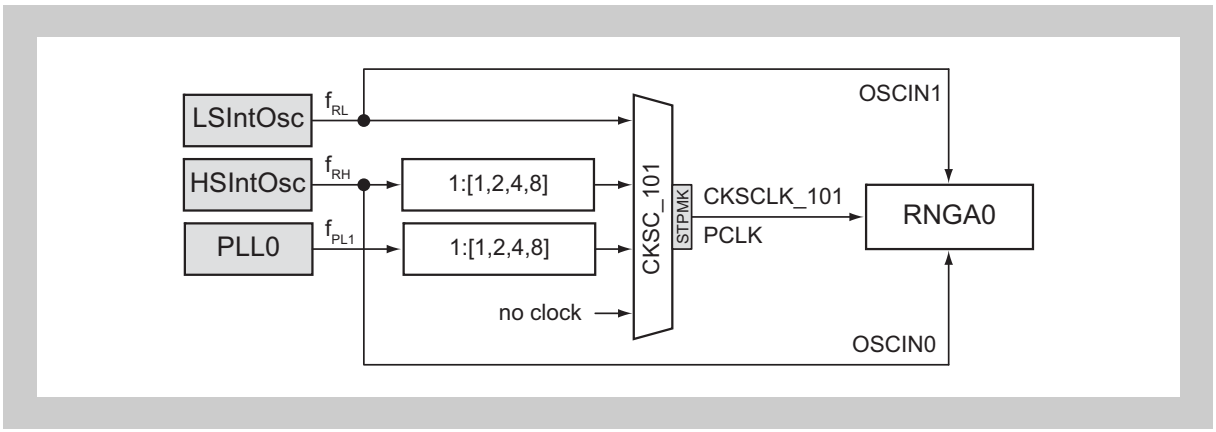


Figure 32-1 RNGA clock supply

RNGA H/W reset The Random Number Generators A and their registers are initialized by the following reset signal:

Table 32-4 RNGAn reset signal

RNGAn	Reset signal
RNGA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

32.2 Functional Overview

Features summary

The Random Number Generator has the following functions/features:

- Generation of a random number sequence passing the FIPS test
- Random number format: 16 bits
- Seed generated by hardware
  The seed is the start value of the random number sequence.

32.3 Functional Description

The Random Number Generator stores the 16-bit random numbers in RNGAnRN[15:00] of the RNGAnRNG register.

32.3.1 RNGA status

Generation of a random number in RNGAnRNG.RNGAnRN[15:00] takes up to 32 PCLK clock cycles.

The status bit RNGAnRNG.RNGAnRNA is set when a new random number is available.

If RNGAnRNG.RNGAnRNA = 0 the random number of the last read access is read again.

After reading the random number register RNGAnRNG, the status bit RNGAnRNGH.RNGAnRNA is automatically cleared.

32.3.2 RNGA start and reset

After power-up the Random Number Generator starts automatically to generate random numbers.

The first number RNGAnRNG.RNGAnRN[15:00] is available, when RNGAnRNG.RNGAnRNA = 1.

A reset of the Random Number Generator sets RNGAnRNG.RNGAnRNA = 0, but does not change the random number RNGAnRNG.RNGAnRN[15:00].

32.4 Registers

This section contains a description of all registers of the RNGA.

32.4.1 Registers overview

The Random Number Generator A is controlled and operated by the following registers:

Table 32-5 RNGA register overview

Register name	Shortcut	Address
Random number register	RNGAnRNG	<RNGAn_base>

<RNGAn_base> The base addresses <RNGAn_base> of the RNGAn are defined in the first section of this chapter under the key word “Register addresses”.

32.4.2 Registers details

(1) RNGAnRNG - Random number register

This register stores the random number and holds the indicator for the availability of a new random naumber.

Access The register can be read in 32-bit units.

Address <RNGAn_base>

Initial Value $00xx_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RNGAnRNA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RNGAnRN[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 32-6 RNGAnRNG register contents

Bit position	Bit name	Function
31	RNGAnRNA	Indicates the availability of a new random number: 0: no new random number available, the last random number is read again 1: new random number available RNGAnRNGH.RNGAnRNA is automatically cleared after this register is read.
15 to 0	RNGAn RN[15:00]	Stores the current random number.

Chapter 33 Key Return Function (KR)

This chapter contains a generic description of the Key Return Function.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

33.1 V850E2/Fx4-H KR Features

Instances

This microcontroller has following number of instances of the Key Return Function.

Table 33-1 Instances of KR

Key Return Function	
Instance	1
Name	KR0

Instances index n

Throughout this chapter, the individual instances of a Key Return Function is identified by the index "n" (n = 0), for example KRnM for the Key Return mode register.

Register addresses

All KRn register addresses are given as address offsets to the individual base address <KRn_base>.
The base address <KRn_base> of each KRn is listed in the following table:

Table 33-2 Register base addresses <KRn_base>

KRn instance	<KRn_base> address
KR0	FF82 $B000_H$

Clock supply All Key Return Functions provide one clock input:

Table 33-3 KRn clock supply

KRn instance	KRn clock	Connected to
KR0	PCLK	Clock Controller CKSCLK_A02

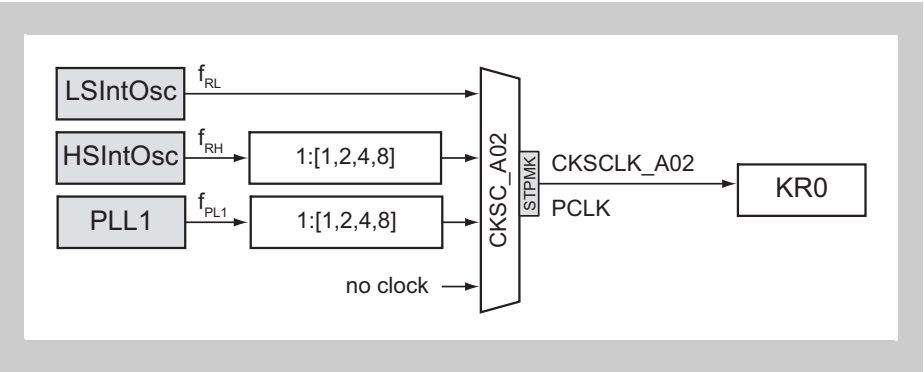


Figure 33-1 KR clock supply

Interrupts The Key Return Function can generate the following interrupt requests:

Table 33-4 KRn interrupt requests

KRn signals	Function	Connected to
KR0TIKR	Communication status interrupt	Interrupt Controller INTKR0[a]

a) This interrupt can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

KR H/W reset The Key Return Function and their registers are initialized by the following reset signal:

Table 33-5 KRn reset signal

KRn	Reset signal
KR0	• Reset Controller: SYSRES

I/O signals The I/O signals of the Key Return Function are listed in the following table.

Table 33-6 KRn I/O signals

KRn signal	Function	Connected to
KR0TPKR7 to KR0TPKR0	Key input signals	Ports KR0[a] to KR7[a]

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

33.2 Functional Overview

Features summary The Key Return Function has the following feature:

A key interrupt request signal (KRnTIKR) can be generated by inputting a low level to any of the eight key input pins (KRnTPKR0 to KRnTPKR7).

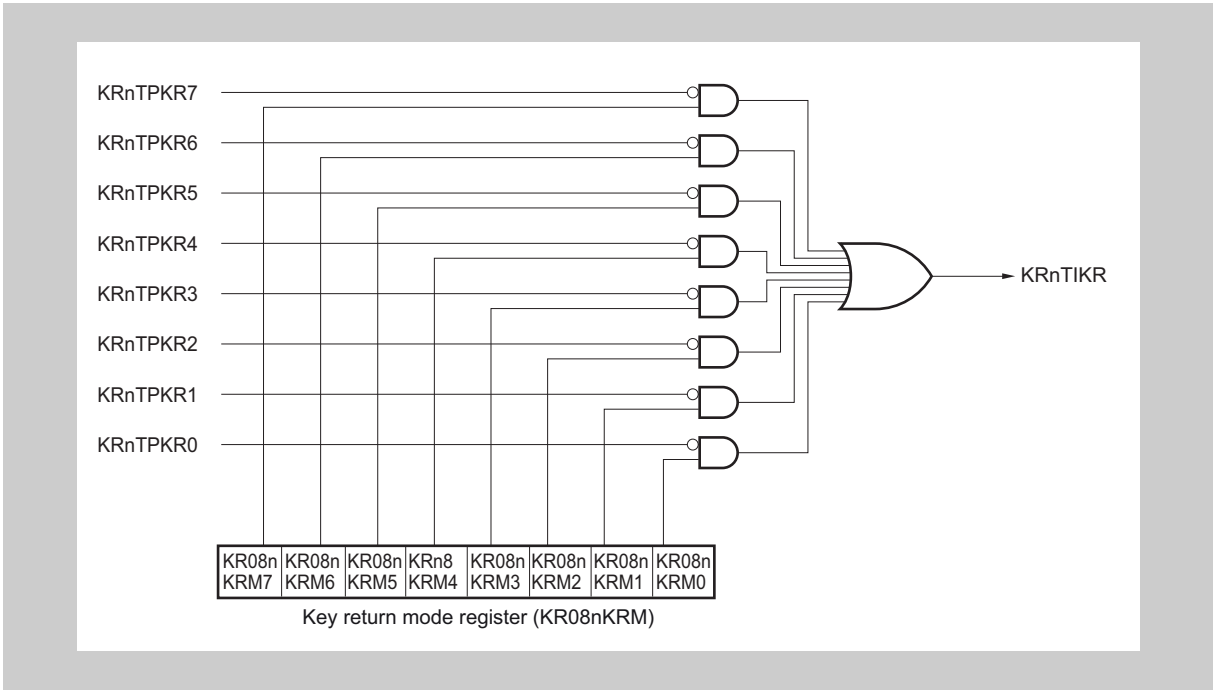


Figure 33-2 Block diagram of the Key Return Function

33.3 Functional Description

33.3.1 Interrupt request KRnTIKR

The interrupt request KRnTIKR is generated when a low level is input to key input pin KRnTPKR[7:0] while input to pin KRnTPKR[7:0] is enabled (KR08nM.KR08nKRM[7:0] = 1).

The following figure shows the interrupt request generation:

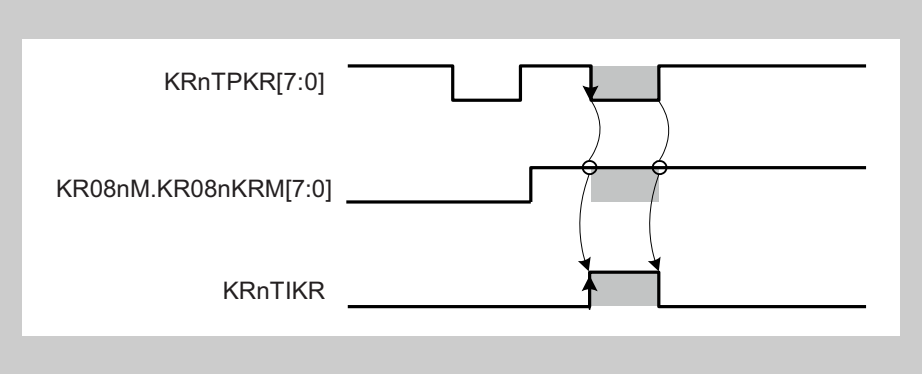


Figure 33-3 Interrupt request generation

Cautions

1. If a low level is input to any key input KRnTPKR[7:0], KRnTIKR is not generated again, even if another key input changes from high to low level.
2. An unintended key interrupt request KRnTIKR may occur under the following conditions:
   - KR08nM.KR08nKRM[7:0] = 1 and the key input KRnTPKR[7:0] level changes from high to low.
   - The key input KRnTPKR[7:0] level is low and KR08nM.KR08nKRM[7:0] is changed from 0 to 1.

   Thus mask (i.e. disable) KRnTIKR in the Interrupt Controller before changing KR08nM.KR08nKRM[7:0] from 0 to 1 or vice versa.

33.4 Registers

This section contains a description of all registers of the Key Return Function.

33.4.1 Key Return Function registers overview

The Key Return Function is controlled and operated by the following registers:

Table 33-7 Key Return Function registers overview

Register name	Shortcut	Address
Key return mode register	KR08nM	<KRn_base>

<KRn_base> <KRn_base> of the KRn are defined in the first section of this chapter under the key word “Register addresses”.

33.4.2 Key Return Function registers details

(1) KR08nM - Key return mode register

This register enables/disables the key input signal detection.

Access This register can be read/written in 8-bit units.

Address <KRn_base>

Initial Value 00_H. This register is cleared by any reset.

7	6	5	4	3	2	1	0
KR08n KRM7	KR08n KRM6	KR08n KRM5	KR08n KRM4	KR08n KRM3	KR08n KRM2	KR08n KRM1	KR08n KRM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 33-8 KR08nM register contents

Bit position	Bit name	Function
7 to 0	KR08n KRMm	Enables/disables the key input signal detection 0: Disabled 1: Enabled

Chapter 34 A/D Converter A (ADCA)

This chapter contains a generic description of the A/D Converter A.

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

34.1 V850E2/Fx4-H ADCA Features

Instances

This microcontroller has following number of instances of the A/D Converter A.

Table 34-1 Instances of ADCA

A/D Converter A	
Instance	2
Name	ADCA0, ADCA1

Instances index n

Throughout this chapter, the individual instances of the ADCA are identified by “n” (n = 0, 1), for example ADCAn, or ADCAnCTL0 for the control register 0 of ADCAn.

Channel group index i

The A/D Converter has 3 A/D conversion channel groups, abbreviated with CG. Throughout this chapter, the individual channel group is identified by the index "i" (i = 0 to 2), for example, ADCAnIOCi for the CGi interrupt controller register.

Channel index m

Each A/D Converter has several A/D conversion channels. Throughout this chapter, the individual channels of each ADCA are identified by the index “m”, for example, ADCAnCmCR for the channel m conversion result register.
The number of channels of the ADCA instances for each device are given in the following table.

Table 34-2 ADCAn channel select indices m

ADCAn instance	Channel select index	
	V850E2/FK4-H	V850E2/FL4-H
ADCA0	m = 0 to 23	m = 0 to 23
ADCA1	m = 0 to 15	m = 0 to 23

A/D Converter resolution

The V850E2/Fx4-H A/D Converter support 10-bit and 12-bit resolution.

Register addresses

All ADCAn register addresses are given as address offsets to the individual base address <ADCAn_base>.
The base address <ADCAn_base> of each ADCAn is listed in the following table:

Table 34-3 Register base addresses <ADCAn_base>

ADCAn instance	<ADCAn_base> address
ADCA0	FF81 $D000_H$
ADCA1	FF81 $E000_H$

Clock supply

All A/D Converters provide one clock input:

Table 34-4 ADCAn clock supply

ADCAn instance	ADCAn clock	Connected to
ADCA0	PCLK	Clock Controller CKSCLK_012
ADCA1	PCLK	Clock Controller CKSCLK_122

Caution

The ADCA internal system clock ADCATCLK, which is derived from the PCLK, must not exceed a certain limit.
Refer to the description of the control bits ADCAnCTL1.ADCAnFR[3:0], which determine the relation between PCLK and ADCATCLK.
For the maximum frequency of ADCATCLK refer to the Electrical Target Specification for details.

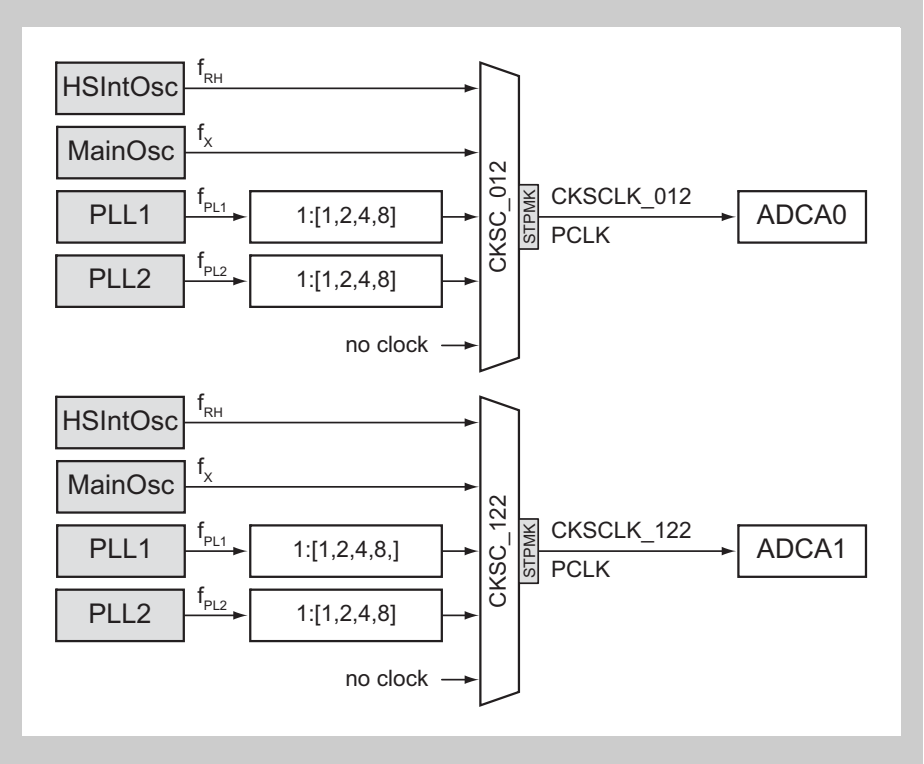


Figure 34-1 ADCA clock supply

Interrupts and DMA/DTS

The A/D Converters can generate the following interrupt and DMA/DTS requests:

Table 34-5 ADCAn interrupt and DMA/DTS requests

ADCAn signals	Function	Connected to
ADCA0:		
INTADCA0T0	End of conversion CG0	Interrupt Controller INTADCA0I0[a] DMA Controller trigger 38 DTS Controller trigger 7
INTADCA0T1	End of conversion CG1	Interrupt Controller INTADCA0I1[a] DMA Controller trigger 39 DTS Controller trigger 8
INTADCA0T2	End of conversion CG2	Interrupt Controller INTADCA0I2[a] DMA Controller trigger 40 DTS Controller trigger 56
INTADCA0LLT	Last conversion	Interrupt Controller INTADCA0LLT[a] DMA Controller trigger 41 DTS Controller trigger 57
INTADCA0ERR	Error interrupt	Interrupt Controller INTADCA0ERR[a]
ADCA1:		
INTADCA1T0	End of conversion CG0	Interrupt Controller INTADCA1I0 DMA Controller trigger 67 DTS Controller trigger 9
INTADCA1T1	End of conversion CG1	Interrupt Controller INTADCA1I1 DMA Controller trigger 68 DTS Controller trigger 10
INTADCA1T2	End of conversion CG2	Interrupt Controller INTADCA1I2 DMA Controller trigger 69 DTS Controller trigger 11
INTADCA1LLT	Last conversion	Interrupt Controller INTADCA1LLT
INTADCA1ERR	Error interrupt	Interrupt Controller INTADCA1ERR

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

ADCA H/W reset

The A/D Converters and their registers are initialized by the following reset signal:

Table 34-6 ADCAn reset signal

ADCAn	Reset signal
ADCA0	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_0 (Isolated-Area-0 wake-up from DEEPSTOP mode)
ADCA1	• Reset Controller: SYSRES • Stand-by Controller: DPSTPWU_1 (Isolated-Area-1 wake-up from DEEPSTOP mode)

I/O signals The I/O signals of the A/D Converters are listed in the following table.

Table 34-7 ADCAn I/O signals

ADCAn signal	Function	Connected to
ADCA0:		
ADCA0Im	Analog inputs	Port ADCA0Im
ADCA0TTRGi	H/W trigger for CGi	H/W trigger expansion i, refer to the section *"H/W Trigger Expansion"* below.
AVREFP	Positive analogue reference voltage	Port AVREF0P
AVREFM	Negative analogue reference voltage	Port AVREF0M
ADCA1:		
ADCA1Im	Analog inputs	Port ADCA1Im
ADCA1TTRGi	H/W trigger for CGi	H/W trigger expansion i, refer to the section *"H/W Trigger Expansion"* below.
AVREFP	Positive analogue reference voltage	Port AVREF1P
AVREFM	Negative analogue reference voltage	Port AVREF1M

ADCA ports If the ports used as ADCAn inputs shall be used as port inputs, set

- ADCA0CTL1.ADCA0GPS = 1 to use port groups P10 or P11 in port mode
- ADCA1CTL1.ADCA1GPS = 1 to use port groups P12 or P13 in port mode

Simultaneous sampling Some channels are equipped with separate Sample & Hold (S&H) circuits, which allow exact simultaneous sampling of these channels.
Simultaneous sampling of multiple A/D Converter input signals is available for the following input channels:

Table 34-8 Simultaneous sampling channels

ADCAn	Simulataneous sampling channels
ADCA0	channels m = 0 to 5
ADCA1	–

34.2 H/W Trigger Expansion

All A/D Converter H/W trigger inputs ADCAnTTRGi are equipped with a H/W trigger expansion module that enables up to 16 signals ADCAnTTINi[15:00] to trigger an A/D conversion process.

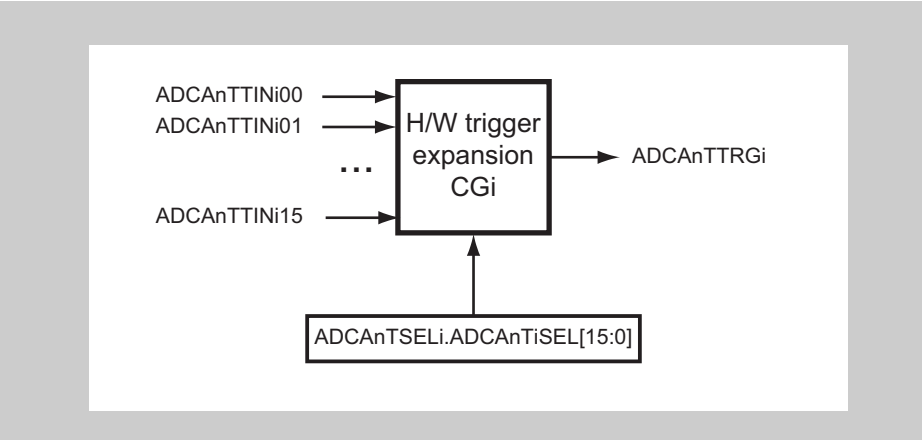


Figure 34-2 H/W trigger expansion module

34.2.1 ADCAn H/W trigger selection

The selection of active H/W triggers is made via the ADCAn register ADCAnTSELi.ADCAnTiSEL[15:0] bits.

Caution It is only allowed to select a single H/W trigger for each ADCAn H/W trigger input ADCAnTTRGi.
Thus do not set more than one bit of each ADCAnTSELi register to “1” at the same time.

34.2.2 ADCAn H/W trigger edge selection

The A/D conversion process of channel group i is started with the rising edge of the H/W triggers ADCAnTTRGi.
The rising edge has to be selection via the ADCAnCTL1 register:

- ADCAnCTL1.ADCAnTiETS0 = 0: no edge is selected, thus ADCAnTTRGi does not trigger an A/D conversion
- ADCAnCTL1.ADCAnTiETS0 = 1: rising edge is selected, thus rising edge of ADCAnTTRGi triggers an A/D conversion

(1) External A/D converter H/W triggers

ADCAnTRG[2:0] The rising edge of the external ADCAn H/W trigger signals ADCAnTRG[2:0] can start and A/D conversion, provided ADCAnCTL1.ADCAnTiETS0 = 1.

INTPx In case an external interrupt signal INTPx is used to start an A/D conversion, the edge of INTPx to trigger A/D conversion can be selected by the port filters:

Table 34-9 ADCAn H/W trigger edge selection for external interrupts INTPx

Control setup	Rising edge	Falling edge	Both edges
ADCAnCTL1.ADCAnTiETS0 =	1		
FCLAnCTLm.FCLAnINTLm =	0		
FCLAnCTLm.FCLAnINTFm =	0	1	1
FCLAnCTLm.FCLAnINTRm =	1	0	1

Note The index “x” stands for the number of the external interrupt signal.
The index “m” denotes the number of the port control register, assigned to INTPx.

For further details about the port filters refer to the *“Port filters”* section in the chapter “Port Functions”.

34.2.3 ADCA0 H/W trigger tables

The available ADCA0 H/W triggers comprises

- external H/W triggers
- external interrupts
- interrupt signals of Timer Array Units A
- interrupt signals of Timer Array Units B
- interrupt signals of Timer Array Units C
- interrupt signals of Timer Array Units J
- interrupt signals of Encoder Timer
- Timer Motor Control Function A/D trigger signals
- PWM Diagnosis A/D trigger signals

Table 34-10 ADCA0 H/W trigger signals (1/2)

ADCA0 channel group	Trigger input signals			ADCA0 trigger signal
	Name	Control bit	Connected to	
CG0	ADCA0TTIN000	ADCA0TSEL0.ADCA0T0SEL[00]	Port ADCA0TRG0[a]	ADCA0TTRG0
	ADCA0TTIN001	ADCA0TSEL0.ADCA0T0SEL[01]	Port INTP2	
	ADCA0TTIN002	ADCA0TSEL0.ADCA0T0SEL[02]	Port INTP5	
	ADCA0TTIN003	ADCA0TSEL0.ADCA0T0SEL[03]	TAUA0 INTTAUA0Im[b]	
	ADCA0TTIN004	ADCA0TSEL0.ADCA0T0SEL[04]	TAUB1 INTTAUB1Im[c]	
	ADCA0TTIN005	ADCA0TSEL0.ADCA0T0SEL[05]	TAUB2 INTTAUB2I15	
	ADCA0TTIN006	ADCA0TSEL0.ADCA0T0SEL[06]	TAUC3 INTTAUC3I15	
	ADCA0TTIN007	ADCA0TSEL0.ADCA0T0SEL[07]	TAUC4 INTTAUC4I15	
	ADCA0TTIN008	ADCA0TSEL0.ADCA0T0SEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA0TTIN009	ADCA0TSEL0.ADCA0T0SEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA0TTIN010	ADCA0TSEL0.ADCA0T0SEL[10]	ENCA0 INTENCA0I1	
	ADCA0TTIN011	ADCA0TSEL0.ADCA0T0SEL[11]	PMCA0 PMCAADTR0	
	ADCA0TTIN012	ADCA0TSEL0.ADCA0T0SEL[12]	PMCA0 PMCAADTR1	
	ADCA0TTIN013	ADCA0TSEL0.ADCA0T0SEL[13]	PMCA0 PMCAADTR2	
	ADCA0TTIN014	ADCA0TSEL0.ADCA0T0SEL[14]	not connected	
	ADCA0TTIN015	ADCA0TSEL0.ADCA0T0SEL[15]	not connected	

Table 34-10 ADCA0 H/W trigger signals (2/2)

ADCA0 channel group	Trigger input signals			ADCA0 trigger signal
	Name	Control bit	Connected to	
CG1	ADCA0TTIN100	ADCA0TSEL1.ADCA0T1SEL[00]	Port ADCA0TRG1[a]	ADCA0TTRG1
	ADCA0TTIN101	ADCA0TSEL1.ADCA0T1SEL[01]	Port INTP1	
	ADCA0TTIN102	ADCA0TSEL1.ADCA0T1SEL[02]	Port INTP4	
	ADCA0TTIN103	ADCA0TSEL1.ADCA0T1SEL[03]	TAUA0 INTTAUA0Im[b]	
	ADCA0TTIN104	ADCA0TSEL1.ADCA0T1SEL[04]	TAUB1 INTTAUB1Im[c]	
	ADCA0TTIN105	ADCA0TSEL1.ADCA0T1SEL[05]	TAUB2 INTTAUB2I15	
	ADCA0TTIN106	ADCA0TSEL1.ADCA0T1SEL[06]	TAUC3 INTTAUC3I15	
	ADCA0TTIN107	ADCA0TSEL1.ADCA0T1SEL[07]	TAUC4 INTTAUC4I15	
	ADCA0TTIN108	ADCA0TSEL1.ADCA0T1SEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA0TTIN109	ADCA0TSEL1.ADCA0T1SEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA0TTIN110	ADCA0TSEL1.ADCA0T1SEL[10]	ENCA0 INTENCA0I1	
	ADCA0TTIN111	ADCA0TSEL1.ADCA0T1SEL[11]	PMCA0 PMCAADTR0	
	ADCA0TTIN112	ADCA0TSEL1.ADCA0T1SEL[12]	PMCA0 PMCAADTR1	
	ADCA0TTIN113	ADCA0TSEL1.ADCA0T1SEL[13]	PMCA0 PMCAADTR2	
	ADCA0TTIN114	ADCA0TSEL1.ADCA0T1SEL[14]	TAPA0 TAPATADOUT0	
	ADCA0TTIN115	ADCA0TSEL1.ADCA0T1SEL[15]	TAPA0 TAPATADOUT1	
CG2	ADCA0TTIN200	ADCA0TSEL2.ADCA0T2SEL[00]	Port ADCA0TRG2[a]	ADCA0TTRG2
	ADCA0TTIN201	ADCA0TSEL2.ADCA0T2SEL[01]	Port INTP0	
	ADCA0TTIN202	ADCA0TSEL2.ADCA0T2SEL[02]	Port INTP3	
	ADCA0TTIN203	ADCA0TSEL2.ADCA0T2SEL[03]	TAUA0 INTTAUA0Im[b]	
	ADCA0TTIN204	ADCA0TSEL2.ADCA0T2SEL[04]	TAUB1 INTTAUB1Im[c]	
	ADCA0TTIN205	ADCA0TSEL2.ADCA0T2SEL[05]	TAUB2 INTTAUB2I15	
	ADCA0TTIN206	ADCA0TSEL2.ADCA0T2SEL[06]	TAUC3 INTTAUC3I15	
	ADCA0TTIN207	ADCA0TSEL2.ADCA0T2SEL[07]	TAUC4 INTTAUC4I15	
	ADCA0TTIN208	ADCA0TSEL2.ADCA0T2SEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA0TTIN209	ADCA0TSEL2.ADCA0T2SEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA0TTIN210	ADCA0TSEL2.ADCA0T2SEL[10]	ENCA0 INTENCA0I1	
	ADCA0TTIN211	ADCA0TSEL2.ADCA0T2SEL[11]	TAPA0 TAPATADOUT0	
	ADCA0TTIN212	ADCA0TSEL2.ADCA0T2SEL[12]	TAPA0 TAPATADOUT1	
	ADCA0TTIN213	ADCA0TSEL2.ADCA0T2SEL[13]	not connected	
	ADCA0TTIN214	ADCA0TSEL2.ADCA0T2SEL[14]	not connected	
	ADCA0TTIN215	ADCA0TSEL2.ADCA0T2SEL[15]	PMCA0 PMCAADTR0	

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

b) The TAUA0 interrupt is selected via the *"PIC0ADTEN40i - TAUA0 interrupt selection register for ADCA0TTRGi"*. Refer to its description below.

c) The TAUA1 interrupt is selected via the *"PIC0ADTEN41i - TAUB1 interrupt selection register for ADCA0TTRGi"*. Refer to its description below.

Caution The initial state of the noise filters of the external ADCA0 trigger signals ADCA0TRG[2:0] block the input signals.
Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

(1) PIC0ADTEN40i - TAUA0 interrupt selection register for ADCA0TTRGi

This register selects a TAUA0 interrupt INTTAUA0Im as a ADCA0 H/W trigger input signal ADCA0TTINi03 for ADCA0 channel group CGi.

Access This register can be read/written in 16-bit units.

Address PIC0ADTEN400 for CG0 ADCA0TTIN003: FF81 C0C4$_H$
PIC0ADTEN401 for CG1 ADCA0TTIN103: FF81 C0C8$_H$
PIC0ADTEN402 for CG2 ADCA0TTIN203: FF81 C0CC$_H$

Initial Value 0000$_H$

15	14	13	12	11	10	9	8
PIC0 ADTEN 40i15	PIC0 ADTEN 40i14	PIC0 ADTEN 40i13	PIC0 ADTEN 40i12	PIC0 ADTEN 40i11	PIC0 ADTEN 40i10	PIC0 ADTEN 40i9	PIC0 ADTEN 40i8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PIC0 ADTEN 40i7	PIC0 ADTEN 40i6	PIC0 ADTEN 40i5	PIC0 ADTEN 40i4	PIC0 ADTEN 40i3	PIC0 ADTEN 40i2	PIC0 ADTEN 40i1	PIC0 ADTEN 40i0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-11 PIC0ADTEN40i register contents

Bit position	Bit name	Function
m	PIC0 ADTEN 40im	Selection of INTTAUA0Im (m = 0 to 15) as ADCA0 CGi H/W trigger ADCA0TTINi03: 0: INTTAUA0Im disabled as ADCA0TTINi03 H/W trigger 1: INTTAUA0Im enabled as ADCA0TTINi03H/W trigger

Cautions
1. It is only allowed to select a single TAUA0 interrupt INTTAUA0Im as a H/W trigger for ADCA0TTINi03.
Thus do not set more than one bit of each PIC0ADTEN40i register to “1” at the same time.
2. Only change register settings while the A/D converter is stopped (ADCA0CTL0.ADCA0CE = 0).

(2) PIC0ADTEN41i - TAUB1 interrupt selection register for ADCA0TTRGi

This register selects a TAUB1 interrupt INTTAUB1Im as a ADCA0 H/W trigger input signal ADCA0TTINi04 for ADCA0 channel group CGi.

Access This register can be read/written in 16-bit units.

Address PIC0ADTEN410 for CG0 ADCA0TTIN003: FF81 C0D0$_H$
PIC0ADTEN411 for CG1 ADCA0TTIN103: FF81 C0D4$_H$
PIC0ADTEN412 for CG2 ADCA0TTIN203: FF81 C0D8$_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8
PIC0 ADTEN 41i15	PIC0 ADTEN 41i14	PIC0 ADTEN 41i13	PIC0 ADTEN 41i12	PIC0 ADTEN 41i11	PIC0 ADTEN 41i10	PIC0 ADTEN 41i9	PIC0 ADTEN 41i8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

7	6	5	4	3	2	1	0
PIC0 ADTEN 41i7	PIC0 ADTEN 41i6	PIC0 ADTEN 41i5	PIC0 ADTEN 41i4	PIC0 ADTEN 41i3	PIC0 ADTEN 41i2	PIC0 ADTEN 41i1	PIC0 ADTEN 41i0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-12 PIC0ADTEN41i register contents

Bit position	Bit name	Function
m	PIC0 ADTEN 41im	Selection of INTTAUB1Im (m = 0 to 15) as ADCA0 CGi H/W trigger ADCA0TTINi04: 0: INTTAUB1Im disabled as CGi ADCA0TTINi04 H/W trigger 1: INTTAUB1Im enabled as CGi ADCA0TTINi04 H/W trigger

Cautions

1. It is only allowed to select a single TAUB1 interrupt INTTAUB1Im as a H/W trigger for ADCA0TTINi04.
   Thus do not set more than one bit of each PIC0ADTEN41i register to “1” at the same time.
2. Only change register settings while the A/D converter is stopped (ADCA0CTL0.ADCA0CE = 0).

34.2.4 ADCA1 H/W trigger selections

The available ADCA0 H/W triggers comprises

- external H/W triggers
- external interrupts
- interrupt signals of Timer Array Units A
- interrupt signals of Timer Array Units B
- interrupt signals of Timer Array Units C
- interrupt signals of Timer Array Units J
- Timer Motor Control Function A/D trigger signals
- PWM Diagnosis A/D trigger signals

Table 34-13 ADCA1 H/W trigger signals (1/2)

ADCA1 channel group	Trigger input signals			ADCA1 trigger signal
	Name	Control bit	Connected to	
CG0	ADCA1TTIN000	ADCA1TSEL0.ADCA1TISEL[00]	Port ADCA1TRG0[a]	ADCA1TTRG0
	ADCA1TTIN001	ADCA1TSEL0.ADCA1TISEL[01]	Port INTP2	
	ADCA1TTIN002	ADCA1TSEL0.ADCA1TISEL[02]	Port INTP5	
	ADCA1TTIN003	ADCA1TSEL0.ADCA1TISEL[03]	TAUA0 INTTAUA0I15	
	ADCA1TTIN004	ADCA1TSEL0.ADCA1TISEL[04]	TAUB1 INTTAUB1I15	
	ADCA1TTIN005	ADCA1TSEL0.ADCA1TISEL[05]	TAUB2 INTTAUB2I15	
	ADCA1TTIN006	ADCA1TSEL0.ADCA1TISEL[06]	TAUC3 INTTAUC3I15	
	ADCA1TTIN007	ADCA1TSEL0.ADCA1TISEL[07]	TAUC4 INTTAUC4I15	
	ADCA1TTIN008	ADCA1TSEL0.ADCA1TISEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA1TTIN009	ADCA1TSEL0.ADCA1TISEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA1TTIN010	ADCA1TSEL0.ADCA1TISEL[10]	not connected	
	ADCA1TTIN011	ADCA1TSEL0.ADCA1TISEL[11]	PMCA0 PMCAADTR3	
	ADCA1TTIN012	ADCA1TSEL0.ADCA1TISEL[12]	PMCA0 PMCAADTR4	
	ADCA1TTIN013	ADCA1TSEL0.ADCA1TISEL[13]	PMCA0 PMCAADTR5	
	ADCA1TTIN014	ADCA1TSEL0.ADCA1TISEL[14]	not connected	
	ADCA1TTIN015	ADCA1TSEL0.ADCA1TISEL[15]	not connected	

Table 34-13 ADCA1 H/W trigger signals (2/2)

ADCA1 channel group	Trigger input signals			ADCA1 trigger signal
	Name	Control bit	Connected to	
CG1	ADCA1TTIN100	ADCA1TSEL1.ADCA1TISEL[00]	Port ADCA1TRG1[a]	ADCA1TTRG1
	ADCA1TTIN101	ADCA1TSEL1.ADCA1TISEL[01]	Port INTP1	
	ADCA1TTIN102	ADCA1TSEL1.ADCA1TISEL[02]	Port INTP4	
	ADCA1TTIN103	ADCA1TSEL1.ADCA1TISEL[03]	TAUA0 INTTAUA0I15	
	ADCA1TTIN104	ADCA1TSEL1.ADCA1TISEL[04]	TAUB1 INTTAUB1I15	
	ADCA1TTIN105	ADCA1TSEL1.ADCA1TISEL[05]	TAUB2 INTTAUB2I15	
	ADCA1TTIN106	ADCA1TSEL1.ADCA1TISEL[06]	TAUC3 INTTAUC3I15	
	ADCA1TTIN107	ADCA1TSEL1.ADCA1TISEL[07]	TAUC4 INTTAUC4I15	
	ADCA1TTIN108	ADCA1TSEL1.ADCA1TISEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA1TTIN109	ADCA1TSEL1.ADCA1TISEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA1TTIN110	ADCA1TSEL1.ADCA1TISEL[10]	not connected	
	ADCA1TTIN111	ADCA1TSEL1.ADCA1TISEL[11]	PMCA0 PMCAADTR3	
	ADCA1TTIN112	ADCA1TSEL1.ADCA1TISEL[12]	PMCA0 PMCAADTR4	
	ADCA1TTIN113	ADCA1TSEL1.ADCA1TISEL[13]	PMCA0 PMCAADTR5	
	ADCA1TTIN114	ADCA1TSEL1.ADCA1TISEL[14]	TAPA0 TAPATADOUT0	
	ADCA1TTIN115	ADCA1TSEL1.ADCA1TISEL[15]	TAPA0 TAPATADOUT1	
CG2	ADCA1TTIN200	ADCA1TSEL2.ADCA1TISEL[00]	Port ADCA1TRG2[a]	ADCA1TTRG2
	ADCA1TTIN201	ADCA1TSEL2.ADCA1TISEL[01]	Port INTP0	
	ADCA1TTIN202	ADCA1TSEL2.ADCA1TISEL[02]	Port INTP3	
	ADCA1TTIN203	ADCA1TSEL2.ADCA1TISEL[03]	TAUA0 INTTAUA0I15	
	ADCA1TTIN204	ADCA1TSEL2.ADCA1TISEL[04]	TAUB1 INTTAUB1I15	
	ADCA1TTIN205	ADCA1TSEL2.ADCA1TISEL[05]	TAUB2 INTTAUB2I15	
	ADCA1TTIN206	ADCA1TSEL2.ADCA1TISEL[06]	TAUC3 INTTAUC3I15	
	ADCA1TTIN207	ADCA1TSEL2.ADCA1TISEL[07]	TAUC4 INTTAUC4I15	
	ADCA1TTIN208	ADCA1TSEL2.ADCA1TISEL[08]	TAUJ0 INTTAUJ0I3	
	ADCA1TTIN209	ADCA1TSEL2.ADCA1TISEL[09]	TAUJ1 INTTAUJ1I3	
	ADCA1TTIN210	ADCA1TSEL2.ADCA1TISEL[10]	not connected	
	ADCA1TTIN211	ADCA1TSEL2.ADCA1TISEL[11]	TAPA0 TAPATADOUT0	
	ADCA1TTIN212	ADCA1TSEL2.ADCA1TISEL[12]	TAPA0 TAPATADOUT1	
	ADCA1TTIN213	ADCA1TSEL2.ADCA1TISEL[13]	not connected	
	ADCA1TTIN214	ADCA1TSEL2.ADCA1TISEL[14]	not connected	
	ADCA1TTIN215	ADCA1TSEL2.ADCA1TISEL[15]	PMCA0 PMCAADTR3	

a) These input signals are passed through a noise filter, refer to the section *"Port Filters"* in the chapter *"Port Functions"*.

Caution The initial state of the noise filters of the external ADCA1 trigger signals ADCA1TRG[2:0] block the input signals.
Thus the filters must be configured (bypassed or activated) in order to let the - filtered or unfiltered - input signals pass.

34.3 Functional Overview

The A/D Converter A (ADCA) converts analog input signals into digital values.

Features summary

The ADCA has the following features:

- support of 10-bit and 12-bit resolution
  The resolution supported by this microcontroller is specified in the first section of this chapter under the key word *"A/D Converter resolution"*.
- A/D conversion based on successive approximation method
- A/D conversion of up to 24 analog input signals
  The number of channels supported by this microcontroller is specified in the first section of this chapter under the key word *"Channel index m"*.
- A/D conversion of up to 3 differently-prioritized groups of channels
- one-shot and continuous A/D conversion modes (continuous conversion mode for channel group CG0 only)
- autorepeat function (repeats the one-shot conversion mode 1 to 4 times)
- exact simultaneous sampling of multiple channels
  Refer to the key word *"Simultaneous sampling"* in the first section of this chapter for information about the channels which can be used for simultaneous sampling.
- software and hardware start trigger modes
  The hardware trigger source can be selected from multiple input signals.
- configurable terms for generation of end interrupts
- three types of result check functions
- discharge function for discharging the capacitor prior to executing conversion with a new sampling value
- on-off switch function for buffer amplifier
- self-diagnosis functions to check the ADCA components

The following figure shows the main components of the ADCA.

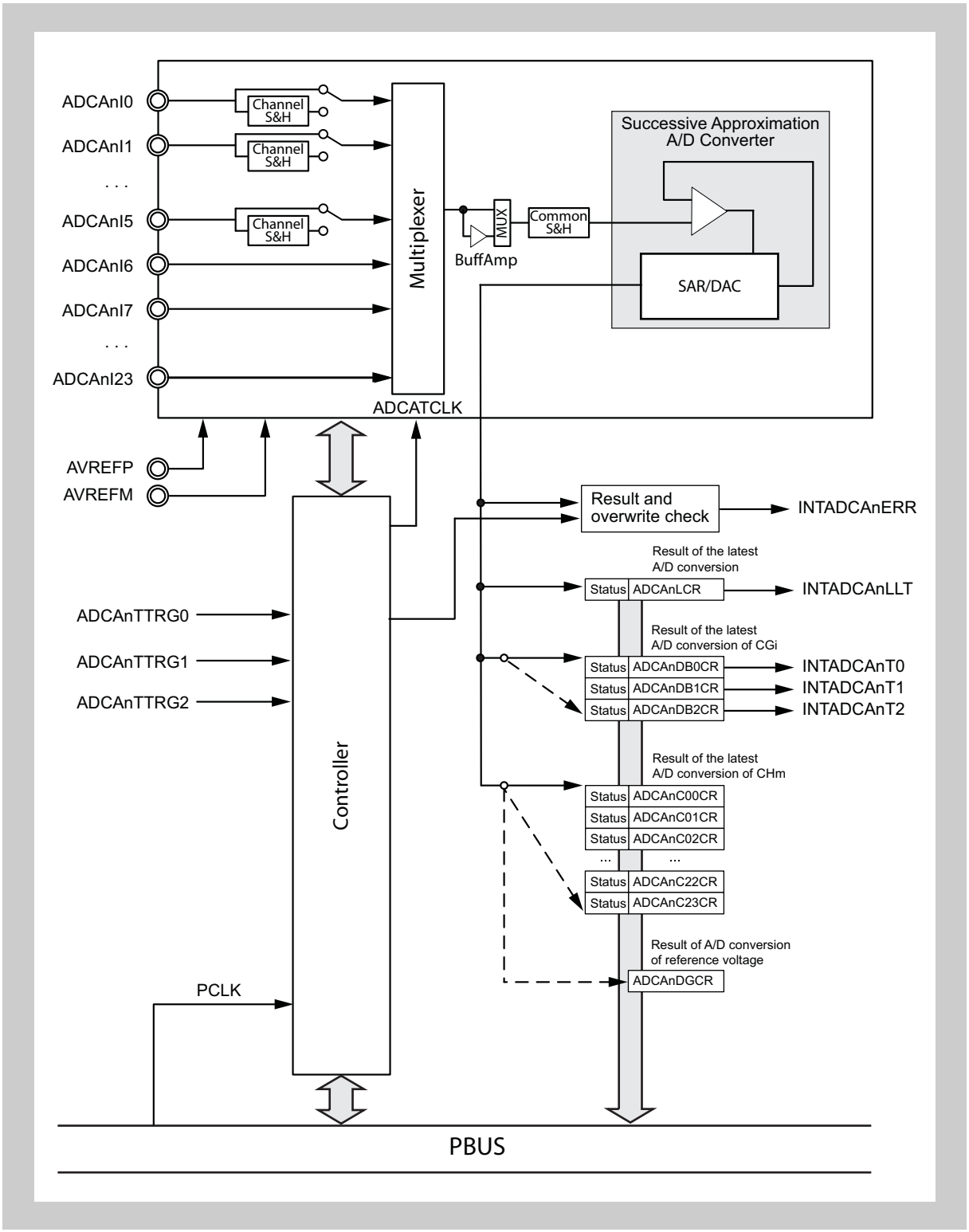


Figure 34-3 Block diagram of the ADCA

34.4 Cautions

Caution Make sure that the voltages input to ADCAnIm do not exceed the rated values. If a voltage higher than AVDD or lower than AVSS is input to a channel, the conversion value sticks to a maximum respectivelly, which are defined in the Electrical Target Specification. The electrical characteristics of the other channels may also be affected.

For further information concerning the electrical characteristics refer to the Electrical Target Specification.

34.5 Functional Description

A/D Converter resolution

The A/D Converter A (ADCA) converts up to 24 analog input signals into digital values and supports 10-bit and 12-bit resolution.
The resolution supported by this microcontroller is specified in the first section of this chapter under the key word *"A/D Converter resolution"*.

Note The 10-bit/12-bit resolution setting is applied in common to all the channels.

Channels and channel groups

Each of the input channels can be assigned to one of three channel groups CGi (i = 0 to 2). The list of the input channels assigned to each CG is called a scan list (including diagnostic A/D conversion of CG0). The scan lists can be set up easily with one register. The scan lists can also be changed during operation. All the A/D conversions for a scan list are called scan list conversion.

The ADCA supports up to three differently-prioritized channel groups and 2 conversion modes:

- Continuous conversion mode:
  repeats conversion of all input channels of the channel group 0 (CG0) scan list.
- One-shot conversion:
  executes conversion of all input channels of the channel group i (CGi) scan list conversion only once. In one-shot conversion mode, scan list conversion is repeated one to four times. The number of repetitions is selectable.

Simultaneous sampling

Channel-related Sample & Hold (S&H) circuits enable 6 analog input signals ACDCAnIm to be sampled simultaneously.
Refer to the key word "Simultaneous sampling" in the first section of this chapter for information about the channels which can be used for simultaneous sampling.

A/D conversion

The A/D conversion can be started using either software or hardware as the start trigger.

A multiplexer selects the channel to be converted and the Sample & Hold circuit holds the voltage input.

The A/D Converter uses the successive approximation method.
The successive approximation register (SAR) stores the D/A converter output voltage to be compared with the analog input voltage as a 10-bit or 12-bit digital value.

After each successful conversion the INTADCAnLLT signal is generated.

A/D conversion result registers

When the A/D conversion is complete, the contents of the SAR register is stored in three registers simultaneously, allowing the conversion result

- of all channels
- of the last converted channel
- of the last of a certain channel group

to be read.

Depending on the configuration, the ADCA generates a conversion end interrupt after the A/D conversion of certain channels or at the end of the A/D conversion of all channels of a channel group.

Optional result check

The results of A/D conversions can be checked with the following functions:

- conversion result overwrite check function
- conversion result upper/lower limit compare function

Optional discharge function

If required, the internal capacitor of the Sample & Hold circuit can be discharged prior to every conversion.

On-off switch function for buffer amplifier

In order to reduce the load of the external analogue signal source, the signal can be connected to an internal buffer amplifier.
The buffer amplifier precharges the internal sampling capacitor before the A/D sampling period.

Self-diagnosis functions

Following self-diagnostic functions are provided to check that the ADCA works correctly and to detect analog input pins that are disconnected.

- A/D conversion circuit diagnosis
- channel multiplexer diagnosis
- open pin diagnosis
- channel S&H diagnosis

Configurable stabilization time

By setting an arbitrary value to the stabilization counter, the optimum stabilization time can be secured after the power is switched on.

34.5.1 Basic Operation

This section describes the basic procedure for an analog to digital conversion. More detailed descriptions are given in the following sections.

1. To optimize the startup time after power on and after standby mode is released, adjust the stabilization time by specifying the stabilization counter ADCAnCNT register.
2. Switch the power of the ADCA on by setting ADCAnCTL1.ADCAnGPSE = 1.
3. Before you enable the A/D converter (ADCA must be disabled by ADCAnCTL0.ADCAnCE = 0), configure the resolution, ADCA clock, trigger mode, conversion mode, interrupt generation, channel groups, etc. in the following registers:
   – ADCAnCTL1
   – ADCAnCGi
   – ADCAnIOCi
   – ADCAnSHCTL
   – ADCAnTSELi
4. If you want to check that the A/D conversion results are within a certain value range, enable the conversion result upper/lower limit compare function for the desired channels (ADCAnCTL2.ADCAnRCKm) and specify the lower and upper limits for the ADCAnLL and ADCAnUL registers.
5. If you want the capacitor of the Sample & Hold circuit to be discharged before sampling a new value, enable the discharge function by writing ADCAnCTL1.ADCAnDISC = 1.
6. If required, enable the buffer amplifier by ADCAnCTL1.ADCAnBPC = 1.
7. Enable the A/D converter by setting ADCAnCTL0.ADCAnCE = 1.
   The A/D converter is ready for A/D conversion after the stabilization time has elapsed after power on or stand-by mode release.
8. Depending on the configured trigger mode, A/D conversion is started by a channel group related start trigger:
   – by a software trigger (ADCAnTRGi.ADCAnSTTi = 1), or
   – by a hardware trigger (input signal ADCAnTTRGi)

   If the A/D conversion starts for multiple CGs, the order of A/D conversion depends on the priority of the CGs.
9. The A/D conversion end interrupt INTADCAnTi is generated when the conversion of the channel set in the ADCAnIOCi register ends.
10. Read the results from the A/D conversion result registers ADCAnLCR, ADCAnDBiCR or ADCAnCmCR.
11. Monitor the following registers:
    – ADCAnSTR1:
      To check whether the A/D conversion results have been overwritten before being read.
    – ADCAnSTR0:
      To check whether the A/D conversion results are within the configured range (only if the conversion result upper/lower limit compare function is enabled).

    Flags to check the above items can also be found in the conversion result registers ADCAnLCR, ADCAnDBiCR or ADCAnCmCR, which thus can be used alternatively.

12. Disable the A/D converter if you want to reconfigure it. To do so, set ADCAnCTL0.ADCAnCE = 0.

34.5.2 Clock usage

The ADCA system clock ADCATCLK is derived from PCLK. The division factor is specified in ADCAnCTL1.ADCAnFR[3:0].

34.5.3 Channels and channel groups

Each input channel can be assigned to a certain channel group (CG). The assigned channels represent the scan list of the channel group.
A scan list can be created for each CG through register settings, and also can be changed during operation. The conversion settings for a CG are applied to all the channels within that group.

The ADCA supports up to three channel groups (CGi, where i = 0 to 2). The channels, i.e. the scan list, of each CGi are specified with the ADCAnCGi register.

Note The conversion of a single channel can be performed by assigning only one input channel to a CG.

(1) Order of A/D conversion

Upon occurrence of the start trigger for a CG, the channels set to its scan list are converted in ascending order, i.e. starting from the lowest channel number.

If A/D conversion requests for multiple CGs are pending, the CGs are converted in the following hierarchical order:

CG2 (highest priority) > CG1 > CG0 (lowest priority)

If the start trigger for a CG with a higher priority, or the trigger for ADCHALT mode is set, the current A/D conversion is halted.

One of the following two methods can be selected for halting A/D conversion, according to the setting of ADCAnCTL1.ADCAnTRMi.

- ADCAnCTL1.ADCAnTRMi = 0:
  The A/D conversion of CG is interrupted immediately.

  The A/D conversion of the interrupted channel is repeated after all pending A/D conversions of higher priority CGs have been finished.
- ADCAnCTL1.ADCAnTRMi = 1
  The A/D conversion of the current channel is completed before the higher priority CG is converted.

  When the A/D conversions of *all* higher priority CGs have been finished, the interrupted A/D conversion is continued from the next channel.

ADCAnSTR2.ADCAnST[2:0] indicates the current conversion status.

Examples The following figures illustrate the different types of conversion interruption; CH3, CH9, and CH20 are assigned to CG0, CH5 and CH9 are assigned to CG2.

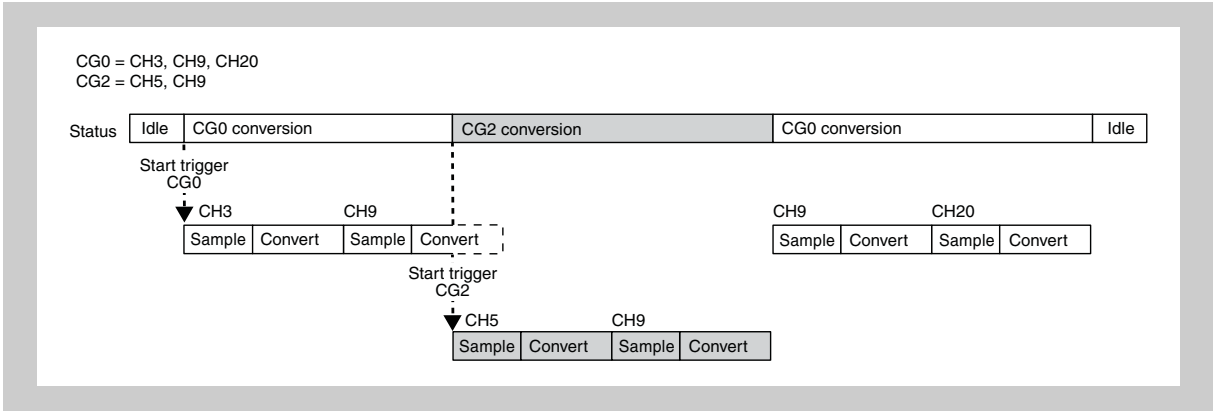


Figure 34-4 Immediate interruption of A/D conversion of CG0 (ADCAnCTL1.ADCAnTRM0 = 0)

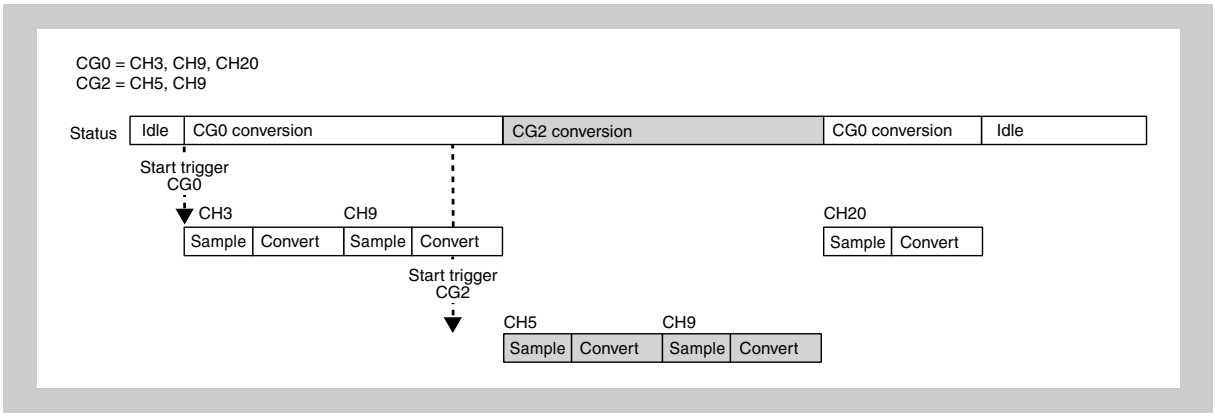


Figure 34-5 Wait until A/D conversion of current channel has been completed (ADCAnCTL1.ADCAnTRM0 = 1)

Note A channel can be assigned to multiple CGs at the same time. This feature should not be used with simultaneous sampling. Otherwise the first sampled value is held over the entire period and converted multiple times (see *34.5.13 "Channel S&H function" on page 2786*).

34.5.4 A/D conversion modes

The A/D converter provides the following A/D conversion modes.

Mode	Operation	Channel group
One-shot conversion mode (ADCAnCTL1.ADCAnMD0 = 0)	Executes CGi scan list conversion only once. The scan list conversion is repeated one to four times. The number of repetitions is selectable by ADCAnCTL0.SCTi[1:0].	CG0, CG1, CG2
Continuous conversion mode (ADCAnCTL1.ADCAnMD0 = 1)	Executes scan list conversion repeatedly.	CG0

Notes
1. A running A/D conversion is interrupted by an A/D conversion request for a higher priority CG and then is automatically continued when all requests of higher priority CGs have been finished (see *1 "Order of A/D conversion" on page 2769*).
2. CG1 and CG2 operate in one-shot conversion mode regardless of the conversion mode setting in ADCAnCTL1.ADCAnMD0.

(1) One-shot conversion mode

In one-shot conversion mode, CGi scan list conversion is executed upon a start trigger. The number of times scan list conversion is repeated can be specified for each CG from 1 to 4, by specifying ADCAnCTL0.ADCAnSCTi[1:0].

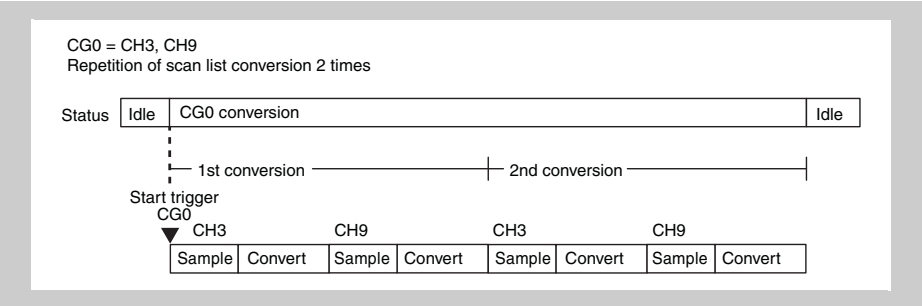


Figure 34-6 2 times repetition of scan list conversion in one-shot conversion mode

(a) Start Trigger for the same CG before end of conversion

The A/D converter can hold one trigger for each CG for starting another conversion prior to the end of conversion of the same CG. Therefore, A/D conversion is performed continuously if one or more subsequent start triggers are input (the second and subsequent start triggers are ignored) before the A/D conversion for the CGi started by the first start trigger ends.

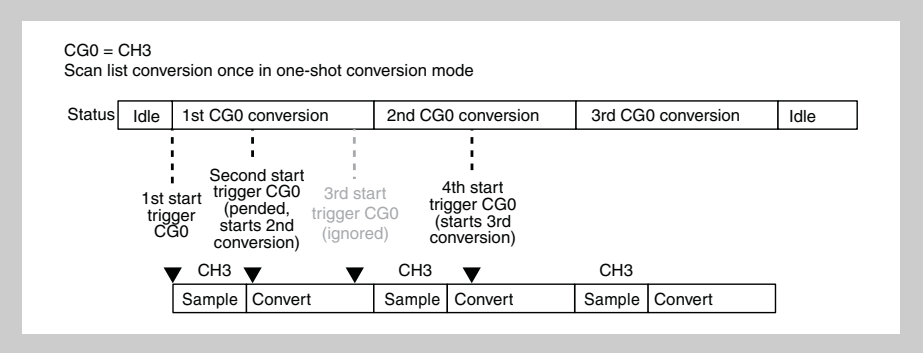


Figure 34-7 Start Trigger for the same CG before end of conversion

(b) Start Trigger for a different CG before end of conversion

Start triggers for lower priority CGs during conversion of a higher priority CG are ignored, i.e.

- CG0 start trigger is ignored during CG1 or CG2 conversion
- CG1 start trigger is ignored during CG2 conversion

Reversely, a start trigger for a higher priority CG during conversion of a lower priority CG is held.

that was input before the start of conversion of a high priority CG is held. When no start trigger is generated, a start-before-conversion-end trigger is accepted even during the conversion of a high priority CG.

- CG2 start trigger is held during CG0 or CG1 conversion
- CG1 start trigger is held during CG0 conversion

During conversion, start triggers before end of conversion for a lower priority CG are ignored.
Reversely, a start trigger before end of conversion for a lower priority CG that was input before the start of conversion of a high priority CG is held. When no start trigger is generated, a start-before-conversion-end trigger is accepted even during the conversion of a high priority CG.

Depending on the ADCAnCTL1.ADCAnTRM0 setting, conversion of the higher priority CG is launched immediately ((ADCAnTRM0 = 0) or after completion of the lower priority CG conversion (ADCAnTRM0 = 1).
Refer also to *34.5.3 “Channels and channel groups” on page 2769* .

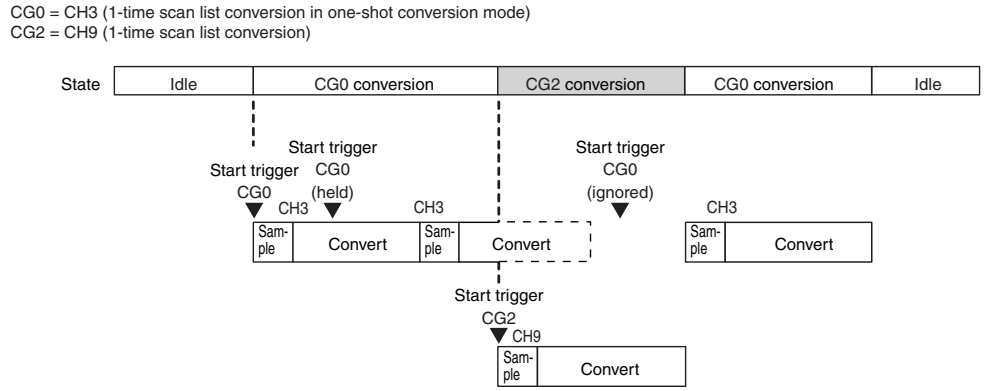


Figure 34-8 Start Trigger for a different CG before end of conversion (ADCAnCTL1.ADCAnTRM0 = 0)

(2) Continuous conversion mode

Continuous conversion mode can be used only for CG0 (ADCAnCTL1.ADCAnMD0 = 1).

In continuous conversion mode, a start trigger causes the channels of CG0 to be sampled and converted repeatedly until a stop trigger is generated or another stop condition occurs (refer to *34.5.6 "Stopping A/D conversion" on page 2776*).

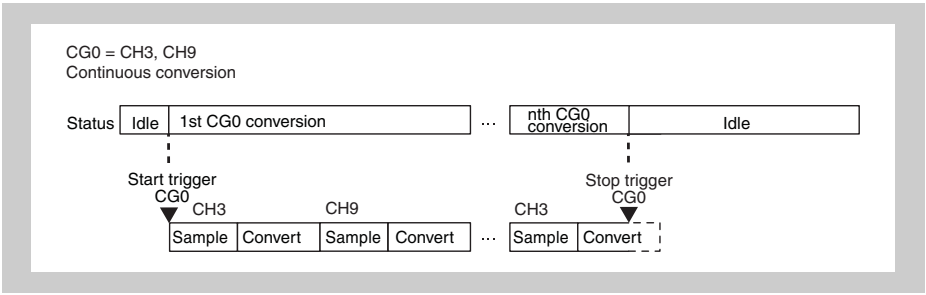


Figure 34-9 Continuous conversion mode

Caution After a stop trigger, the state becomes idle and sampling/conversion cannot be performed.

Note Additional start triggers for CG0 are ignored in continuous conversion mode.

34.5.5 Starting A/D conversion (start trigger modes)

A/D conversion can be started by a software or a hardware trigger, as specified by ADCAnCTL1.ADCAnMD1.

If A/D conversion is started for multiple CGs, the conversion order depends on the priority of the CGs (refer to *1 "Order of A/D conversion" on page 2769*).

Notes

1. If no channels are assigned to the CGi scan list (ADCAnCGi = 0000 0000_H), start triggers for that CGi are ignored
2. In one-shot conversion mode, the A/D converter can hold only one start trigger. Additional start triggers are ignored (refer to *Figure 34-7 "Start Trigger for the same CG before end of conversion" on page 2772* and *34.5.13 "Channel S&H function" on page 2786*).
3. In continuous conversion mode, additional start triggers before a stop trigger has been generated, are ignored.

(1) Software start trigger

The A/D conversion of CGi is started by setting ADCAnTRGi.ADCAnSTTi = 1, provided that the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1).

Timing example of software start trigger

The following figure shows the timing of a software start trigger with the following conditions:

- ADCATCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)

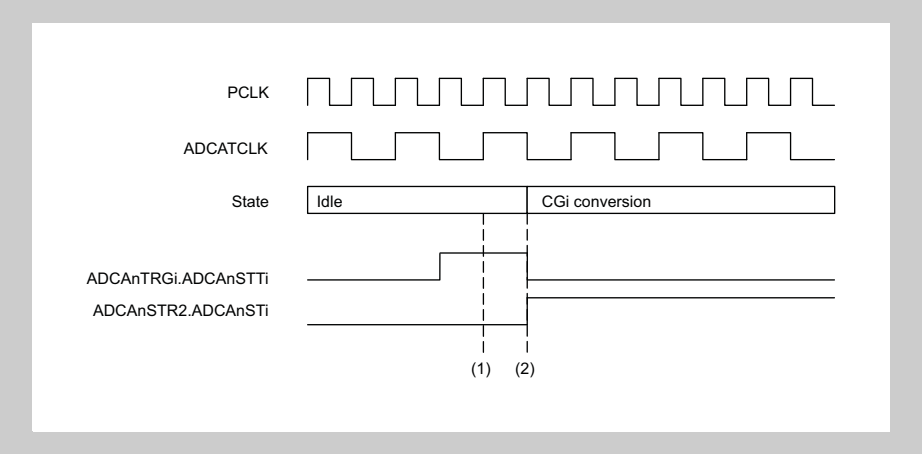


Figure 34-10 Software start trigger timing example

1. Software trigger for CGi is written.
2. A/D conversion starts at the next falling edge of ADCATCLK. The status bit ADCAnSTR2.ADCAnSTi is set, indicating that A/D conversion of CGi is running.

(2) Hardware start trigger

The A/D conversion of CGi is started upon detection of the valid edge of the ADCAnTTRGi signal, provided that the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1) and hardware trigger mode is set (ADCAnCTL1.ADCAnMD1 = 1).

The valid edge is specified in ADCAnCTL1.ADCAnTiETS[1:0] for every CG individually.

H/W trigger expansion

If this microcontroller supports a hardware trigger expansion, up to 16 hardware trigger sources can be specified for each ADCAnTTRGi signal input. ADCAnTSELi specifies the input signals to be used as the ADCAnTTRGi signal.

Note For details about the hardware start trigger function expansion refer to “H/W trigger expansion” in the first section of this chapter.

Timing of hardware start trigger

The A/D converter starts A/D conversion upon detection of the valid edge of the ADCAnTTRGi signal.

The following figure shows the timing of a hardware start trigger with the following conditions:

- ADCATCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)
- Valid edge of ADCAnTTRGi is rising edge (ADCAnCTL1.ADCAnTiETS[1:0] = 01_B)

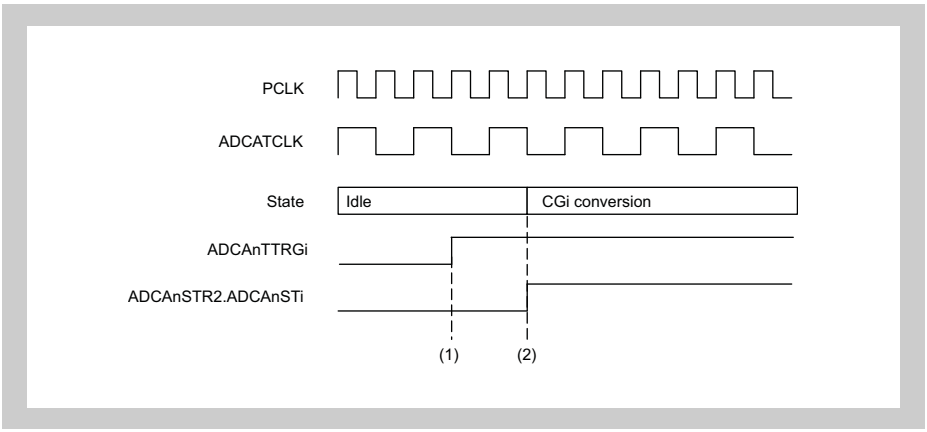


Figure 34-11 Hardware start trigger timing

1. Input signal ADCAnTTRGi rises.
2. A/D conversion starts at the next falling edge of ADCATCLK.

34.5.6 Stopping A/D conversion

(1) Stop trigger

Setting the stop trigger bit for CGi to 1 (ADCAnTRG4+i.ADCAnSPi = 1) stops the A/D conversion for the CGi.

If the stop trigger is generated before the A/D conversion ends, the A/D conversion end interrupt signal INTADCAnTi is not generated and the A/D conversion result register is not updated.

If the start trigger is generated again after the A/D conversion is stopped by a stop trigger, scan list conversion is executed from the beginning.

Follow the stop trigger procedure below when using the hardware start trigger.

1. Stop hardware start trigger generation.
2. Set the stop trigger bit (ADCAnTRG4+i.ADCAnSPi = 1).
3. Check the status of ADCAnSTR2.ADCAnSTi.

If the above procedure is not followed, the hardware start trigger and the stop trigger may conflict with each other and A/D conversion may not stop.

Timing of stop trigger

1. Stop trigger for CGi is written.
2. A/D conversion of CGi stops at the next falling edge of ADCATCLK.
   The status bit ADCAnSTR2.ADCAnSTi is cleared, indicating that A/D conversion of CGi is stopped.
   If the digital value of ADCAnIm is already available, the operation is as follows:
   – All A/D conversion result registers are updated.
   – The conversion end interrupt INTADCAnTi is generated as configured in ADCAnIOCi (refer to *34.5.10 “Interrupt generation” on page 2780*).
   – If configured in ADCAnCTL2, the A/D conversion result is checked to see whether it is in the specified value range (refer to *34.5.12 “Result check functions” on page 2784*).

   The A/D converter proceeds with pending A/D conversion requests of other CGs - if there are any.

The following figures show the timing of a stop trigger with the following conditions:

- ADCATCLK = PCLK / 2 (ADCAnCTL1.ADCAnFR[3:0] = 000_B)
- The A/D conversion end interrupt INTADCAnTi is generated at the end of the A/D conversion of CGi (ADCAnIOCi = 0000 0000_H)

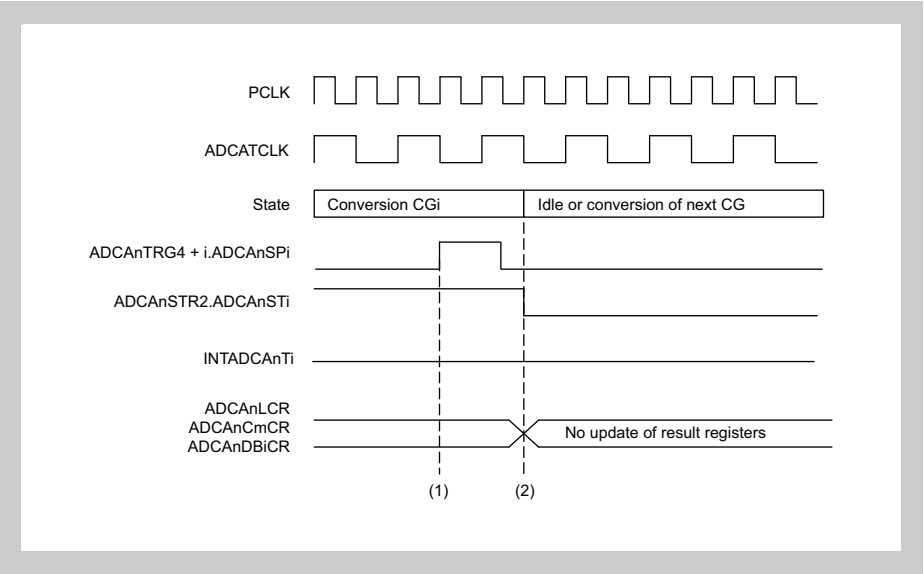


Figure 34-12 Stop trigger timing, if generated before A/D conversion end

1. The stop trigger (ADCAnTRG4+i.ADCAnSPi) is set to 1.
2. The status bit (ADCAnSTR2.ADCAnSTi) is cleared.
   The A/D conversion end interrupt INTADCAnTi is not generated and the A/D conversion result register is not updated.

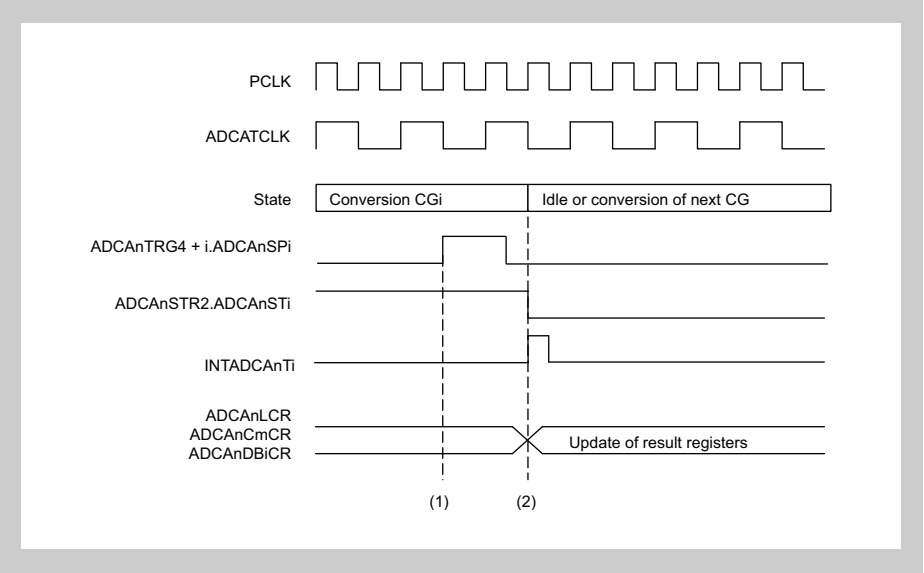


Figure 34-13 Stop trigger timing, if generated after A/D conversion end

1. The stop trigger (ADCAnTRG4+i.ADCAnSPi) is set to 1.
2. The status bit (ADCAnSTR2.ADCAnSTi) is cleared.
   The A/D conversion end interrupt (INTADCAnTi) is generated and the A/D conversion result register is updated.

(2) Other stop conditions

In addition to the software stop trigger, A/D conversion is stopped when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

With ADCAnCTL0.ADCAnCE = 0 conversions are stopped immediately and no A/D conversion results are stored in the conversion result registers.

34.5.7 Stand-by mode

Stand-by mode is entered, when the related system stand-by mode becomes active.

The A/D converter is automatically disabled (ADCAnCTL1.ADCAnCE = 0).

To leave stand-by mode:

1. Release related system stand-by mode.
2. Enable the A/D converter by setting ADCAnCTL1.ADCAnCE to 1.

Note After stand-by mode is released, a start trigger is accepted but conversion does not start until the stabilization time elapses (stabilization counter ADCAnCNT = 00_H).
Refer to *34.5.17 “Stabilization control” on page 2801* for details.

34.5.8 Pausing and resuming A/D conversion (ADCHALT mode)

The A/D Converter allows the A/D conversion (of all CGs) to be paused/halted. The halted A/D conversion can be continued with a resume trigger.

Procedure:

1. Set ADCAnTRG3 = 1 to go into ADCHALT mode (refer to *1 “Order of A/D conversion” on page 2769* for details about the halt operation).
   – Start triggers are ignored in ADCHALT mode.
   – The internal circuits are stopped and the power consumption is reduced.
   – The analog input pins ADCAnIm can be used for other functions in ADCHALT mode.
2. Set ADCAnTRG7.ADCAnSP3 = 1 to release ADCHALT mode and start A/D conversion.

Note ADCHALT has the highest priority and overrules all CGi conversions.

34.5.9 Resolution, sampling and conversion times

The total conversion time comprises initialization, sampling, A/D conversion and result storage times.

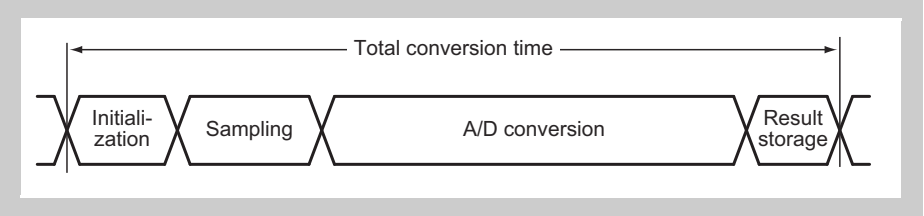


Figure 34-14 Total conversion time

The *initialization time* takes a fixed number of clock cycles to initialize the A/D Converter for sampling.

The *sampling time* is the time of connecting an analog input voltage to the Sample & Hold circuit. The *A/D conversion time* is the time required to obtain one digital value out of an analog input voltage.

The *A/D conversion time*, and thus the *total conversion time,* depends on the conversion resolution.

The *results storage time* takes a fixed number of clock cycles to store the conversion result in the respective registers.

Table 34-14 Sampling and conversion times (discharge function and buffer amplifier disabled)

Conversion resolution	Initialization and result storage time	Sampling time	A/D conversion time	Total conversion time
10 bit (ADCAnCTL1. ADCAnCTYP = 1)	3 clocks[a]	6.5 clocks[a]	11.5 clocks[a]	21 clocks[a]
12 bit (ADCAnCTL1. ADCAnCTYP = 0)			13.5 clocks[a]	23 clocks[a]

a) clocks = ADCA system clock ADCATCLK periods

Discharge function

Using the discharge function (ADCAnCTL1.ADCAnDISC = 1) increases the total conversion time by 1 ADCATCLK cycle.
Refer to see *34.5.15 “Discharge function” on page 2800* for details.

Buffer amplifier

Using the buffer amplifier (ADCAnCTL1.ADCAnBPC = 1) increases the total conversion time by 4 ADCATCLK cycles.
Refer to *34.5.16 “Buffer amplifier function” on page 2801* for details.

ADCATCLK limits

The ADCA system clock ADCATCLK must not exceed a certain limit.
The limits of the ADCATCLK are calculated as follows:

$$ADCATCLK_{max}[MHz] = \frac{ADCATCT[clocks]}{TCON_{min}[\mu s]}$$

$$ADCATCLK_{min}[MHz] = \frac{ADCATCT[clocks]}{TCON_{max}[\mu s]}$$

with TCONmin is the minimum and TCONmax the maximum total conversion time, which are specified in the Electrical Target Specification.

34.5.10 Interrupt generation

(1) A/D conversion end interrupt INTADCAnTi

The INTADCAnTi interrupt indicates that the new A/D conversion result is saved to the conversion result register.

An A/D conversion end interrupt is generated upon completion of the A/D conversion for any channel of CGi specified in the ADCAnIOCi register.

If nothing else is configured (ADCAnIOCi = 0000 0000_H), INTADCAnTi is generated at the end of the A/D conversion of CGi.

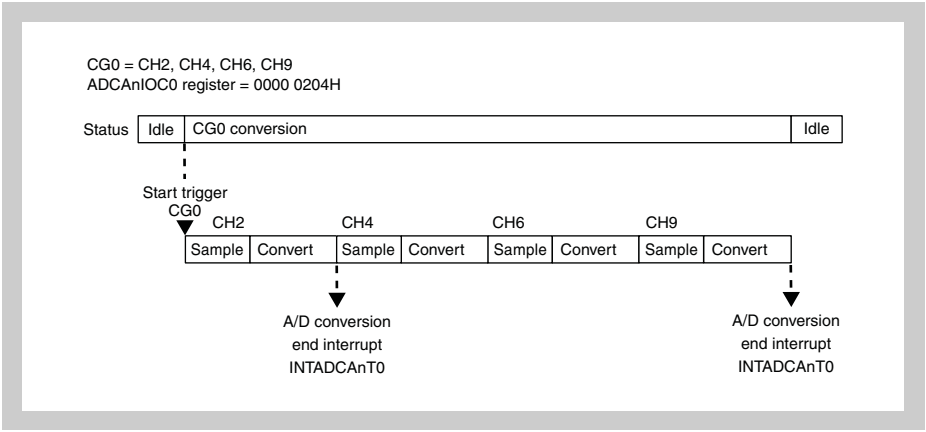


Figure 34-15 Generation of A/D conversion end interrupt INTADCAnTi

Notes

1. ADCAnIOCi can be written at any time even when the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1). The new value takes effect after the current A/D conversion of CGi has been completed.
2. ADCAnIOCi is associated with ADCAnCGi and their buffer registers should be updated simultaneously. As the update time depends on writing ADCAnCGi, always write ADCAnIOCi before ADCAnCGi if you want to change the interrupt generation for a CG.

(2) Error interrupt INTADCAnERR

The interrupt INTADCAnERR is generated in the following cases:

- The result of the A/D conversion of a specified channel is out of the specified range, when the conversion result upper/lower limit compare function is enabled.
- An A/D conversion result in ADCAnLCR, ADCAnDBiCR, or ADCAnCmCR has been overwritten before it was read.
  The generation of the INTADCAnERR error interrupt upon register overwrite can be controlled for each register by setting ADCAnCTL0.ADCAnOEM[4:0].

Refer to *34.5.12 “Result check functions” on page 2784* for details.

(3) Last conversion interrupt INTADCAnLLT

The interrupt INTADCAnLLT is generated after each successful conversion.

34.5.11 Storage of A/D conversion result

(1) A/D conversion result registers

The A/D conversion result is stored in the following registers:

- ADCAnLCR register
  This register stores the latest A/D conversion result.
- ADCAnDBiCR register
  This register stores the latest A/D conversion result for CGi.
- ADCAnCmCR register
  This register stores the latest A/D conversion results for channel m.

Note The A/D conversion result registers remain unchanged until the next conversion is completed and the content of the successive approximation register SAR is copied to the conversion result registers. Thus during conversion the conversion result registers content is not changing.

Each register also stores status flags of the A/D conversion result (see *34.5.12 "Result check functions" on page 2784*).

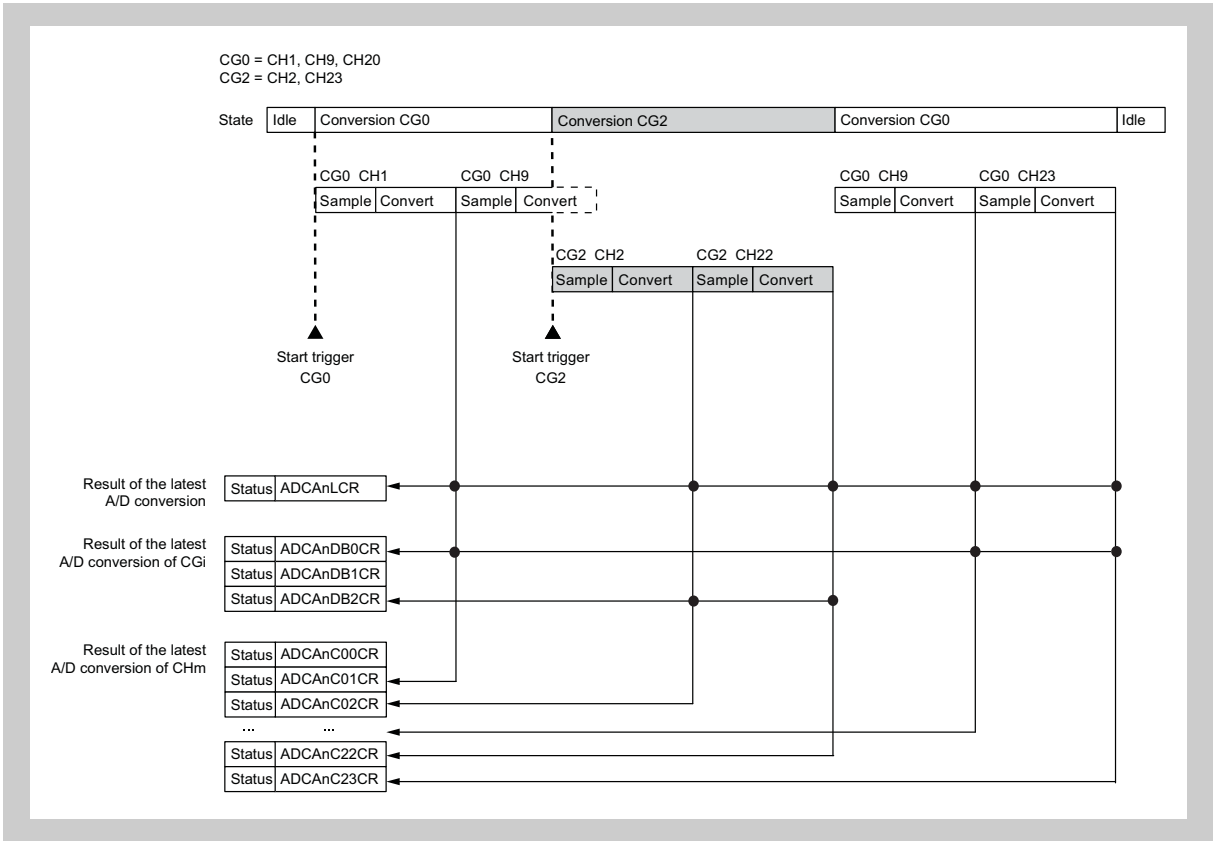


Figure 34-16 Storage of A/D conversion result

(2) Configuration related to A/D conversion result storage

Bit position ADCAnCTL1.ADCAnCRAC specifies whether the 12-bit or 10-bit A/D conversion result is right aligned (ADCAnCRAC = 0) or left aligned (ADCAnCRAC = 1).

Read & clear function ADCAnCTL1.ADCAnRCL specifies whether the A/D conversion result ADCAnCmCR is retained after reading it or cleared by reading it.

(3) Relationship between analog input voltage and A/D conversion result

The relationship between the analog input voltage input to the analog input pin (ADCAnIm) and the A/D conversion results (the values ADCAnLCR[15:00], ADCAnCmCR[15:00], and ADCAnDBiCR[15:00]) is expressed as follows:

$$\text{A/D conversion result value} = \text{INT}\left(\frac{V_{IAN} - AV_{REFM}}{AV_{REFP} - AV_{REFM}} \times 2^k + 0.5\right)$$ sdfasfd

or

$$(\text{A/D conversion result} - 0.5) \times \frac{AV_{REFP} - AV_{REFM}}{2^k} \leq V_{IAN} - AV_{REFMn} < (\text{A/D conversion result} + 0.5) \times \frac{AV_{REFP} - AV_{REFM}}{2^k}$$

INT():	Function that returns the integer part of the value in parentheses
V_{IAN}:	Analog input voltage
AV_{REFP}:	AVREFP pin voltage
AV_{REFM}:	AVREFM pin voltage
A/D conversion result:	Values of the ADCAnLCR[15:00], ADCAnCmCR[15:00], and ADCAnDBiCR[15:00] bits
k:	Resolution

The figures below show the relationship between the analog input voltage and A/D conversion result.

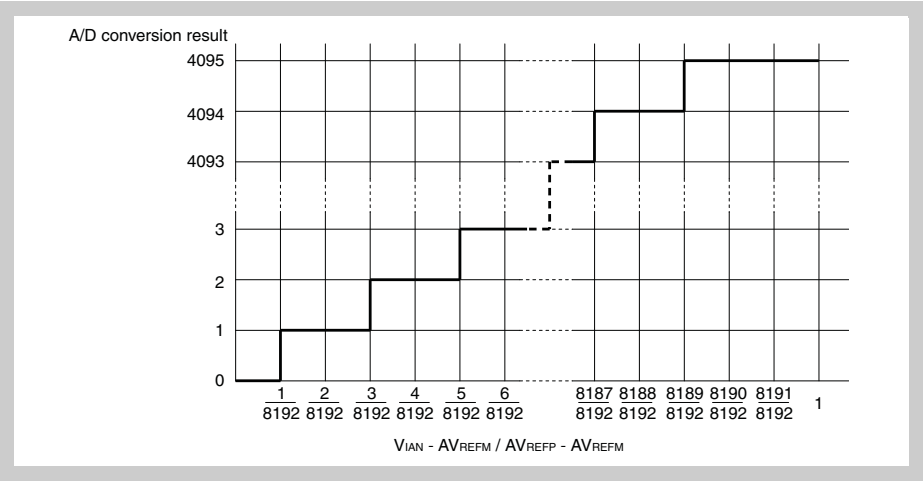


Figure 34-17 Conversion characteristics of 12-bit A/D converter (ADCAnCTL1.ADCAnCTYP = 0)

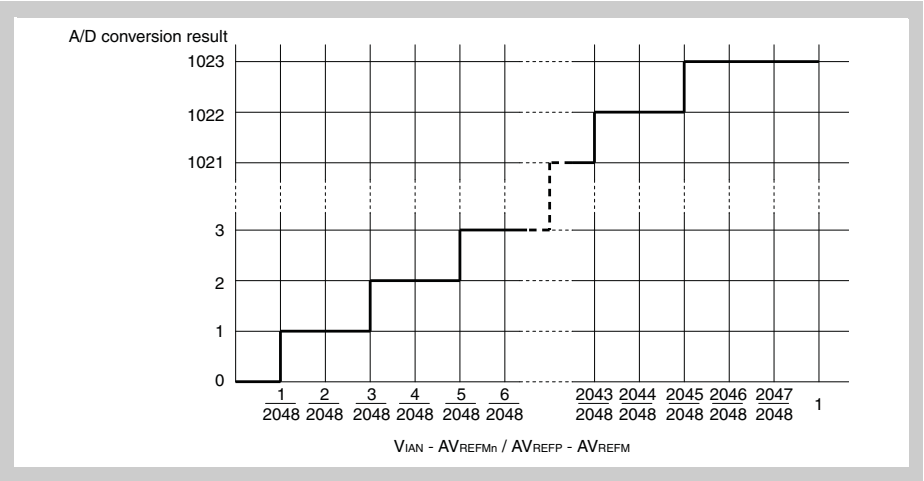


Figure 34-18 Conversion characteristics of 10-bit A/D converter (ADCAnCTL1.ADCAnCTYP = 1)

34.5.12 Result check functions

ADCA allows checking of the A/D conversion result with the following functions:

- Conversion result overwrite check function
- Conversion result read flag function
- Conversion result upper/lower limit compare function

(1) Conversion result overwrite check function

The ADCA allows to check if an A/D conversion result has been overwritten before it was read.

Error flags

The A/D conversion result registers have the following overwrite error flags:

- ADCAnLCR.ADCAnLER1
- ADCAnDBiCR.ADCAnDBiER1
- ADCAnCmCR.ADCAnCmER1

Setting of the overwrite error flag indicates, that the conversion result stored in the respective register was overwritten before it is read.

The overwrite flag ADCAnCmCR.ADCAnCmER1 is reflected in ADCAnSTR1.ADCAnOWEm, which holds overwrite flags for all channels.

Error interrupt

If the A/D conversion results in the ADCAnLCR, ADCAnDBiCR, and ADCAnCmCR registers are overwritten before they were read, an error interrupt INTADCAnERR is generated.

Generation of an INTADCAnERR interrupt can be masked separately for each conversion result register type, so that the interrupt does not become effective:

Table 34-15 Overwrite error interrupt masking

Result register	Mask bit in ADCAnCTL0
ADCAnLCR	ADCAnOEM4
ADCAnDB2CR	ADCAnOME3
ADCAnDB1CR	ADCAnOEM2
ADCAnDB0CR	ADCAnOEM1
ADCAnCmCR	ADCAnOEM0

If a mask bit is set to 1, the interrupt is not effective.

Note Overwrite error interrupt generation for conversion result registers that are not to be read shall be masked by setting the appropriate ADCAnCTL0.ADCAnOEM[4:0] = 1.

(2) Conversion result read flag function

An update status flag indicates whether the A/D conversion result in the conversion result register has already been read or is a new one.

Status flags

The update status flags are provided in the A/D conversion result registers:

- ADCAnLCR.ADCAnLUR
- ADCAnDBiCR.ADCAnDBiUR
- ADCAnCmCR.ADCAnCmUR

If an update status flag is set to 1, the A/D conversion result is new.

The update status flags are cleared after they are read.

(3) Conversion result upper/lower limit compare function

The ADCA can check whether an A/D conversion results lies within a configurable value range.

The result check function can be enabled for every channel individually in ADCAnCTL2.

When enabled, the A/D conversion result ADCAnCmCR is compared with the specified lower limit (ADCAnLL) and upper limit (ADCAnUL), each time the channel result is converted respectively updated.

Error flags

If the A/D conversion result of a specified channel is either lower than the lower limit value ADCAnLL or higher than the upper limit value ADCAnUL, the ADCAnSTR0.ADCAnRCE[23:00] error flag corresponding to that channel is set.

ADCAnSTR0 indicates the error status of the latest A/D conversion upper/lower limit check for every channel. By use of this register it is possible to evaluating which A/D conversion results are outside the specified range.

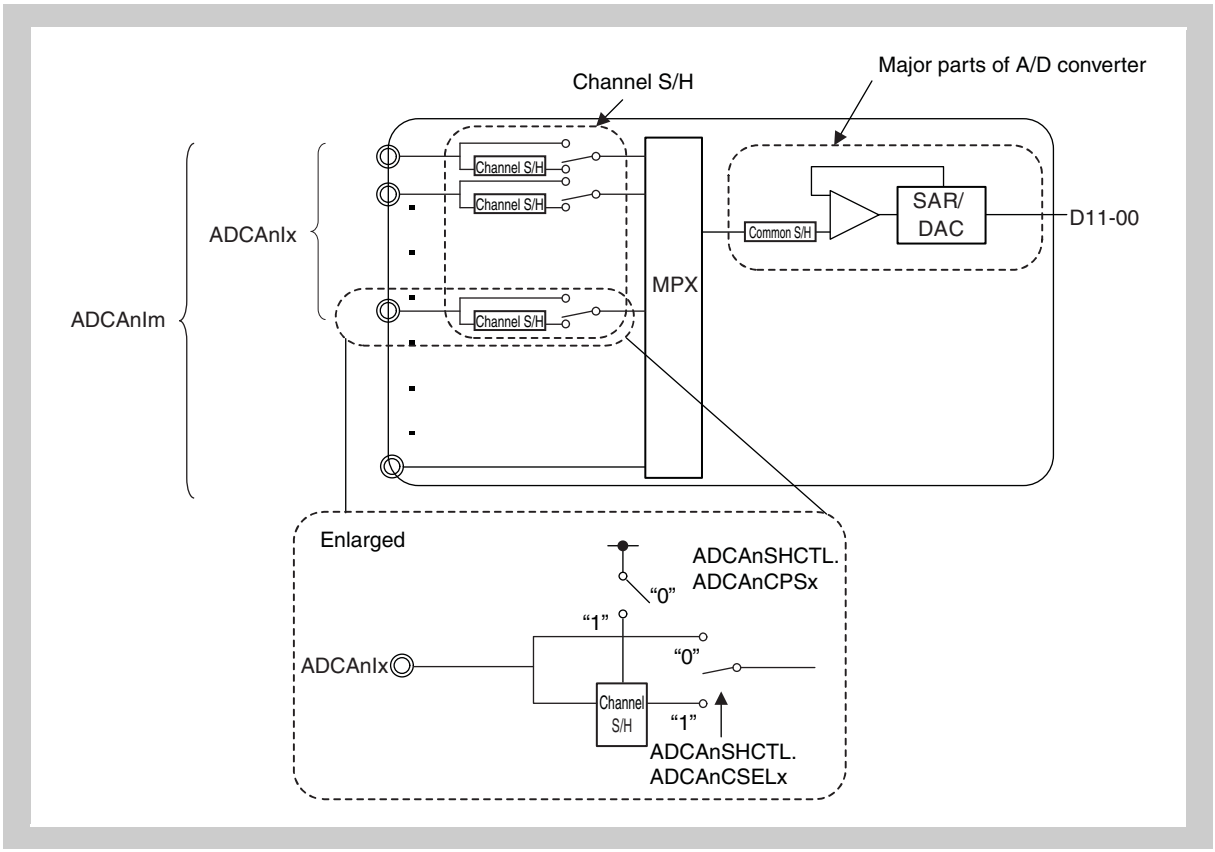
The result check error flag is also mirrored by the A/D conversion result registers ADCAnLCR, ADCAnDBiCR, and ADCAnCmCR.

The result check error flags ADCAnSTR0.ADCAnRCE[23:00] are reflected in ADCAnCmCR.ADCAnCmER0 of the respective channel.

Error interrupt

If the A/D conversion result for the specified channel is out of the setting range, the INTADCAnERR error interrupt is generated.

34.5.13 Channel S&H function

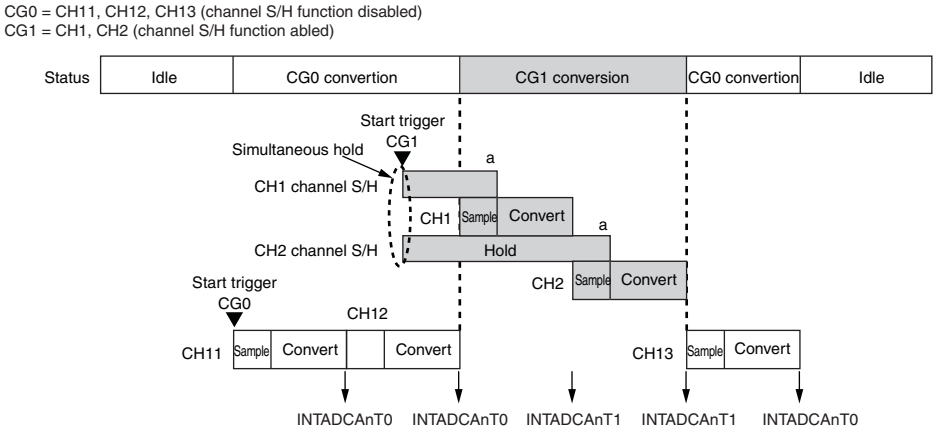


Channel S&H can be used in CG0's one-shot conversion mode (no repetition) and CG1, 2 (no repetition).

Power supply to the channel S&H circuit is controlled with ADCAnSHCTL. ADCAnCPSx, and the channel S&H function is enabled/disabled with ADCAnSHCTL. ADCAnCSELx.

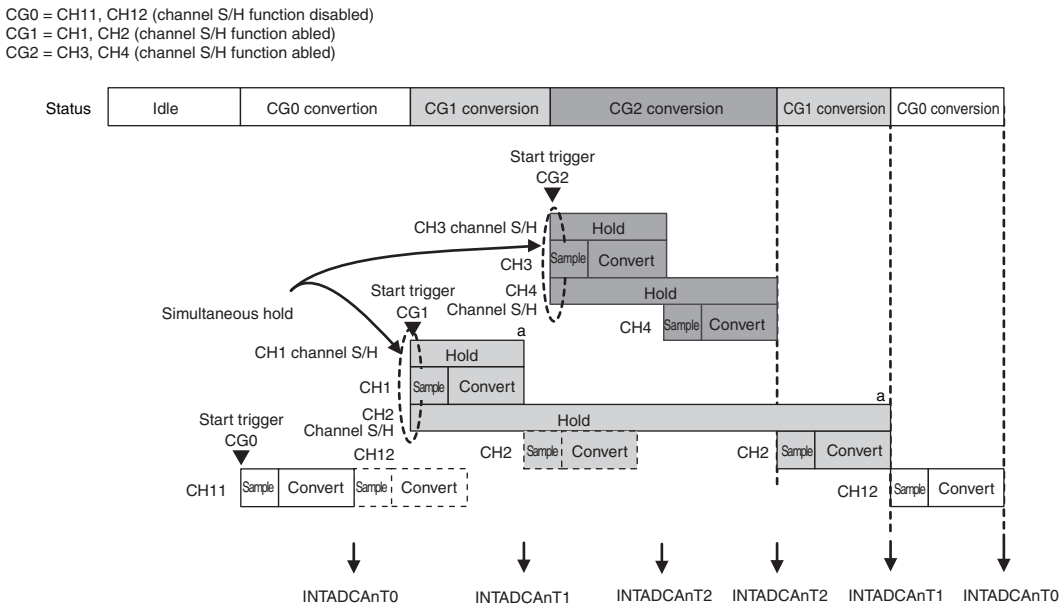
When a hardware start trigger or software start trigger is generated, the analog input signal of the channel for which the channel S&H function is enabled with ADCAnSHCTL.ADCAnCSELx is held with the channel S&H circuit. Next, scan list conversion starts according to the ADCAnCTL1.ADCAnTRMi setting.

Example 1: A start trigger with a high priority is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 1.



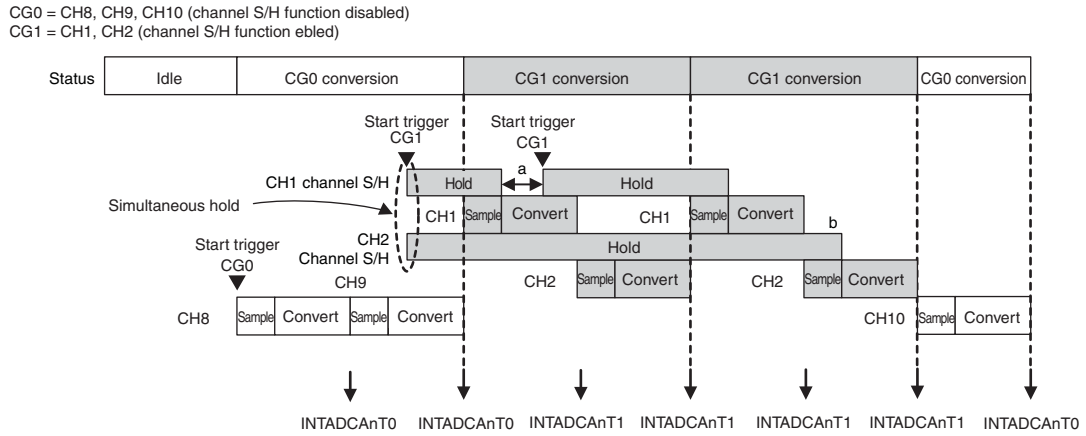
a) When ADCAnTRM1 bit = 1, the channel S&H circuit switches to sampling at this timing.

Example 2: A start trigger with a high priority is generated while ADCAnTRM0 = 0 and ADCAnTRM1 = 0.



a) When ADCAnTRM1 bit = 0, the channel S&H circuit switches to sampling at this timing.

Example 3: A subsequent start trigger is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 1.

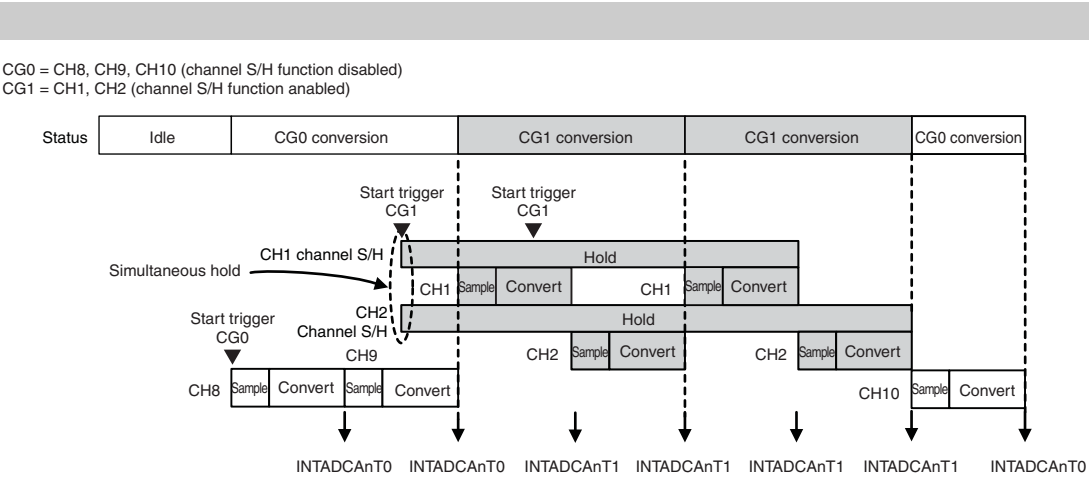


a) The sampling time in the channel S&H circuit must be taken into consideration. Secure 1.5 µs or more.

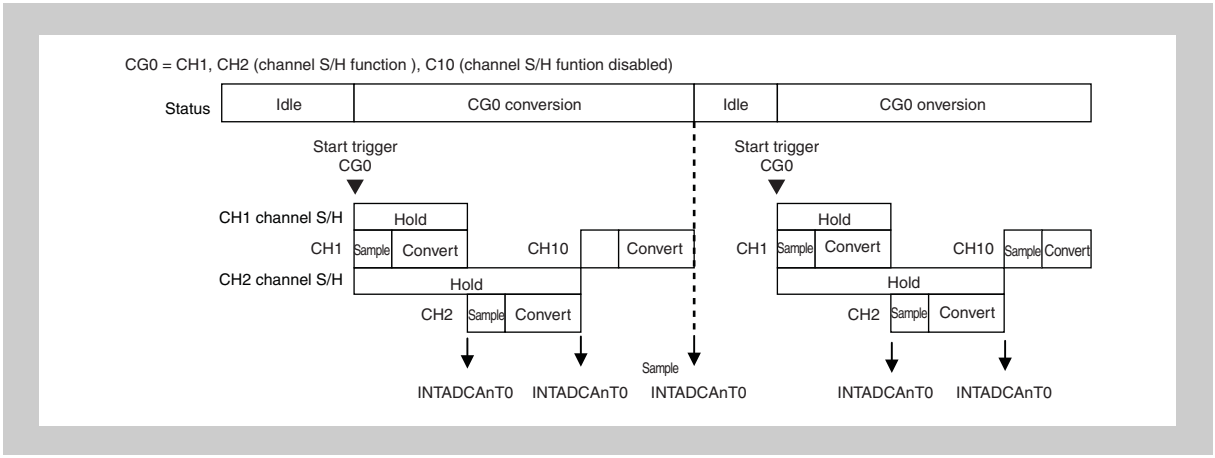
b) When the start trigger is generated, the channel S&H circuit continues to hold the current analog input until the next conversion.

Example 4: A subsequent start trigger is generated while ADCAnTRM0 = 1 and ADCAnTRM1 = 0.

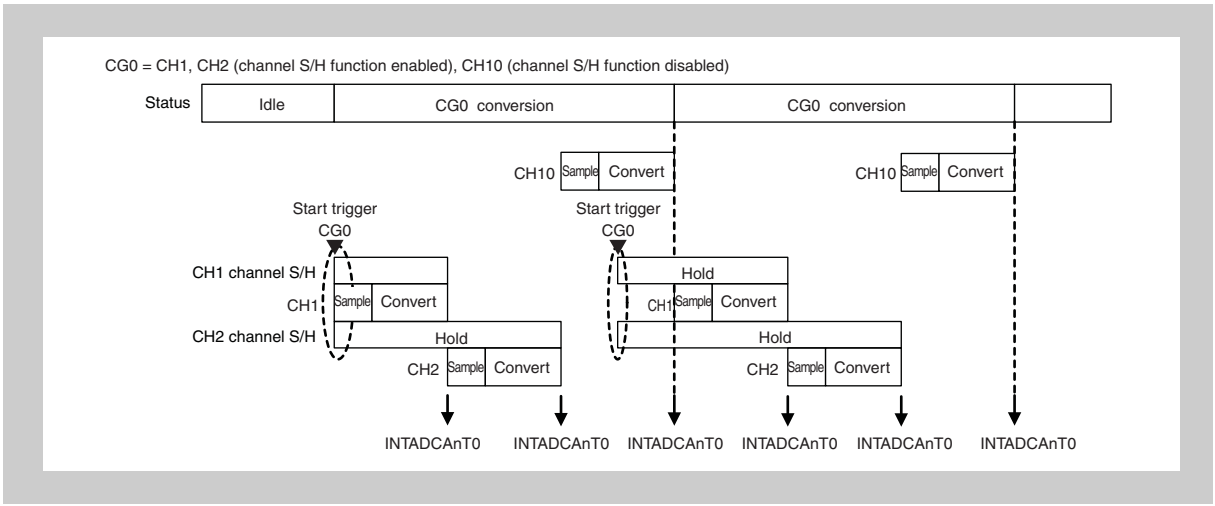
When a subsequent start trigger is generated during A/D conversion, the channel S&H circuit does not newly perform sampling.



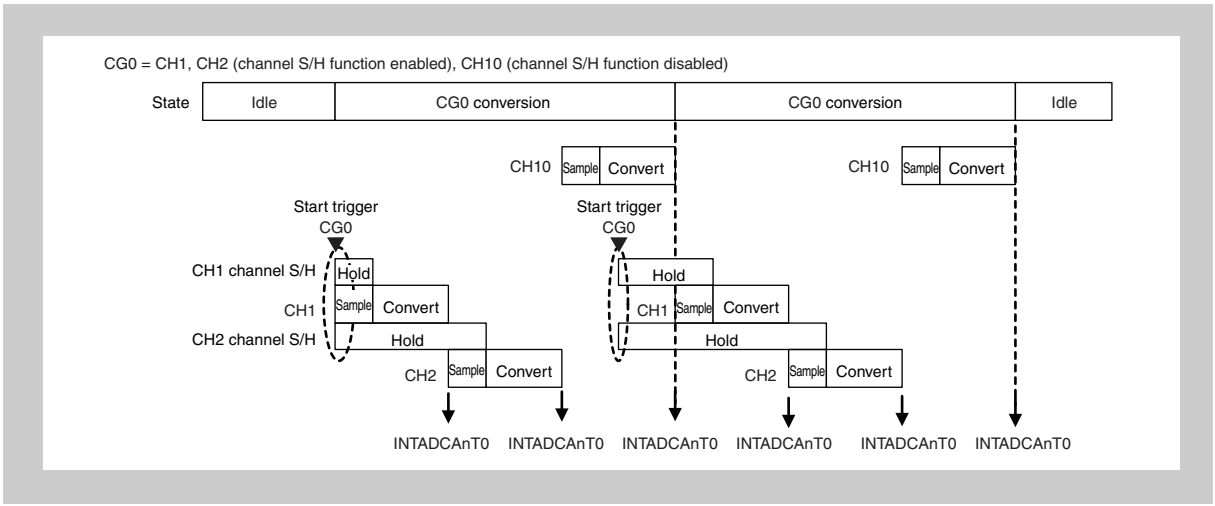
Example 5: CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 0.



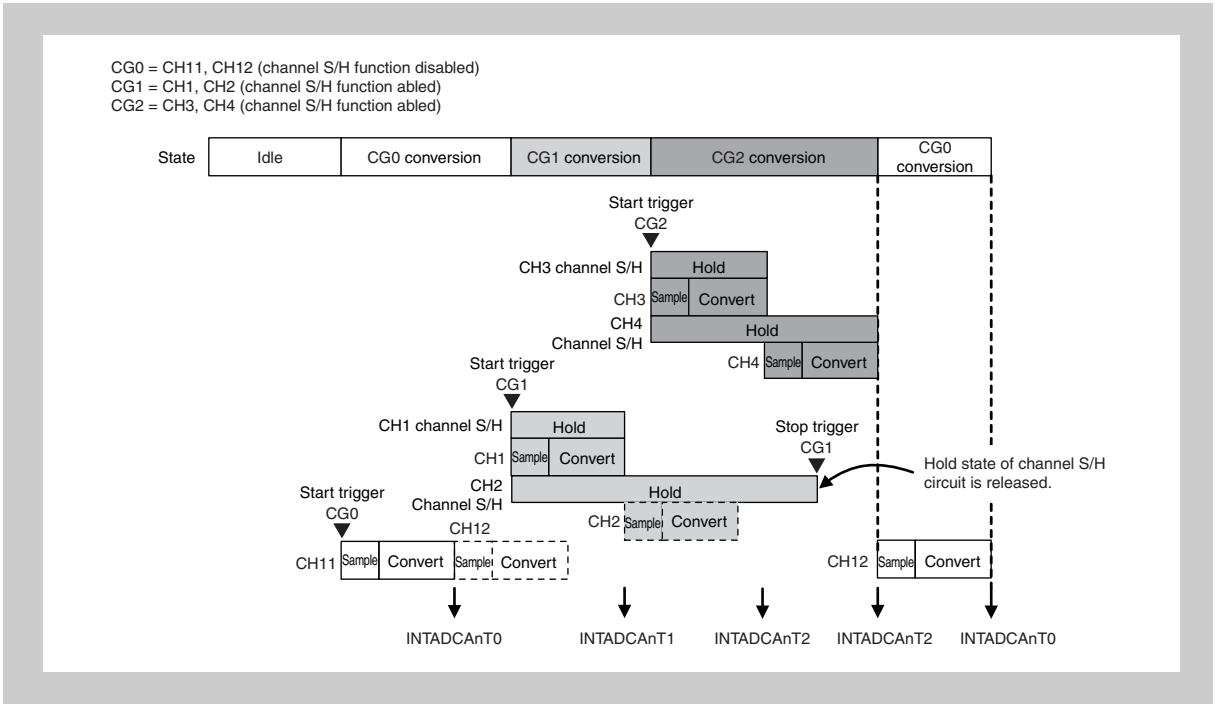
Example 6: A subsequent start trigger is generated while CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 0.



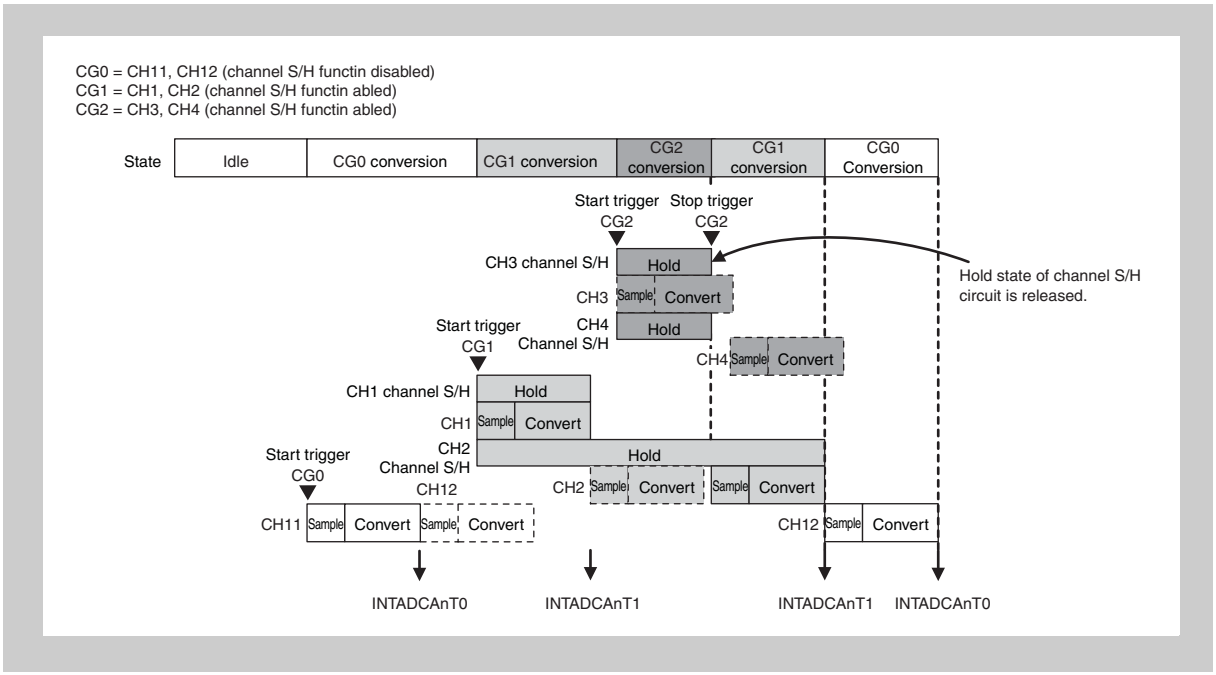
Example 7: A subsequent start trigger is generated while CG0 is in one-shot conversion mode (no repetition) and ADCAnTRM0 = 1.



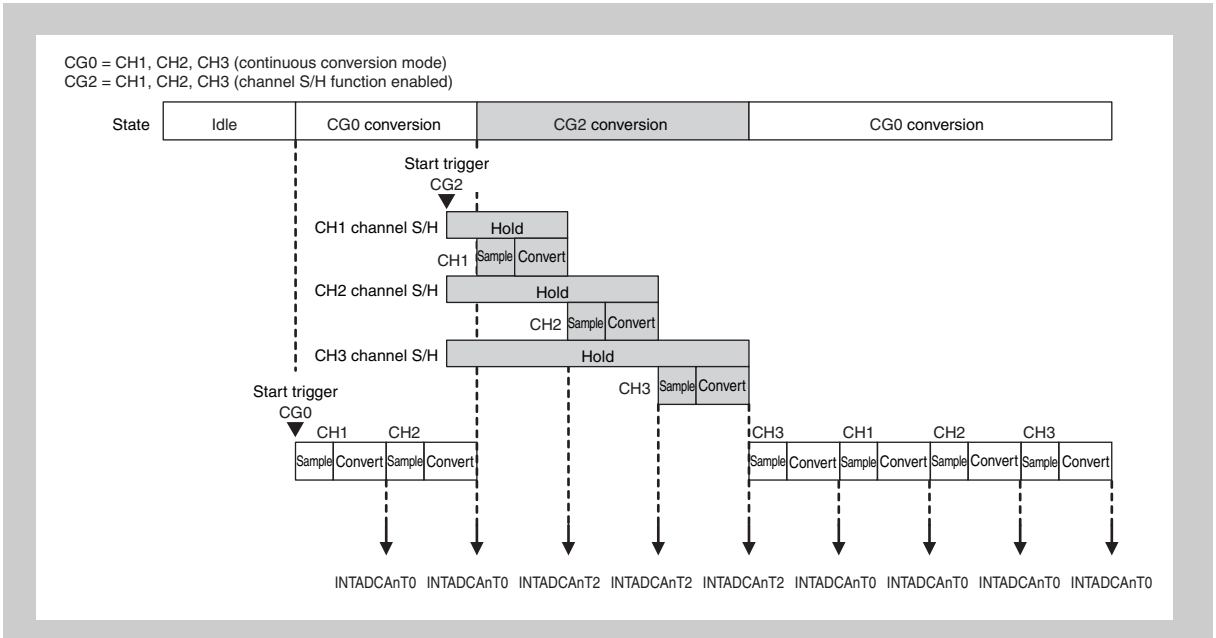
Example 8: The CG1 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



Example 9: The CG2 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



Example 10: The CG2 stop trigger is generated while different channels are set for CG1 and CG2, ADCAnTRM0 = 0, and ADCAnTRM1 = 0.



(1) Restrictions when using the channel S&H function

1. Do not set a channel that will use the channel S&H function to multiple CGs.

Example 1: The channel S&H function is used for CG0, CG1, and CG2 (CG0 = one-shot conversion mode).

The following combinations can be set.

- CG0 (CH1 selected), CG1 (CH2 selected), CG2 (CH3 selected)

Setting the following combinations is prohibited.

- CG0 (CH1 selected), CG1 (CH1 selected), CG2 (CH2, 3 selected)

Example 2: The channel S&H function is used for CG1 and CG2 (CG0 = continuous conversion mode).

The following combinations can be set.

- CG0 (CH1 selected), CG1 (CH1, 2 selected), CG2 (CH3 selected)
- CG0 (CH1, 2, 3 selected), CG1 (CH1 selected), CG2 (CH2, 3 selected)

Setting the following combinations is prohibited.

- CG0 (CH1, 2, 3 selected), CG1 (CH1, 2 selected), CG2 (CH2, 3 selected)

2. To change a scan list during operation for a CGi that uses the channel S&H function, specify the settings so that the target channels that use the channel S&H function do not change.

Example: CH1, 2, and 3 use the channel S&H function.

The following combinations can be set.

- CG0 (CH1, 2, 3) changed to CG0 (CH1, 2, 3, 10, 11)
- CG0 (CH1, 2, 3, 10, 11) changed to CG0 (CH1, 2, 3)
- CG0 (CH7, 8, 9) changed to CG0 (CH10, 11, 12)

Setting the following combinations is prohibited.

- CG0 (CH1, 2, 3) changed to CG0 (CH1, 2)
- CG0 (CH1) changed to CG0 (CH1, 2, 3)
- CG0 (CH7, 8, 9) changed to CG0 (CH1, 7, 8, 9)
- CG0 (CH1, 2, 9) changed to CG0 (CH9, 10, 11)

3. The repetition function is prohibited for a CG that uses the channel S&H function. If the channel S&H function is used in CGi, set the number of repetitions setting bits (ADCAnCTL0.ADCAnSCTi[1:0]) for that CG to 00_B.

34.5.14 Self-diagnosis functions

The following self-diagnosis functions can be used to verify that the ADCA works properly:

- Diagnosis of the A/D conversion circuit
- Diagnosis of the channel multiplexer
- Diagnosis for open pins
- Channel S&H circuit diagnosis

The figure below outlines the self-diagnosis functions which are explained in detail in the following sections.

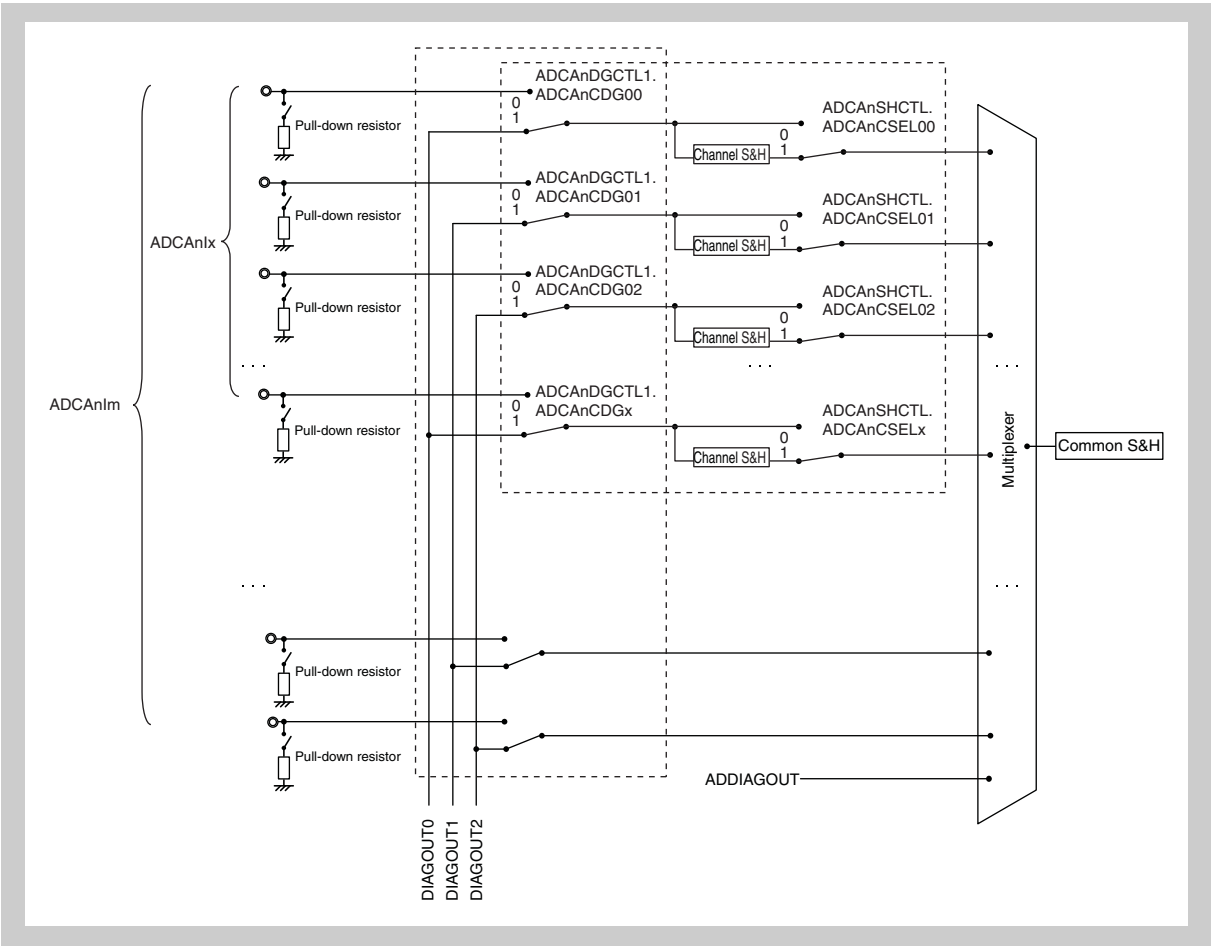


Figure 34-19 Outline of self-diagnosis functions

- The internal reference voltage ADDIAGOUT is used as input for a diagnostic A/D conversion to check the proper function of the A/D conversion circuit.
- The internal reference voltages DIAGOUT0, DIAGOUT1, and DIAGOUT2 are used to test the multiplexer.
- The pull-down resistors are connected to detect open input pins.

(1) Diagnosis of the A/D conversion circuit

The operation of the A/D conversion circuit can be diagnosed.

The diagnosis of the A/D conversion circuit can be performed during normal A/D conversion. Following the end of A/D conversion of CG0, the ADDIAGOUT reference voltage signal is converted. If this diagnosis A/D conversion result differs greatly from the expected value, a hardware anomaly or a malfunction may occur.

The diagnostic A/D conversion is enabled by ADCAnCG0.ADCAnDIAG = 1.

Note The A/D conversion circuit diagnosis is available for CG0 only.

The diagnostic A/D conversion is started after the A/D conversion of the last channel of CG0 has been completed:

- The A/D conversion results of CG0 are stored in the “normal” A/D conversion result registers (refer to *1 “A/D conversion result registers” on page 2781*).
- The result of the diagnostic A/D conversion is stored in ADCAnDGCR.

Diagnosis procedure

1. Switch the power of the ADCA on by setting ADCAnCTL1.ADCAnGPS = 1.
2. Configure CG0 and the A/D conversion as follows:
   – Ensure that ADCAnCG0.ADCAnDIAG is set to 1 to enable the diagnostic A/D conversion of the reference voltage.
   For example, write 8000 0007_H to first convert the analog input voltages of CH0, CH1, and CH2, and then the reference voltage ADDIAGOUT for diagnostic purposes.
   – Set ADCAnI0C0.ADCAnCG0IDG = 1 to generate the A/D conversion end interrupt INTADCAnT0 on finishing the diagnostic A/D conversion.
3. Specify the reference voltage signal ADDIAGOUT via ADCAnDGCTL0.ADCAnPSEL[2:0].
   For example, set ADCAnDGCTL0.ADCAnPSEL[2:0] to 010_B to apply the reference voltage 1/2 AVDD.
4. Enable the ADCA by setting ADCAnCTL0.ADCAnCE = 1.
5. Generate software or hardware start triggers to start the A/D conversion.
6. When the A/D conversion end interrupt INTADCAnT0 is generated, read the A/D conversion results of the diagnostic A/D conversion in ADCAnDGCR.

ADDIAGOUT change ADCAnDGCTL0.ADCAnPSEL[2:0] can be written - and thus the reference voltage ADDIAGOUT can be changed - even during A/D conversion, as shown in the diagram below.

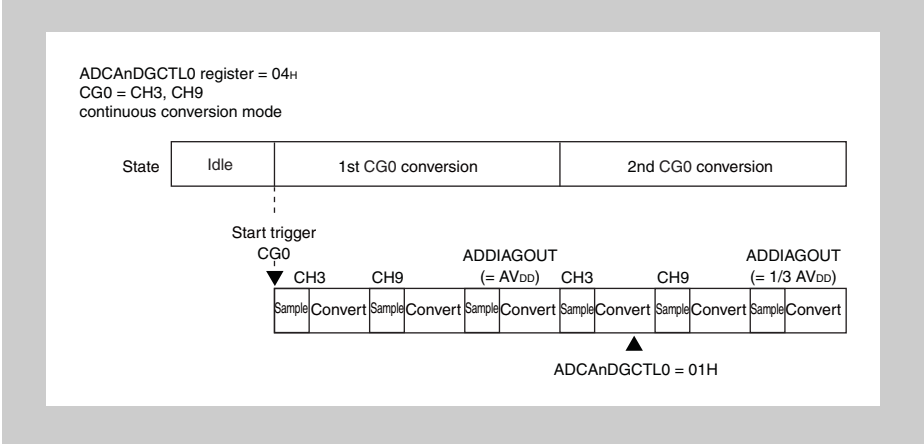


Figure 34-20 ADDIAGOUT change during A/D conversion

The value set with ADCAnDGCTL0.ADCAnPSEL[2:0] is applied upon completion of the conversion of the current channel. Therefore, set the reference voltage of the next diagnosis A/D conversion before that diagnosis A/D conversion starts.

(2) Diagnosis of the channel multiplexer

Table 34-16 Channel assignment of reference voltages

Reference voltage	Assigned channels
DIAGOUT0	21, 18, 15, 12, 9, 6, 3, 0
DIAGOUT1	22, 19, 16, 13, 10, 7, 4, 1
DIAGOUT2	23, 20, 17, 14, 11, 8, 5, 2

To test the channel multiplexer, different reference voltages can be input to the channels. This ensures that during the test, different voltages are applied and not the same voltage by accident.

Note The channel multiplexer check is available for CG0 only. At least one channel of DIAGOUT0, DIAGOUT1, and DIAGOUT2 should be assigned to CG0, such as CH0, CH1, and CH2 as in the example below.

Diagnosis procedure

1. Switch the power of the ADCA on by setting ADCAnCTL1.ADCAnGPS to 1.
2. Specify the channels of CG0 in ADCAnCG0.
   For example, write 0000 0007_H to use CH0, CH1, and CH2 for diagnosis.
3. Configure ADCAnI0C0 to generate the A/D conversion end interrupt INTADCAnTi at the end of all A/D conversions for CG0.
4. Configure the other A/D conversion settings as desired.
5. Specify the reference voltages in ADCAnDGCTL0.ADCAnPSEL[2:0].
   For example, set ADCAnDGCTL0.ADCAnPSEL[2:0] = 010_B to use the following reference voltages:
   – DIAGOUT0 = 1/2 AVDD
   – DIAGOUT1 = 2/3 AVDD
   – DIAGOUT2 = 1/3 AVDD

– Configure to which channels of CG0 a reference voltage is applied (instead of the analog input voltage ADCAnIm) in ADCAnDGCTL1. For example, set ADCAnDGCTL1 to 0000 0006_H:
– Analog input voltage will be applied to CH0.
– DIAGOUT1 ($2/3\ AV_{DD)}$ will be applied to CH1.
– DIAGOUT2 ($1/3\ AV_{DD)}$ will be applied to CH2.

6. Enable the ADCA by setting ADCAnCTL0.ADCAnCE = 1.
7. Generate software or hardware start triggers to start the A/D conversion.
8. When receiving the A/D conversion end interrupt INTADCAnT0, read the A/D conversion results of CG0.

Caution If the ADCA is enabled (ADCAnCTL0.ADCAnCE = 1), changing the setting of the ADCAnDGCTL1 register is prohibited.

(3) Diagnosis for open pins

Open input pins result in erroneous A/D conversion results. Especially in safety-relevant applications it might be necessary to check for open pins regularly.

The internal pull-down resistors can be connected to test the analog inputs ADCAnIm.

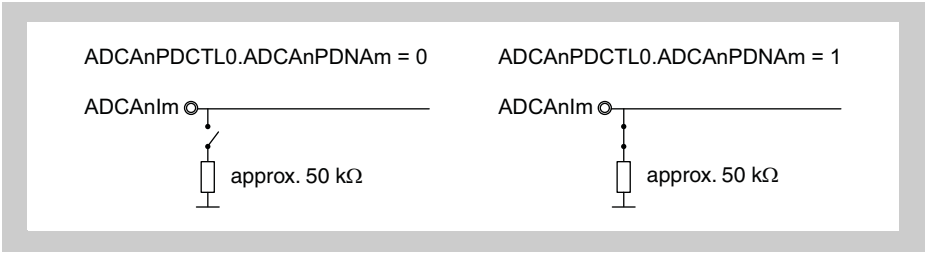


Figure 34-21 Internal pull-down resistors

When an internal pull-down resistor is connected to the ADCAnIm analog input (ADCAnPDCTL0.ADCAnPDNAm = 1) and ADCAnIm is open, the A/D conversion result declines to almost 0 V.

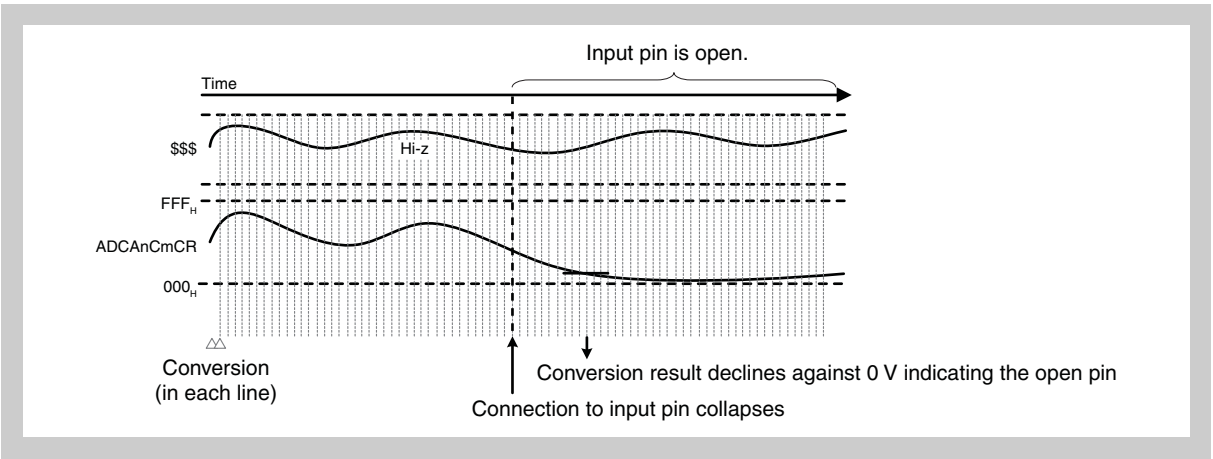


Figure 34-22 Detection of an open input pin

Notes

1. This diagnosis gives valid results only if the ADCAnIm input voltage has a certain minimum input voltage (refer to the Electrical Target Specification).
2. The internal pull-down resistors must not be connected during normal A/D conversion operation. Connecting internal pull-down resistors may lead to a drop in the input voltage and may result in imprecise A/D conversion results.

Diagnosis procedure

To diagnose open input pins:

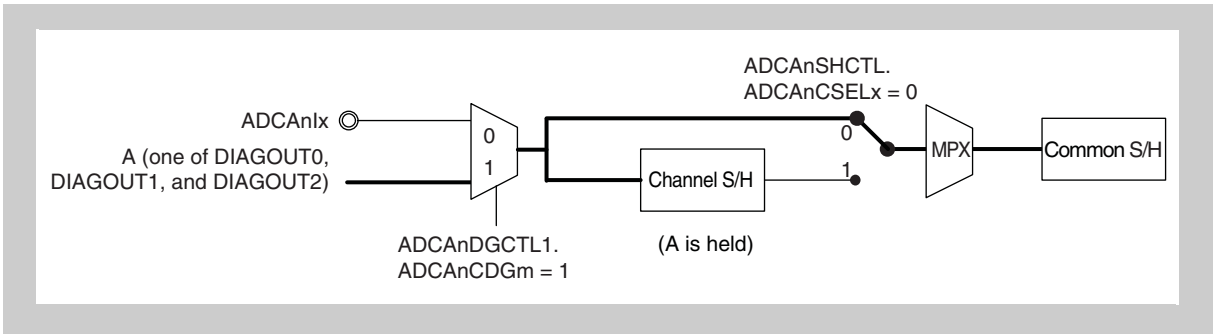
1. Configure the CG and A/D conversion settings as usual.
2. Configure the internal pull-down resistors to be connected in ADCAnPDCTL0.
3. Start A/D conversions multiple times.
4. Monitor the channel's A/D conversion results and check if any result declines to almost 0 V.

(4) Channel S&H circuit diagnosis

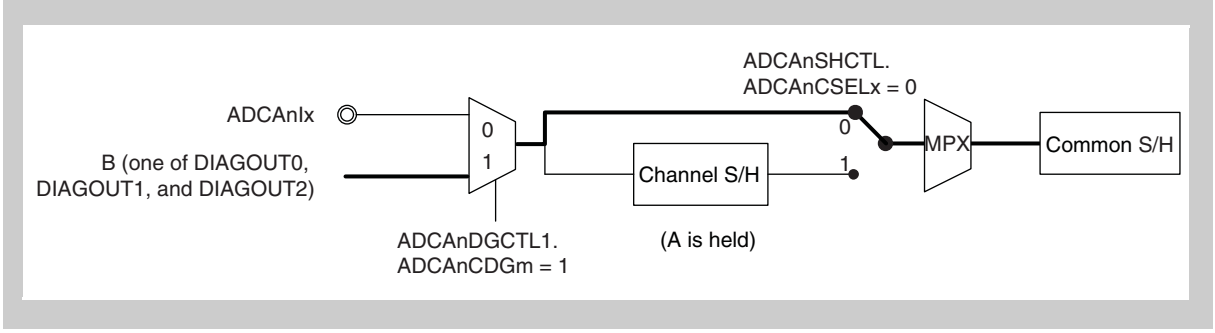
Refer to *34.5.13 "Channel S&H function"* for details about the channel S&H function.

The channel S&H circuit can be diagnosed. An overview is given below.

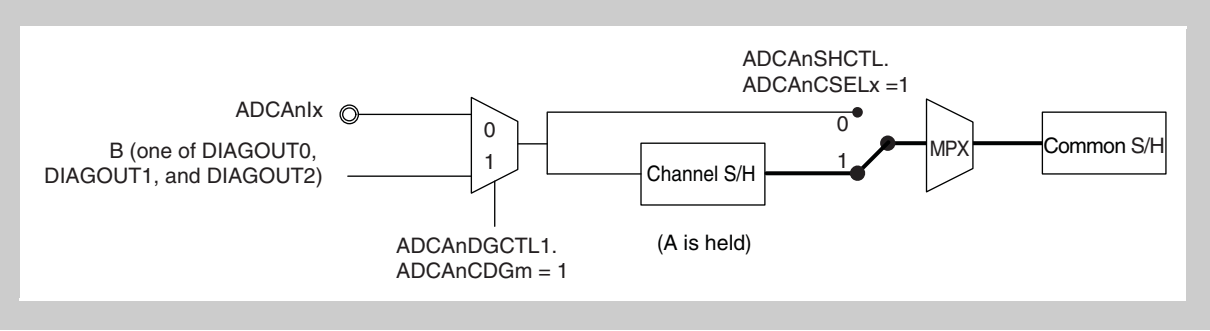
1. Use one of the reference voltage signals DIAGOUT0 to DIAGOUT2.
2. Set voltage "A" for the selected reference voltage signal (DIAGOUT0, DIAGOUT1, or DIAGOUT2). Conversion is performed by holding "A" in the channel S&H circuit, without using the channel S&H circuit.



3. Set voltage "B" for DIAGOUT0, DIAGOUT1, or DIAGOUT2. Conversion is performed without using the channel S&H circuit. The conversion result becomes "B".
   The channel S&H circuit continues to hold "A".



4. Conversion is performed using the channel S&H circuit.
   The channel S&H circuit continues to hold "A". The conversion result becomes “A”.



5. The result is as follows.
   – The value of the first conversion result (3) is “B”.
   – The value of the second conversion result (4) is “A”.

The diagnosis flow of the channel S&H circuit is shown below.

Initial settings

The following example uses the ADCAnI1 pin.

- Set ADCAnCTL1.ADCAnGPS to 1.
- Specify software trigger for the start trigger, and one-shot conversion mode (number of repetitions: 1) for the A/D conversion mode.
- Set the ADCAnCG0 register to 0000 0002_H. (ADCAnI1 with channel S&H circuit is selected.)
- Set the ADCAnIOC0 register to 0000 0002_H to generate the A/D conversion end interrupt INTADCAnT0. (An interrupt must be generated upon conversion end.)
- Set ADCAnDGCTL0.ADCAnPSEL[2:0] to 001_B. ($1/3AV_{DD}$ is selected for reference voltage DIAGOUT1.)
- Set ADCAnDGCTL1.ADCAnCDG01 to 1.
- Set ADCAnCTL0.ADCAnCE to 1.

Operation flow

[Step 1]

Input the software start trigger for CG0 (first time). ($1/3AV_{DD}$ is held in the channel S&H circuit.

Next, change the ADCAnPSEL[2:0] to 010_B. (Following completion of the first A/D conversion, $2/3AV_{DD}$ is selected for the DIAGOUT1 reference voltage.

Input the software start trigger for CG0 (second time). (The start trigger is maintained.)

The procedure up to this point must be performed until the first A/D conversion ends.

[Step 2]

The first A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

Change the ADCAnCSEL01 to 1. (The voltage held in the channel S&H circuit is converted at the next A/D conversion.)

Input the software start trigger for CG0 (third time). (The start trigger is maintained.)

The procedure up to this point must be performed until the second A/D conversion ends.

[Step 3]

The second A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

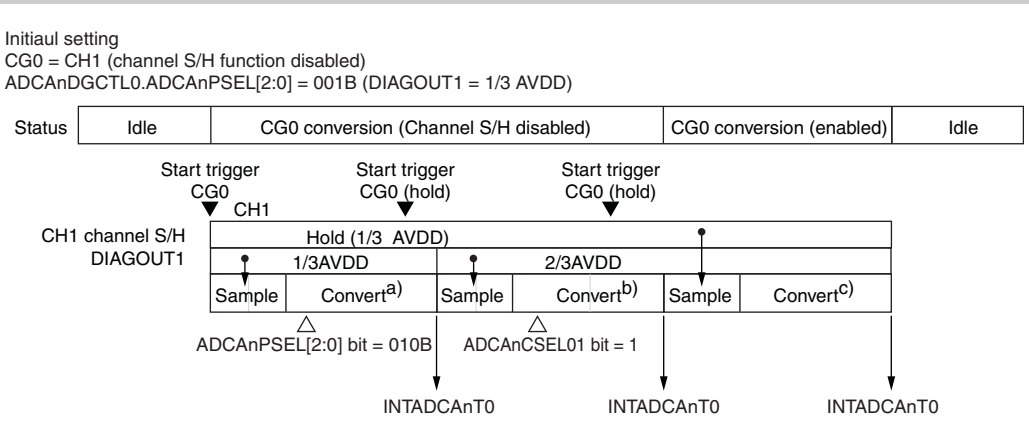
Read the ADCAnC01CR register and check whether it stores the expected value ($2/3AV_{DD}$).

[Step 4]

The third A/D conversion ends and the A/D conversion end interrupt INTADCAnT0 is generated.

Read the ADCAnC01CR register and check whether it stores the expected value ($1/3AV_{DD}$).

A timing example of the channel S&H circuit diagnosis is shown below.



a) DIAGOUT1 is converted. The conversion result is 1/3 AV_{DD}.
b) DIAGOUT1 is converted. The conversion result is 2/3 AV_{DD}.
c) The input voltage held in channel S&H is converted. The conversion result is 1/3 AV_{DD}.

34.5.15 Discharge function

If required, the internal capacitor of the common Sample & Hold circuit can be discharged prior to every conversion. This ensures that the capacitor is always empty before the new sample value is stored.

Note Using the discharge function increases the total conversion time by 1 ADCATCLK cycle (see *34.5.9 “Resolution, sampling and conversion times” on page 2779*).

Configuration The discharge function is enabled by setting ADCAnCTL1.ADCAnDISC to 1.

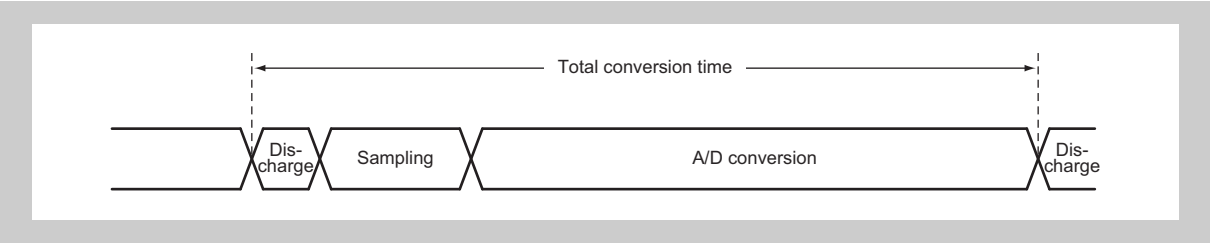


Figure 34-23 Timing when discharge function is enabled

34.5.16 Buffer amplifier function

If required, the analogue input signal can be connected to an internal buffer amplifier. This way the load of the external signal source is decreased and signal drops during discharge of the sample capacitor with high-impedance signal sources can be minimized.

The buffer amplifier function is enabled by setting ADCAnCTL1.ADCAnBPC to 1.

Note Using the buffer amplifier function increases the total conversion time by 4 ADCATCLK cycles (see *34.5.9 “Resolution, sampling and conversion times” on page 2779*).

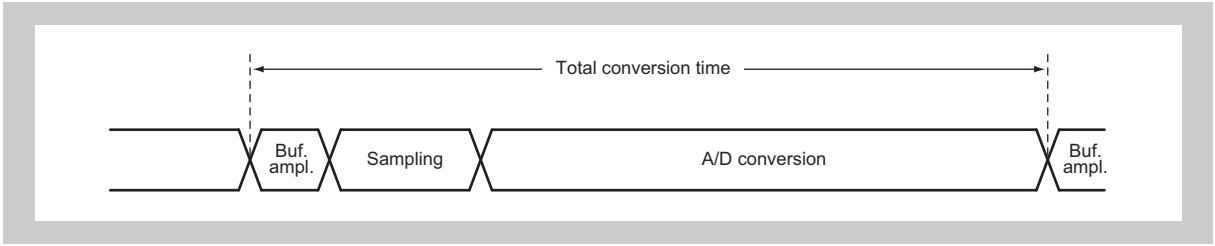


Figure 34-24 Timing when buffer amplifier function is enabled

34.5.17 Stabilization control

The A/D converter needs time for stabilization purposes in the following cases:

- The A/D converter has been switched on (ADCAnCTL1.ADCAnGPS = 1)
- The stand-by mode has been terminated.

A start trigger is accepted during the stabilization time, but the conversion does not start before the stabilization time has elapsed.

In order to secure the minimum stabilization time, the stabilization time counter ADCAnCNT must be set (refer to the Electrical Target Specification).

34.6 Registers

This section contains a description of all the registers of the ADCA.

34.6.1 ADCA registers overview

The ADCA is controlled and operated by the registers in the following table.

- Where there is one register per channel, this is indicated by an “m”. The number “m” of available channels are specified in the first section of this chapter under the key word “Channel index m”.
- Where there is one register per CG, this is indicated by an “i” (i = 0 to 2).

Table 34-17 ADCA registers overview (1/2)

Register name	Shortcut	Address
Control registers		
A/D converter mode control register 0	ADCAnCTL0	<ADCAn_base> + 100_H
A/D converter mode control register 1	ADCAnCTL1	<ADCAn_base> + 104_H
A/D converter CG register i	ADCAnCGi	<ADCAn_base> + i x 4_H
A/D converter interrupt control register i	ADCAnIOCi	<ADCAn_base> + C_H + i x 4_H
A/D converter trigger select control register i	ADCAnTSELi	<ADCAn_base> + 108_H + i x 4_H
A/D converter stabilization counter	ADCAnCNT	<ADCAn_base> + 114_H
A/D converter channel S&H control register	ADCAnSHCTL	<ADCAn_base> + 118_H
Conversion status registers		
A/D converter overwrite error flag register	ADCAnSTR1	<ADCAn_base> + 28_H
ADCAnSTR1 flag clear register	ADCAnSTC1	<ADCAn_base> + 34_H
A/D converter status flag register 2	ADCAnSTR2	<ADCAn_base> + $2C_H$
ADCAnSTR2 flag clear register	ADCAnSTC2	<ADCAn_base> + 38_H
S/W trigger registers		
A/D converter S/W trigger register i	ADCAnTRGi	<ADCAn_base> + $A4_H$ + i x 4_H
A/D converter S/W trigger register 3	ADCAnTRG3	<ADCAn_base> + $B0_H$
A/D converter S/W trigger register 4+i	ADCAnTRG4 + i	<ADCAn_base> + $B4_H$ + i x 4_H
A/D converter S/W trigger register 7	ADCAnTRG7	<ADCAn_base> + $C0_H$
A/D conversion result registers		
A/D converter latest conversion result register	ADCAnLCR	<ADCAn_base> + $A0_H$
A/D converter conversion result register m	ADCAnCmCR	<ADCAn_base> + $3C_H$ + m x 4_H
A/D converter CGi buffer register i	ADCAnDBiCR	<ADCAn_base> + $C4_H$ + i x 4_H
A/D converter CGi buffer result register i	ADCAnDBiCRL	<ADCAn_base> + $D0_H$ + i x 4_H
Diagnostic conversion result register	ADCAnDGCR	<ADCAn_base> + $9C_H$
A/D conversion result upper/lower limit comparison registers		
A/D converter result check register	ADCAnCTL2	<ADCAn_base> + 18_H
A/D converter result check (upper limit)	ADCAnUL	<ADCAn_base> + $1C_H$
A/D converter result check (lower limit)	ADCAnLL	<ADCAn_base> + 20_H
A/D converter result check error flag	ADCAnSTR0	<ADCAn_base> + 24_H
ADCAnSTR0 flag clear register	ADCAnSTC0	<ADCAn_base> + 30_H

Table 34-17 ADCA registers overview (2/2)

Register name	Shortcut	Address
Diagnose functions control registers		
Self-diagnosis function control register 0	ADCAnDGCTL0	<ADCAn_base> + DC_H
Self-diagnosis function control register 1	ADCAnDGCTL1	<ADCAn_base> + $11C_H$
Pull down resistance control register 0	ADCAnPDCTL0	<ADCAn_base> + 120_H
Emulation register		
Emulation register	ADCAnEMU	<ADCAn_base> + 128_H

<ADCAn_base> The base addresses <ADCAn_base> of the ADCAn is defined in the first section of this chapter under the key word “Register addresses”.

34.6.2 Control registers details

(1) ADCAnCTL0 – A/D converter mode control register 0

This register enables/disables the A/D converter. Additionally, it specifies the number of repetitions for one-shot conversion mode, and also whether to generate an error interrupt request if the A/D conversion result is overwritten before it is read.

Access This register can be read/written in16-bit units.

Address <ADCAn_base> + 100_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	ADCAn OEM4	ADCAn OEM3	ADCAn OEM2	ADCAn OEM1	ADCAn OEM0	ADCAn CE	0	ADCAn SCT2[1:0]		ADCAn SCT1[1:0]		ADCAn SCT0[1:0]	
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-18 ADCAnCTL0 register contents (1/2)

Bit position	Bit name	Function
12	ADCAn OEM4	Specifies whether to generate the INTADCAnERR error interrupt when an A/D conversion result is overwritten in ADCAnLCR before it was read. 0: generate INTADCAnERR if the A/D conversion result is overwritten 1: do not generate INTADCAnERR Refer to *34.5.12 "Result check functions" on page 2784* for details.
11	ADCAn OEM3	Specifies whether to generate the INTADCAnERR error interrupt when an A/D conversion result is overwritten in ADCAnDB2CR before it was read. 0: generate INTADCAnERR if the A/D conversion result is overwritten 1: do not generate INTADCAnERR Refer to *34.5.12 "Result check functions" on page 2784* for details.
10	ADCAn OEM2	Specifies whether to generate the INTADCAnERR error interrupt when an A/D conversion result is overwritten in ADCAnDB1CR before it was read. 0: generate INTADCAnERR if the A/D conversion result is overwritten 1: do not generate INTADCAnERR Refer to *34.5.12 "Result check functions" on page 2784* for details.
9	ADCAn OEM1	Specifies whether to generate the INTADCAnERR error interrupt when an A/D conversion result is overwritten in ADCAnDB0CR before it was read. 0: generate INTADCAnERR if the A/D conversion result is overwritten 1: do not generate INTADCAnERR Refer to *34.5.12 "Result check functions" on page 2784* for details.
8	ADCAn OEM0	Specifies whether to generate the INTADCAnERR error interrupt when an A/D conversion result is overwritten in ADCAnCmCR before it was read. 0: generate INTADCAnERR if the A/D conversion result is overwritten 1: do not generate INTADCAnERR Refer to *34.5.12 "Result check functions" on page 2784* for details.
7	ADCAn CE	This bit enables/disables the A/D converter. 0: disable A/D converter 1: enable A/D converter Note that A/D conversion only starts on hardware trigger or software trigger, when ADCAnCTL0.ADCAnCE = 1. Note also that the A/D converter needs time to stabilize after it has been enabled. A start trigger is accepted even immediately after power-on. After the values of the stabilization counter ADCAnCNT changes to 00_H, the A/D conversion starts.

Table 34-18 ADCAnCTL0 register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>5 to 4</td><td>ADCAn SCT2[1:0]</td><td>Number of repetitions of scan list conversion for CG2 in one-shot conversion mode.
<table>
<tr><th>ADCAn SCT21</th><th>ADCAn SCT20</th><th>Number of repetitions of scan list conversion for CG2</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>0</td><td>3</td></tr>
<tr><td>1</td><td>1</td><td>4</td></tr>
</table></td></tr>
<tr><td>3 to 2</td><td>ADCAn SCT1[1:0]</td><td>Number of repetitions of scan list conversion for CG1 in one-shot conversion mode.
<table>
<tr><th>ADCAn SCT11</th><th>ADCAn SCT10</th><th>Number of repetitions of scan list conversion for CG1</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>0</td><td>3</td></tr>
<tr><td>1</td><td>1</td><td>4</td></tr>
</table></td></tr>
<tr><td>1 to 0</td><td>ADCAn SCT0[1:0]</td><td>Number of repetitions of scan list conversion for CG0 in one-shot conversion mode.
<table>
<tr><th>ADCAn SCT01</th><th>ADCAn SCT00</th><th>Number of repetitions of scan list conversion for CG0</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>2</td></tr>
<tr><td>1</td><td>0</td><td>3</td></tr>
<tr><td>1</td><td>1</td><td>4</td></tr>
</table></td></tr>
</table>

(2) ADCAnCTL1 – A/D converter mode control register 1

This register specifies the conversion mode and controls the conversion operations.

Access This register can be read/written in 32-bit units.

Address <ADCAn_base> + 104_H

Initial Value $0100\ 0008_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	ADCAn T2ETS0	0	ADCAn T1ETS0	0	ADCAn T0ETS0	0	ADCAn CRAC	0	0	ADCAn MD1	ADCAn MD0	0	0	ADCAn DISC	ADCAn RCL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAn CTYP	0	0	0	ADCAn FR[3:0]				0	ADCAn TRM2	ADCAn TRM1	ADCAn TRM0	ADCAn BPC	0	0	ADCAn GPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-19 ADCAnCTL1 register contents (1/3)

Bit position	Bit name	Function
31	ADCAn T2ETS0	Specifies the valid edge of the H/W trigger ADCAnTTRGi. 0: no edge selected (no trigger accepted) 1: rising edge selected For further information refer to the section *"H/W trigger expansion"* in this chapter.
29	ADCAn T1ETS0	
27	ADCAn T0ETS0	
24	ADCAn CRAC	Specifies the alignment of the A/D and diagnosis conversion result: 0: right-aligned 1: left-aligned
21	ADCAn MD1	Specifies the A/D conversion start trigger for all CGs. 0: software trigger 1: hardware trigger and software trigger This configuration is valid for all CGs. The A/D converter only detects triggers when the A/D converter is enabled. Refer to *34.5.5 "Starting A/D conversion (start trigger modes)" on page 2774* for details.
20	ADCAn MD0	Specifies the A/D conversion mode for CG0: 0: one-shot conversion mode Number of repetitions of scan list conversion is configured in ADCAnCTL0.ADCAnSCTi[1:0] for each CG individually. 1: continuous conversion mode This configuration applies to the A/D conversion of CG0 only. CG1 and CG2 are always operated in one-shot conversion mode. Refer to *34.5.4 "A/D conversion modes" on page 2771* for details.
17	ADCAn DISC	Enables/disables the discharge function: 0: disable 1: enable Refer to *34.5.15 "Discharge function" on page 2800* for details.
16	ADCAn RCL	Specifies whether the A/D conversion results ADCAnCmCR and ADCAnDBiCR are retained after they are read. 0: A/D conversion results are retained until they are overwritten by the next A/D conversion results 1: A/D conversion results are cleared after they are read

Table 34-19 ADCAnCTL1 register contents (2/3)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15</td><td>ADCAn CTYP</td><td>Specifies the resolution mode:
0: 12-bit resolution
1: 10-bit resolution

Caution The resolution supported by this microcontroller is specified in the first section of this chapter under the key word “A/D Converter resolution”. In case this microcontroller supports only 10-bit resolution, ADCAnCTYP must be set to 1.</td></tr>
<tr><td>11 to 8</td><td>ADCAn FR[3:0]</td><td>Specifies the ADCA system clock ADCATCLK
<table>
<tr><th>ADCAnFR[3:0]</th><th>ADCA system clock ADCATCLK</th></tr>
<tr><td>0000_B</td><td>PCLK / 2</td></tr>
<tr><td>0001_B</td><td>PCLK / 3</td></tr>
<tr><td>0010_B</td><td>PCLK / 4</td></tr>
<tr><td>0011v</td><td>PCLK / 5</td></tr>
<tr><td>0100_B</td><td>PCLK / 6</td></tr>
<tr><td>0110_B</td><td>PCLK / 8</td></tr>
<tr><td>1000_B</td><td>PCLK / 10</td></tr>
<tr><td>1010_B</td><td>PCLK / 12</td></tr>
<tr><td>1100_B</td><td>PCLK / 14</td></tr>
<tr><td>1110_B</td><td>PCLK / 16</td></tr>
<tr><td>all others</td><td>setting prohibited</td></tr>
</table>
Caution The ADCA system clock ADCATCLK must not exceed a certain limit, refer to the Electrical Target Specification for details.</td></tr>
<tr><td>6</td><td>ADCAn TRM2</td><td>Specifies the behaviour of CG2 conversion when transition to ADCHALT mode is requested.[a]
0: Halt the current A/D conversion of CG2 immediately, and enter ADCHALT mode.
1: Finish conversion of the currently converted channel of CG2, halt A/D conversion of CG2, and enter ADCHALT mode.
A/D conversion of CG2 is continued as soon as ADCHALT mode has been terminated.</td></tr>
<tr><td>5</td><td>ADCAn TRM1</td><td>Specifies the behaviour of CG1 conversion when a start trigger for a CG2 A/D conversion occurs or when transition to ADCHALT mode is requested.[a]
0: Halt the current A/D conversion of CG1 immediately, and start the A/D conversion of CG2 or enter ADCHALT mode respectively.
1: Finish conversion of the currently converted channel of CG1, halt A/D conversion of CG1, and start the A/D conversion of CG2 or enter ADCHALT mode respectively.
A/D conversion of CG1 is continued as soon as all pending A/D conversions of CG2 have been completed or the ADCHALT mode has been terminated.</td></tr>
</table>

Table 34-19 ADCAnCTL1 register contents (3/3)

Bit position	Bit name	Function
4	ADCAn TRM0	Specifies the behaviour of CG0 conversion when a start trigger for a CG1 or CG2 A/D conversion occurs or when transition to ADCHALT mode is requested.[a] 0: Halt the current A/D conversion of CG0 immediately, and start the A/D conversion of CG2 or CG1 or enter ADCHALT mode respectively. 1: Finish conversion of the currently converted channel of CG0, halt A/D conversion of CG0, and start the A/D conversion of CG2 or CG1 or enter ADCHALT mode respectively. A/D conversion of CG0 is continued as soon as all pending A/D conversions of CG2 or CG1 have been completed or the ADCHALT mode has been terminated.
3	ADCAn BPC	Enables/disables the buffer amplifier function: 0: disable 1: enable Refer to *34.5.16 “Buffer amplifier function” on page 2801* for details.
0	ADCAn GPS	Switches the power of the ADCA off/on: 0: power off 1: power on The A/D converter needs a stabilization time after A/D power on (see *34.5.17 “Stabilization control” on page 2801*).

a) The behaviour follows the priority scheme: ADCHALT > CG2 > CG1 > CG0. Refer to 1 “Order of A/D conversion” on page 7 for details.

(3) ADCAnCGi – A/D converter channel group register i

This register creates the scan list for each CG. The channels set to the scan list of CGi are converted in ascending order, i.e. starting from the lowest channel number.
For details, refer to *34.5.3 “Channels and channel groups” on page 2769* .

Diagnosis conversion

Additionally the diagnosis A/D conversion channel can be added to the CG0 scan list via the ADCAnCG0 register.
For details, refer to *34.5.14 “Self-diagnosis functions” on page 2793* .

Multiple assignments

A certain analog input channel ADCAnIm can must not be assigned to CG1 and CG2.
However it is possible to share channels between

- CG0 and CG1 or
- CG0 and CG2.

Register update

The ADCAnCGi registers can be written at any time, even when the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1). A new ADCAnCGi value takes effect

- after the current A/D conversion of CGi has been completed.
- when CGi is not currently undergoing A/D conversion, the new value becomes effective immediately.
- when CGi is currently undergoing A/D conversion, the new value becomes effective upon completion of the scan list conversion of CGi currently being executed.
- when the stop trigger bit (ADCAnTRG4+i.ADCAnSPi) of CGi is set, the new value becomes effective when A/D conversion is stopped.

The ADCAnCGi registers can be overwritten at any time. The last written value is valid to become effective.

Access This register can be read/written in 32-bit units.

Address <ADCAn_base> + i x 4_H

Initial Value 0000 0000_H

ADCAnCG0:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAn DIAG	0	0	0	0	0	0	0	ADCAnCG0S[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG0S[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ADCAnCG1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCG1S[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG1S[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ADCAnCG2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCG2S[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG2S[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-20 ADCAnCGi register contents

Bit position	Bit name	Function
31	ADCAn DIAG	Enables/disables the diagnostic A/D conversion of the reference voltage ADDIAGOUT at the end of the A/D conversion of CG0: 0: disable A/D conversion of ADDIAGOUT 1: convert ADDIAGOUT This bit is only available in ADCAnCG0. Refer to *34.5.14 "Self-diagnosis functions" on page 2793* for details.
23 to 00	ADCAn CGiS[23:00]	Specifies the analog inputs to be converted for CGi: 0: do not convert analog input ADCAnIm 1: convert analog input ADCAnIm

(4) ADCAnIOCi – A/D converter interrupt control register i

The A/D conversion end interrupt INTADCAnTi can be generated when the A/D conversion of a certain channel has been completed.

This register specifies the channel in each CGi scan list, for which INTADCAnTi is generated on A/D conversion completion.

If ADCAnIOCi = 0000 0000_H, INTADCAnTi is automatically generated at the completion of A/D conversion of CGi scan list.

Register update

The ADCAnIOCi registers can be written at any time, even when the A/D converter is enabled (ADCAnCTL0.ADCAnCE = 1). A new ADCAnIOCi value takes effect concurrently with a new scan list set up, i.e. when ADCAnCGi was written and becomes effective.
Thus always write ADCAnIOCi before ADCAnCGi if you want to change the interrupt generation for a CG.
The ADCAnIOCi registers can be overwritten at any time. The last written value is valid to become effective.

Refer also to the key word “Register update” in ADCAnCGi register description.

Access

This register can be read/written in 32-bit units.
The new value takes effect after the current A/D conversion of CGi has been completed.

Address

<ADCAn_base> + $0C_H$ + i x 4_H

Initial Value

0000 0000_H

ADCAnIOC0:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ADCAnCG0IDG	0	0	0	0	0	0	0	ADCAnCG0I[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG0I[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ADCAnIOC1:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCG1I[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG1I[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ADCAnIOC2:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCG2I[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCG2I[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-21 ADCAnIOCi register contents

Bit position	Bit name	Function
31	ADCAn CG0IDG	Specifies whether INTADCAnT0 is generated on completion of the A/D conversion of the reference voltage when the diagnostic mode is enabled for CG0 (ADCAnCG0.ADCAnDIAG = 1): 0: do not generate INTADCAnT0 1: generate INTADCAnT0 This bit is only available in ADCAnIOC0. Refer to *34.5.14 "Self-diagnosis functions" on page 2793* for details.
23 to 00	ADCAn CGil[23:00]	Specifies whether the interrupt INTADCAnTi is generated on A/D conversion completion of channel m: 0: Do not generate INTADCAnTi 1: Generate INTADCAnTi **Note:** ADCAnCGil bits, which are assigned to not available channels, must be set to 0.

(5) ADCAnCNT – A/D converter stabilization counter

This register specifies the stabilization time.

Access This register can be read/written in 8-bit units.

Address <ADCAn_base> + 114_H

Initial Value 00_H

7	6	5	4	3	2	1	0
ADCAnCNT[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-22 ADCAnCNT register contents

Bit position	Bit name	Function
7 to 0	ADCAn CNT[7:0]	Specifies the stabilization counter: Stabilization time = ADCAnCNT[7:0] * PCLK clock cycles

(6) ADCAnSHCTL – A/D converter channel S&H control register

This register specifies the channels that will be sampled simultaneously.

Additionally this register can be used to switch the power of the channel-related Sample & Hold circuits on/off, for example to save power when simultaneous sampling is not required.

The configuration is valid for all CGs.

Access This register can be read/written in 32-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + 118_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	ADCAnCPS[05:00]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	ADCAnCSEL[05:00]					
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-23 ADCAnSHCTL register contents

Bit position	Bit name	Function
21 to 16	ADCAn CPSm (m = 00 to 05)	Enables/disables the channel-related Sample & Hold circuit for channel m: 0: disable Sample & Hold circuit (for power save mode) 1: enable Sample & Hold circuit
5 to 0	ADCAn CSELm (m = 00 to 05)	Specifies whether the value held in the Sample & Hold circuit of channel m is converted: 0: do not use value Sample & Hold circuit 1: use value of Sample & Hold circuit

Note ADCAnCSELm take effect only when the power of the corresponding Sample & Hold circuit is switched on (ADCAnCPSm = 1).

(7) ADCAnTSELi – A/D converter trigger select control register i

This register specifies the input signals to be used in combination with hardware start trigger signals ADCAnTTRGi.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + 108_H + i x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnTiSEL[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-24 ADCAnTSELi register contents

Bit position	Bit name	Function
15 to 0	ADCAn TiSEL[15:00]	Specifies whether the corresponding input signal is to be used as hardware start trigger 0: do not use as hardware start trigger 1: use as hardware start trigger

Note For details about the hardware start trigger function expansion refer to “H/W trigger expansion” in the first section of this chapter.

34.6.3 Conversion status registers

(1) ADCAnSTR1 – A/D converter overwrite error flag register

This register indicates for every channel whether the latest A/D conversion result has been overwritten before it was read.

Access This register can be read in 32-bit units.

Address <ADCAn_base> + 28_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnOWE[23:16]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnOWE[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 34-25 ADCAnSTR1 register contents

Bit position	Bit name	Function
23 to 0	ADCAn OWE[23:00]	Indicates whether the A/D conversion result of channel m has been overwritten before it was read: 0: not overwritten 1: overwritten This error flag is cleared by setting ADCAnSTR1.ADCAnOWECm to 1. All flags are also cleared by enabling and disabling the ADCA (ADCAn.ADCAnCE= 0 and then ADCAn.ADCAnCE= 1).

Note ADCAnSTR1.ADCAnOWEm is mirrored by the following overwrite error flags:

- Error flag in register with the latest A/D conversion result of CHm (ADCAnCmCR.ADCAnCmER1)

(2) ADCAnSTC1 – ADCAnSTR1 flag clear register

This register is the clear control register of ADCAnSTR1.

Access This register can be written in 32-bit units.

Address <ADCAn_base> + 34_H

Initial Value Reading this register returns always 0000 $0000_{H.}$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnOWEC[23:16]							
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnOWEC[15:00]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 34-26 ADCAnSTC1 register contents

Bit position	Bit name	Function
23 to 0	ADCAn OWEC[23:00]	Clear overwrite error flags in ADCAnSTR1: 0: no function 1: clears the corresponding ADCAnSTR1.ADCAnOWEm

(3) ADCAnSTR2 – A/D converter status flag register 2

This register indicates the current conversion status.

Access This register can be read in 16-bit units.

Address <ADCAn_base> + $2C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	ADAAn RQT3	ADAAn RQ2	ADAAn RQ1	ADAAn RQ0	0	0	0	0	ADAAn ST3	ADAAn ST2	ADAAn ST1	ADAAn ST0
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 34-27 ADCAnSTR2 register contents

Bit position	Bit name	Function
11	ADCAn RQ3	Indicates whether a ADCHALT request is pending: 0: ADCHALT request is not pending 1: ADCHALT request is pending
10	ADAAn RQ2	Indicates whether A/D conversion request for CG2 is pending: 0: A/D conversion request for CG2 is not pending 1: A/D conversion request for CG2 is pending
9	ADAAn RQ1	Indicates whether A/D conversion request for CG1 is pending: 0: A/D conversion request for CG1 is not pending 1: A/D conversion request for CG1 is pending
8	ADAAn RQ0	Indicates whether A/D conversion request for CG0 is pending: 0: A/D conversion request for CG0 is not pending 1: A/D conversion request for CG0 is pending
3	ADCAn ST3	Indicates whether the A/D conversion is currently in the ADCHALT state due to a software trigger (ADCAnTRG3.ADCAnSTT3). 0: not in ADCHALT state 1: in ADCHALT state This bit is cleared when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).
2	ADAAn ST2	Indicates whether A/D conversion of CG2 is currently performed: 0: A/D conversion is not currently performed 1: A/D conversion is currently performed This bit is cleared when the A/D converter is disabled (ADAAnCTL0.ADAAnCE = 0).
1	ADAAn ST1	Indicates whether A/D conversion of CG1 is currently performed: 0: A/D conversion is not currently performed (including halt caused by A/D conversion of CG with higher priority) 1: A/D conversion is currently performed This bit is cleared when the A/D converter is disabled (ADAAnCTL0.ADAAnCE = 0).
0	ADAAn ST0	Indicates whether A/D conversion of CG0 is currently performed: 0: A/D conversion is not currently performed (including halt caused by A/D conversion of CG with higher priority) 1: A/D conversion is currently performed This bit is cleared when the A/D converter is disabled (ADAAnCTL0.ADAAnCE = 0).

(4) ADCAnSTC2 – A/D converter status flag clear register 2

This register is used to clear the overwrite and result check status flags of ADCAnLCR and ADCAnDBiCR.

Access This register can be written in 8-bit units.

Address <ADCAn_base> + 38_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
ADCAn LERC1	ADCAn LERC0	ADCAn DB2ERC1	ADCAn DB2ERC0	ADCAn DB1ERC1	ADCAn DB1ERC0	ADCAn DB0ERC1	ADCAn DB0ERC0
W	W	W	W	W	W	W	W

Table 34-28 ADCAnSTC2 register contents

Bit position	Bit name	Function
7	ADCAn LERC1	Clears the overwrite flag ADCAnLCR.ADCAnLER1: 0: no function 1: ADCAnLCR.ADCAnLER1 is cleared
6	ADCAn LERC0	Clears the result check error flag ADCAnLCR.ADCAnLER0: 0: no function 1: ADCAnLCR.ADCAnLER0 is cleared
5, 3, 1	ADCAn DBiERC1	Clears the overwrite flag ADCAnDBiCR.ADCAnDBiER1: 0: no function 1: ADCAnDBiCR.ADCAnDBiER1 is cleared
4, 2, 0	ADCAn DBiERC0	Clears the result check error flag ADCAnDBiCR.ADCAnDBiER0: 0: no function 1: ADCAnDBiCR.ADCAnDBiER0 is cleared

34.6.4 S/W trigger registers details

(1) ADCAnTRGi – A/D converter software trigger register i

This register is the trigger register to *start* the A/D conversion of CGi.

Access This register can be written in 8-bit units.

Address <ADCAn_base> + $A4_H$ + i x 4_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn STTi
W	W	W	W	W	W	W	W

Table 34-29 ADCAnTRGi register contents

Bit position	Bit name	Function
0	ADCAn STTi	Starts the A/D conversion of CGi: 0: no function 1: starts A/D conversion of CGi

Refer to *34.5.5 “Starting A/D conversion (start trigger modes)” on page 2774* for details.

(2) ADCAnTRG3 – A/D converter software trigger register 3

This register is the trigger register for transition to ADCHALT mode.

Access This register can be written in 8-bit units.

Address <ADCAn_base> + $B0_H$

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn STT3
W	W	W	W	W	W	W	W

Table 34-30 ADCAnTRG3 register contents

Bit position	Bit name	Function
0	ADCAn STT3	0: no function 1: transition to ADCHALT mode

Refer to *34.5.8 “Pausing and resuming A/D conversion (ADCHALT mode)” on page 2778* for details.

(3) ADCAnTRG4+i – A/D converter S/W trigger register 4+i

This register is the software trigger register to *stop* the A/D conversion of CGi.

Access This register can be written in 8-bit units.

Address <ADCAn_base> + $B4_H$ + i x 4_H

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn SPi
W	W	W	W	W	W	W	W

Table 34-31 ADCAnTRG4 + i register contents

Bit position	Bit name	Function
0	ADCAn SPi	0: no function 1: stops the A/D conversion of CGi

Refer to *34.5.6 “Stopping A/D conversion” on page 2776* for details.

Note If a H/W start trigger for CGi occurs simultaneously with a CGI A/D conversion stop by ADCAnTRG4+i, the H/W start trigger has higher priority.

(4) ADCAnTRG7 – A/D converter S/W trigger register 7

This register is the S/W trigger register to resume the A/D conversion after a ADCHALT.

Access This register can be written in 8-bit units.

Address <ADCAn_base> + $C0_H$

Initial Value Reading this register returns always 00_H.

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	ADCAn SP3
W	W	W	W	W	W	W	W

Table 34-32 ADCAnTRG7 register contents

Bit position	Bit name	Function
0	ADCAn SP3	0: no function 1: resume A/D conversion

Refer to *34.5.8 “Pausing and resuming A/D conversion (ADCHALT mode)” on page 2778* for details.

34.6.5 ADCA conversion result registers details

(1) ADCAnLCR – A/D converter latest conversion result register

This register stores the result and the status of the latest A/D conversion.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.

Address <ADCAn_base> + $A0_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn LCG[1:0]		ADCAn LER1	ADCAn LER0	ADCAn LUR	ADCAn LCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnLCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 34-33 ADCAnLCR register contents (1/2)

Bit position	Bit name	Function
25 to 24	ADCAn LCG[1:0]	Indicates the CG to which the result in ADCAnLCR[15:00] belongs. ADCAn LCG1 / ADCAn LCG0 / Channel group 0 / 0 / CG0 0 / 1 / CG1 1 / 0 / CG2 1 / 1 / None
23	ADCAn LER1	Indicates the overwrite error status. 0: not overwritten 1: overwritten This error flag is cleared when ADCAnSTC2.ADCAnLERC1 is set to 1.
22	ADCAn LER0	Indicates the A/D conversion result upper/lower limit comparison status. 0: The conversion result is within the setting range. 1: The conversion result is out of the setting range. This error flag is cleared when ADCAnSTC2.ADCAnLERC0 is set to 1.
21	ADCAn LUR	Indicates the update status of the A/D conversion result. 0: The A/D conversion result has been read from the ADCAnLCR register. 1: 1: The A/D conversion result is new and has not been read from ADCAnLCR yet. This bit is cleared after the read operation.
20 to 16	ADCAn LCN[4:0]	Indicates the channel number corresponding to the conversion result stored in the ADCAnLCR[15:00]. 00000 = CH0 00001 = CH1 ... = ... 10111 = CH23

Table 34-33 ADCAnLCR register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>15 to 0</td><td>ADCAn LCR[15:00]</td><td>Result of the A/D conversion.
The resolution and alignment are determined by ADCAnACTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTYP</th><th>ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>ADCAnLCR[11:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>ADCAnLCR[15:04]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>ADCAnLCR[09:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>ADCAnLCR[15:06]</td></tr>
</table>
</td></tr>
</table>

Note If A/D conversion is performed using the internal reference voltage, the A/D conversion result is stored in ADCAnDGCR, not in the ADCAnLCR, ADCAnCmCR, and ADCAnDBiCR.

(2) ADCAnCmCR – A/D converter conversion result register for channel m

This register stores the result and the status of the latest A/D conversion of channel m.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.

Address <ADCAn_base> + $3C_H$ + m x 4_H

Initial Value $0300\ 0000_H$ + m x $0001\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn CmCG[1:0]		ADCAn CmER1	ADCAn CmER0	ADCAn CmUR	ADCAn CmCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCmCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Notes

1. The functions of the individual bits are identical to those of the ADCAnLCR register, except that each of the m registers hold the latest result for the specific channel m rather than the latest A/D conversion result for any channel.
2. If ADCAnCTL1.ADCAnRCL = 0, the A/D conversion result in ADCAnCmCR[15:00] is kept until it is overwritten by the next A/D conversion result.

   If ADCAnCTL1.ADCAnRCL = 1, ADCAnCmCR[15:00] is cleared by reading it.

Table 34-34 ADCAnCmCR register contents (1/2)

Bit position	Bit name	Function
25 to 24	ADCAn CmCG[1:0]	Indicates the CG to which the result in ADCAnCmCR[15:00] belongs.
23	ADCAn CmER1	Indicates the overwrite error status. 0: not overwritten 1: overwritten This error flag reflects the value of ADCAnSTR1.ADCAnOWEm. It is cleared when ADCAnSTC1.ADCAnQWECm is set to 1.
22	ADCAn CmER0	Indicates the A/D conversion result upper/lower limit comparison status. 0: The conversion result is within the setting range. 1: The conversion result is out of the setting range. This error flag reflects the value of ADCAnSTR0.ADCAnRCEm. It is cleared when ADCAnSTC0.ADCAnRCECm is set to 1.

ADCAn CmCG1	ADCAn CmCG0	Channel group
0	0	CG0
0	1	CG1
1	0	CG2
1	1	None

Table 34-34 ADCAnCmCR register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>21</td><td>ADCAn
CmUR</td><td>Indicates the update status of the A/D conversion result.
0: The A/D conversion result has been read from the ADCAnCmCR register.
1: The A/D conversion result is new and has not been read from ADCAnCmCR yet.
This bit is cleared after the read operation.</td></tr>
<tr><td>20 to 16</td><td>ADCAn
CmCN[4:0]</td><td>Indicates the channel number corresponding to the conversion result stored in the ADCAnCmCR[15:00].
00000 = CH0
00001 = CH1
... = ...
10111 = CH23</td></tr>
<tr><td>15 to 0</td><td>ADCAn
CmCR[15:00]</td><td>Result of the A/D conversion.
The resolution and alignment are determined by ADCAnACTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTYP</th><th>ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>ADCAnCmCR[11:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>ADCAnCmCR[15:04]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>ADCAnCmCR[09:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>ADCAnCmCR[15:06]</td></tr>
</table></td></tr>
</table>

Note If A/D conversion is performed using the internal reference voltage, the A/D conversion result is stored in ADCAnDGCR, not in the ADCAnLCR, ADCAnCmCR, and ADCAnDBiCR.

(3) ADCAnDBiCR – A/D converter CGI buffer register

This register stores the result and the status of the latest A/D conversion of CGi. It allows the A/D conversion results for all channels of CGi to be read continuously.

Access This register can be read in 32-bit units.

- The upper 16 bits store the A/D conversion result status.
- The lower 16 bits store the A/D conversion result.
  The conversion result can also be read via the ADCAnDBiCRL register.

Address <ADCAn_base> + $C4_H$ + i x 4_H

Initial Value 0000 0000_H + i x 0100 0000_H

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	ADCAn DBiCG[1:0]		ADCAn DBiER1	ADCAn DBiER0	ADCAn DBiUR	ADCAn DBiCN[4:0]				
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnDBiCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Note The functions of the individual bits are identical to those of the ADCAnLCR register, except that each of the i registers hold the latest result for the specific channel group i rather than the latest A/D conversion result for any channel.

Table 34-35 ADCAnDBiCR register contents (1/2)

Bit position	Bit name	Function
25 to 24	ADCAn DBiCG[1:0]	Indicates the CG to which the result in ADCAnDBiCR[15:00] belongs. ADCAn DBiCG1 / ADCAn DBiCG0 / Channel group 0 / 0 / CG0 0 / 1 / CG1 1 / 0 / CG2 1 / 1 / None
23	ADCAn DBiER1	Indicates the overwrite error status. 0: not overwritten 1: overwritten This error flag is cleared when ADCAnSTC2.ADCAnDBiERC1 is set to 1.
22	ADCAn DBiER0	Indicates the A/D conversion result upper/lower limit comparison status. 0: The conversion result is within the setting range. 1: The conversion result is out of the setting range. This error flag is cleared when ADCAnSTC2.ADCAnDBiERC0 is set to 1.
21	ADCAn DBiUR	Indicates the update status of the A/D conversion result. 0: The A/D conversion result has been read from the ADCAnDBiCR register. 1: 1: The A/D conversion result is new and has not been read from ADCAnDBiCR yet. This bit is cleared after the read operation.

Table 34-35 ADCAnDBiCR register contents (2/2)

<table>
<tr><th>Bit position</th><th>Bit name</th><th>Function</th></tr>
<tr><td>20 to 16</td><td>ADCAn DBiCN[4:0]</td><td>Indicates the channel number corresponding to the conversion result stored in the ADCAnDBiCR[15:00].
00000 = CH0
00001 = CH1
... = ...
10111 = CH23</td></tr>
<tr><td>15 to 0</td><td>ADCAn DBiCR[15:00]</td><td>Result of the A/D conversion.
The resolution and alignment are determined by ADCAnACTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.
<table>
<tr><th>ADCAn CTYP</th><th>ADCAn CRAC</th><th>Resolution and alignment</th><th>Bit positions of A/D conversion result</th></tr>
<tr><td>0</td><td>0</td><td>12-bit resolution, right-aligned</td><td>ADCAnDBiCR[11:00]</td></tr>
<tr><td>0</td><td>1</td><td>12-bit resolution, left-aligned</td><td>ADCAnDBiCR[15:04]</td></tr>
<tr><td>1</td><td>0</td><td>10-bit resolution, right-aligned</td><td>ADCAnDBiCR[09:00]</td></tr>
<tr><td>1</td><td>1</td><td>10-bit resolution, left-aligned</td><td>ADCAnDBiCR[15:06]</td></tr>
</table></td></tr>
</table>

Note If A/D conversion is performed using the internal reference voltage, the A/D conversion result is stored in ADCAnDGCR, not in the ADCAnLCR, ADCAnCmCR, and ADCAnDBiCR.

(4) ADCAnDBiCRL – A/D converter CGI buffer result register

This register stores the result of the latest A/D conversion of CGi.

The content of this register is identical to the lower 16 bit of the ADCAnDBiCR register (ADCAnDBiCR[15:00]).

Access This register can be read in 16-bit units.

Address <ADCAn_base> + $D0_H$ + i x 4_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnDBiCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

(5) ADCAnDGCR – Diagnostic conversion result register

This register stores the A/D conversion result of the reference voltage ADDIAGOUT signal (when ADCAnCG0.ADCAnDIAG = 1).

The diagnosis conversion starts after the A/D conversion of the last channel of CG0 has been completed.

Access This register can be read in 16-bit units.

Address <ADCAn_base> + $9C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnDGCR[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 34-36 ADCAnDGCR register contents

Bit position	Bit name	Function
15 to 0	ADCAn DGCR[15:00]	Result of the diagnostic A/D conversion. The resolution and alignment are determined by ADCAnACTL1.ADCAnCTYP and ADCAnCTL1.ADCAnCRAC, as follows.

ADCAn CTYP	ADCAn CRAC	Resolution and alignment	Bit positions of A/D conversion result
0	0	12-bit resolution, right-aligned	ADCAnDBiCR[11:00]
0	1	12-bit resolution, left-aligned	ADCAnDBiCR[15:04]
1	0	10-bit resolution, right-aligned	ADCAnDBiCR[09:00]
1	1	10-bit resolution, left-aligned	ADCAnDBiCR[15:06]

34.6.6 A/D conversion result upper/lower limit comparison registers details

(1) ADCAnCTL2 – A/D converter result check register

This register can enable/disable the conversion result upper/lower limit comparison function for each channel.

Refer to *34.5.12 "Result check functions" on page 2784* for details.

Access This register can be read/written in 32-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + 18_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCK[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCK[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-37 ADCAnCTL2 register contents

Bit position	Bit name	Function
23 to 00	ADCAn RCK[23:00]	Enables/disables result upper/lower limit comparison for CHm. 0: do not perform upper/lower limit comparison for A/D conversion of CHm 1: perform upper/lower limit comparison for A/D conversion of CHm

Note The settings are valid for A/D conversions of every CG.

(2) ADCAnUL – A/D converter result check upper limit register

This register specifies the upper limit of the A/D conversion result.

Refer to *34.5.12 "Result check functions" on page 2784* for details.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + $1C_H$

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnUL[11:00]												0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-38 ADCAnUL register contents

Bit position	Bit name	Function
15 to 4	ADCAn UL[11:00]	Specifies the upper limit of the A/D conversion result. In the case of the 10-bit resolution, use ADCAUL[11:02] and set ADCAUL[01:00] = 00_B.

(3) ADCAnLL – A/D converter result lower limit register

This register specifies the lower limit of the A/D conversion result.

Refer to *34.5.12 "Result check functions" on page 2784* for details.

Access This register can be read/written in 16-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + 20_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnLL[11:00]												0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-39 ADCAnLL register contents

Bit position	Bit name	Function
15 to 4	ADCAn LL[11:00]	Specifies the lower limit of the A/D conversion result. In the case of the 10-bit resolution, use ADCAnLL[11:02] and set ADCALL[01:00] = 00_B.

(4) ADCAnSTR0 – A/D converter result check error flag register

This register indicates the error status of the latest A/D conversion result upper/lower limit comparison for the channel set in the ADCAnCTL2 register. By evaluating this register it is possible to deduce which A/D conversion results are outside the specified range.

Refer to *34.5.12 "Result check functions" on page 2784* for details.

Access This register can be read in 32-bit units.

Address <ADCAn_base> + 24_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCE[23:16]							
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCE[15:00]															
R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 34-40 ADCAnSTR0 register contents

Bit position	Bit name	Function
23 to 0	ADCAn RCE[23:00]	Indicates whether the A/D conversion result is within the specified value range: 0: conversion results within the specified range 1: at least one Conversion result out of the specified range This error flag is cleared by setting ADCAnSTC0.ADCAnRCECm to 1. All flags are also cleared by enabling and disabling the ADCA (ADCAn.ADCAnCE= 0 and then ADCAn.ADCAnCE= 1).

Note ADCAnSTR0.ADCAnRCEm is mirrored by the following A/D conversion result error flag:

- Error flag in A/D converter conversion result register for channel m (ADCAnCmCR.ADCAnCmER0)

(5) ADCAnSTC0 – ADCAnSTR0 flag clear register

This register is the clear control register of ADCAnSTR0.

Access This register can be written in 32-bit units.

Address <ADCAn_base> + 30_H

Initial Value Reading this register always returns 0000 0000_H.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnRCEC[23:16]							
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnRCEC[15:00]															
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Table 34-41 ADCAnSTC0 register contents

Bit position	Bit name	Function
23 to 0	ADCAn RCEC[23:00]	0: no function 1: clears the corresponding ADCAnSTR0.ADCAnRCEm

34.6.7 Diagnose functions registers

(1) ADCAnDGCTL0 – Self-diagnosis function control register 0

This register specifies the reference voltage to be applied to diagnose the A/D conversion circuit operation.

Access This register can be read/written in 16-bit units.

Address <ADCAn_base> + DC_H

Initial Value 0000_H

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	ADCAnPSEL[2:0]		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-42 ADCAnDGCTL0 register contents

Bit position	Bit name	Function
2 to 0	ADCAn PSEL[2:0]	Specifies the reference voltages

ADCAn PSEL2	ADCAn PSEL1	ADCAn PSEL0	ADDIAGOUT	DIAGOUT2	DIAGOUT1	DIAGOUT0
0	0	0	AVSS	2/3 AVDD	1/2 AVDD	1/3 AVDD
0	0	1	1/3 AVDD	1/2 AVDD	1/3 AVDD	2/3 AVDD
0	1	0	1/2 AVDD	1/3 AVDD	2/3 AVDD	1/2 AVDD
0	1	1	2/3 AVDD	Hi-Z	Hi-Z	Hi-Z
1	0	0	AVDD	2/3 AVDD	1/2 AVDD	1/3 AVDD
1	0	1	AVDD	1/2 AVDD	1/3 AVDD	2/3 AVDD
1	1	0	AVDD	1/3 AVDD	2/3 AVDD	1/2 AVDD
1	1	1	AVDD	Hi-Z	Hi-Z	Hi-Z

When Hi-Z is selected and converted, the A/D conversion result is undefined.

Refer to *34.5.14 “Self-diagnosis functions” on page 2793* for details.

(2) ADCAnDGCTL1 – Self-diagnosis function control register 1

This register specifies the channels to which a internal reference voltage is applied (instead of the analog input signal ADCAnIm).

Access This register can be read/written in 32-bit units.
It can only be written when the A/D converter is disabled (ADCAnCTL0.ADCAnCE = 0).

Address <ADCAn_base> + $11C_H$

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnCDG[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnCDG[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-43 ADCAnDGCTL1 register contents

Bit position	Bit name	Function
23 to 0	ADCAn CDGm	Specifies the input voltage: 0: Use analog input voltage ADCAnIm 1: Use the following reference voltage: DIAGOUT0 for m = 21, 18, 15, 12, 9, 6, 3, 0 DIAGOUT1 for m = 22, 19, 16, 13, 10, 7, 4, 1 DIAGOUT2 for m = 23, 20, 17, 14, 11, 8, 5, 2

(3) ADCAnPDCTL0 – Pull down resistance control register 0

This register specifies the channels to which an internal pull-down resistor for the ADCAnIm pin is to be connected.
For details, refer to *34.5.14 “Self-diagnosis functions” on page 2793* .

Access This register can be read/written in 32-bit units.

Address <ADCAn_base> + 120_H

Initial Value $0000\ 0000_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	ADCAnPDNA[23:16]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADCAnPDNA[15:00]															
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-44 ADCAnPDCTL0 register contents

Bit position	Bit name	Function
23 to 0	ADCAn PDNAm	Specifies whether an internal pull-down register is connected to CHm: 0: Do not connect an internal pull-down resistor 1: Connect an internal pull-down resistor

34.6.8 Emulation register

(1) ADCAnEMU - Emulation register

This register controls whether the A/D Converter can be stopped during emulation, for instance upon a breakpoint hit.

Access This register can be read/written in 8-bit units.

Address <ADCAn_base> + 128_H

Initial Value 00_H

7	6	5	4	3	2	1	0
ADCAn SVSDIS	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Table 34-45 ADCAnEMU register contents

Bit position	Bit name	Function
7	ADCAn SVSDIS	Emulation control 0: ADCAn can be stopped during emulation 1: ADCAn continuous operating during emulation

Chapter 35 Voltage Comparator (VCPC)

This chapter contains a generic description of the Voltage Comparator (VCPC).

The first section describes all properties specific to the V850E2/Fx4-H, such as instances, register base addresses, input/output signal names, etc.
The subsequent sections describe the features that apply to all implementations.

35.1 V850E2/Fx4-H VCPC Features

Instances

This microcontroller has following number of instances of the Voltage Comparator.

Table 35-1 Instances of VCPC

Voltage Comparator	
Instances	1
Names	VCPC0

Instances index n

Throughout this chapter, the individual instances of a Voltage Comparator are identified by the index “n” (n = 0), for example, VCPCnSTRm for the VCPCn control register 0.

Channel index m

Each Voltage Comparator has 2 channels. Throughout this chapter, the individual channels are identified by the index “m” (m = 0 to 1).

Register addresses

All VCPCn register addresses are given as address offsets from the individual base address <VCPCn_base>.
The base address <VCPCn_base> of each VCPCn is listed in the following table:

Table 35-2 Register base addresses <VCPCn_base>

VCPCn instance	<VCPCn_base> address
VCPC0	FF81 8000_H

Clock supply All Voltage Comparators provide one clock input.

Table 35-3 VCPCn clock supply

VCPCn instance	VCPCn clock	Connected to
VCPC0	PCLK	Clock Controller CKSCLK_A02

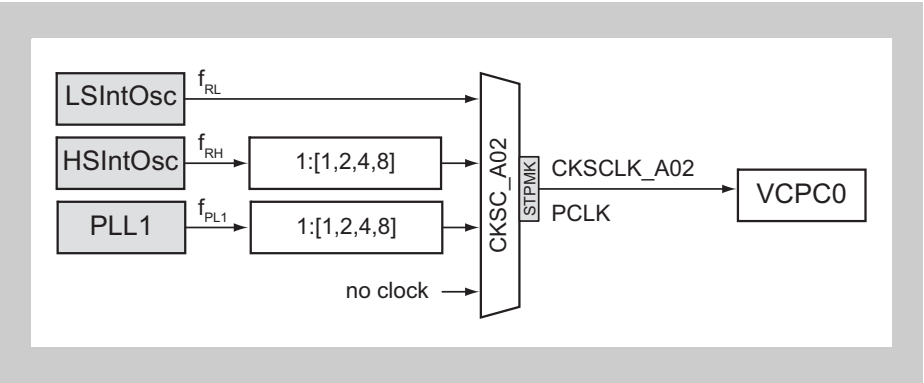


Figure 35-1 VCPC clock supply

Interrupts The Voltage Comparator can generate following interrupt requests:

Table 35-4 VCPC interrupt requests

VCPC0 signals	Function	Connected to
INTVCPC0	Channel 0 interrupt request	Interrupt Controller INTVCPC0[a]
INTVCPC1	Channel 1 interrupt request	Interrupt Controller INTVCPC1[a]

a) These interrupts can be used as a wake-up source from any stand-by mode. Refer to the chapter *"Stand-by Controller (STBC)"* for details.

VCPC H/W reset The Voltage Comparators and their registers are initialized by the following reset signal:

Table 35-5 VCPCn reset signal

VCPCn	Reset signal
VCPC0	• Reset Controller: SYSRES

I/O signals

The I/O signals of the Voltage Comparator are listed in the table below.

Table 35-6 VCPCn I/O signals

VCPC0 signals	Function	Connected to
VCPCIN0	Channel 0 compare voltage input	Port VCPC0IN
VCPCIN1	Channel 1 compare voltage input	Port VCPC1IN
VCPCOUT0	Channel 0 compare output signal	Port VCPC0OUT
VCPCOUT1	Channel 1 compare output signal	Port VCPC1OUT

35.2 Overview

The Voltage Comparator comprises two identical channels m = 0, 1.

Features summary The Voltage Comparator has the following features:

- Comparison of an external voltage with two internal reference voltages for switching with hysteresis
- Stand-by function in order to achieve minimized stand-by current
- Comparator status information:
  - status flag in register
  - interrupt request INTVCPCm generation upon falling and rising input voltage V_{CMP}

Note For details on the input and reference voltage levels refer to the Data Sheet.

35.2.1 Description

Each Voltage Comparator channel m consists of two separate voltage comparators CMPR and CMPF, which compare the input voltage V_{CMP} against two different internal reference voltages V_{CMPR} and V_{CMPF}.

The figure below shows a block diagram of channel m of the Voltage Comparator.

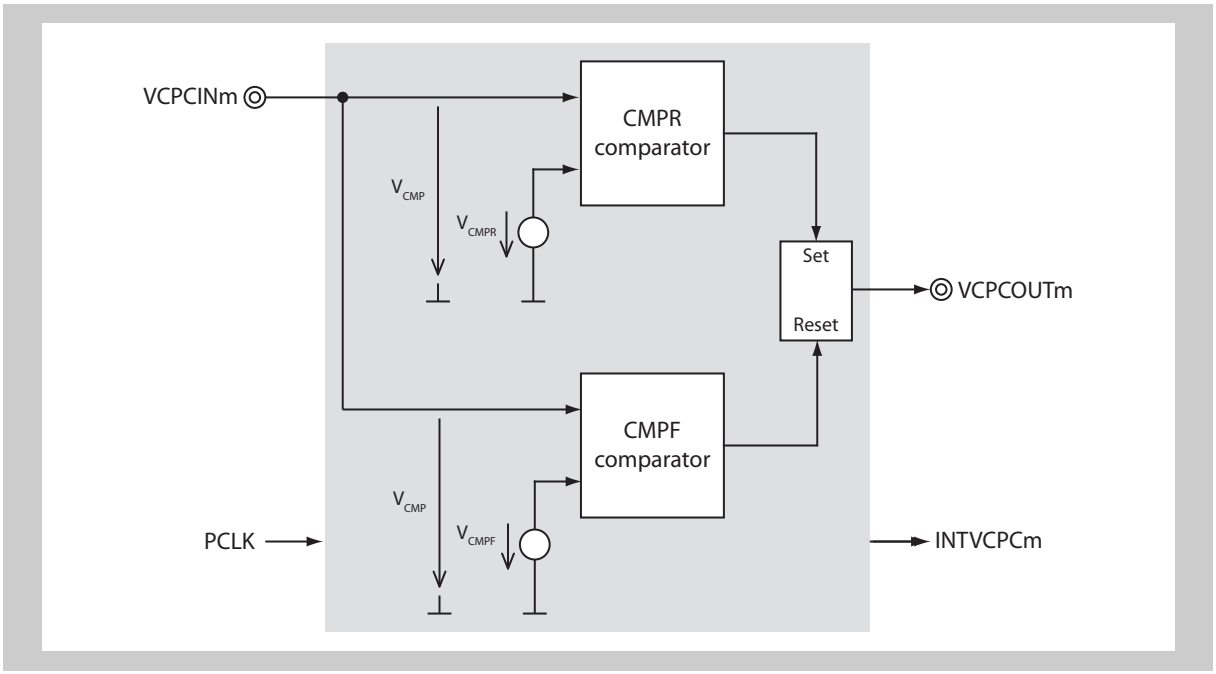


Figure 35-2 Voltage Comparator channel m block diagram

Output and status flag

The CMPR and CMPF comparator's outputs effect the following:

- If $V_{CMP} > V_{CMPR}$:
  The output signal VCPCOUTm becomes high and the status flag VCPCnSTRm.VCPCnSFm is set.
- $V_{CMP} < V_{CMPF}$:
  The output signal VCPCOUTm becomes low and the status flag VCPCnSTRm.VCPCnSFm is cleared.

Hysteresis

The reference voltages are different: $V_{CMPR} > V_{CMPF}$.
This way a hysteresis is introduced.

Interrupt

Depending on the CMPR and CMPF comparator's outputs the interrupt INTVCPCm can be generated.

It can be selected whether the interrupt shall be generated during rise or fall of V_{CMP}:

- VCPCnCTLm.VCPCnESm[1:0] = 00_B:
  INTVCPCm generated if V_{CMP} falls below V_{CMPF}
- VCPCnCTLm.VCPCnESm[1:0] = 01_B:
  INTVCPCm generated if V_{CMP} rises above V_{CMPR}
- VCPCnCTLm.VCPCnESm[1:0] = 11_B:
  INTVCPCm generated if V_{CMP} falls below V_{CMPF} or rises above V_{CMPR}

Note The interrupt generation requires the clock PCLK to operate. The comparison result VCPCOUTm is also output if PCLK is stopped.

The following figure shows the behaviour of the Voltage Comparator.
In this example, following settings are assumed:

- channel m is enabled (VCPCnCTLm.VCPCnOEm = 1)
- channel m interrupt INTVCPCIm at falling and rising input voltage VCPCINm (VCPCnCTLm.VCPCnESm[1:0] = 11_B)

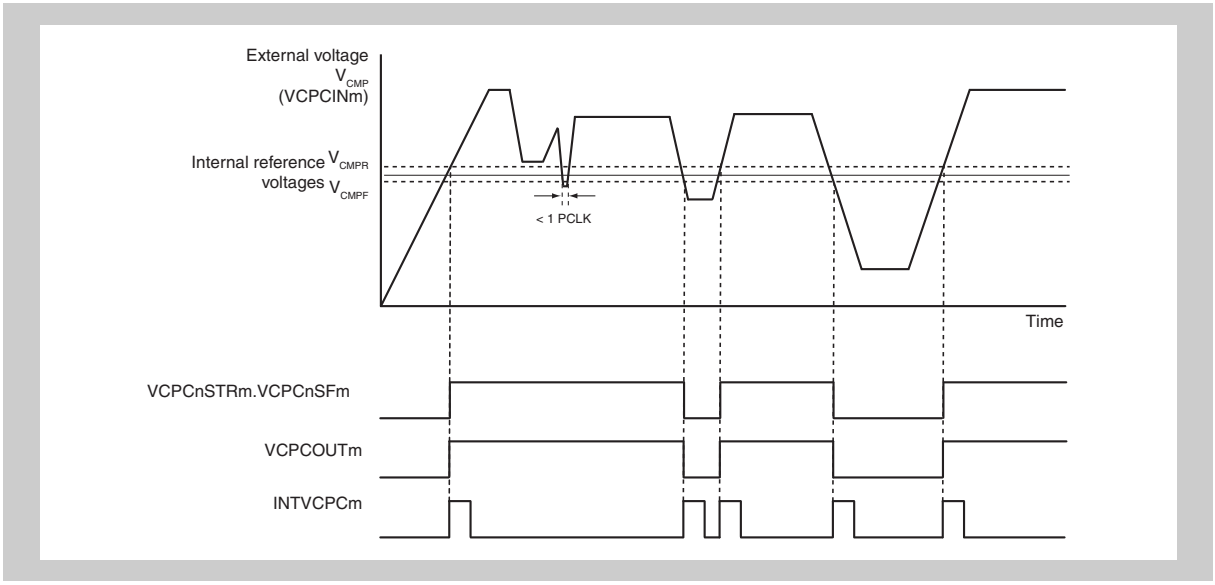


Figure 35-3 Voltage Comparator behaviour

Delay time Refer to the Data Sheet for the delay time between crossing of V_{CMP} the V_{CMPR}/V_{CMPF} level and change of VCPCOUTm.

Cautions

1. A V_{CMP} pulse, crossing of the V_{CMPR}/V_{CMPF}, with a width of less than 1 PCLK period might be ignored. In that case it would neither change VCPCOUTm nor asserts the interrupt INTVCPCm.
2. The power-on stabilization time must be passed after having enabled a Voltage Comparator and before being able to read the correct status of the Voltage Comparator's status flag. This also affects interrupt generation which means the Voltage Comparator interrupts must be masked / ignored during the power-on stabilization time.
   For details on the power-on stabilization time refer to the Data Sheet.

35.2.2 Stand-by mode

In order to reduce power consumption during stand-by modes, the Voltage Comparator channels m can be separately set into stand-by mode. This is done by setting VCPCnCTLm.VCPCnOEm = 0.

If the Voltage Comparator channel m is set in stand-by mode it assumes that $V_{CMP} < V_{CMPF}$ (VCPCnSTRm.VCPCnSFm = 0 and VCPCOUTm = low level).

35.3 Voltage Comparator Registers

The Voltage Comparator is controlled by means of the following registers:

Table 35-7 Voltage Comparator registers overview

Register name	Shortcut	Address
Control register 0	VCPCnCTL0	<VCPCn_base>
Control register 1	VCPCnCTL1	<VCPCn_base> + 4_H
Status register 0	VCPCnSTR0	<VCPCn_base> + 8_H
Status register 1	VCPCnSTR1	<VCPCn_base> + C_H

<VCPCn_base> The base addresses <VCPCn_base> of the VCPCn is defined in the first section of this chapter under the key word “Register addresses”.

(1) VCPCnCTLm - Voltage Comparator control registers

These registers control whether the Voltage Comparator channel m is operating or in stand-by mode. Further it specifies whether an interrupt is generated when V_{CMP} rises above V_{CMPR} or falls below V_{CMPF} or at both transitions.

Access These registers can be read/written in 8-bit units.

Address VCPCnCTL0: <VCPCn_base>
VCPCnCTL1: <VCPCn_base> + 4_H

Initial Value 00_H

7	6	5	4	3	2	1	0
VCPCn OEm	0	0	0	0	0	VCPCn ESm[1:0]	
R/W	R	R	R	R	R	R/W	R/W

Table 35-8 VCPCnCTLm registers contents

Bit position	Bit name	Function
7	VCPCn OEm	Enable Voltage Comparator channel m 0: VCPCm in stand-by mode (VCPCnSTRm.VCPCnSFm = 0, VCPCOUTm = low level) 1: VCPCm is operating
1 to 0	VCPCn ESm[1:0]	Voltage transition direction selection for INTVCPCm interrupt assertion

VCPCnESm1	VCPCnESm0	INTVCPCm assertion upon
0	0	V_{CMP} falls below V_{CMPF}
0	1	V_{CMP} rises above V_{CMPR}
1	0	reserved
1	1	V_{CMP} falls below V_{CMPF} or rises above V_{CMPR}

Caution Do not change the voltage transition direction selection VCPCnCTLm.VCPCnESm[1:0] while the Voltage Comparator channel m is operating (VCPCnSTRm.VCPCnOEm = 1).

(2) VCPCnSTRm - Voltage Comparator status register

These registers reflect the result of the voltage comparison of channel m.

Access These registers are read-only in 8-bit units.

Address VCPCnSTR0: <VCPCn_base> + 8_H
VCPCnSTR1: <VCPCn_base> + C_H

Initial Value 00_H

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	VCPCn SFm
R	R	R	R	R	R	R	R

Table 35-9 VCPCnSTRm registers contents

Bit position	Bit name	Function
0	VCPCn SFm	Voltage Comparator status flag 0: Input voltage V_{CMP} is below reference voltage V_{CMPF}. 1: Input voltage V_{CMP} is above reference voltage V_{CMPR}. If the Voltage Comparator channel m is in stand-by (VCPCnCTLm.VCPCnOEm = 0) VCPCnSFm is also 0.

Chapter 36 On-chip Debug Unit (OCD)

This microcontroller has a debug function on-chip. By using the on-chip debug emulator, programs can be debugged with the microcontroller mounted in the target system.

The debug functions incorporated in this microcontroller conform to IEEE-ISTO 5001™-2003 Class 1, a Nexus debug interface standard.

Caution The debug functions described in this chapter are supported by the microcontroller but whether they are usable depends on the debugger. For details of debugging, see the user's manual of the debugger.

36.1 V850E2/Fx4-H On-chip Debug Features

36.1.1 Modules behaviour during emulation break

This section specifies, which modules

- are always stopped (unconditional emulation break)
- can optionally be stopped (emulation break function)
- continue operation,

if the debugger stops the microcontroller's operation in case of an emulation break.

Emulation break An emulation break refers to a

- breakpoint hit
- manual break

during a debug session.

(1) Modules with unconditional emulation break

Following list shows all modules which are always stopped upon a emulation break:

Table 36-1 Modules with unconditional emulation break

Module
Window Watchdog Timer A (WDTA)

(2) Modules with optional emulation break

Modules with optional emulation break provide a control bit, that allows the application software to make the module stop or continue upon emulation break.

These modules support the emulation stop function.

Following list shows all modules which can optionally be stopped or continue upon an emulation break:

Table 36-2 Modules with optional emulation break

Module	Control register bit
OS Timer (OSTM)	OSTMnEMU.OSTMnSVSDIS: 0: stop during break 1: continue during break
Timer Array Unit A (TAUA)	TAUAnEMU.TAUAnSVSDIS: 0: stop during break 1: continue during break
Timer Array Unit B (TAUB)	TAUBnEMU.TAUBnSVSDIS: 0: stop during break 1: continue during break
Timer Array Unit C (TAUC)	TAUCnEMU.TAUCnSVSDIS: 0: stop during break 1: continue during break
Timer Array Unit J (TAUJ)	TAUJnEMU.TAUJnSVSDIS: 0: stop during break 1: continue during break
Real-Time Clock (RTCA)	RTCAnEMU.RTCAnSVSDIS: 0: stop during break 1: continue during break
Clocked Serial Interface G (CSIG)	CSIGnEMU.CSIGnSVSDIS: 0: stop during break 1: continue during break
Clocked Serial Interface G (CSIH)	CSIHnEMU.CSIHnSVSDIS: 0: stop during break 1: continue during break
Asynchronous Serial Interface E (URTE)	URTEnEMU.URTEnSVSDIS: 0: stop during break 1: continue during break
I^2C Interface (IICB)	IICBnEMU.IICBnSVSDIS: 0: stop during break 1: continue during break
Timer Motor Control Function (TAPA)	TAPAnEMU.TAPAnSVSDIS: 0: stop during break 1: continue during break
Encoder Timer (ENCA)	ENCAnEMU.ENCAnSVSDIS: 0: stop during break 1: continue during break
A/D Converter A (ADCA)	ADCAnEMU.ADCAnSVSDIS: 0: stop during break 1: continue during break

(3) Modules continuing operation at emulation break

Following list shows all modules continuing operation upon an emulation break:

Table 36-3 Modules continuing operation at emulation break

Module
CAN Controller (FCN)
Diagnostic CAN Controller (DCN)
LIN Master Controller (LMA)
FlexRay™ (FLX)
Random Number Generator A (RNGA)
PWM Delay (DLYA)
PWM Diagnostic (PMCA)
Ethernet Controller (ETHA)
Key Return Function (KR)
Voltage Comparator (VCPC)

36.1.2 Signal masking

Following V850E2/Fx4-H external signals can be masked, so that they don't have any effect, while the microcontroller is controlled by the On-chip Debug Unit:

- $\overline{\text{RESET}}$
- NMI

36.2 Functional Overview

The On-Chip Debug functions are outlined below.

(1) Debug interface

This interface is used to communicate with the host by using the $\overline{\text{DCUTRST}}$, DCUTCK, DCUTMS, DCUTDI, DCUTDO, and $\overline{\text{DCUTRDY}}$ signals via the on-chip debug emulator.

(2) Debug monitoring function

The basic debug functions below can be used by running a monitoring program in a memory space for debugging while execution of the user-created program is paused.

- downloading the user-created program
- reading and writing the memory and registers
- running the user-created program starting at any address

(3) Hardware break function

Up to four breakpoints can be specified for instructions and data. If a breakpoint is specified for an instruction, execution can be interrupted at the specified address. If a breakpoint is specified for an address, execution can be interrupted when data at the specified address is accessed.

In addition, break conditions can be combined by using a sequence of up to two levels.

(4) Software break function

Execution of the user-created program stored in the RAM can be interrupted at the specified address.

(5) Forced break function

Execution of the user-created program can be interrupted forcibly.

(6) Forced reset function

The microcontroller can be reset forcibly.

(7) Real-time RAM monitoring (RRM)

The memory can be read during program execution. Because this read access uses debug-dedicated DMA, it has minimal effect on program execution.

(8) Dynamic memory modification (DMM)

The memory can be written during program execution. Because this write access uses debug-dedicated DMA, it has minimal effect on program execution.

(9) Timer function

Using a 32-bit counter, the time for running the user-created program can be measured based on the clock obtained by dividing the DCUTCK signal frequency by 2.

(10) Mask function

Some dedicated external signals can be masked, so that they don't have any effect, while the microcontroller is controlled by the On-Chip Debug Unit.

These signals are listed in the section *"Signal masking"* above in this chapter.

(11) Modules run/stop selection during a break

Upon a breakpoint hit the microcontroller's modules behave as follows:

- module always stops operation during break
- behaviour of the module during break is an user's option and can be specified by use of the module's emulation register.
  This emulation break function needs to be generally enabled, refer to *36.3 "Emulation Break Control" on page 2850* for details.
- module always stays in operation during break

The bahaviour of the modules of this microcontroller is described in the section *"Modules behaviour during emulation break"* above in this chapter.

(12) Hot attach function

The On-Chip Debug emulator can be connected and start debugging without resetting the CPU while it is running.

(13) Security function

To prevent the contents of the flash memory from being read by an unauthorized person, a 96-bit ID code can be written to the microcontroller. If the code the user inputs when starting a debugger does not match the ID code written to the microcontroller, the flash memory cannot be accessed. If the OCDID[95] bit of the On-Chip Debug ID register OCDIDH is set to 0, the flash memory cannot be accessed even if the ID code matches.

For details of how to set the ID code, see the user's manual of the used software tools and refer to section *"On-Chip Debug Interface Protection"* in the chapter *"Code Protection and Security"* in this manual.

(14) On-Chip Debug and Stand-by modes

Setting the device in STOP or DEEPSTOP mode is not possible, if the microcontroller is operated under control of an On-Chip Debugger.

36.3 Emulation Break Control

The emulation break function generates a stop request to the modules of the microcontroller, if the debugger is taking over the control of the microcontroller, for instance upon a breakpoint hit.

Those modules stop their operation during emulation break, if

- the module supports the emulation break function (for a list of these modules refer to *"Modules behaviour during emulation break"* in the first section of this chapter)
- emulation break is enabled in the module (by setting the module's emulation register).

Debugger support

In general the debugger provides an option to enable the emulation break function. In this case the application program does not need to take care for that.

36.4 Connection with On-Chip Debug Emulator

Following signals are used to connect the debugger with the microcontroller:

Table 36-4 Signals used to connect On-Chip Debug emulator

Pin name	Description
VDD	Signal used to detect power supply to the target system, or the power supply for the buffers in the On-Chip Debug emulator
$\overline{\text{DCUTRST}}$	Signal to asynchronously reset the debug functions of the microcontroller
DCUTCK	Clock signal used for debugging
DCUTMS	Signal to select the transfer mode for data communication
DCUTDI	Data signal input to the microcontroller
DCUTDO	Data signal output from the microcontroller
$\overline{\text{DCUTRDY}}$	Synchronization signal for data communication
FLMD0	Mode signal used to rewrite the flash memory in the microcontroller

Refer to the section *"Operation Modes"* in chapter *"CPU System Functions"* for details.

36.5 Cautions on using On-Chip Debugging

(1) Handling of device that was used for debugging

Do not mount a device that was used for debugging on a mass-produced product, because the flash memory was rewritten during debugging and thus the number of flash memory rewrites cannot be guaranteed.

Chapter 37 Boundary Scan

37.1 Outline

Boundary Scan is a test method defined by IEEE Std 1149.1.

It is used for testing interconnections between devices mounted on printed circuits boards.

V850E2/Fx4-H complies with the IEEE Std 1149.1-2001.

Note Boundary Scan generic issues (e.g. explanation of each instruction, state transition diagram of TAP controller, etc.) are not described in this document.

37.2 JTAG interface

Communication with a Boundary Scan host test system is executed via the JTAG interface, as described in the following table:

Table 37-1 JTAG interface

Port	Signal name	Recommendation if unused
JP0_0	DCUTDI	Open (if $\overline{\text{DCUTRST}}$ at low level)
JP0_1	DCUTDO	Open
JP0_2	DCUTCK	Open
JP0_3	DCUTMS	Open
JP0_4	$\overline{\text{DCUTRST}}$	Connect to VSS via an appropriate resistor

37.3 Entering Boundary Scan mode

Boundary Scan mode is entered by setting following pins at reset release:

- FLMD0 = VDD
- FLMD1 (P0_1) = VDD
- MODE0 (P0_2) = VSS
- MODE1 (P0_3) = VSS

Refer also to the sections *"Operation Modes"* and *"Mode pins and JP0 connections"* in chapter *"CPU System Function"* for details.

37.4 Boundary scan features

The table below shows the IEEE1149.1-2001 instructions and instruction codes for V850E2/Fx4-H.

Table 37-2 Boundary scan instruction codes

Instruction	Instruction code		Notes
	bit[20:16]	bit[15:0]	
BYPASS	1 1111	$1111\ 1111\ 1111\ 1111_B$	
EXTEST	1 1111	$1111\ 1111\ 1110\ 1000_B$	
SAMPLE	1 1111	$1111\ 1111\ 1111\ 1000_B$	same code as PRELOAD
PRELOAD	1 1111	$1111\ 1111\ 1111\ 1000_B$	same code as SAMPLE
IDCODE	1 1111	$1111\ 1111\ 1111\ 1110_B$	

37.5 Boundary Scan applicable pins

Note The ports of the Boundary Scan applicable pins don't need to be configured by use of any port configuration register during Boundary Scan.

Boundary Scan (EXTEST) is applicable to all pins except pins indicated in the following table.

Table 37-3 Boundary Scan inapplicable pins

Type	Pin name
JTAG interface	• JP0_0 to JP0_4
Power control output	• WAKE • PTCTL1
Analog inputs/ outputs	• VCPCnIN, VCPCnOUT • ADCA0I0 to ADCA0I5
Clock inputs/ outputs	• X1, X2 • XT1, XT2
Power supply	• REGnVDD, REGnC, CVDD, FVDDn, EnVDD, BnVDD, OSCVDD, AnVDD, AnVREF0P, AnVREF0M
Ground	• REGnVSS, CVSS, VSS, EnVSS, BnVSS, OSCVSS, AnVSS

For following signals, boundary scan is implemented as sampling only:

Table 37-4 Boundary scan (sampling only) applicable pins

Type	Pin name
Reset	$\overline{\text{RESET}}$
Mode	FLMD0
Power control	PWGD

Following pins are shared with analog buffers, differential buffers, etc. So, boundary scan is applied only to the general purpose I/O part.

Table 37-5 Boundary Scan (only general purpose I/O part) applicable pins

Type	Pin name
ADCA0 inputs	P10_6 to P10_15, P11_0 to P11_7
ADCA1 inputs	P12_0 to P12_15, P13_0 to P13_7

37.6 DID - Boundary scan ID register

The V850E2/Fx4-H specific ID code is provided by the device identification register DID. The value of DID can be read with the JTAG IDCODE instruction.

Access This register can not be accessed by the CPU.

Initial Value xxxx xxxx xxxx 0100 0100 0111_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RN[3:0]				PN[15:04]											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PN[03:00]				MID[10:0]											1

Table 37-6 DID register contents

Bit position	Bit name	Function
31 to 28	RN[3:0]	Revision number
27 to 12	PN[15:00]	Product number
11 to 1	MID[10:0]	Manufacturer ID, constant 01000100011_B

Chapter 38 Power Supply

The V850E2/Fx4-H devices provide separate power supply pins for the digital and analog circuits of the functional modules on the different power domains and for several groups of port I/O buffers.

38.1 Power supply pins naming

In this section some general information is given about the naming of the power supply pins.

In general the name of a power supply pin is composed of up to four fields with the following meaning:

Supply type	Prefix	Kind of supply	Suffix
Symbol, representing the purpose of the supply	Consecutive number for separate supplies[a]	VDD or VSS	Consecutive number for different pins for same supply[a]

a) Prefix and suffix number may be omitted, if only a single supply pin is used.

The different supply types are defined in the following table.

Table 38-1 Supply types symbols

Symbol	Explanation
A	Analog circuits supply (e.g. analog parts of the A/D Converter)
B	Standard I/O buffer supply
E	Standard I/O buffer supply
F	Flash memory supply
OSC	Oscillator supply
REG	Internal voltage regulator input voltage

Examples:

- E0VDD: supply for standard I/O buffers group “0” via single pin
- B1VDD2: supply for standard I/O buffers group “1” via several pins (this is the second pin)
- REG0VSS: input voltage for on-chip voltage regulator REG0

38.2 Power supply schemes

The following sections show the power supply schemes, i.e. the power supply pins and the modules they supply.

Information about the voltage ranges of the power supply pins and all conditions related to them are provided in the Electrical Target Specification. The Electrical Target Specification provides also details about the electrical properties of the port pins.

38.2.1 Power supply of the digital circuits

The digital circuits of the different power domains are supplied as follows:

(1) Always-On-Area power supply

The power supply for the digital circuits of the Always-On-Area V_{REG0} is generated from the external voltage REG0VDD by the on-chip voltage regulator REG0 and an associated on-chip power regulation transistor.

(2) Isolated areas power supply

The power supply for the digital circuits of the isolated areas V_{REGISO} is provided externally via CVDD. The on-chip voltage regulator REG1 generates the control voltage PTCTL1 for an external power regulation transistor. Alternatively an external voltage regulator can be used to generated CVDD.

The following diagram shows the power supply of the digital circuits of all power domains.

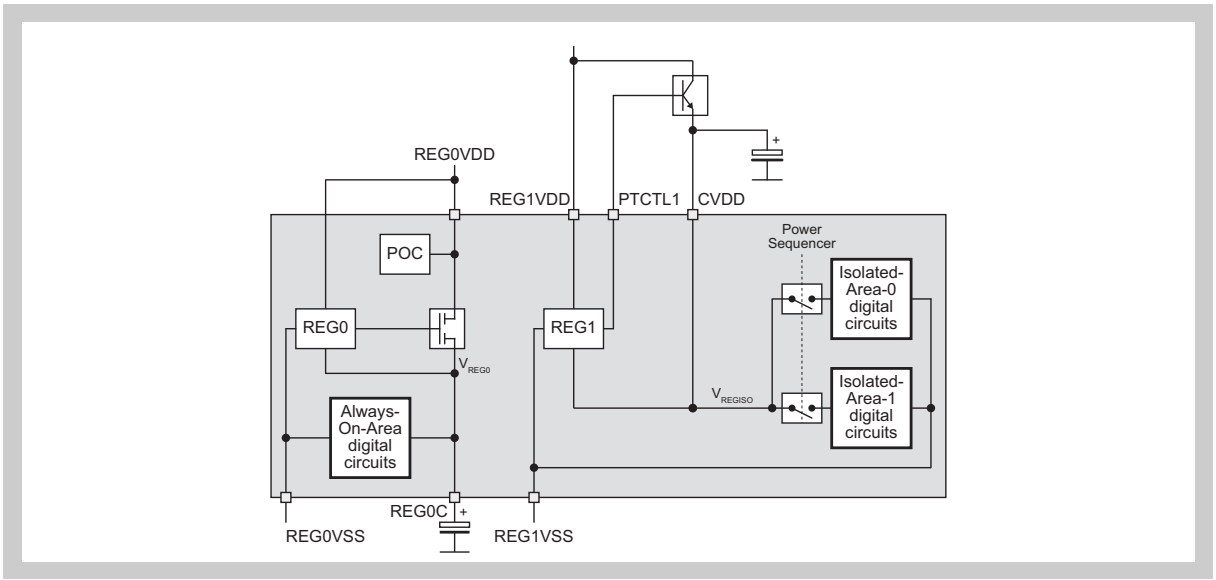


Figure 38-1 Digital circuits power supply scheme

38.2.2 V850E2/FK4-H power supply scheme

The following table and diagram shows the power supply scheme of the oscillators, flash memory, analog circuitry and the port buffers of the V850E2/FK4-H devices.

For information about the power supply of the digital circuits of the different power domains refer to section *"Power supply of the digital circuits"* above in this chapter.

Table 38-2 V850E2/FK4-H power supply pins

Pin	Modules	Port buffers
• REGnVDD/REGnVSS/ REGnC[a] • CVDD	Voltage supply of the digital circuits of the different power domains, refer to section *"Power supply of the digital circuits""* above in this chapter.	–
E0VDD/E0VSS	analog circuits of Voltage Comparators VCPC	JP0, P0
E1VDD/E1VSS	–	P1, P2, P3, P4
B0VDD/B0VSS	–	P21, P24, P25, P27
OSCVDD/OSCVSS	MainOsc, SubOsc	–
A0VDD/A0VSS/ A0VREFP/A0VREFM	analog circuits of A/D Converter ADCA0	P10, P11
A1VDD/A1VSS/ A1VREFP/A1VREFM	analog circuits of A/D Converter ADCA1	P12, P13
FVDD/VSS	Flash memory	–

a) n = 0, 1

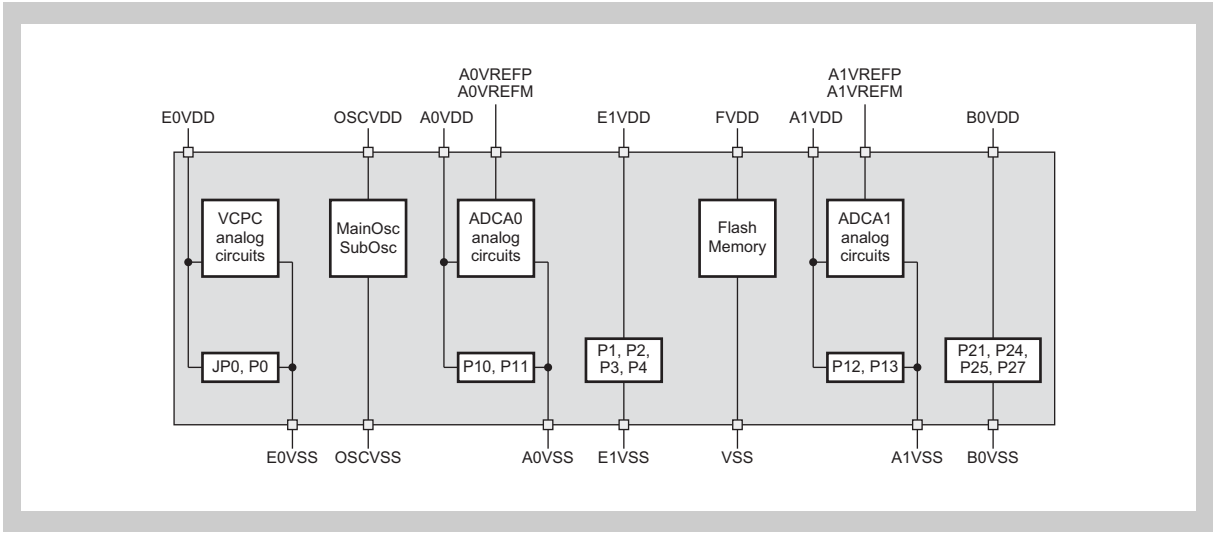


Figure 38-2 V850E2/FK4-H power supply scheme

38.2.3 V850E2/FL4-H power supply scheme

The following table and diagram shows the power supply scheme of the oscillators, flash memory, analog circuitry and the port buffers of the V850E2/FK4-H devices.

For information about the power supply of the digital circuits of the different power domains refer to section *"Power supply of the digital circuits"* above in this chapter.

Table 38-3 V850E2/FL4-H power supply pins

Pin	Modules	Port buffers
• REGnVDD/REGnVSS/ REGnC[a] • CVDD	Voltage supply of the digital circuits of the different power domains, refer to section *"Power supply of the digital circuits""* above in this chapter.	–
E0VDD/E0VSS	analog circuits of Voltage Comparators VCPC	JP0, P0
E1VDD/E1VSS	–	P1, P2, P3, P4
B0VDD/B0VSS	–	P21, P24, P25, P27
OSCVDD/OSCVSS	MainOsc, SubOsc	–
A0VDD/A0VSS/ A0VREFP/A0VREFM	analog circuits of A/D Converter ADCA0	P10, P11
A1VDD/A1VSS/ A1VREFP/A1VREFM	analog circuits of A/D Converter ADCA1	P12, P13
FVDD/VSS	Flash memory	–

a) n = 0, 1

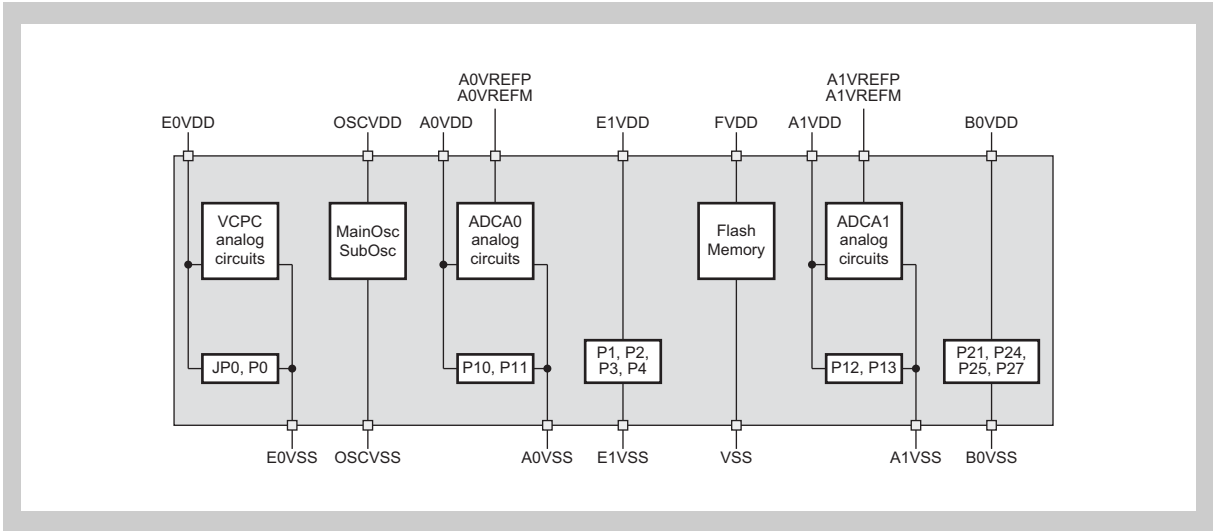


Figure 38-3 V850E2/FL4-H power supply scheme

38.3 Power-up and down procedures

This section describes the power-up and power-down procedures.

Cautions

1. The Power Sequencer power up mode control flash mask option OPBT0.OPBT0[0] must be set to 1.
2. All figures are only for explanatory purposes without any relevance to the real hardware implementation.

For detailed information about voltage levels, time and frequency values refer to the Electrical Target Specification.

The following figure outlines the power control.

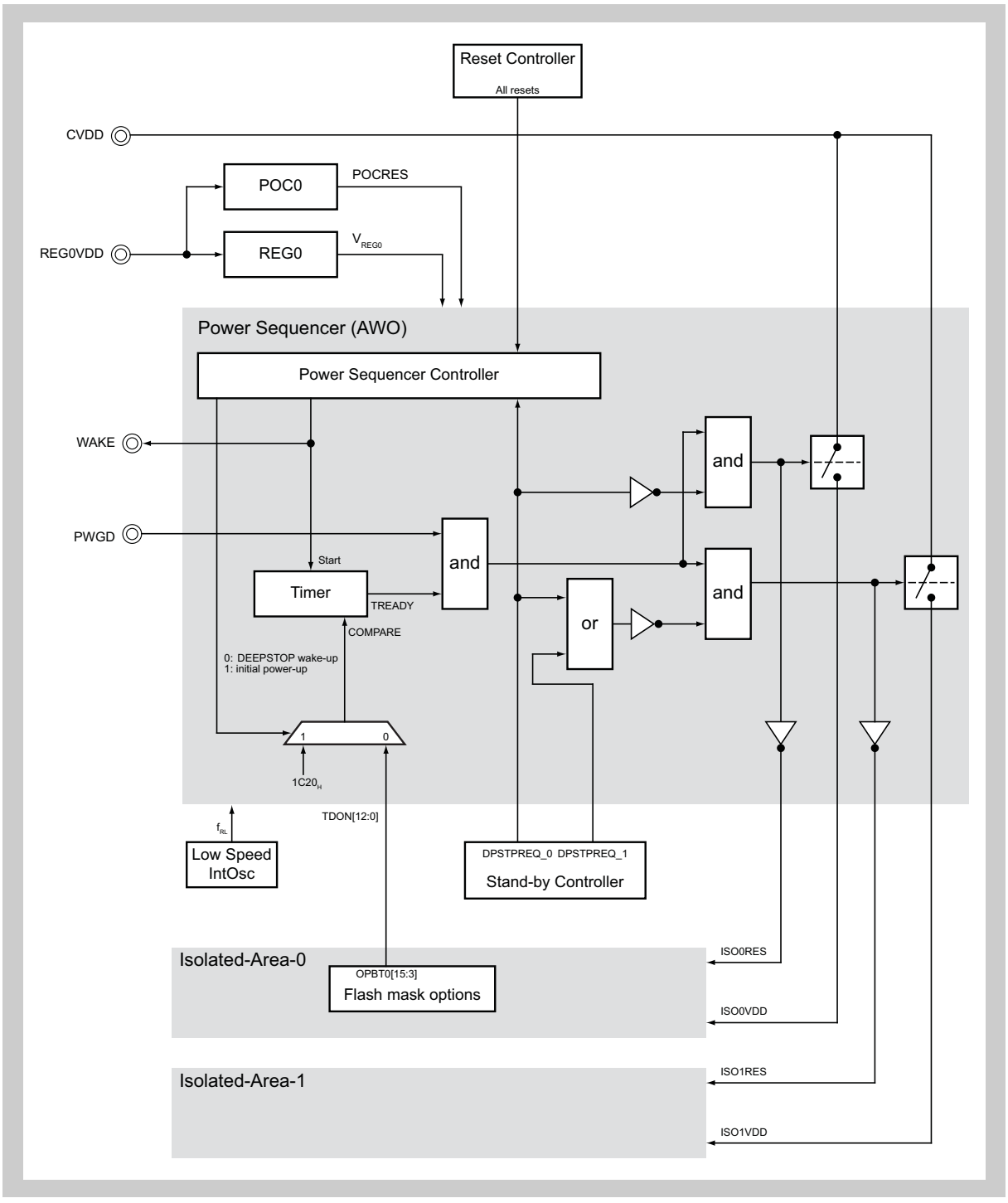


Figure 38-4 Power controller scheme

(1) Always-On-Area supply

The supply voltage V_{REG0} of the Always-On-Area is generated by the on-chip voltage regulator REG0 from the input voltage REG0VDD.

REG0VDD is observed by the Power-On-Clear circuit POC, that keeps the Always-On-Area in reset state (POCRES) as long as REG0VDD voltage is below the POC level V_{POC}.

(2) Isolated areas supply

The isolated areas supply voltages ISO0VDD and ISO1VDD are supplied by the external CVDD.

Power-up and -down of the isolated areas is necessary in two situations:

- initial power-up and final power-down
  The entire microcontroller is powered up respectively down, including the Always-On-Area.
- DEEPSTOP mode entry and wake-up
  In DEEPSTOP mode only the supply of the isolated areas ISO0VDD/ ISO1VDD are switched off, while the power supply of the Always-On-Area remains active.
  DEEPSTOP mode entry and wake-up is signalled by the Stand-by Controller signals
  - DPSTPREQ_0: asserted in DEEPSTOP mode
  - DPSTPREQ_1: asserted in Isolated-Area-1 DEEPSTOP mode

Isolated areas supply during reset

Any reset switches off the power supplies of the isolated areas 0 and 1. Thus after a reset the isolated areas have the same status as after initial power-up or after wake-up from DEEPSTOP mode, i.e. they have to be completely re-initialized.

Note Be aware that also after any reset the internal CPU RAM content is undefined.

38.3.1 Power Sequencer

The Power Sequencer controls the correct power-up and -down of the isolated areas 0 and 1.

It is located on the Always-On-Area and starts operation if the AWO power supply V_{REG0} is stable and Power-On-Clear reset POCRES is released.

For controlling the isolated areas the Power Sequencer

- switches on the isolated areas power supply ISO0VDD/ISO1VDD if the isolated area supply CVDD is stable, respectively switches off ISO0VDD/ ISO1VDD when CVDD is not stable
- controls the isolated areas reset ISO0RES/ISO1RES in order to avoid operation of the isolated areas, if their power supply is not stable.

The Power Sequencer assumes the external isolated areas power supply CVDD as stable, if

- the Power Sequencer timer has elapsed
  The Power Sequencer timer ensures a minimum power-up time T_{PUMIN}.
- the external PWGD signal is set to high level
  The isolated area power source can signal stable CVDD by asserting the PWGD signal.

(1) Power Sequencer timer

The Power Sequencer involves a timer, that counts up Low Speed IntOsc periods with f_{RL} to a certain COMPARE value and generates the TREADY signal.

The timer count time is

$$T_{PU} = COMPARE \times 1/f_{RL}$$

Thus the COMPARE value sets the minimum power-up time T_{PUMIN} between POCRES release and power-up of the isolated areas.

COMPARE The COMPARE value depends on the kind of power-up:

- Initial power-up
  The COMPARE value is fixed to $1C20_H$ = 7200. At initial power-up the Low Speed IntOsc is not trimmed and with COMPARE = 7200 the initial run time $T_{PU} = T_{PUINIT}$ of the Power Sequencer timer during initial power-up is minimum 10 ms.
- DEEPSTOP wake-up
  The COMPARE value is TDON[12:0].
  TDON[12:0] is a flash mask option and can be set by the user.
  That means the wake-up time from DEEPSTOP mode can be adjusted to the application environment and thus be minimized ($T_{PU} = T_{PUMIN)}$.

(2) PWGD signal

The PWGD is asserted by the external isolated area CVDD power source to signal stable CVDD voltage.

PWGD is also used to detect a fail of the isolated areas power supply.
Refer to the section *"Power-Fail"* below in this chapter.

Caution If the external CVDD source does not provide a status signal to be connected to PWGD, PWGD has to be connected to E0VDD.

(3) WAKE signal

The WAKE output signals to the external isolated area voltage supply CVDD that it

- must be switched on and become stable, if WAKE is high level
- can be switched off, if WAKE is low level

38.3.2 Initial power-up and final power-down

The following diagram shows the principle flow of an initial power-up and final power-down.

Note The diagram shows only the functional dependencies and disregards all chip-internal delay and stabilization times.
Refer to the Electrical Target Specification for details about these times.

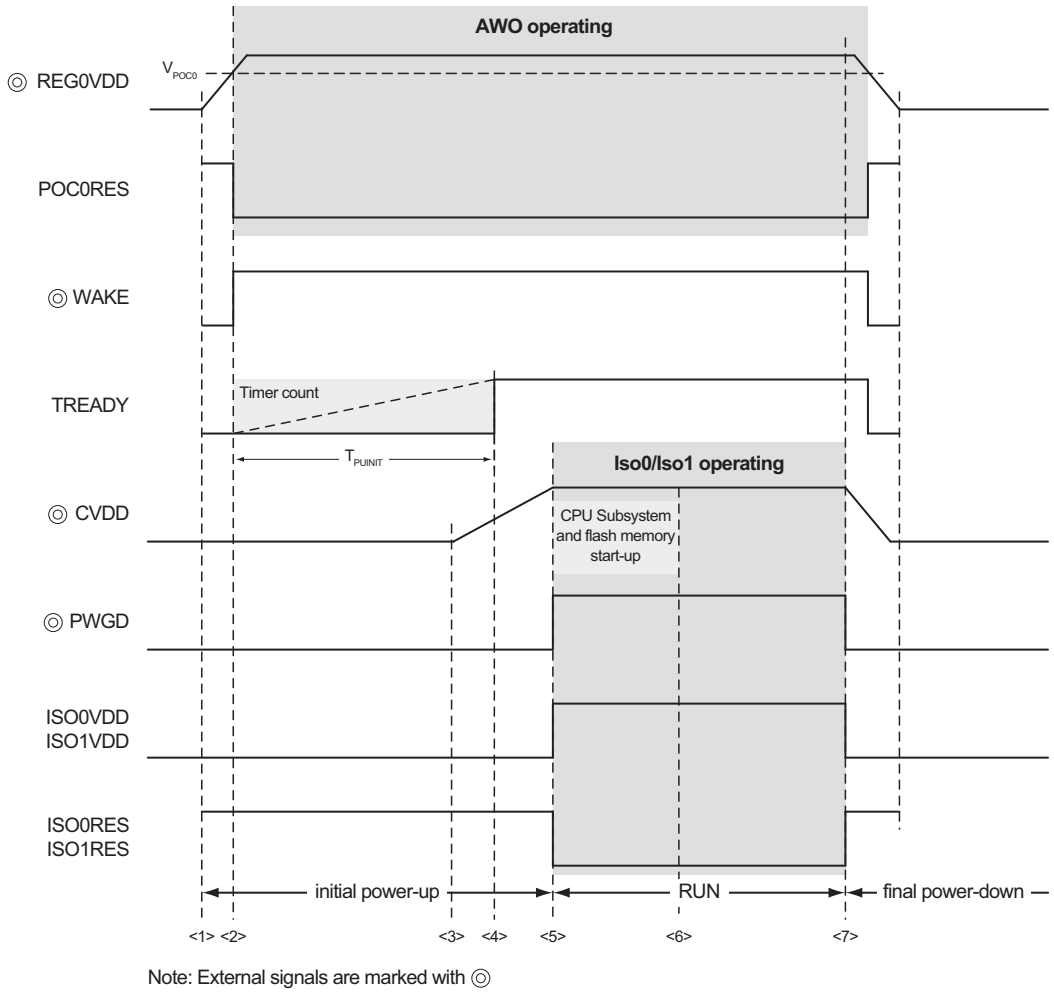


Figure 38-5 Initial power-up and final power-down

<1> The REG0 voltage regulator input REG0VDD ramps up, while the POC0RES keeps the microcontroller in reset state.

<2> If REG0VDD exceeds the POC0 voltage level V_{POC0}, POC0RES is released and the Always-On-Area is in operation.
The Power Sequencer starts the isolated areas power-up process by starting its timer and asserting the WAKE signal.

<3> CVDD ramps up.

<4> The Power Sequencer initial power-up time T_{PUINIT} has elapsed.

<5> If CVDD is stable, the external CVDD source changes PWGD to high level. The Power Sequencer switches on ISO0VDD/ISO1VDD and releases ISO0RES/ISO1RES.
The isolated areas are in operation.

<6> After start-up of the CPU Subsystem and the flash memory the CPU fetches its first instruction.

<7> For final power-down all power supplies are powered down.

Cautions

1. At initial power-up the external isolated areas supply CVDD must not be applied before the REG0VDD voltage for the Always-On-Area.
2. At final power-down the Always-On-Area supply REG0VDD must not be switched off before the external isolated areas supply CVDD.

Refer to the Electrical Target Specification for details about the timing relation between REG0VDD and CVDD.

38.3.3 DEEPSTOP entry and wake-up

In DEEPSTOP modes the power supply of the isolated areas are switched off:

- DEEPSTOP mode:
  power supply of Isolated-Area-0 and Isolated-Area-1 are switched off
- RUN mode (ISO1 DEEPSTOP) and STOP mode (ISO1 DEEPSTOP):
  power supply of Isolated-Area-1 is switched off

In either case the Always-On-Area remains active.

The Stand-by Controller generates indication signals, which depend on the stand-by mode:

- DPSTREQ_0 indicates Isolate-Area-0 and Isolated-Area-1 DEEPSTOP request
- DPSTREQ_1 indicates Isolated-Area-1 DEEPSTOP request.

(1) Voltage regulator control in DEEPSTOP mode

During DEEPSTOP mode (ISO0VDD/ISO1VDD switched off) the external voltage supply CVDD can be powered down as well and the on-chip voltage regulator REG0 can be set into an especial power save mode.

Powering down the external CVDD voltage and setting REG0 into power save mode during DEEPSTOP reduces power consumption to minimum, but lengthens the wake-up time, since the output voltage stabilization time of the REG0 regulator has to be regarded.

In order to minimize the wake-up time, the external CVDD supply can be kept active and the on-chip voltage regulator REG0 can stay active in DEEPSTOP. However keeping CVDD and REG0 active during DEEPSTOP increases the DEEPSTOP power consumption.

The bit PSC1.PSC1REGSTP determines the behaviour of the voltage regulator REG0 and of the WAKE output signal during DEEPSTOP modes:

- PSC1.PSC1REGSTP = 0 (refer to case (a) in the figure below):
  - WAKE remains on high level in DEEPSTOP:
    CVDD must not be powered down.
  - On-chip regulator REG0 remains in normal operation mode.
  - At wake-up only the minimum run time of the Power Sequencer timer T_{PUMIN} has to pass, before the voltage supply of the isolated areas is switched on.
- PSC1.PSC1REGSTP = 1 (refer to case (b) in the figure below):
  - WAKE turns to low level in DEEPSTOP:
    CVDD can be powered down.
  - On-chip regulator is switched into power save mode.
  - At wake-up the concurrent
    - stabilization of the on-chip regulator REG0
    - run time T_{PUMIN} of the Power Sequencer timer
    - power-up of CVDD
    determine the time for switching on the isolated areas supply voltage.

The figure below illustrates the behaviour in both cases.

Note Refer to the Electrical Target Specification for the specification of

- the regulator stabilization time (T_{RAA})
- the minimum Power Sequencer run time (T_{PUMIN}), refer also to *"Power Sequencer timer"* below.

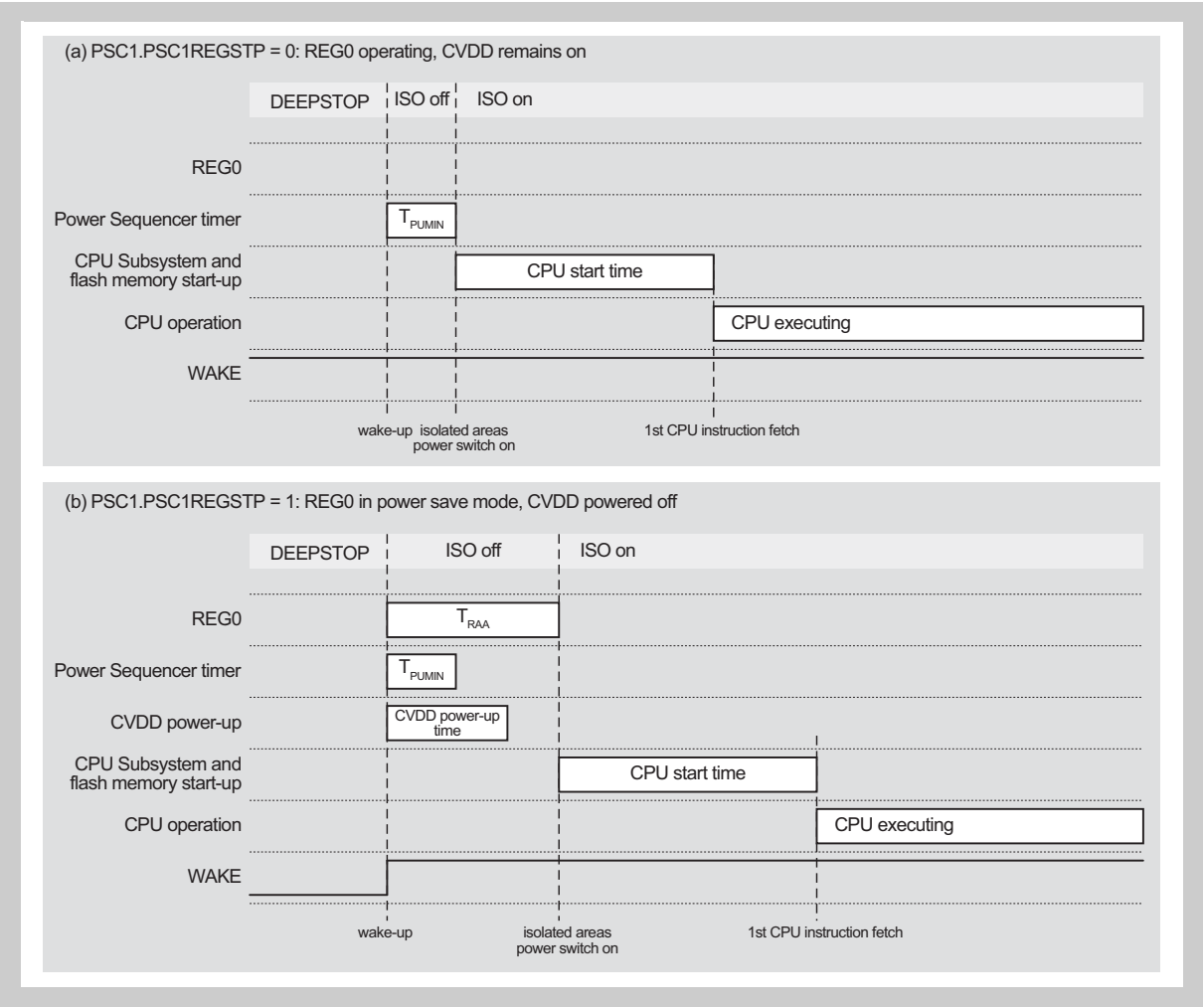


Figure 38-6 Voltage regulator control and DEEPSTOP wake-up time

Notes

1. In RUN mode (ISO1 DEEPSTOP) and STOP mode (ISO1 DEEPSTOP) REG1VDD must not be powered down (WAKE signal remains on high level), since the Isolated-Area-0 is active.
2. The default value “0” of bit 4 of the PSC0 register must be changed to “1” before entering DEEPSTOP mode.

Refer to the description of the *“PSC1 – Power save control register 1”* in the *“Stand-by Controller”* chapter.

(2) Power Sequencer timer

The Power Sequencer timer must span the minimum timer run time

T_{PUMIN} = TDON[12:0] x $1/f_{RLmax}$

Refer to the Electrical Target Specification for f_{RLmax}.

Caution Make sure to set the flash mask option TDON[12:0] correctly in order to guarantee a minimum Power Sequencer timer run time T_{PUMIN}.

For instance, with T_{PUMIN} = 1 ms and f_{RLmax} = 260 kHz,

TDON[12:0] = 1 ms x 260 kHz = 260 = 104_H.

(3) DEEPSTOP entry and wake-up

The following diagram shows the principle DEEPSTOP entry and wake-up flow, when the isolated areas voltage supply CVDD is powered down during DEEPSTOP (PSC1.PSC1REGSTP = 1).

Note The diagram shows only the functional dependencies and disregards all chip-internal delay and stabilization times.
Refer to the Electrical Target Specification for details about these times.

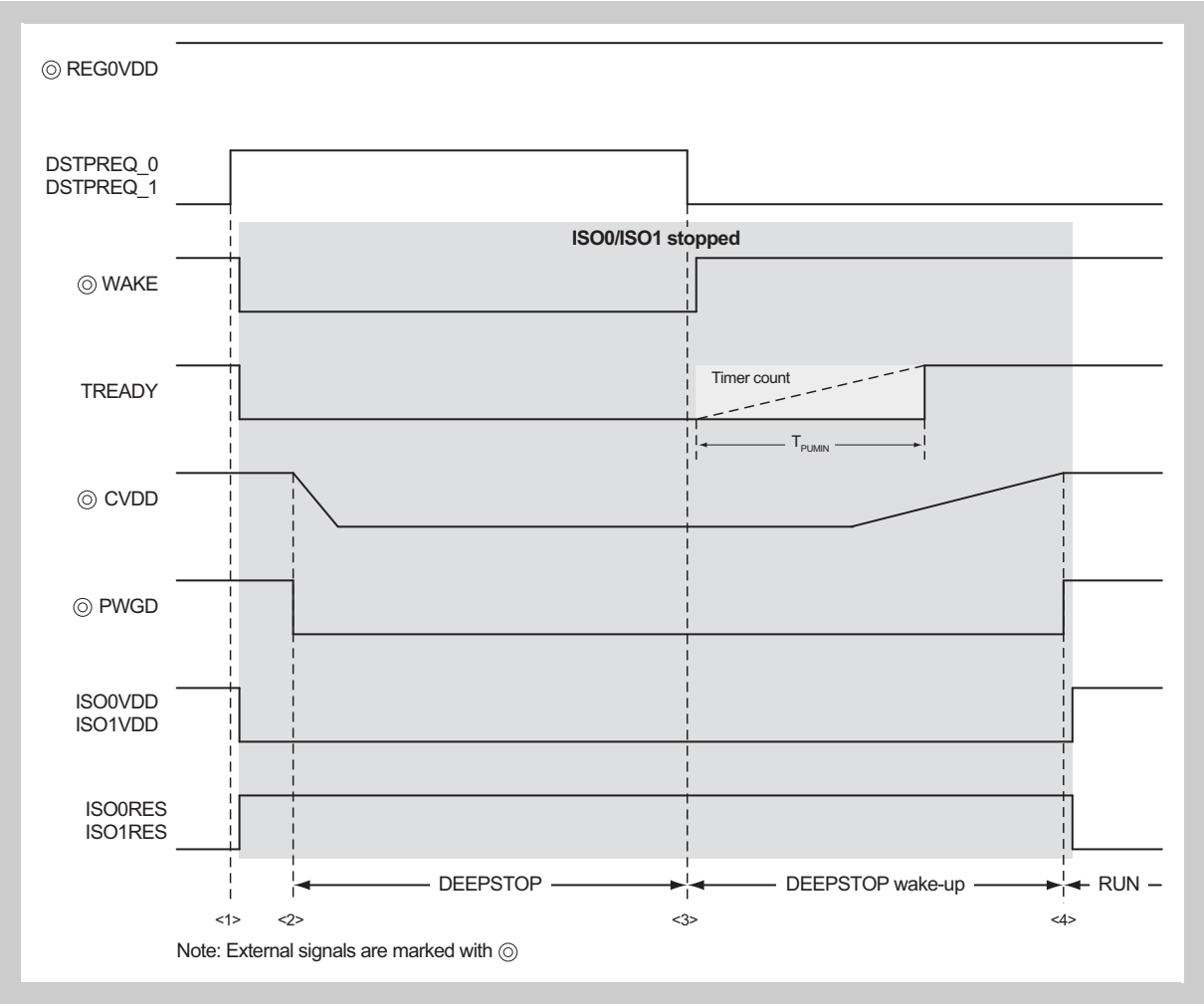


Figure 38-7 DEEPSTOP entry and wake-up ((PSC1.PSC1REGSTP = 1)

<1> Upon a DEEPSTOP request (DPSTPREQ_0 = DPSTPREQ_1 = 1)

- the WAKE signal is set to low level
- the isolated areas voltage supply ISO0VDD/ISO1VDD are switched off, resets are asserted (ISO0RES = ISO1RES = 1).

<2> The external voltage CVDD can be powered down and the PWGD signal can be set to low level.

<3> A wake-up event de-asserts DPSTPREQ_0/DPSTPREQ_1, causing the Power Sequencer to start the isolated areas power-up process by asserting the WAKE signal and starting the Power Sequencer timer.
The count time T_{PUMIN} is determined by the TDON[12:0] value, and thus by the flash mask option OPBT0.OPBT0[15:3].
CVDD must be powered up and stabilized.

<4> The Power Sequencer minimum wake-up time T_{PUMIN} has elapsed, PWGD is set to high level, CVDD is stable, the isolated areas supplies ISO1VDD/ISO2VDD are switched on and their resets ISO1RES/ISO2RES are released.

DEEPSTOP with PSC1REGSTP = 0

In case DEEPSTOP mode is entered with PSC1.PSC1REGSTP = 0, the WAKE remains at high level. This can be used to avoid power down of the external CVDD supply and to keep the on-chip regulator in normal operation.

38.3.4 Power-Fail

The PWGD is asserted by the external isolated area CVDD power source to signal stable CVDD voltage.

During operation of the isolated areas the Power Sequencer observes the PWGD signal. If the CVDD supply fails for any reason and the PWGD signal changes to low level, a power fail status is detected and

- the isolated areas power supplies ISO0VDD/ISO1VDD are switched off
- the isolated areas resets ISO0RES/ISO1RES are asserted

This mode is referred to as Power-Fail mode.
Refer to the section *"DEEPSTOP mode"* in the *"Stand-by Controller (STBC)"* chapter for further details.

After CVDD is applied again, the microcontroller behaves like after a wake-up from DEEPSTOP mode, i.e. the CPU starts from its reset state.

Note If the external CVDD source does not provide a status signal to be connected to PWGD, PWGD has to be connected to E0VDD.
Consequently power fail detection is not possible.

Note The diagram shows only the functional dependencies and disregards all chip-internal delay and stabilization times.
Refer to the Electrical Target Specification for details about these times.

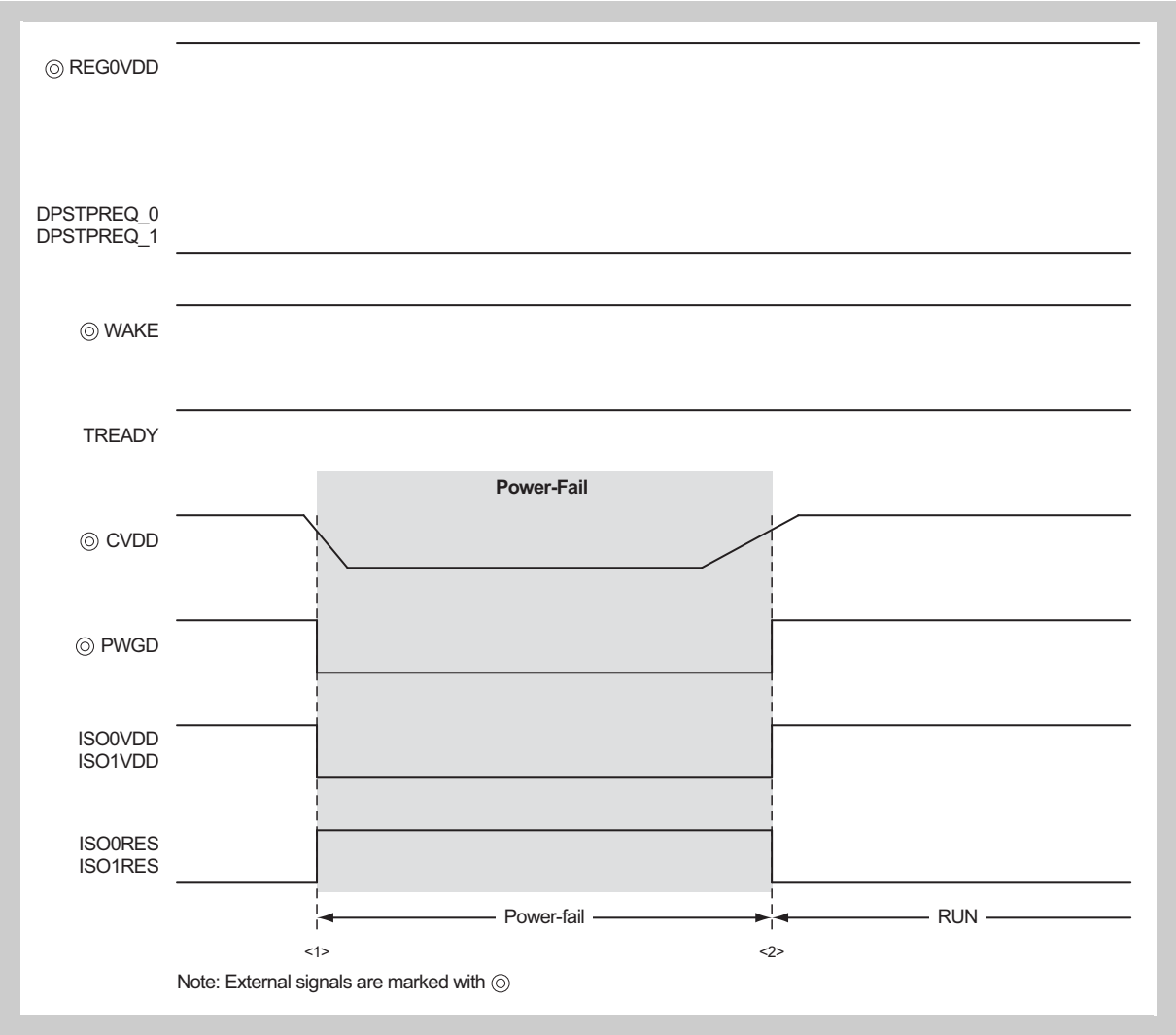


Figure 38-8 Power-Fail

<1> The external voltage supply CVDD is powered down due to a power fail and PWGD turns to low level.
The isolated areas power supply ISO0VDD/ISO1VDD are switched off and these power domains are set in reset (ISO0RES = ISO1RES = 1).

<2> If CVDD is powered up again, indicated by PWGD high level, isolated areas power supply ISO0VDD/IOS1VDD are switched on, and their resets are released.
The CPU starts operation as after reset.

38.3.5 Other power supplies

This section describes how to handle the power supplies, which are not used to power the Always-On-Area and the isolated areas.

The following table summarizes all power supplies and their power-up and -down conditions and options.

Table 38-4 Power supplies

Pin	Supply	Condition
REG0VDD	• Always-On-Area digital circuits	• all must be supplied simultaneously • must not be stopped in DEEPSTOP mode
FVDD	• flash memory	
OSCVDD	• MainOsc, SubOSC	
E0VDD	• analog circuits of Voltage Comparator • port groups JP0, P0 buffers	
E1VDD	• port groups P0 to P4 buffers	
B0VDD	• port groups P21, P24 to P29 buffers	
A0VDD	• analog circuits of ADCA0 • port groups P10, P11	• must not be supplied before REG0VDD • must be supplied and stable if ADCA0 is used • may be stopped in DEEPSTOP mode
A1VDD	• analog circuits of ADCA1 • port groups P12, P13	• must not be supplied before REG0VDD • must be supplied and stable if ADCA1 is used • may be stopped in DEEPSTOP mode
CVDD	• Isolated areas digital circuits	• must not be supplied before REG0VDD • may be stopped in DEEPSTOP mode

Note For details concerning the power supply pins handling refer to the Electrical Target Specification.

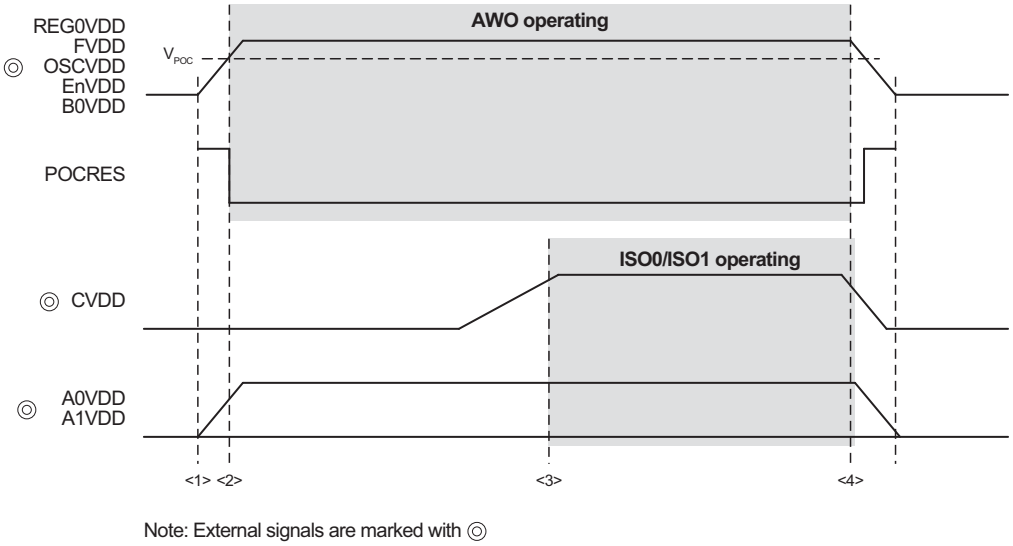


Figure 38-9 Other power supplies

<1> The REG0VDD, FVDD, OSCVDD, EnVDD, B0VDD ramp up, while the POCRES keeps the microcontroller in reset state.

<2> If REG0VDD exceeds the POC voltage level V_{POC}, POCRES is released and the Always-On-Area is in operation.
The Power Sequencer starts the isolated areas power-up process, as described in the section *"Power Sequencer"* above.

<3> CVDD is stable and the isolated ares are switched on.

<4> For finally power down the entire microcontroller, CVDD is switched off, the isolated areas set in reset (ISO0RES/ISO1RES = 1) and switched off.
REG0VDD can be switched off as well with CVDD, but not before.
If REG0VDD drops below V_{POC0}, the entire microcontroller is reset (POCRES = 1) and the Always-On-Area switched off.

The A/D Converters power supplies A0VDD/A1VDD can be applied and switched off upon demand
However they must not be active, when REG0VDD is switched off.

Revision History

The table below gives an overview about the revision history of this document.

Revison history overview (1/3)

Rev.	Date	Summary
0.01	May 05, 2010	first edition
0.02	Dec 22, 2010	• Introduction - product name register added • Port Functions - section "Non-port input/output signals" added - section “Recommended connection of unused pins” revised • CPU System Functions - section "Mode pins and JP0 connections" added • External Memory Controller (MEMC) - chapter completely revised • DMA/DTS Controller (DMAC) - DTS controller description added • Clock Controller - description concerning Clock Controller reset/stand-by behaviour added - CLMATERR signal removed from Clock Monitor description • Stand-by Controller - “Stand-by Contoller signal conenctions” and “Oscillators wake-up” added • OS Timer (OSTM) - external signals for count enable, synchronization and count output removed • A/D Conterter (ADCA) - ADCA0 H/W triggers completed • Power Supply - chapter completely revised • several corrections and additions

Revison history overview (2/3)

Rev.	Date	Summary
1.00	Aug 04, 2011	• Introduction - V850E2/FM4-H products removed from entire document • CPU System Functions - section “Data flash wait cycle control” added • Interrupt Functions - section “Exceptions and Interrupts” improved - section “V850E2/Dx4(-H) Exceptions” added - section "Code flash error correction" added • Clock Controller - stand-by stop acknowledge bits from oscillators and PLL status registers removed • Stand-by Controller (STBC) - complete revision of the chapter • Reset Controller - Power-Fail description, incl. PWGDF/PWGDFC registers, moved to chapter “Power Supply” • Motor Control - section “Basic structure of motor control” added - section “Timer Motor Control Function (TAPA)” completely revised - section “Three-Phase PWM Output with Dead Time” added - section “High-accuracy Triangle PWM Output with Dead Time” added - section ”Delay Pulse Output with Dead Time” added • Encoder Timer - section “Basic structure of Encoder Timer” added - section “Encoder Timer (ENCA)” completely revised - section “A/D Trigger Encoder Capture” added - section “Synchronized Timer Operation” added - section “Trigger Pulse Width Measurement” added • PWM Diagnostic - basic funcional description added - “PWM Delay (DLYA)” and “PWM Diagnostic (PMCA)” merged into new “PWM Diagnostic” chapter • CAN Controller (FCN) - section "CAN Controller (FCN) baudrate and time quanta" added - error counter clear procedure changed - sleep mode and dominant bus behaviour defined • Diagnostic CAN Controller (DCN) - section "Diagnostic CAN Controller (DCN) baudrate and time quanta" added • On-Chip Debug Unit (OCD) - Emulation break control register EPC removed (function can be executed via the debugger) • Power Supply - complete revision of section “Voltage regulator control in DEEPSTOP mode” - Power-Fail description added

Revison history overview (3/3)

Rev.	Date	Summary
1.01	Mar 15, 2012	• Introduction - V850E2/FK4-3M, V850E2/FK4-4M, V850E2/FL4-3M, V850E2/FL4-4M, products removed from entire document • Port Functions - port filters behaviour with respect to stand-by modes revised • CPU System Functions - section "Module wait clocks insertion" added • Stand-by Controller (STBC) - correction: set CPUCLK to IntOsc clock source before entering STOP mode • Reset Controller - section "MainOsc during RESET" added • Asynchronous Serial Interface E (URTE) - receive/status interrupt INTUAEnTRA completely removed • I^2C Interface (IICB) - status interrupt connection to Interrupt Controller added • On-Chip Debug Unit (OCD) - Emulation break control register EPC removed (function can be executed via the debugger)
1.02	Nov 07, 2012	• Introduction - TAUC8 removed from FL4-H - 208-pin QFP package removed for FL4-H • CPU System Functions - for boundary scan mode setting: P0_4 must have low level • Stand-by Controller (STBC) - STOP and DEEPSTOP flows corrected with respect to oscillators re-start - Precaution: "Clock generators and early wake-up" added • Timer Array Unit B (TAUB) - TAUB1 input selection added • A/D Converter A (ADCA) - new specification of H/W trigger edge selection • Voltage Comparator (VCPC) - chapter completely revised • Power Supply - power fail indicator PWGDF removed
2.00	Apr 29, 2013	• Clock Controller - PLL SSCG downspread mode removed

For details about the modifications in each revision refer to the related section below.

Revision History Rev. 0.02

This revision history list shows all functional changes of

Rev. 0.02 compared to Rev. 0.01.

Note The chapter and page numbers in the table below refer to the older Rev. 0.02 and thus may not be valid for the current revision.

Revision history of Rev. 0.02 (1/2)

Chapter	Page	Description
1	53	PRDNAME register moved to "Introduction" chapter
1	53	Product name register register added
2	58	description for PMSRn/PMCSRn usage added
2	60	note added concerning usage of PSRn, PNOTn
2	62	registers access width corrected
2	62	JPFCEn registers removed
2	83	"V850E2/Fx4 Port Group Configuration" reordered
2	84	"Initialization of port control registers" added
2	86	ADCA register settings added to use ADCA ports in input port mode
2	141	section concerning recommendation of unused pins connection revised
2	145	caution added concerning filter bypass
2	156	caution added concerning filter bypass
3	166	caution added for "Memory Protection Unit (MPU)"
3	184	section "Mode pins and JP0 connections" added
3	186	correction: R3 resistors at DCUTDO and DCUTRDY
3	200	correct setting for ATEAPMI added
3	212	description of ATEAPMI register corrected
3	212	ATEAPMI description corrected
3	221	caution added concerning write to PROTSn registers
3	222	bitnames from CLMAnPCMD removed
3	223	PPCMDn access width changed to 8-bit
3	223	caution added concerning write to PPROTSn registers
3	224	bitnames from FLMDPCMD removed
4	225	"External Memory Controller (MEMC)" revised
5	322	"Restore from EI level maskable interrupt (EIINT)" diagram corrected
7	474	self-programming library RAM changed to 4 KB
9	494	section "General Description of Clock Generation and Control" completely revised
9	495	description concerning Clock Controller reset/stand-by behaviour added
9	515	caution added concerning clock selection changes
9	515	step 2 added in clock selector ID process
9	551	duty cycle calculation added
9	558	section "CLMA enable and suspend/resume" revised
9	560	removed CLMATERR signal
9	587	caution added: SSCG not for PLL1

Revision history of Rev. 0.02 (2/2)

Chapter	Page	Description
10	598	section "Stand-by Controller signal connections" added
10	615	DEEPSTOP exit procedure corrected
12	648	VLVI explanation improved
12	648	correction: VLVI only for BURAM
13	664	count enable, synchronization and count output removed
14	681	"WDTA Start-up Options" revised
14	681	Varying activation code feature added in "Window Watchdog Timer (WDTA)" chapter
15	703	TAUAn ADC H/W triggers corrected
15	704	"TAUA Input Selection" completely revised
15	900	slave channel 3 output mode for "Delay Pulse Output Function" corrected
15	900	removed limitation about TAUAnCDRm registers content
25	1779	setting of URTEnCTL1.URTEnSLBM corrected
25	1803	32-bit register set added
26	1836	message buffer register addresses corrected
26	1855	cautions added to FCNnCMCLCTL register description
26	1885	figure changed and note added in "Receive data read" section
26	1925	"Wait for 1CAN data bits" box removed
29	2124	CSIHnCTL1.CSIHnCKR bit added
29	2141	caution added to CSIHnTX0W register description
29	2142	caution added to CSIHnTX0H register description
34	2679	RNG Maurer test removed
36	2690	caution removed: rising or falling edge selection possible
36	2691	ADCAn H/W triggers completely revised
36	2697	added references for simultaneous sampling channels
36	2715	total conversion time corrected
36	2753	explanation of ADCAnRCK[23:00] corrected

Revision History Rev. 1.00

This revision history list shows all functional changes of

Rev. 1.00 compared to Rev. 0.02.

Note The chapter and page numbers in the table below refer to the older Rev. 1.00 and thus may not be valid for the current revision.

Revision history of Rev. 1.00 (1/4)

Chapter	Page	Description
2	88	port group 13 removed from FK4-H
2	121	"Port and pin function during and after reset" added
2	128	port filters for CSIG2 signals added
2	134	calculations of filter characteristics corrected
3	149	PBUS/HBUS byte/halfword access descriptions corrected
3	153	"Overall PBUS latencies" added
3	157	instruction fetch from data flash, PBUS I/F and HBUS I/F removed
3	158	caution added concerning instruction fetch from data RAM
3	161	section "Data flash wait cycle control" added
3	170	note added concerning debugger I/F ports in debug mode
3	192	ETAERR register added
3	193	ETAWLEA register added
3	193	ETAWLEM register added
3	194	ETARCFGn register description corrected
3	198	ATEEEA register added
3	199	ATELEA register added
3	200	ATESR register added
3	201	ATESHL register added
5	254	section "Exception and interrupts" improved
5	257	new section "Exceptions"
5	262	section "Code flash error correction" added
6	347	DSAn 32-bit access description added
6	354	DDAn 32-bit access description added
7	450	data flash reading also via data flash access library
7	459	"Version acquisition" command removed
7	469	WDTA VAC flash mask options added
9	488	impact of DEEPSTOP on clock generators corrected
9	503	PLLCk.PLLCkP[1:0] settings corrected
9	506	Clock switching timing formulas corrected
9	507	PLL dividers 3, 5, 6, 10 removed from all clock selections
9	511	correction: STPMK_000 available, but msut remain 0
9	535	"Clock change delay" description added
9	538	CLMAn error signals added
9	538	correction: CLMA0 interrupt is not connected

Revision history of Rev. 1.00 (2/4)

Chapter	Page	Description
9	555	MOSCSSTPACK bit removed
9	560	SOSCSSTPACK bit removed
9	562	initial value of ROSCE corrected
9	564	ROSCSSTPACK bit removed
9	566	PLLSkSTPACK bit removed
9	571	caution added concerning CKSC_1m and PROTCMD1/PROTS1 access in Iso1 DEEPSTOP mode
9	573	caution added concerning CSCSTAT_1m access in Iso1 DEEPSTOP mode
10	579	chapter "Stand-by Controller functions" completely revised
10	584	description about "Module clocks during transition to stand-by mode" added
10	589	section "Oscillators wake-up" revised
10	596	caution added concerning Iso1 register access during Iso1 STOP
10	599	caution added concerning Iso1 register access during Iso1 STOP
10	602	caution added concerning Iso1 register access during Iso1 DEEPSTOP
10	606	caution added concerning Iso1 register access during Iso1 DEEPSTOP
10	611	correction: PSC0IOHLDMSK bit can be read back
10	613	correction: PSC01HLDMSK bit can be read back
10	613	function of PSC1REGSTP corrected
10	614	correction: PSC1IOHLDCLR bit read returns always 0
12	628	PowerGood description, incl. PWGDF/PWGDFC registers, move to chapter "Power Supply"
12	629	section added concerning isolated areas power supply during reset
12	642	S/W reset control bit name corrected to SWRESA0
14	667	exchanged "varying activation code" by "variable activation code" and wording changed
15	1055	TAUAnBRS register width corrected to 8 bit
15	1062	TAUAnCMURm register width corrected to 8 bit
15	1063	TAUAnCSRm register width corrected to 8 bit
15	1064	TAUAnCSCm register width corrected to 8 bit
16	1296	TAUBnCMURm register width corrected to 8 bit
16	1297	TAUBnCSRm register width corrected to 8 bit
16	1298	TAUBnCSCm register width corrected to 8 bit
18	1383	section "TAUJ Input Selection" completely revised
18	1484	TAUJnBRS register width corrected to 8 bit
18	1491	TAUJnCMURm register width corrected to 8 bit
18	1492	TAUJnCSRm register width corrected to 8 bit
18	1493	TAUJnCSCm register width corrected to 8 bit
18	1493	TAUJnTS register width corrected to 8 bit
18	1494	TAUJnTE register width corrected to 8 bit
18	1494	TAUJnTT register width corrected to 8 bit
18	1495	TAUJnTO register width corrected to 8 bit
18	1495	TAUJnTOE register width corrected to 8 bit

Revision history of Rev. 1.00 (3/4)

Chapter	Page	Description
18	1496	TAUJnTOM register width corrected to 8 bit
18	1497	TAUJnTOC register width corrected to 8 bit
18	1497	TAUJnTOL register width corrected to 8 bit
18	1498	TAUJnRDE register width corrected to 8 bit
18	1498	TAUJnRDM register width corrected to 8 bit
18	1499	TAUJnRDT register width corrected to 8 bit
18	1499	TAUJnRSF register width corrected to 8 bit
20	1558	chapter "Motor Control" completely revised
20	1585	section "Three-Phase PWM Output with Dead Time" added
20	1612	section "High-accuracy Triangle PWM Output with Dead Time" added
20	1651	section "Delay Pulse Output with Dead Time" added
21	1676	chapter "Encoder Timer" completely revised
21	1709	Added 'ENCA0 start by ADC trigger' discription
21	1709	section "A/D Trigger Encoder Capture" added
21	1716	section "Synchronized Timer Operation" added
21	1721	section "Trigger Pulse Width Measurement" added
25	1876	FCN baudrate setting table added
25	1895	caution added concerning FCNnGMCSPRE settings
25	1903	caution added concerning limited usage of FCNnCMCLERCF
25	1903	previous last note removed, previous first note covered by new caution, new note added
25	1903	reference to figure corrected
25	1904	note added concerning sleep mode and dominant blocked bus
25	1912	caution added concerning FCNnCMBRPRS settings
25	1913	time names in "Data bit time" diagram modified
25	1914	caution added concerning FCNnCMBTCTL settings
25	1933	section "Resetting error counter FCNnCMERCNT of FCN module" removed
25	1938	example for "Mask function" corrected
25	1962	baud rate settings conditions corrected
25	1970	S/W reset from initialization flow removed
25	1970	CAN initialization flow modified
25	1971	figure "Re-initialization without Software Reset function" corrected
25	1971	reference to figure corrected
25	1975	flow diagram "Message buffer redefinition during transmission" corrected
26	2024	figure "Re-initialisation of the DIAG_CH CAN module using 128 buffers without software reset" corrected
26	2026	figure "Re-initialisation of the DIAG_CH CAN Module using first 112 buffers only without software reset" corrected
26	2071	caution added concerning limited usage of DCNnCRCLERCF
26	2071	previous note 1 removed, note 4 added
26	2073	note added concerning sleep mode and dominant blocked bus
26	2085	DCNnCRBSSTR address corrected

Revision history of Rev. 1.00 (4/4)

Chapter	Page	Description
27	2097	usage about CSIGn handshake function described
29	2221	section "I2C Interface Port Settings" added
29	2237	behaviour of the acknowledge in slave mode corrected
29	2241	IICB clock setting formulas corrected
34	2744	Added caution for ADCA0TTRGi register changes
34	2748	ADCA chapter revised
34	2756	section"Start Trigger for the different CG before end of conversion" added
34	2757	caution added concerning stop trigger in countinuous mode
34	2766	section "Relationship between analog input voltage and A/D conversion result" added
34	2780	caution added concerning change of diagnosic setup while ADCA enabled
34	2793	sharing of channels between channel groups corrected
34	2810	ADCAnDBiCRL register added
34	2819	ADCAnSVSDIS bit position corrected
35	2820	number of Voltage Comparator channels for V850E2/FG4 corrected
35	2828	level of VCPCnSFm corrected in stand-by mode (VCPCnOEm = 0)
37	2837	IDCODE instruction added
38	2845	"Isolated areas power supply during reset" added
38	2849	section "Voltage regulator control in DEEPSTOP mode" completely revised
38	2853	section "Power-Fail" added
38	2855	Power-Fail flag indicator register PWGDF moved from "Rest Controller" to "Power Supply"

Revision History Rev. 1.01

This revision history list shows all functional changes of

Rev. 1.01 compared to Rev. 1.00.

Note The chapter and page numbers in the table below refer to the older Rev. 1.01 and thus may not be valid for the current revision.

Revision history of Rev. 1.01 (1/2)

Chapter	Page	Description
2	68	PBDCn register description corrected
2	72	read value of PSRn/JPSR0 register corrected
2	131	port filters behaviour with respect to stand-by modes revised
2	134	description of digital filter characteristics improved
2	135	caution added concerning undersampling with digital port filters
3	154	section "Module wait clocks insertion" added
3	164	DCLKWAIT register added
3	187	description of "Direct mode" improved
3	189	ETAWLEM register removed
3	193	ETAERR register width corrected: 16-bit only
3	193	ETAERR.ERE bit added
3	194	ETAEREA register added
3	195	initial value of ETAWLEA corrected: undefined
3	197	initial value of ETAADRSn corrected: undefined
3	198	initial value of ETAMASKn corrected: undefined
3	210	addresses of JPPCMD0/JPPROTS0 registers corrected
5	256	section "Exceptions and interrupts" improved
5	264	section "Code flash error correction" added
6	381	address of DTFR15 register corrected
6	408	diagram "Configuration of TI: corrected
6	433	TI fetch address description corrected
9	492	note added concening MainOsc and RESET
9	507	Clock switching timing formulas corrected
9	544	clock monitor formulas corrected
9	545	CLMAnOSEL setting corrected
9	557	description of MOSCCAMPSEL[1:0] modified
9	557	MOSCCSHTSTBY description improved
9	558	caution added concerning MainOsc operation during external RESET
10	586	"CPU clock source before STOP and DEEPSTOP mode" added
10	597	correction: set CPUCLK to IntOsc clock source before entering STOP mode
10	598	oscillators operation after STOP mode early wake-up described
10	603	correction: set CPUCLK to IntOsc clock source before entering DEEPSTOP mode
10	604	PLLk stop by S/W during DEEPSTOP entry removed
10	605	oscillators operation after DEEPSTOP mode early wake-up described

Revision history of Rev. 1.01 (2/2)

Chapter	Page	Description
10	612	application hints concerning wake-up handling added
12	644	description of "MainOsc during RESET" added
13	664	16-bit access to OSTMnCMP removed
13	665	16-bit access to OSTMnCNT removed
14	673	OPWDINT signal level corrected
15	727	correction: TAUAnTOE.TOEm can be written during count operation
16	1114	correction: TAUBnTOE.TOEm can be written during count operation
16	1297	initial value of TAUBnTPS corrected
17	1339	correction: TAUCnTOE.TOEm can be written during count operation
17	1375	initial value of TAUBnTPS corrected
18	1414	correction: TAUJnTOE.TOEm can be written during count operation
21	1686	Encoder timer basic structure diagram corrected
21	1690	caution added concerning initial filter state
24	1856	note added concerning missing or incomplete response receptions
24	1863	32-bit control and status register names corrected
25	1889	some description concerning CAN RAM added
25	1901	initial values of FCNnGMCLCTL register defined more precisely
25	1976	section "Representative examples of baud rate settings" removed and "Clock prescaler and baudrate generator settings" added
25	1995	Tx request abort processing flow corrected
27	2099	port configuration information added for data consistency check
27	2119	note added concerning overrun error in master mode
27	2129	data and clock phase description corrected
28	2138	port configuration information added for data consistency check
29	2223	status interrupt connection to Interrupt Controller added
30	2463	FLXnWRDSm access changed to read/write
34	2766	total conversion time specification corrected

Revision History Rev. 1.02

This revision history list shows all functional changes of

Rev. 1.02 compared to Rev. 1.01.

Note The chapter and page numbers in the table below refer to the older Rev. 1.02 and thus may not be valid for the current revision.

Revision history of Rev. 1.02 (1/2)

Chapter	Page	Description
1	45	number of TAUC units of FK4-H reduced to 5
2	84	corrrection: ETH0MDI/ETH0MDO operation with PIPn_m = 1
3	169	P0_4 pin added for boundary scan mode selection
3	171	P0_4 added to mode settings
3	212	caution added concerning PROTCMD1 access during Isolated-Area-1 DEEPSTOP mode.
3	213	caution added concerning PROTS1 access during Isolated-Area-1 DEEPSTOP mode.
4	232	"Multiplexed bus mode" diagram corrected
4	243	"Idle cycle" diagram corrected
5	258	section "Exception handler switching" added
9	506	caution corrected concerning clock selection changes
9	507	clock switching time calculations imporved
9	535	caution added concerning EMI behaviour of CSCXFOUT
10	585	Stand-by status overview table corrected
10	587	description about "Clock generators in stand-by mode" added and "CPU clock source before STOP and DEEPSTOP mode" removed
10	598	caution added concerning clock generators in STOP mode added
10	599	caution added concerning clock generators in STOP mode added
10	605	caution added concerning clock generators in DEEPSTOP mode added
10	606	caution added concerning clock generators in DEEPSTOP mode added
10	613	"Precaution: Clock generators and early wake-up" added
12	638	operation mode control added to Reset Controller
12	645	VLVI timing diagram improved
14	685	WDTA behaviour after error detection described
15	730	new terms for channel output modes defined (Direct, Independent, Synchronous Channel Output Mode)
16	1099	TAUB1 input selection options corrected
16	1121	new terms for channel output modes defined (Direct, Independent, Synchronous Channel Output Mode)
17	1346	new terms for channel output modes defined (Direct, Independent, Synchronous Channel Output Mode)
18	1421	new terms for channel output modes defined (Direct, Independent, Synchronous Channel Output Mode)
22	1765	DLYAn PWM signals connections corrected
23	1796	caution added concerning initial filter state

Revision history of Rev. 1.02 (2/2)

Chapter	Page	Description
24	1846	order of LMAnCTLL and LMAnCTLH setting corrected
24	1854	order of LMAnCTLL and LMAnCTLH setting corrected
24	1872	note added concerning LMAnCTLL change
25	1895	several namings correct (e.g. "module" to "channel", "FCN module system clock" to "CAN channel clock")
28	2206	description of CSIHnIDLx improved
30	2384	description improved
30	2388	description improved
30	2392	description improved
30	2421	initial value of FLXnMHDC corrected
30	2552	cautions concerning FlexRay usage added
33	2748	register and bit names corrected
34	2756	H/W trigger selection description improved
34	2805	description of ADCAnTiETS bits changed
34	2827	ADCAnCTL2 register description corrected
35	2838	Voltage Comparator chapter completely revised
38	2868	power fail indicator PWGDF removed

Revision History Rev. 2.00

This revision history list shows all functional changes of

Rev. 2.00 compared to Rev. 1.02.

Revision history of Rev. 2.00

Chapter	Page	Description
9	494	"Short stabilization time mode" description corrected
9	570	down spread dithering PLL mod removed
12	655	LVICNT[2:0] settings corrected
17	1348	"Independent Channel Output Mode 1" removed
22	1781	TRGSEL0 description corrected
22	1782	TRGSEL1 description corrected
22	1783	TRGSEL2 description corrected
22	1784	TRGSEL3 description corrected
22	1785	TRGSEL4 description corrected
23	1826	data consistency error detection timing specified more precisely

Index

Numerics

75% interrupt output (WDTA) 687

A

A/D Converter A (ADCA) 2752
<ADCAn_base> 2753
ADCA1 H/W trigger selections 2762
ADCAnSTR0 flag clear register (ADCAnSTC0) 2831
ADCAnSTR1 flag clear register (ADCAnSTC1) 2816
ADCHALT mode 2778
Base addresses 2753
Buffer amplifier function 2801
CGi buffer register (ADCAnDBiCR) 2825
CGi buffer result register (ADCAnDBiCRL) 2826
Channel and channel groups (CG) 2769
Channel group index i 2752
Channel group register i (ADCAnCGi) 2809
Channel index m 2752
Channel S&H control register (ADCAnSHCTL) 2813
Channel S&H function 2786
Clock supply 2753
Continuous conversion mode 2773
Conversion end interrupt INTADCAnTi 2780
Conversion modes 2771
Conversion result register for channel m (ADCAnCmCR) 2823
Conversion time 2779
Diagnosis for open pins 2796
Diagnosis of the A/D conversion circuit 2794
Diagnosis of the channel multiplexer 2795
Diagnostic conversion result register (ADCAnDGCR) 2827
Discharge function 2800
Emulation register (ADCAnEMU) 2835
Error interrupt INTADCAnERR 2780
Functional Description 2766
I/O signals 2755
Instances 2752
Instances index n 2752
Interrupt control register i (ADCAnIOCi) 2811
Interrupts 2780
Interrupts and DMA/DTS 2754
Last conversion interrupt INTADCAnLLT 2781
Latest conversion result register (ADCAnLCR) 2821
Mode control register 0 (ADCAnCTL0) 2804
Mode control register 1 (ADCAnCTL1) 2806
One-shot conversion mode 2771
Overwrite check function 2784
Overwrite error flag register (ADCAnSTR1) 2815
Pull down resistance control register 0 (ADCAnPDCTL0) 2834
Pull-down resistors 2796
Registers 2802
Reset 2754
Resolution 2752, 2779
Result check error flag register (ADCAnSTR0) 2830
Result check functions 2784
Result check register (ADCAnCTL2) 2828
Result lower limit register (ADCAnLL) 2829
Result read flag function 2785
Result registers 2781
Result upper limit register (ADCAnUL) 2829
Result upper/lower limit compare function 2785
Sampling time 2779
Self-diagnosis function control register 0 (ADCAnDGCTL0) 2832
Self-diagnosis function control register 1 (ADCAnDGCTL1) 2833
Self-diagnosis functions 2793
Simultaneous sampling 2755
Stabilization control 2801
Stabilization counter (ADCAnCNT) 2812
Stand-by mode 2778
Start triggers 2774
Status flag clear register 2 (ADCAnSTC2) 2818
Status flag register 2 (ADCAnSTR2) 2817
Stop triggers 2776
Successive approximation register (SAR) 2767
SW trigger register 3 (ADCAnTRG3) 2819
SW trigger register 4+i (ADCAnTRG4+i) 2820

SW trigger register 7 (ADCAnTRG7) 2820
SW trigger register i (ADCAnTRGi) 2819
TAUA0 interrupt selection register for ADCA0TTRGi (PIC0ADTEN40i) 2760
TAUB1 interrupt selection register for ADCA0TTRGi (PIC0ADTEN41i) 2761
Trigger select control register i (ADCAnTSELi) 2814
A/D Trigger Encoder Capture 1727
ADCA
see A/D Converter A (ADCA)
ADCAnCGi 2809
ADCAnCmCR 2823
ADCAnCNT 2812
ADCAnCTL0 2804
ADCAnCTL1 2806
ADCAnCTL2 2828
ADCAnDBiCR 2825
ADCAnDBiCRL 2826
ADCAnDGCR 2827
ADCAnDGCTL0 2832
ADCAnDGCTL1 2833
ADCAnEMU 2835
ADCAnIOCi 2811
ADCAnLCR 2821
ADCAnLL 2829
ADCAnPDCTL0 2834
ADCAnSHCTL 2813
ADCAnSTC0 2831
ADCAnSTC1 2816
ADCAnSTC2 2818
ADCAnSTR0 2830
ADCAnSTR1 2815
ADCAnSTR2 2817
ADCAnTRG3 2819
ADCAnTRG4+i 2820
ADCAnTRG7 2820
ADCAnTRGi 2819
ADCAnTSELi 2814
ADCAnUL 2829
ADDIAGOUT 2794
Address maps 180
Address space 178
CPU 178
Physical 178
Aggregated Channel Status (FLXnACS) 2445
Alphabetic pin function list 111
Analog filter type A 125
Analog filter type B 126
Analog filter type C 126
Analog filters 132
Control registers 132
Filter characteristic 132
Stand-by mode 133
Type A 133
Type B 134
Asynchronous Serial Interface E
see UARTEn
Asynchronous Serial Interface E (URTE)
Baud rate generator 1835
BF reception 1824
BF transmission 1823
BF transmission/reception format 1821
Configuration 1800
Continuous transmission procedure 1828
Control register 0 (URTEnCTL0) 1802
Control register 1 (URTEnCTL1) 1804
Control register 2 (URTEnCTL2) 1807
Data formats 1819
Emulation register(URTEnEMU) 1816
Functional Overview 1799
Interrupt requests 1817
Operation 1819
Parity types and operations 1833
Receive data noise filter 1834
Receive data register(URTEnRX) 1814
Reception 1830
Reception errors 1832
Reception interrupt (INTUAEnTIR) 1818
Registers 1801
Status clear register(URTEnSTC) 1813
Status interrupt (INTUAEnTIS) 1819
Status register 0(URTEnSTR0) 1810
Status register 1(URTEnSTR1) 1811
Transmission 1827
Transmission interrupt (INTUAEnTIT) 1817
Transmit data register(URTEnTX) 1815
Trigger register(URTEnTRG) 1808
ATEAPMI 205
ATEBSR 204
ATEEEA 201
ATELEA 202
ATESHL 204
ATESR 203
AWC0, AWC1 229

B

Back-up RAM (BURAM) 184
Access 184

Access error clear register (BURAEC) 186
Access error register (BURAE) 186
Address 184
Clock supply 184
Control register (BURC) 185
Write permission 184
Write protection 185
BCT0, BCT1 224
Boundary Scan 2852
Applicable pins 2853
Device ID register (DID) 2854
Features 2853
Instruction codes 2853
JTAG interface 2852
BSC 222
BURAE 186
BURAEC 186
BURAM
see Back-up RAM (BURAM)
BURC 185

C

CAN (Controller area network) 1888, 2019
CAN Controller
Time stamp 1892
CAN Controller (FCN) 1888
<FCNn_base> 1889
Base addresses 1889
Baudrate settings 1981
Bit set/clear function 1908
CAN RAM 1898
Channel bit rate prescaler register (FCNnCMBRPRS) 1931
Channel bit rate register (FCNnCMBTCTL 1932
Channel control register (FCNnCMCLCTL) 1921
Channel data byte register (FCNnMmDATxB/H/W) 1941
Channel data length register m (FCNnMmDTLGB) 1942
Channel error counter register (FCNnCMERCNT) 1928
Channel information register (FCNnCMINSTR) 1927
Channel interrupt enable register (FCNnCMIECTL) 1929
Channel interrupt status register (FCNnCMISCTL) 1930
Channel last error information register (FCNnCMLCSTR) 1926
Channel last- in-pointer register (FCNnCMLISTR) 1934
Channel last out-pointer register (FCNnCMLOSTR) 1936
Channel mask control register (FCNnCMMKCTLaH) 1919
Channel receive history list register (FCNnCMRGRX) 1934
Channel time stamp register (FCNnCMTSCTL) 1939
Channel transmit history list register (FCNnCMTGTX) 1937
Clock supply 1889
Configuration 1898
Control registers 1910
Data new bit monitor registers (FCNnDNBMRXk) 1918
Diagnosis functions 1976
FCN0 signal connection selection register (ITGSLFC0) 1894
FCN0/FCN1 connection 1893
Global automatic block transmission control register (FCNnGMABCTL) 1915
Global automatic block transmission delay register (FCNnGMADCTL) 1917
Global clock selection register (FCNnGMCSPRE) 1914
Global control register (FCNnGMCLCTL) 1910
I/O signals 1891
Initialization 1950
Instances 1888
Instances index n 1888
Internal registers 1899
Interrupt function 1975
Interrupts and DMA 1890
Message buffers index m 1888
Message configuration register m (FCNnMmSTRB) 1943
Message control register m (FCNnMmCTL) 1947
Message ID register m (FCNnMmMIDxH/W) 1945
Message reception 1953
Message transmission 1962
Operation 1987
Overview of functions 1896
Power saving modes 1970
Register bit configuration 1903
Reset 1891
Special operational modes 1976
Time stamp function 1980
Transition from initialization mode to opera-

tion mode 1952
CC Error Vector (FLXnCCEV) 2437
CC Status Vector (FLXnCCSV) 2434
CDEDADR 268
CECCER 266
CECCERC 266
CG (ADCA) 2769
CKSC_mn 574
CLMA
see Clock Monitor A (CLMA)
CLMAn protection command register (CLMAnPCMD) 215
CLMAn protection status register (CLMAnPS) 215
CLMAnCMPH 553
CLMAnCMPL 552
CLMAnCTL0 550
CLMAnCTL1 551
CLMAnEMU0 554
CLMAnPCMD 215, 554
CLMAnPS 215
$\overline{\text{CLMAnRES}}$) 650
Clock Controller 482
Clock selector control registers (CKSC_mn) 574
Clock selector status registers (CSCSTAT_mn) 576
Clock switching 507
High Speed Internal Oscillator (High Speed IntOsc) 499
High Speed IntOsc enable register (ROSCE) 565
High Speed IntOsc status register (ROSCS) 567
Illegal ID 507
Low Speed Internal Oscillator (Low Speed IntOsc) 498
Main Oscillator (MainOsc) 493
MainOsc control register (MOSCC) 559
MainOsc enable register (MOSCE) 557
MainOsc stabilization time register (MOSCST) 560
MainOsc status register (MOSCS) 558
Phase-Locked Loop generators (PLL) 502
PLLk control registers (PLLCk) 570
PLLk enable register (PLLEk) 568
PLLk stabilization time register (PLLSTk) 573
PLLk status registers (PLLSk) 569
Register write protection 555
Registers 555
Sub Oscillator (SubOsc) 496
SubOsc enable register (SOSCE) 562
SubOsc stabilization time register (SOSCST) 564
SubOsc status register (SOSCS) 563
Clock Domain Figures 527
Clock domain index n 482
Clock Monitor A
Clock supply 540
Clock Monitor A (CLMA) 539
<CLMAn_base> 539
Base addresses 539
CLMAnPCMD 215
CLMAnPS 215
Comparison register H (CLMAnCMPH) 553
Comparison register L (CLMAnCMPL) 552
Control register 0 (CLMAnCTL0) 550
Control register 1(CLMAnCTL1) 551
Emulation register 0 (CLMAnEMU0) 554
Enable/disable 547
Instances 539
Instances index n 539
Interrupts and reset outputs 541
Protection command register (CLMAnPCMD) 554
Register write protection 549
Registers 549
Reset 541
Start-up options 542
Suspend/resume 548
Clock Monitors reset ($\overline{\text{CLMAnRES}}$) 650
Clock switching 507
Clocked Serial Interface (CSIG)
EDL 2117
Clocked Serial Interface G (CSIG) 2104
<CSIGn_base> 2104
Base addresses 2104
Clock supply 2105
Communication in slave mode 2120
Configuration register 0 (CSIGnCFG0) 2138
Control register 0 (CSIGnCTL0) 2131
Control register 1 (CSIGnCTL1) 2132
Control register 2 (CSIGnCTL2) 2134
Data consistency check 2109, 2127
Data length select function 2117
Data transfer modes 2116
Emulation register (CSIGnEMU) 2141
Error detection 2127
Extended data length 2117

Functional description 2112
Functional overview 2110
Hand-shake function 2123
I/O signals 2107
Instances 2104
Instances index n 2104
Internal signals 2107
Interrupts 2121
Interrupts and DMA 2106
Loop-back mode 2126
Master/slave connections 2113
Master/slave mode 2112
Overrun error 2129
Parity check 2128
Reception register 0 (CSIGnRX0) 2141
Registers 2130
Reset 2106
Rx-only mode control register 0 (CSIGnBCTL0) 2137
Serial data direction select function 2119
Status clear register 0 (CSIGnSTCR0) 2136
Status register 0 (CSIGnSTR0) 2135
Transmission clock selection 2115
Transmission register 0 for half word access (CSIGnTX0H) 2140
Transmission register 0 for word access (CSIGnTX0W) 2140
Clocked Serial Interface H (CSIH) 2144
Memory control register 1 (CSIHnMCTL1) 2200
Memory control register 2 (CSIHnMCTL2) 2201
<CSIHn_base> 2144
Base addresses 2144
Broadcast (RCB) 2155
Buffer memory configurations 2160
Chip select index m 2144
Chip select priority 2155
Chip select timing 2158
Chip selection 2155
Clock supply 2145
Communication in slave mode 2167
Configuration register x (CSIHnCFGx) 2205
Control register 0 (CSIHnCTL0) 2190
Control register 1 (CSIHnCTL1) 2192
Control register 2 (CSIHnCTL2) 2194
Control Registers 2189
Data consistency check 2148, 2179
Data length select function 2163
Data transfer modes 2162
Emulation register (CSIHnEMU) 2212
Error detection 2179
Extended data length (EDL) 2164
Functional description 2151
Functional overview 2149
Hand-shake function 2176
I/O signals 2147
Instances 2144
Instances index n 2144
Internal signals 2146
Interrupt requests 2168
Interrupts and DMA/DTS 2145
Job concept 2157
Loop-back mode 2188
Master/slave connections 2153
Master/slave mode 2152
Memory control register 0 (CSIHnMCTL0) 2199
Memory read/write pointer register 0 (CSIHnMRWP0) 2203
Operating Procedures 2213
Overflow error 2183
Overrun error 2185
Parity error 2181
Procedures in direct access mode 2213
Procedures in dual buffer mode 2222
Procedures in FIFO mode 2228
Procedures in transmit-only buffer mode 2218
Receive data register 0 for half word access (CSIHnRX0H) 2212
Receive data register 0 for word access (CSIHnRX0W) 2211
Registers details 2190
Reset 2146
Serial data direction select function 2166
Status clear register 0 (CSIHnSTCR0) 2198
Status register 0 (CSIHnSTR0) 2195
Time-out error 2182
Transmission clock selection 2159
Transmit data register 0 for half word access (CSIHnTX0H) 2210
Transmit data register 0 for word access (CSIHnTX0W) 2209
CNTAm 1842, 1843
Configuration register (CNTAm) 1843
Control register (CNTAm) 1843
Registers 1842

Code flash error correction 265
Double-bit error detection address register (CDEDADR) 268
Error flag clear register (CECCERC) 266
Error flag register (CECCER) 266
Single-bit error correction address register (CSECADR) 267
Code Protection and Security 631
Control settings (FLXnCS) 2379
Controller Information (FLXnCI) 2377
CPU
Address space 178
CPU Access Bus Structures and Latencies 149
CPU Subsystem
HBUS bridge 187
Reset 156
CPU Subsystem modules access 149
CPU System Functions 143
CSCSTAT_mn 576
CSECADR 267
CSIG
see Clocked Serial Interface G (CSIG)
CSIGnBCTL0 2137
CSIGnCFG0 2138
CSIGnCTL0 2131
CSIGnCTL1 2132
CSIGnCTL2 2134
CSIGnEMU 2141
CSIGnRX0 2141
CSIGnSTCR0 2136
CSIGnSTR0 2135
CSIGnTX0H 2140
CSIGnTX0W 2140
CSIH
see Clocked Serial Interface H (CSIH)
CSIHnCFGx 2205
CSIHnCTL0 2190
CSIHnCTL1 2192
CSIHnCTL2 2194
CSIHnEMU 2212
CSIHnMCTL0 2199
CSIHnMCTL1 2200
CSIHnMCTL2 2201
CSIHnMRWP0 2203
CSIHnRX0H 2212
CSIHnRX0W 2211
CSIHnSTCR0 2198
CSIHnSTR0 2195
CSIHnTX0H 2210
CSIHnTX0W 2209

D

Data consistency check (CSIG) 2109
Data consistency check (CSIH) 2148
Data CRC Function A (DCRA) 474
Base addresses 474
<DCRAn_base> 474
Clock supply 475
CRC control register (DCRAnCTL) 481
CRC data register (DCRAnCOUT) 480
CRC input register (DCRAnCIN) 479
Functional Description 477
Functional Overview 476
Instances 474
Instances index n 474
Registers 478
Reset 475
Data flash wait cycle control register (DCLKWAIT) 165
Data flash wait cycles 164
Data space 178
DBRES 650
DCLKWAIT 165
DCN
Data new bit monitor registers (DCNnDNBMRXk) 2080
DCN transfer ID mask registers (DCNnTIDMTX0H/1H) 2080
DCN transfer ID reference registers (DCNnTIDRTXxH/yW) 2077
Global automatic block transmission delay register (DCNnGMADCTL) 2075, 2076
Global clock selection register (DCNnGMCSPRE) 2075
Global control register (DCNnGMCLCTL) 2075
Message configuration register m (DCNnMmSTRB) 2098
Message control register m (DCNnMmCTL) 2100
Message ID register m (DCNnMmMIDxH/W) 2098
Module bit rate prescaler register (DCNnCMBRPRS) 2081
Module bit rate register (DCNnCMBTCTL 2081
Module control register (DCNnCMCLCTL) 2081
Module error counter register (DCNnCMERCNT) 2081
Module information register (DCNnCMINSTR) 2081

Module interrupt enable register (DCNnCMIECTL) 2081
Module interrupt status register (DCNnCMISCTL) 2081
Module last error information register (DCNnCMLCSTR) 2081
Module last- in-pointer register (DCNnCMLISTR) 2082
Module last out-pointer register (DCNnCMLOSTR) 2082
Module mask control register (DCNnCMMKCTLaH) 2081
Module receive history list register (DCNnCMRGRX) 2082
Module RXONLY_CH bit rate prescaler register (DCNnCRBRPRS) 2091
Module RXONLY_CH bit rate register (DCNnCRBTCTL 2092
Module RXONLY_CH bus selector register (DCNnCRBSSTR) 2097
Module RXONLY_CH control register (DCNnCRCLCTL) 2083
Module RXONLY_CH error counter register (DCNnCRERCNT) 2088
Module RXONLY_CH interrupt enable register (DCNnCRIECTL) 2089
Module RXONLY_CH interrupt status register (DCNnCRISCTL) 2090
Module RXONLY_CH last error information register (DCNnCRCSTR) 2087
Module RXONLY_CH last-in-pointer register (DCNnCRLISTR) 2094
Module RXONLY_CH time stamp register (DCNnCRTSCTL) 2095
Module time stamp register (DCNnCMTSCTL) 2082
Module transmit history list register (DCNnCMTGTX) 2082
Register bit configuration 2067
see Diagnostic CAN Controller (DCN)
DCNnCMBRPRS 2081
DCNnCMBTCTL 2081
DCNnCMCLCTL 2081
DCNnCMERCNT 2081
DCNnCMIECTL 2081
DCNnCMINSTR 2081
DCNnCMISCTL 2081
DCNnCMLCSTR 2081
DCNnCMLISTR 2082
DCNnCMLOSTR 2082
DCNnCMMKCTLaH 2081
DCNnCMRGRX 2082
DCNnCMTGTX 2082
DCNnCMTSCTL 2082
DCNnCRBRPRS 2091
DCNnCRBSSTR 2097
DCNnCRBTCTL 2092
DCNnCRCLCTL 2083
DCNnCRCSTR 2087
DCNnCRERCNT 2088
DCNnCRIECTL 2089
DCNnCRISCTL 2090
DCNnCRLISTR 2094
DCNnCRTSCTL 2095
DCNnDNBMRXk 2080
DCNnGMADCTL 2075, 2076
DCNnGMCLCTL 2075
DCNnGMCSPRE 2075
DCNnMmCTL 2100
DCNnMmMIDxH/W 2098
DCNnMmSTRB 2098
DCNnTIDMTX0H/1H 2080
DCNnTIDRTXxH/yW 2077
DCRA
Base addresses 2743
see Data CRC Function A (DCRA)
DCRAnCIN 479
DCRAnCOUT 480
DCRAnCTL 481
DDAnH 359
DDAnL 357
DDCn 360
Debugger reset (DBRES) 650
DEC 223
DEEPSTOP mode 605
Delay Pulse Output with Dead Time 1669
DHC 227
Diagnostic CAN Controller (DCN) 2019
<DCNn_base> 2019
Base addresses 2019
Clock supply 2020
I/O signals 2021
Instances 2019
Instances index n 2019
Interrupts and DMA/DTS 2020
Message buffers index m 2019
Reset 2020
DID 2854
Digital filter type D 127
Digital filters 135
Control registers 137
Filter characteristic 135
Filter groups 136

Stand-by mode 137
Type D 138
DLYAEN 1769
DMA 324
DMA address map 180
DMA channels 324
DMA trigger factors 325
DMA/DTS Controller
Channel index n 324
DMA channels 324
DMA trigger factors 325
DMA/DTS Controller (DMAC) 324
Clock supply 324
DMA address map 324
DMA destination address register H (DDAnH) 359
DMA destination address register L (DDAnL) 357
DMA destination chip select register (DDCn) 360
DMA next destination address register H (DNDAnH) 362
DMA next destination address register L (DNDAnL) 361
DMA next destination chip select register (DNDCn) 363
DMA next source address register H (DNSAnH) 355
DMA next source address register L (DNSAnL) 354
DMA next source chip select register (DNSCn) 356
DMA next transfer count register (DNTCn) 365
DMA request check register (DRQSTR) 384
DMA request clear register (DRQCLR) 383
DMA source address register H (DSAnH) 352
DMA source address register L (DSAnL) 350
DMA source chip select register (DSCn) 353
DMA transfer control register (DTCTn) 367
DMA transfer count compare register (DTCCn) 366
DMA transfer count register (DTCn) 364
DMA transfer request control register (DTRCx) 348
DMA transfer request select register (DTRSn) 349
DMA transfer status register (DTSn) 369
DMA/DTS data transfer unit (DMAT) 330
DTFRn register (DTFRn) 382
DTS active channel register (DTS0ACR) 395
DTS base table register (DTS0BTR) 393
DTS base vector register (DTS0BVR) 394
DTS control information register (DTS0CIR) 405
DTS destination address count size register (DTS0DCS) 404
DTS destination address register (DTS0DAR) 400
DTS extension address count size/repeat address registers (DTS0ECSRA) 407
DTS hold enable register (DTSHENm) 443
DTS initialization control register (DTS0ICR) 391
DTS source address count size register (DTS0SCS) 403
DTS source address register (DTS0SAR) 399
DTS TI hold channel number register (DTS0HC) 397
DTS TI hold status register (DTS0TST) 396
DTS transfer counter or else address register (DTS0TCEA) 401
DTS transfer enable register (DTSENn) 442
DTS transfer request control register (DTS0TRC) 390
DTS transfer status register (DTS0TSR) 389
Reset 324
DMAC (DMA/DTS Controller) 324
DMAINTSL0 296
DMAINTSL1 296
DMAT 330
DNDAnH 362
DNDAnL 361
DNDCn 363
DNFAnCTL 141
DNFAnEN 142
DNSAnH 355
DNSAnL 354
DNSCn 356
DNTCn 365
DPSTPREQ_0 583
DPSTPREQ_1 583
DPSTPWU_0 584
DPSTPWU_1 584
DRQCLR 383

DRQSTR 384
DSAnH 352
DSAnL 350
DSC 228
DSCn 353
DTCCn 366
DTCn 364
DTCTn 367
DTFRn 382
DTRCx 348
DTRSn 349
DTS 324
DTS path through interrupts 292
DTS0ACR 395
DTS0BTR 393
DTS0BVR 394
DTS0CIR 405
DTS0DAR 400
DTS0DCS 404
DTS0ECSRA 407
DTS0HC 397
DTS0ICR 391
DTS0SAR 399
DTS0SCS 403
DTS0TCEA 401
DTS0TRC 390
DTS0TSR 389
DTS0TST 396
DTSENn 442
DTSHENm 443
DTSn 369
DWC0, DWC1 225

E

Edge detection configuration 297
EDL 2164
EH_BASE 259
EH_CFG 259
EH_RESET 259
Emulation break control 2850
ENCA
    see Encoder Timer (ENCA)
ENCA input selection register (TISLENCBYPS) 1733
ENCAnCCR0 1709
ENCAnCCR1 1710
ENCAnCNT 1710
ENCAnCTL 1703
ENCAnFGC 1709
ENCAnFLG 1708
ENCAnIOC0 1705
ENCAnIOC1 1706
ENCAnTE 1711
ENCAnTS 1711
ENCAnTT 1712
Encoder Timer 1694
Encoder Timer (ENCA) 1696
    <ENCAn_base> 1696
    Base address 1696
    Block diagram 1700
    Capture compare register 0 (ENCAnCCR0) 1709
    Capture compare register 1 (ENCAnCCR1) 1710
    Clock supply 1697
    Control register (ENCAnCTL) 1703
    Counter register (ENCAnCNT) 1710
    Functional description 1713
    Functional overview 1699
    I/O control register 0 (ENCAnIOC0) 1705
    I/O control register 1 (ENCAnIOC1) 1706
    I/O signals 1698
    Instances 1696
    Instances index n 1696
    Internal signals 1698
    Interrupts 1697
    Registers 1702
    Reset 1698
    Status flag clear register (ENCAnFGC) 1709
    Status flag register (ENCAnFLG) 1708
    Timer enable status register (ENCAnTE) 1711
    Timer start trigger register (ENCAnTS) 1711
    Timer stop trigger register (ENCAnTT) 1712
Error detection (WDTA) 685
Error Interrupt Enable Reset (FLXnEIER) 2401
Error Interrupt Enable Set (FLXnEIES) 2401
Error Interrupt Line Select (FLXnEILS) 2397
Error Interrupt Register (FLXnEIR) 2389
ETACFG 191
ETACMD 192
ETAEREA 195
ETAERR 194
ETARADRSn 198
ETARCFGn 197
ETARMASKn 199
ETAWLEA 196
ETAWRL 193

ETHA
see Ethernet Controller (ETHA)
Ethernet Controller
see Ethernet Controller (ETHA)
Ethernet Controller (ETHA) 2555
<ETHAn_base> 2555
Base addresses 2555
Clock supply 2555
Configuration 2559
Functions 2558
I/O signals 2557
Instances 2555
Instances index n 2555
Interrupt requests and sources 2561
Interrupts and DMA/DTS 2556
Registers 2562
Reset 2556
Even Sync ID (FLXnESIDn) 2447
EWC 232
Exception handler address switching function 323
Exceptions 260
External Memory Controller (MEMC) 218
Address wait configuration registers 0 and 1 (AWC0, AWC1) 229
Bus cycle type configuration registers 0 and 1 (BCT0, BCT1) 224
Bus size configuration register (BSC) 222
Clock supply 218
Data endian configuration register (DEC) 223
Data hold wait configuration register (DHC) 227
Data setup wait configuration register (DSC) 228
Data wait configuration registers 0 and 1 (DWC0, DWC1) 225
External wait error configuration register (EWC) 232
Idle cycle configuration registers 0 and 1 (ICC0, ICC1) 230
Registers 221
External Memory Controller (MEMC)Reset 218

F

FCLAnCTLm 140
FCN
see CAN Controller (FCN)
FCNnCMBRPRS 1931
FCNnCMBTCTL 1932
FCNnCMCLCTL 1921
FCNnCMERCNT 1928
FCNnCMIECTL 1929
FCNnCMINSTR 1927
FCNnCMISCTL 1930
FCNnCMLCSTR 1926
FCNnCMLISTR 1934
FCNnCMLOSTR 1936
FCNnCMMKCTLaH 1919
FCNnCMRGRX 1934
FCNnCMTGTX 1937
FCNnCMTSCTL 1939
FCNnDNBMRXk 1918
FCNnGMABCTL 1915
FCNnGMADCTL 1917
FCNnGMCLCTL 1910
FCNnGMCSPRE 1914
FCNnMmCTL 1947
FCNnMmDATxB/H/W 1941
FCNnMmDTLGB 1942
FCNnMmMIDxH/W 1945
FCNnMmSTRB 1943
FENMIF 290
FENMIFC 295
FIC 308
FIFO Rejection Filter (FLXnFRF) 2454
Flash Mask Options
Flash mask option register 0 (OPBT0) 472
Flash Memory 446
FLMD protection command register (FLMDPCMD) 217
FLMD protection error status register (FLMDPS) 217
Flash memory
Configuration area 448
Self-Programming 448
Self-programming 463
Serial-Programming 448
Flash programmer
Communication mode 454
Pin connection 455
Flash Programmer and Self-Programming Protection 632
Flash programming
Mode 170
FlexRay
Appendix 2545
FLXnVI 2363
Functional Description 2492
Programmer’s model 2372
FlexRay (FLX)
Message buffer size 2364

FlexRay™ (FLX) 2363
<FLXn_base> 2363
Base addresses 2363
Clock supply 2364
I/O signals 2365
Instances 2363
Instances index n 2363
Interrupts and DMA/DTS 2365
Reset 2365
Vendor information 2363
FLMD control register (FLMDCNT) 464
FLMDCNT 464
FLMDPCMD 217
FLMDPS 217
FLX
see FlexRay™ (FLX)
FLXnACS 2445
FLXnCCEV 2437
FLXnCCSV 2434
FLXnCI 2377
FLXnCS 2379
FLXnEIER 2401
FLXnEIES 2401
FLXnEILS 2397
FLXnEIR 2389
FLXnESIDn 2447
FLXnFRF 2454
FLXnGTUC1 2423
FLXnGTUC10 2432
FLXnGTUC11 2433
FLXnGTUC2 2424
FLXnGTUC3 2425
FLXnGTUC4 2426
FLXnGTUC5 2427
FLXnGTUC6 2428
FLXnGTUC7 2429
FLXnGTUC8 2430
FLXnGTUC9 2431
FLXnIBCM 2478
FLXnIBCR 2479
FLXnILE 2405
FLXnLCK 2388
FLXnMBS 2485
FLXnMBSC1 2468
FLXnMBSC2 2468
FLXnMBSC3 2468
FLXnMBSC4 2468
FLXnMHDC 2422
FLXnMHDS 2457, 2470
FLXnMRC 2451
FLXnMTCCV 2439
FLXnNDAT1 2466
FLXnNDAT2 2466
FLXnNDAT3 2466
FLXnNDAT4 2466
FLXnNEMC 2418
FLXnNMVn 2450
FLXnOBCM 2489
FLXnOBCR 2490
FLXnOCV 2440
FLXnOSIDn 2449
FLXnPRTC1 2419
FLXnPRTC2 2421
FLXnRCV 2439
FLXnRDDSn 2481
FLXnRDHS1 2482
FLXnRDHS2 2483
FLXnRDHS3 2484
FLXnSCV 2438
FLXnSFS 2441
FLXnSIER 2403
FLXnSIES 2403
FLXnSILS 2399
FLXnSIR 2393
FLXnSTPW 2408, 2409
FLXnSUCC1 2410
FLXnSUCC2 2416
FLXnSUCC3 2417
FLXnSWNIT 2443
FLXnT0C 2406
FLXnT1C 2407
FLXnTEST1 2382, 2386
FLXnTEST2 2388
FLXnTXRQm 2464
FLXnVI 2378
FLXnWRDSn 2473
FLXnWRHS1 2474
FLXnWRHS2 2476
FLXnWRHS3 2477
FNC 307
FOUT
see Frequency Output Function (FOUT)
FOUTDIV 537, 538
Frequency Output Function (FOUT) 536
Clock divider 537
FOUT clock divider (FOUTDIV) 537
FOUT divisor register (FOUTDIV) 538

G

GTU Configuration Register 1

(FLXnGTUC1) 2423
GTU Configuration Register 10 (FLXnGTUC10) 2432
GTU Configuration Register 11 (FLXnGTUC11) 2433
GTU Configuration Register 2 (FLXnGTUC2) 2424
GTU Configuration Register 3 (FLXnGTUC3) 2425
GTU Configuration Register 4 (FLXnGTUC4) 2426
GTU Configuration Register 5 (FLXnGTUC5) 2427
GTU Configuration Register 6 (FLXnGTUC6) 2428
GTU Configuration Register 7 (FLXnGTUC7) 2429
GTU Configuration Register 8 (FLXnGTUC8) 2430
GTU Configuration Register 9 (FLXnGTUC9) 2431

H

HBUS bridge 187
- Area n address register (ETARADRSn) 198
- Area n mask register (ETARMASKn) 199
- Area n setting register (ETARCFGn) 197
- Calculation of area n address range 200
- Command register (ATEAPMI) 205
- Command register (ATEBSR) 204
- Command register (ETACMD) 192
- Configuration register (ETACFG) 191
- HBUS error response error address register (ETAEREA) 195
- Master I/F error flag register (ETAERR) 194
- Master I/F wait insertion limit error address register (ETAWLEA) 196
- Registers 190
- Slave I/F CPU lock hold time limit register (ATESHL) 204
- Slave I/F CPU Subsystem error address register (ATEEEA) 201
- Slave I/F error flag register (ATESR) 203
- Slave I/F lock hold time error address register (ATELEA) 202
- Wait insertion limit register (ETAWRL) 193

HBUS Cross Connection 166
- Arbitration 169
- Clock supply 166
- Connection matrix 168
- Data ports 168

HBUS-RAM 183
High Speed IntOsc 499
High-accuracy Triangle PWM Output with Dead Time 1630
Hi-Z output control register n (PIC0HIZCEN0) 1582
Hi-Z output control register n (PIC0HIZCENn) 1629, 1668, 1693

I

I^2C Interface (IICB) 2232
- <IICBn_base> 2232
- Base addresses 2232
- Clock supply 2233
- Control register 0 (IICBnCTL0) 2248
- Control register 1 (IICBnCTL1) 2251
- Data register (IICBnDAT) 2246
- Emulation register (IICBnEMU) 2269
- High-level width setting register (IICBnWH) 2256
- I/O signals 2234
- Instances 2232
- Instances index n 2232
- Interrupts and DMA 2233
- Low level width setting register (IICBnWL) 2253
- Port Settings 2234
- Reset 2233
- Slave address register (IICBnSVA) 2247
- Status clear register (IICBnSTRC) 2268
- Status register 0 (IICBnSTR0) 2261
- Status register 1 (IICBnSTR1) 2267
- Trigger register (IICBnTRG) 2257

ICC0, ICC1 230
ICn 299
ICSR 306
IDMODI 637
IICB
- see I^2C Interface (IICB)

IICBnCTL0 2248
IICBnCTL1 2251
IICBnDAT 2246
IICBnEMU 2269
IICBnSTR0 2261
IICBnSTR1 2267
IICBnSTRC 2268
IICBnSVA 2247
IICBnTRG 2257
IICBnWH 2256
IICBnWL 2253
Illegal ID 507

IMRm 301
Input Buffer Command Mask (FLXnIBCM) 2478
Input Buffer Command Request (FLXnIBCR) 2479
INTADCAnERR 2780
INTADCAnLLT 2781
INTADCAnTi 2780
Interrupt functions 257
Acknowledgement 258
Acknowledgement condition 258
Control registers 299
DMA interrupt selection register 0 (DMAINTSL0) 296
DMA interrupt selection register 1 (DMAINTSL1) 296
DTS path through interrupts 292
Edge detection configuration 297
EI level interrupt control registers (ICn) 299
EI level interrupt mask registers (IMRm) 301
EI level maskable interrupt sharing 292, 295
EI level maskable interrupts 270
Exception handler address switching function 323
Exceptions 260
Exceptions and Interrupts 257
FE level INT status register (FIC) 308
FE level maskable interrupts 270
FE level NMI status register (FNC) 307
FE level non-maskable interrupt sharing 290
FE level non-maskable interrupts 269
FENMI factor clear register (FENMIFC) 295
FENMI factor register (FENMIF) 290
In-service priority clear function 323
In-service priority clear register (ISPC) 304
Interrupt acknowledgment and restoring 309
Interrupt controller status register (ICSR) 306
Interrupt operation 316
Interrupt path through mode register (PTHMD0) 294
Interrupt priority level judgment 316
Interrupt request sources 269
Interrupt types 269
Mask function of EI level maskable interrupt 316
Memory error exceptions MEP 260
Or-ed interrupts 292
Pending interrupt report function 322
Priority mask function 322
Priority mask register (PMR) 303
Priority order 258
Request 258
Restore 259
Resume 259
Select channel hold register (SCR) 305
System error control register (SEG_CONT) 263
System error exceptions SYSERR 261
System error flag register (SEG_FLAG) 264
TAPA interrupt seletor register (TAPAINTSL0) 295
Interrupt Line Enable (FLXnILE) 2405
Interrupt operation 316
Interrupt request sources 269
INTUAEnTIR (URTE) 1818
INTUAEnTIS (URTE) 1819
INTUAEnTIT (URTE) 1817
ISPC 304
ITGSLFC0 1894

J

JP0 70
JPBDC0 68
JPD0 74
JPDSC0 75
JPFC0 66
JPFCE0 67
JPIBC0 65
JPIS0 77
JPISE0 78
JPM0 63
JPMC0 60
JPMCSR0 61
JPMSR0 64
JPNOT0 71
JPODC0 76
JPPR0 69
JPSR0 72
JPU0 73
JTAG ports 52

K

Key Interrupt Function 2749
Key Return Function
Functional Description 2750
Key Return Function (KR) 2747
<KRn_base> 2747

Base addresses 2747
Clock supply 2748
I/O signals 2748
Instances 2747
Instances index n 2747
Interrupts 2748
Reset 2748
Key Return Function (KR08) 2749
Key return mode register
KR08nKRM 2751
Key return mode register (KR08nKRM) 2751
KR
see Key Return Function (KR)
KR08nKRM 2751

L

LIN Master Controller (LMA) 1836
<CNTAm_base> 1837
<LMAn_base> 1836
Base addresses 1836
Clock supply 1837
CNTAm base addresses 1837
CNTAm instances 1836
CNTAm instances index m 1836
Compare register H (LMAnCMPH) 1882
Compare register L (LMAnCMPL) 1882
Control register H (LMAnCTLH) 1875
Control register L (LMAnCTLL) 1873
Instances 1836
Instances index n 1836
Internal signals 1841
Interrupts and DMA/DTS 1839
Registers 1871
Reset 1840
Rx control register H (LMAnRCTLH) 1887
Rx control register L (LMAnRCTLL) 1886
Status clear register H (LMAnSTCH) 1881
Status clear register L (LMAnSTCL) 1880
Status register H (LMAnSTRH) 1877
Status register L (LMAnSTRL) 1876
Tx control register H (LMAnTCTLH) 1885
Tx control register L (LMAnTCTLL) 1883
LIN Master Scheduler Counter (CNTA) 1842
LMA
see LIN Master Controller (LMA)
LMAn
Control register L (LMAnCTLL) 1843
LMAnCMPH 1882
LMAnCMPL 1882
LMAnCTLH 1875
LMAnCTLL 1843, 1873
LMAnRCTLH 1887
LMAnRCTLL 1886
LMAnSTCH 1881
LMAnSTCL 1880
LMAnSTRH 1877
LMAnSTRL 1876
LMAnTCTLH 1885
LMAnTCTLL 1883
Lock Register (FLXnLCK) 2388
Low Speed IntOsc 498
Low-Voltage Indicator (LVI) 644
LVI 644
LVICNT 655

M

Macrotick and Cycle Counter Value (FLXnMTCCV) 2439
MainOsc 493
MEMC
see External Memory Controller (MEMC)
Memory
Areas 182
Back-up RAM areas 183
External memory areas areas 183
HBUS-RAM areas 183
Internal code flash areas 182
Internal data flash areas 182
Internal local RAM areas 182
Memory Protection Unit (MPU) 148
MEP (Memory error exceptions) 260
Message Buffer Status (FLXnMBS) 2485
Message Buffer Status Changed 1 (FLXnMBSC1) 2468
Message Buffer Status Changed 2 (FLXnMBSC2) 2468
Message Buffer Status Changed 3 (FLXnMBSC3) 2468
Message Buffer Status Changed 4 (FLXnMBSC4) 2468
Message Handler Status (FLXnMHDS) 2457, 2470
Message RAM Configuration (FLXnMRC) 2451
MHD Configuration Register (FLXnMHDC) 2422
MOSCC 559
MOSCE 557
MOSCS 558
MOSCST 560
Motor Control 1576
MPU

see Memory Protection Unit (MPU)

N

NEM Configuration Register (FLXnNEMC) 2418
Network Management Vector FLXnNMVn) 2450
New Data 1 (FLXnNDAT1) 2466
New Data 2 (FLXnNDAT2) 2466
New Data 3 (FLXnNDAT3) 2466
New Data 4 (FLXnNDAT4) 2466
Noise elimination 125
Normal operation mode 170

O

OCD
see On-Chip Debug Unit (OCD)
OCDIDL/M/H 636
Odd Sync ID (FLXnOSIDn) 2449
Offset Correction Value (FLXnOCV) 2440
On-Chip Debug Interface Protection 633
Control register (IDMODI) 637
Control registers 635
Enable flag 633
ID code 634
ID registers (OCDIDL/M/H) 636
Protection levels summary 634
On-Chip Debug Unit (OCD) 2845
Debug interface 2847
Debug monitoring function 2847
Debugger connection 2851
Dynamic memory modification (DMM) 2848
Emulation break 2845
Emulation break control 2850
Forced break function 2848
Forced reset function 2848
Hardware break function 2848
Hot attach function 2849
Mask function 2848
Modules run/stop selection during a break 2848
Real-time RAM monitoring (RRM) 2848
Security function 2849
Software break function 2848
Timer function 2848
OPBT0 472
Operation modes 170
Boundary Scan mode 171
Normal operation mode 170
Serial flash programming mode 170
Ordering information 48
Or-ed interrupts 292
OS Timer 1756, 1764
OS Timer (OSTM) 658, 1756
<OSTMn_base> 658
Base addresses 658
Clock supply 659
Compare register (OSTMnCMP) 669
Control register (OSTMnCTL) 672
Count enable status register (OSTMnTE) 671
Count start trigger register (OSTMnTS) 671
Count stop trigger register (OSTMnTT) 672
Counter register (OSTMnCNT) 670
Emulation register (OSTMnEMU) 673
Functional Description 660
Instances 658
Interrupts 659
Registers 668
Reset 659
OS Timer (OSTM)Instances index n 658
OSCWUFMSK 592, 630
OSTM
see OS Timer (OSTM)
OSTM (OS Timer) 1756
OSTMnCMP 669
OSTMnCNT 670
OSTMnCTL 672
OSTMnEMU 673
OSTMnTE 671
OSTMnTS 671
OSTMnTT 672
Output Buffer Command Mask (FLXnOBCM) 2489
Output Buffer Command Request (FLXnOBCR) 2490

P

PBDCn 68
PBUS/HBUS modules access 150
PDn 74
PDSCn 75
Peripheral Protection Unit (PPU) 145
<PPU_base> 145
Areas and registers 145
Base address 145
PFCEn 67
PFCn 66
Physical address space 178

PIBCn 65
PIC0ADTEN40i 2760
PIC0ADTEN41i 2761
PIC0HIZCEN0 1582
PIC0HIZCENn 1629, 1668, 1693
PIC0REG2n0 1657
PIC0REG2n1 1658
PIC0REG2n2 1624, 1661, 1688
PIC0REG2n3 1663
PIC0REG30 1732
PIC0REG31 1404, 1750
PIC0SSER0 1737
PIC0SSER2 1738
PIC0SST 1738
PIPCn 62
PISEn 78
PISn 77
PLL 502
PLLCk 570
PLLEk 568
PLLSk 569
PLLSTk 573
PMCA
see PWM Diagnostic (PMCA)
PMCAnCMPm 1792
PMCAnCTL0 1790
PMCAnCTL1 1791
PMCn 60
PMCSRn 61
PMn 63
PMR 303
PMSRn 64
Pn 70
PNOTn 71
POC 643
POCRES 643
PODCn 76
Port bi-direction control register (PBDCn, JPBDC0) 68
Port drive strength control register (PDSCn, JPDSC0) 75
Port filters 125
Analog filter type A 125
Analog filter type B 126
Analog filter type C 126
Analog filters 132
Clock supply 129
Control registers 139
Digital filter type D 127
Digital filters 135
Digital noise elimination control register (DNFAnCTL) 141
Digital noise elimination enable register (DNFAnEN) 142
Filter control register (FCLAnCTLm) 140
Filters assignment 125
Reset 131
Port filters assignment 125
Port function
PBDCn, JPBDC0 68
PDn, JPD0 74
PDSCn, JPDSC0 75
PFCEn, JPFCE0 67
PFCn, JPFC0 66
PIBCn, JPIBC0 65
PIPCn 62
PISEn, JPISE0 78
PISn, JPIS0 77
PMCn, JPMC0 60
PMCSRn, JPMCSR0 61
PMn, JPM0 63
PMSRn, JPMSR0 64
Pn, JP0 70
PNOTn, JPNOT0 71
PODCn, JPODC0 76
PPCMDn 216
PPRn, JPPR0 69
PPROTSn 216
PSRn, JPSR0 72
PUn, JPU0 73
Register write protection 58
Port function control expansion register (PFCEn, JPFCE0) 67
Port function control register (PFCn, JPFC0) 66
Port Functions
Port groups 50
Port functions 50
<JPORT0_base> 50
<PORTn_base> 50
Base addresses 50
Direct I/O control (PIPC) 83
Functions during and after reset 122
Functions in stand-by mode 122
Permanent inputs 83
Port groups index n 50
Port group configuration 80
Alphabetic pin function list 111
Port input buffer control register (PIBCn, JPIBC0) 65
Port input selection expansion register (PISEn, JPISE0) 78

Port input selection register (PISn, JPIS0) 77
Port IP control register (PIPCn) 62
Port mode control register (PMCn, JPMC0) 60
Port mode control set reset register (PMCSRn, JPMCSR0) 61
Port mode register (PMn, JPM0) 63
Port mode set reset register (PMSRn, JPMSR0) 64
Port NOT register (PNOTn, JPNOT0) 71
Port open drain control register (PODCn, JPODC0) 76
Port pin read register (PPRn, JPPR0) 69
Port protection command register(PPCMDn) 216
Port protection status register (PPROTSn) 216
Port register (Pn, JP0) 70
Port register protection 58
Port set reset register (PSRn, JPSR0) 72
Power domain index m 482
Power Supply Scheme 2855
Power-On-Clear (POC) 643
PPCMDn 216
PPRn 69
PPROTSn 216
PPU
see Peripheral Protection Unit (PPU)
PRDNAME 49
Product name register (PRDNAME) 49
Program space 178
PROTCMDm 213
PROTSm 214
PRT Configuration Register 1 FLXnPRTC1) 2419
PRT Configuration Register 2 (FLXnPRTC2) 2421
PSC0 621
PSC1 623
PSRn 72
PTHMD0 294
Pull-down option register (PDn, JPD0) 74
Pull-up option register (PUn, JPU0) 73
PUn 73
PURES 640
PWM Delay (DLYA) 1764
<DLYAn_base> 1764
Base addresses 1764
Channel group index p 1764
Channel index q 1764
Clock supply 1765
DLYA bypass control 1768
DLYA bypass enable register (DLYAEN) 1769
DLYA count clock CNTCLK 1765
Instances 1764
Instances index n 1764
PWM signals connection 1766
Reset 1766
PWM Diagnostic (PMCA) 1777
<PMCAn_base> 1777
Base addresses 1777
Clock supply 1778
I/O signals 1778
Instances 1777
Instances index n 1777
Internal signals 1779
PMCA comparison registers (PMCAnCMPm) 1792
PMCA control register 0 (PMCAnCTL0) 1790
PMCA control register 1 (PMCAnCTL1) 1791
PMCACUP input selection register (TRGSEL4) 1785
Reset 1778
Trigger input selection register 0 (TRGSEL0) 1781
Trigger input selection register 1 (TRGSEL1) 1782
Trigger input selection register 2 (TRGSEL2) 1783
Trigger input selection register 3 (TRGSEL3) 1784
PWS0 625
PWS1 626

R

Random Number Generator (RNG) 2743
Random Number Generator A (RNGA)
Functional Description 2745
Functional Overview 2745
Random number register (RNGAnRNG) 2746
Registers 2746
Rate Correction Value (FLXnRCV) 2439
RCB 2155
Read Data Section (FLXnRDDSn) 2481
Read Header Section 1 (FLXnRDHS1) 2482
Read Header Section 2 (FLXnRDHS2) 2483
Read Header Section 3 (FLXnRDHS3) 2484
Real-Time Clock
Registers 1771, 1789
Real-Time Clock (RTCA) 1519
<RTCAn_base> 1519

Alarm day of the week setting register (RTCAnALW) 1562, 1563
Alarm hour setting register (RTCAnALH) 1561
Alarm minute setting register (RTCAnALM) 1560
Base addresses 1519
Buffer register (RTCAnMIN) 1544
Calendar count buffer register (RTCAnCAL) 1559
Calendar count register (RTCAnCALC) 1558
Clock error correction register (RTCAnSRBU) 1539
Clock supply 1520
Compare register (RTCAnSCMP) 1540
Control register 0 (RTCAnCTL0) 1533
Control register 1 (RTCAnCTL1) 1534
Control register 2 (RTCAnCTL2) 1535
Day of the month count buffer register (RTCAnDAY) 1551
Day of the month count register (RTCAnDAYC) 1550
Day of the week count buffer register (RTCAnWEEK) 1549
Day of the week count register (RTCAnWEEKC) 1548
Functional description 1523
Hour count buffer register (RTCAnHOUR) 1547
Hour count register (RTCAnHOURC) 1545
I/O signals 1521
Instances 1519
Instances index n 1519
Interrupts 1520
Minute count register (RTCAnMINC) 1543
Month count buffer register (RTCAnMONTH) 1553
Month count register (RTCAnMONC) 1552
One second output (RTCA1HZ) 1528
Registers 1531
Reset 1520
Seconds count buffer register (RTCAnSEC) 1542
Seconds count register (RTCAnSECC) 1541
Sub-count register (RTCAnSUBC) 1537
Sub-count register read buffer (RTCAnSRBU) 1538
Time count buffer register (RTCAnTIME) 1557
Time count register (RTCAnTIMEC) 1556
Year count buffer register (RTCAnYEAR) 1555
Year count register (RTCAnYEARC) 1554
Related documents 48
$\overline{\text{RESET}}$ 648
Reset Controller 638
Clock Monitors reset ($\overline{\text{CLMAnRES}}$) 650
Debugger reset (DBRES) 650
External $\overline{\text{RESET}}$ 648
Low-Voltage Indicator (LVI) 644
LVI control register (LVICNT) 655
Power-On-Clear reset (POCRES) 643
Power-up reset (PURES) 640
Register write protection 651
Registers 651
Reset factor clear register (RESFC) 653
Reset factor register (RESF) 652
Software reset (SWRES) 650
Software reset register (SWRESA) 654
System reset (SYSRES) 640
Very-Low-Voltage flag clear register (VLVFC) 656
Very-Low-Voltage flag register (VLVF) 656
Very-Low-Voltage Indicator (VLVI) 646
Watchdog Timers reset (WDTAnRES) 650
RESF 652
RESFC 653
RNGA
<RNGAn_base> 2743
Clock supply 2744
Instances 2743
Instances index n 2743
Reset 2744
RNGAnRNG 2746
ROSCE 565
ROSCS 567
RTCA
see Real-Time Clock (RTCA)
RTCA1HZ 1528
RTCAnALH 1561
RTCAnALM 1560
RTCAnALW 1562, 1563
RTCAnCAL 1559
RTCAnCALC 1558
RTCAnCTL0 1533
RTCAnCTL1) 1534
RTCAnCTL2 1535
RTCAnDAY 1551
RTCAnDAYC 1550
RTCAnHOUR 1547

RTCAnHOURC 1545
RTCAnMIN 1544
RTCAnMINC 1543
RTCAnMONC 1552
RTCAnMONTH 1553
RTCAnSCMP 1540
RTCAnSEC 1542
RTCAnSECC 1541
RTCAnSRBU 1538
RTCAnSUBC 1537
RTCAnSUBU 1539
RTCAnTIME 1557
RTCAnTIMEC 1556
RTCAnWEEK 1549
RTCAnWEEKC 1548
RTCAnYEAR 1555
RTCAnYEARC 1554
RUN mode (Isolated-Area-1 DEEPSTOP) 610
RUN mode (Isolated-Area-1 STOP) 602

S

SAR (Successive approximation register) 2767
SCR 305
SEG_CONT 263
SEG_FLAG 264
Self-Programming library (SPL) 463
Simultaneous start control register 0 (PIC0SSER0) 1737
Simultaneous start control register 2 (PIC0SSER2) 1738
Simultaneous start trigger control register (PIC0SST) 1738
Slot Counter Value (FLXnSCV) 2438
Software reset (SWRES) 650
SOSCE 562
SOSCS 563
SOSCST 564
SPL (Self-Programming library) 463
SRP
    see System Register Protection (SRP)
Stand-by Controller (STBC) 578
    CAN FCNnRX wake-up 589
    Clock generators in stand-by 587
    DEEPSTOP mode flow 605
    DPSTPREQ_0 583
    DPSTPREQ_1 583
    DPSTPWU_0 584
    DPSTPWU_1 584
    External interrupts INTPm wake-up 589
    Functional modules interrupt wake-up 589
    HALT mode wake-up 589
    Mode control 584
    Mode transitions 597
    On-Chip Debug wake-up 589
    Oscillator wake-up mask registers (OSCWUFMSK) 630
    Oscillators wake-up 592
    Power save control register 0 (PSC0) 621
    Power save control register 1 (PSC1) 623
    Power save status 585
    Power status register 0 (PWS0) 625
    Power status register 1 (PWS1) 626
    Reset 578
    RUN mode (Isolated-Area-1 DEEPSTOP) flow 610
    RUN mode (Isolated-Area-1 STOP) flow 602
    S/W wake-up 589
    Signal connections 583
    Stand-by mode transitions 597
    Stand-by modes 586
    Stand-by modes flows 598
    STOP mode 599
    STPREQ_0 583
    STPREQ_1 584
    Wake-up 588
    Wake-up control 590
    Wake-up events 588
    Wake-up factor clear registers (WUFC) 629
    Wake-up factor mask registers (WUFMSK) 628
    Wake-up factor registers (WUF) 627
    Wake-up factors 578
    WUOSCSTA 584
Stand-by modes flows 598
Status Interrupt Enable Reset (FLXnSIER) 2403
Status Interrupt Enable Set (FLXnSIES) 2403
Status Interrupt Line Select (FLXnSILS) 2399
Status Interrupt Register (FLXnSIR) 2393
STBC
    see Stand-by Controller (STBC)
STOP mode 599
Stop Watch Register (FLXnSTPW) 2408, 2409
STPREQ_0 583
STPREQ_1 584
SubOsc 496
SUC Configuration Register 1 (FLXnSUCC1) 2410
SUC Configuration Register 2 (FLXnSUCC2) 2416

SUC Configuration Register 3 (FLXnSUCC3) 2417
Successive approximation register 2767
SWRES 650
SWRESA 654
Symbol Window and NIT Status (FLXnSWNIT) 2443
Sync Frame Status (FLXnSFS) 2441
Synchronized Timer Operation 1734
SYSERR (System error exceptions) 261
SYSRES 640
System Register Protection (SRP) 144

T

TAPA
see Timer Motor Control Function (TAPA)
TAPAINTSL0 295
TAPAnACTS 1589
TAPAnACTT 1589
TAPAnACWE 1588
TAPAnCTL0 1586
TAPAnCTL1 1587
TAPAnFLG 1588
TAPAnOPHS 1590
TAPAnOPHT 1590
TAUA
Assigning DMA Window Addresses 750
Channel Output Modes 731, 1347
Concepts of Synchronous Channel Operation 716, 1337
Functional Description 712, 1334
Functional Overview 709, 1332
General Operating Procedure 714, 1335
Independent functions 751, 1354
Interrupt Generation upon Overflow 744
Operation Modes 715, 1336
Registers 1065, 1382
see Timer Array Unit A (TAUA)
Simultaneous rewrite 719, 1340
Start Timing of Operating Modes 741, 1351
Synchronous functions 875, 1370
TAUAnTTINm Edge Detection 749
TAUAnTTOUTm Output and INTTAUAnIm Generation when Counter Starts or Restarts 743, 1353
TAUA input selection register (TISLTA0) 1628, 1692
TAUA input selection register (TISLTA0BYPS1) 1626, 1666, 1690
TAUA input selection register (TSOSLTA0) 1627, 1667, 1691
TAUA0 input selection register (TISLTA0) 1754
TAUA0 input selection register (TISLTA0BYPS0) 1752
TAUA0 reception input selection register (TRXSLTA0) 1753
TAUAnBRS 1069
TAUAnCDRm 1070, 1385
TAUAnCMORm 1073, 1387
TAUAnCMURm 1076
TAUAnCNTm 1071, 1386
TAUAnCSCm 1078, 1388
TAUAnCSRm 1077
TAUAnDASi 1090
TAUAnDWRm 1091
TAUAnEMU 1092, 1395
TAUAnRDC 1088, 1393
TAUAnRDE 1087, 1392
TAUAnRDM 1087, 1392
TAUAnRDS 1088, 1393
TAUAnRDT 1089, 1394
TAUAnRSF 1089, 1394
TAUAnTDE 1082
TAUAnTDL 1083
TAUAnTDM 1082
TAUAnTE 1079, 1389
TAUAnTME 1085
TAUAnTO 1086, 1391
TAUAnTOC 1081
TAUAnTOE 1080, 1390
TAUAnTOL 1086, 1391
TAUAnTOM 1080, 1390
TAUAnTPS 1067, 1383
TAUAnTRC 1084
TAUAnTRE 1084
TAUAnTRO 1085
TAUAnTS 1078, 1388
TAUAnTT 1079, 1389
TAUB
see Timer Array Unit B (TAUB)
TAUBnCDRm 1308
TAUBnCMORm 1311
TAUBnCMURm 1314
TAUBnCNTm 1309
TAUBnCSCm 1316
TAUBnCSRm 1315
TAUBnEMU 1325
TAUBnRDC 1323
TAUBnRDE 1322
TAUBnRDM 1322
TAUBnRDS 1323

TAUBnRDT 1324
TAUBnRSF 1324
TAUBnTDE 1320
TAUBnTDL 1320
TAUBnTDM 1320
TAUBnTE 1317
TAUBnTO 1321
TAUBnTOC 1319
TAUBnTOE 1318
TAUBnTOL 1321
TAUBnTOM 1318
TAUBnTPS 1306
TAUBnTS 1316
TAUBnTT 1317
TAUC
see Timer Array Unit C (TAUC)
TAUJ
see Timer Array Unit J (TAUJ)
TAUJ input selection register (TISLTJ) 1755
TAUJnBRS 1501
TAUJnCDRm 1502
TAUJnCMORm 1505
TAUJnCMURm 1509
TAUJnCNTm 1503
TAUJnCSCm 1511
TAUJnCSRm 1510
TAUJnEMU 1518
TAUJnRDE 1516
TAUJnRDM 1516
TAUJnRDT 1517
TAUJnTE 1512
TAUJnTO 1513
TAUJnTOC 1515
TAUJnTOE 1513
TAUJnTOL 1515
TAUJnTOM 1514
TAUJnTPS 1499
TAUJnTS 1511
TAUJnTT 1512
Test Register 1 (FLXnTEST1) 2382, 2386
Test Register 2 (FLXnTEST2) 2388
Three-Phase PWM Output with Dead Time 1603
Timer 0 Configuration (FLXnT0C) 2406
Timer 1 Configuration (FLXnT1C) 2407
Timer Array Unit A
TAUAnBRS 1069
TAUAnCDRm 1070, 1385
TAUAnCMORm 1073, 1387
TAUAnCMURm 1076
TAUAnCNTm 1071, 1386
TAUAnCSCm 1078, 1388
TAUAnCSRm 1077
TAUAnDASi 1090
TAUAnDWRm 1091
TAUAnEMU 1092, 1395
TAUAnRDC 1088, 1393
TAUAnRDE 1087, 1392
TAUAnRDM 1087, 1392
TAUAnRDS 1088, 1393
TAUAnRDT 1089, 1394
TAUAnRSF 1089, 1394
TAUAnTDE 1082
TAUAnTDL 1083
TAUAnTDM 1082
TAUAnTE 1079, 1389
TAUAnTME 1085
TAUAnTO 1086, 1391
TAUAnTOC 1081
TAUAnTOE 1080, 1390
TAUAnTOL 1086, 1391
TAUAnTOM 1080, 1390
TAUAnTPS 1067, 1383
TAUAnTRC 1084
TAUAnTRE 1084
TAUAnTRO 1085
TAUAnTS 1078, 1388
TAUAnTT 1079, 1389
Timer Array Unit A (TAUA) 697
<TAUAn_base> 697
Base addresses 697
Channel index m 697
Clock supply 698
I/O signals 699
Instances 697
Instances index n 697
Interrupts and DMA/DTS 698
Reset 699
TAUA0 input selection register (TSOSLTA0) 707
TAUA0 input selections 701
TAUA0 odd input selection register (TISLTA0) 705
TAUA0 receive input selection register (TRXSLTA0) 706
Timer Array Unit A channel counter register (TAUAnCNTm) 1071, 1386
Timer Array Unit A channel data register (TAUAnCDRm) 1070, 1385
Timer Array Unit A channel dead time output enable register (TAUAnTDE) 1082

Timer Array Unit A channel dead time output level register (TAUAnTDL) 1083
Timer Array Unit A channel dead time output mode register (TAUAnTDM) 1082
Timer Array Unit A channel enable status register (TAUAnTE) 1079, 1389
Timer Array Unit A channel mode OS register (TAUAnCMORm) 1073, 1387
Timer Array Unit A channel mode user register (TAUAnCMURm) 1076
Timer Array Unit A channel modulation output enable register (TAUAnTME) 1085
Timer Array Unit A channel output configuration register (TAUAnTOC) 1081
Timer Array Unit A channel output enable register (TAUAnTOE) 1080, 1390
Timer Array Unit A channel output level register (TAUAnTOL) 1086, 1391
Timer Array Unit A channel output mode register (TAUAnTOM) 1080, 1390
Timer Array Unit A channel output register (TAUAnTO) 1086, 1391
Timer Array Unit A channel real-time control register (TAUAnTRC) 1084
Timer Array Unit A channel real-time output enable register (TAUAnTRE) 1084
Timer Array Unit A channel real-time output register (TAUAnTRO) 1085
Timer Array Unit A channel reload data control channel select register (TAUAnRDS) 1088, 1393
Timer Array Unit A channel reload data control register (TAUAnRDC) 1088, 1393
Timer Array Unit A channel reload data enable register (TAUAnRDE) 1087, 1392
Timer Array Unit A channel reload data mode register (TAUAnRDM) 1087, 1392
Timer Array Unit A channel reload data trigger register (TAUAnRDT) 1089, 1394
Timer Array Unit A channel reload status register (TAUAnRSF) 1089, 1394
Timer Array Unit A channel start trigger register (TAUAnTS) 1078, 1388
Timer Array Unit A channel status clear register (TAUAnCSCm) 1078, 1388
Timer Array Unit A channel status register (TAUAnCSRm) 1077
Timer Array Unit A channel stop trigger register (TAUAnTT) 1079, 1389
Timer Array Unit A DMA window address setting register i (TAUAnDASi) 1090
Timer Array Unit A DMA window register m (TAUAnDWRm) 1091
Timer Array Unit A emulation register (TAUAnEMU) 1092, 1395
Timer Array Unit A prescaler baud rate setting register (TAUAnBRS) 1069
Timer Array Unit A prescaler clock select register (TAUAnTPS) 1067, 1383
Timer Array Unit B (TAUB) 1093
<TAUBn_base> 1093
Assigning DMA Window Addresses 1133
Base addresses 1093
Channel counter register (TAUBnCNTm) 1309
Channel data register (TAUBnCDRm) 1308
Channel dead time output enable register (TAUBnTDE) 1320
Channel dead time output level register (TAUBnTDL) 1320
Channel dead time output mode register (TAUBnTDM) 1320
Channel enable status register (TAUBnTE) 1317
Channel index m 1093
Channel mode OS register (TAUBnCMORm) 1311
Channel mode user register (TAUBnCMURm) 1314
Channel output configuration register (TAUBnTOC) 1319
Channel output enable register (TAUBnTOE) 1318
Channel output level register (TAUBnTOL) 1321
Channel output mode register (TAUBnTOM) 1318
Channel Output Modes 1122
Channel output register (TAUBnTO) 1321
Channel reload data control channel select register (TAUBnRDS) 1323
Channel reload data control register (TAUBnRDC) 1323
Channel reload data enable register (TAUBnRDE) 1322
Channel reload data mode register (TAUBnRDM) 1322
Channel reload data trigger register (TAUBnRDT) 1324
Channel reload status register (TAUBnRSF) 1324
Channel start trigger register (TAUBnTS) 1316
Channel status clear register (TAUBnCSCm) 1316
Channel status register

(TAUBnCSRm) 1315
Channel stop trigger register (TAUBnTT) 1317
Clock supply 1094
Concepts of Synchronous Channel Operation 1109
Emulation register (TAUBnEMU) 1325
Functional Description 1105
Functional Overview 1102
General Operating Procedure 1107
Independent functions 1134
Instances 1093
Instances index n 1093
Interrupt Generation upon Overflow 1132
Interrupts and DMA/DTS 1095
Operation Modes 1108
Prescaler clock select register (TAUBnTPS) 1306
Registers 1305
Reset 1096
Simultaneous rewrite 1112
Start Timing of Operating Modes 1129
Synchronous functions 1225
TAUB1 receive input selection register (TRXSLTA1) 1101
TAUB1TTIN input selections 1100
TAUBnTTINm Edge Detection 1133
TAUBnTTOUTm Output and INTTAUBnIm Generation when Counter Starts or Restarts 1131
Timer Array Unit B (TAUBn)
I/O signals 1097
Timer Array Unit C
I/O signals 1329
Timer Array Unit C (TAUC) 1326
<TAUCn_base> 1326
Base addresses 1326
Channel index m 1326
Clock supply 1327
Instances 1326
Instances index n 1326
Interrupts 1328
Reset 1328
Timer Array Unit J (TAUJ) 1396
<TAUJn_base> 1396
Base addresses 1396
Channel counter register (TAUJnCNTm) 1503
Channel data register (TAUJnCDRm) 1502
Channel enable status register (TAUJnTE) 1512
Channel index m 1396
Channel mode user register (TAUJnCMURm) 1509
Channel output configuration register (TAUJnTOC) 1515
Channel output enable register (TAUJnTOE) 1513
Channel output level register (TAUJnTOL) 1515
Channel output mode register (TAUJnTOM) 1514
Channel Output Modes 1421
Channel output register (TAUJnTO) 1513
Channel reload data enable register (TAUJnRDE) 1516
Channel reload data mode register (TAUJnRDM) 1516
Channel reload data trigger register (TAUJnRDT) 1517
Channel reload status register (TAUJnRSF) 1517
Channel start trigger register (TAUJnTS) 1511
Channel status clear register (TAUJnCSCm) 1511
Channel status register (TAUJnCSRm) 1510
Channel stop trigger register (TAUJnTT) 1512
Clock supply 1397
Concepts of Synchronous Channel Operation 1413
Emulation register (TAUJnEMU) 1518
Functional Description 1409
Functional Overview 1406
General Operating Procedure 1411
hannel mode OS register (TAUJnCMORm) 1505
I/O signals 1399
Independent Channel Interrupt Functions 1435
Independent Channel Signal Measurement Functions 1449
Input selections 1400
Instances 1396
Instances index n 1396
Interrupt Generation upon Overflow 1429
Interrupts and DMA/DTS 1398
Operation Modes 1412
Other Independent Channel Functions 1479
Prescaler baud rate setting register

(TAUJnBRS) 1501
Prescaler clock select register (TAUJnTPS) 1499
Registers 1498
Reset 1398
Simultaneous Rewrite 1416
Start Timing of Operating Modes 1426
Synchronous PWM Signal Functions Triggered at Regular Intervals 1486
TAUJ0/TAUJ1 input selection 1400
TAUJnTTINm Edge Detection 1434
TAUJnTTOUTm Output and INTTAUJnIm Generation when Counter Starts or Restarts 1428
Timer input selection register J (TISLTJ) 1403
Timer I/O control register 2n0 (PIC0REG2n0) 1657
Timer I/O control register 2n1 (PIC0REG2n1) 1658
Timer I/O control register 2n2 (PIC0REG2n2) 1624, 1661, 1688
Timer I/O control register 2n3 (PIC0REG2n3) 1663
Timer I/O control register 30 (PIC0REG30) 1732
Timer I/O control register 31 (PIC0REG31) 1404, 1750
Timer Motor Control Function (TAPA) 1578
<TAPAn_base> 1578
A/D conversion trigger selection function 1599
Asynchronous Hi-Z control function 1591
Asynchronous Hi-Z control start trigger register (TAPAnACTS) 1589
Asynchronous Hi-Z control stop trigger register (TAPAnACTT) 1589
Asynchronous Hi-Z control write enable register (TAPAnACWE) 1588
Base address 1578
Clock supply 1579
Control register 0 (TAPAnCTL0) 1586
Control register 1 (TAPAnCTL1) 1587
Flag register (TAPAnFLG) 1588
Functional overview 1583
Hi-Z control input selection 1581
Hi-Z start trigger register (TAPAnOPHS) 1590
Hi-Z stop trigger register (TAPAnOPHT) 1590
Instances 1578
Instances index n 1578
Internal signals 1580
Interrupt signal output selection function 1598
Interrupts 1579
Registers 1585
Reset 1579
Timing Supervision Unit (TSU) 148
<TSU_base> 148
Base address 148
TISLENCBYPS 1733
TISLTA0 705, 1628, 1692, 1754
TISLTA0BYPS0 1752
TISLTA0BYPS1 1626, 1666, 1690
TISLTJ 1403, 1755
Transmission Request n (FLXnTXRQm) 2464
TRGSEL0 1781
TRGSEL1 1782
TRGSEL2 1783
TRGSEL3 1784
TRGSEL4 1785
Trigger Pulse Width Measurement 1739
TRXSLTA0 706, 1753
TRXSLTA1 1101
TSOSLTA0 707, 1627, 1667, 1691
TSU
see Timing Supervision Unit (TSU)
TTAUJnRSF 1517

U

UARTEn
<URTEn_base> 1794
Base addresses 1794
Baudrate measurement 1798
Clock supply 1795
I/O signals 1797
Instances 1794
Instances index n 1794
Interrupts 1797
Reset 1797
URTEnCTL0 1802
URTEnCTL1 1804
URTEnCTL2 1807
URTEnEMU 1816
URTEnRX 1814
URTEnSTC 1813
URTEnSTR0 1810
URTEnSTR1 1811
URTEnTRG 1808
URTEnTX 1815

V

Variable activation code (WDTA) 684
VCPC
see Voltage Comparator (VCPC)
VCPCnCTLm 2843
VCPCnSTRm 2844
Vendor Information (FLXnVI) 2378
Very-Low-Voltage Indicator (VLVI) 646
VLVF 656
VLVFC 656
VLVI 646
Voltage Comparator (VCPC) 2836
<VCPCn_base> 2836
Base addresses 2836
Channel index m 2836
Clock supply 2837
Control registers (VCPCnCTLm) 2843
Description 2839
I/O signals 2838
Instances 2836
Instances index n 2836
Interrupts 2837
Overview 2839
Registers 2842
Reset 2837
Stand-by mode 2841
Status registers (VCPCnSTRm) 2844

W

Watchdog status (WDTA) 689
Watchdog Timers reset (WDTAnRES) 650
WDTA
see Window Watchdog Timer A (WDTA)
WDTAnEVAC 693
WDTAnMD 695
WDTAnREF 694
WDTAnRES 650
WDTAnWDTE 691
Window function (WDTA) 688
Window Watchdog Timer A (WDTA) 674
<WDTAn_base> 674
75% interrupt output 687
Base addresses 674
Clock supply 675
Enable register (WDTAnWDTE) 691
Enable VAC register (WDTAnEVAC) 693
Error detection 685
Functional description 680
Instances 674
Instances index n 674
Interrupts and reset outputs 676
Mode register (WDTAnMD) 695
Reference value register (WDTAnREF) 694
Registers 690
Reset 676
Start modes 681
Start-up options 677
Variable activation code 684
Watchdog status 689
WDTA trigger 684
Window function 688
Write Data Section (FLXnWRDSn) 2473
Write Header Section 1 (FLXnWRHS1) 2474
Write Header Section 2 (FLXnWRHS2) 2476
Write Header Section 3 (FLXnWRHS3) 2477
Write protected registers 206
Clock monitors protection cluster registers 215
Clock/stand-by/reset protection cluster registers 213
Control protection clusters registers 213
Port protection cluster registers 216
Protection command register m (PROTCMDm) 213
Protection registers overview 211
Protection status register m (PROTSm) 214
Register protection clusters 206
Self-programming protection cluster registers 217
WUF 627
WUFC 629
WUFLm 630
WUFMSK 628
WUOSCSTA 584

V850E2/Fx4-H User Manual

Publication Date: Rev. 2.00 April 29, 2013

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
7F, No. 363 Fu Shing North Road Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
1 harbourFront Avenue, #06-10, keppel Bay Tower, Singapore 098632
Tel: +65-6213-0200, Fax: +65-6278-8001

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.0

V850E2/Fx4-H

Renesas Electronics Corporation

R01UH0002ED0200