To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

RENESAS

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.

User's Manual

32

H8SX/1520R Group

Hardware Manual

Renesas 32-Bit CISC
Microcomputer
H8SX Family H8SX/1500 Series

H8SX/1527R R5F61527R
H8SX/1525R R5F61525R

Renesas Electronics
www.renesas.com

Rev.1.00 2006.03

Keep safety first in your circuit designs!

1. Renesas Technology Corp. puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage.
   Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of nonflammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

1. These materials are intended as a reference to assist our customers in the selection of the Renesas Technology Corp. product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Renesas Technology Corp. or a third party.
2. Renesas Technology Corp. assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
3. All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Renesas Technology Corp. without notice due to product improvements or other reasons. It is therefore recommended that customers contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor for the latest product information before purchasing a product listed herein.
   The information described here may contain technical inaccuracies or typographical errors. Renesas Technology Corp. assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.
   Please also pay attention to information published by Renesas Technology Corp. by various means, including the Renesas Technology Corp. Semiconductor home page (http://www.renesas.com).
4. When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Renesas Technology Corp. assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
5. Renesas Technology Corp. semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Renesas Technology Corp. or an authorized Renesas Technology Corp. product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
6. The prior written approval of Renesas Technology Corp. is necessary to reprint or reproduce in whole or in part these materials.
7. If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
   Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
8. Please contact Renesas Technology Corp. for further details on these materials or the products contained therein.

General Precautions on Handling of Product

1. Treatment of NC Pins

Note: Do not connect anything to the NC pins.
The NC (not connected) pins are either not connected to any of the internal circuitry or are used as test pins or to reduce noise. If something is connected to the NC pins, the operation of the LSI is not guaranteed.

2. Treatment of Unused Input Pins

Note: Fix all unused input pins to high or low level.
Generally, the input pins of CMOS products are high-impedance input pins. If unused pins are in their open states, intermediate levels are induced by noise in the vicinity, a pass-through current flows internally, and a malfunction may occur.

3. Processing before Initialization

Note: When power is first supplied, the product's state is undefined.
The states of internal circuits are undefined until full power is supplied throughout the chip and a low level is input on the reset pin. During the period where the states are undefined, the register settings and the output state of each pin are also undefined. Design your system so that it does not malfunction because of processing while it is in this undefined state. For those products which have a reset function, reset the LSI immediately after the power supply has been turned on.

4. Prohibition of Access to Undefined or Reserved Addresses

Note: Access to undefined or reserved addresses is prohibited.
The undefined or reserved addresses may be used to expand functions, or test registers may have been be allocated to these addresses. Do not access these registers; the system's operation is not guaranteed if they are accessed.

Configuration of This Manual

This manual comprises the following items:

1. General Precautions on Handling of Product
2. Configuration of This Manual
3. Preface
4. Contents
5. Overview
6. Description of Functional Modules
   - CPU and System-Control Modules
   - On-Chip Peripheral Modules

   The configuration of the functional description of each module differs according to the module. However, the generic style includes the following items:

   i) Feature
   ii) Input/Output Pin
   iii) Register Description
   iv) Operation
   v) Usage Note

When designing an application system that includes this LSI, take notes into account. Each section includes notes in relation to the descriptions given, and usage notes are given, as required, as the final part of each section.

7. List of Registers
8. Electrical Characteristics
9. Appendix
10. Main Revisions and Additions in this Edition (only for revised versions)

The list of revisions is a summary of points that have been revised or added to earlier versions. This does not include all of the revised contents. For details, see the actual locations in this manual.

11. Index

Preface

The H8SX/1520R Group is a single-chip microcomputer made up of the high-speed internal 32-bit H8SX CPU as its core, and the peripheral functions required to configure a system. The H8SX CPU is upward compatible with the H8/300, H8/300H, and H8S CPUs.

Target Users: This manual was written for users who will be using the H8SX/1520R Group in the design of application systems. Target users are expected to understand the fundamentals of electrical circuits, logical circuits, and microcomputers.

Objective: This manual was written to explain the hardware functions and electrical characteristics of the H8SX/1520R Group to the target users.
Refer to the H8SX Family Software Manual for a detailed description of the instruction set.

Notes on reading this manual:

In order to understand the overall functions of the chip

Read the manual according to the contents. This manual can be roughly categorized into parts on the CPU, system control functions, and peripheral functions.

In order to understand the details of the CPU's functions

Read the H8SX Family Software Manual.

In order to understand the details of a register when its name is known

Read the index that is the final part of the manual to find the page number of the entry on the register. The addresses, bits, and initial values of the registers are summarized in section 20, List of Registers.

Examples:		
	Register name:	The following notation is used for cases when the same or a similar function, e.g. 16-bit timer pulse unit or serial communication interface, is implemented on more than one channel: XXX_N (XXX is the register name and N is the channel number)
	Bit order:	The MSB is on the left and the LSB is on the right.
	Number notation:	Binary is B'xxxx, hexadecimal is H'xxxx, decimal is xxxx.
	Signal notation:	An overbar is added to a low-active signal: $\overline{\text{xxxx}}$

Related Manuals: The latest versions of all related manuals are available from our web site.
Please ensure you have the latest versions of all documents you require.
http://www.renesas.com/

H8SX/1520R Group manuals:

Document Title	Document No.
H8SX/1520R Group Hardware Manual	This manual
H8/SX Family Software Manual	REJ09B0102

Contents

Section 1 Overview 1
1.1 Features 1
1.2 Block Diagram 2
1.3 Pin Assignments 4
1.3.1 Pin Assignments 4
1.3.2 Pin Configuration in Each Operating Mode 6
1.3.3 Pin Functions 10

Section 2 CPU 19
2.1 Features 19
2.2 CPU Operating Modes 21
2.2.1 Normal Mode 21
2.2.2 Middle Mode 23
2.2.3 Advanced Mode 24
2.2.4 Maximum Mode 25
2.3 Instruction Fetch 27
2.4 Address Space 27
2.5 Registers 28
2.5.1 General Registers 29
2.5.2 Program Counter (PC) 30
2.5.3 Condition-Code Register (CCR) 30
2.5.4 Extended Control Register (EXR) 32
2.5.5 Vector Base Register (VBR) 32
2.5.6 Short Address Base Register (SBR) 32
2.5.7 Multiply-Accumulate Register (MAC) 33
2.5.8 Initial Values of CPU Registers 33
2.6 2Data Formats 33
2.6.1 General Register Data Formats 33
2.6.2 Memory Data Formats 35
2.7 Instruction Set 36
2.7.1 Instructions and Addressing Modes 38
2.7.2 Table of Instructions Classified by Function 42
2.7.3 Basic Instruction Formats 53
2.8 Addressing Modes and Effective Address Calculation 54
2.8.1 Register Direct—Rn 55
2.8.2 Register Indirect—@ERn 55

2.8.3 Register Indirect with Displacement—@(d:2, ERn), @(d:16, ERn), or @(d:32, ERn) ... 55
2.8.4 Index Register Indirect with Displacement—@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L) ... 56
2.8.5 Register Indirect with Post-Increment, Pre-Decrement, Pre-Increment, or Post-Decrement—@ERn+, @–ERn, @+ERn, or @ERn– ... 56
2.8.6 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32 ... 57
2.8.7 Immediate—#xx ... 58
2.8.8 Program-Counter Relative—@(d:8, PC) or @(d:16, PC) ... 59
2.8.9 Program-Counter Relative with Index Register— @(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC) ... 59
2.8.10 Memory Indirect—@@aa:8 ... 59
2.8.11 Extended Memory Indirect—@@vec:7 ... 60
2.8.12 Effective Address Calculation ... 60
2.8.13 MOVA Instruction ... 62
2.9 Processing States ... 63

Section 3 MCU Operating Modes ... 65
3.1 Operating Mode Selection ... 65
3.2 Register Descriptions ... 66
3.2.1 Mode Control Register (MDCR) ... 66
3.2.2 System Control Register (SYSCR) ... 67
3.3 Operating Mode Descriptions ... 69
3.3.1 Mode 1 ... 69
3.3.2 Mode 2 ... 69
3.3.3 Mode 3 ... 69
3.4 Address Map ... 70
3.4.1 Address Map (Advanced Mode) ... 70

Section 4 Exception Handling ... 71
4.1 Exception Handling Types and Priority ... 71
4.2 Exception Sources and Exception Handling Vector Table ... 72
4.3 Reset ... 74
4.3.1 Reset Exception Handling ... 74
4.3.2 Interrupts after Reset ... 75
4.3.3 On-Chip Peripheral Functions after Reset Release ... 75
4.4 Traces ... 76
4.5 Address Error ... 77
4.5.1 Address Error Source ... 77

4.5.2 Address Error Exception Handling 78
4.6 Interrupts 79
4.6.1 Interrupt Sources 79
4.6.2 Interrupt Exception Handling 80
4.7 Instruction Exception Handling 81
4.7.1 Trap Instruction 81
4.7.2 Exception Handling by Illegal Instruction 82
4.8 Stack Status after Exception Handling 83
4.9 Usage Note 84

Section 5 Interrupt Controller 85
5.1 Features 85
5.2 Input/Output Pins 86
5.3 Register Descriptions 87
5.3.1 Interrupt Control Register (INTCR) 87
5.3.2 CPU Priority Control Register (CPUPCR) 88
5.3.3 Interrupt Priority Registers A to G, I, K to O, Q, and R (IPRA to IPRG, IPRI, IPRK to IPRO, IPRQ, and IPRR) 90
5.3.4 IRQ Enable Register (IER) 92
5.3.5 IRQ Sense Control Registers H and L (ISCRH and ISCRL) 94
5.3.6 IRQ Status Register (ISR) 99
5.3.7 Software Standby Release IRQ Enable Register (SSIER) 100
5.4 Interrupt Sources 101
5.4.1 External Interrupts 101
5.4.2 Internal Interrupts 102
5.5 Interrupt Exception Handling Vector Table 103
5.6 Interrupt Control Modes and Interrupt Operation 110
5.6.1 Interrupt Control Mode 0 110
5.6.2 Interrupt Control Mode 2 112
5.6.3 Interrupt Exception Handling Sequence 114
5.6.4 Interrupt Response Times 115
5.6.5 DMAC Activation by Interrupt 116
5.7 CPU Priority Control Function Over DMAC 118
5.8 Usage Notes 121
5.8.1 Conflict between Interrupt Generation and Disabling 121
5.8.2 Instructions that Disable Interrupts 122
5.8.3 Times when Interrupts are Disabled 122
5.8.4 Interrupts during Execution of EEPMOV Instruction 122
5.8.5 Interrupts during Execution of MOVMD and MOVSD Instructions 122
5.8.6 Interrupt Flags of Peripheral Modules 123

Section 6 Bus Controller (BSC) 125
6.1 Features 125
6.2 Register Descriptions 126
6.2.1 Bus Control Register 2 (BCR2) 126
6.3 Bus Configuration 127
6.4 Multi-Clock Function 128
6.5 Internal Bus 129
6.5.1 Access to Internal Address Space 129
6.6 Write Data Buffer Function 130
6.6.1 Write Data Buffer Function for Peripheral Module 130
6.7 Bus Arbitration 131
6.7.1 Operation 131
6.7.2 Bus Transfer Timing 131
6.8 Bus Controller Operation in Reset 132
6.9 Usage Notes 132

Section 7 DMA Controller (DMAC) 133
7.1 Features 133
7.2 Register Descriptions 136
7.2.1 DMA Source Address Register (DSAR) 137
7.2.2 DMA Destination Address Register (DDAR) 138
7.2.3 DMA Offset Register (DOFR) 139
7.2.4 DMA Transfer Count Register (DTCR) 140
7.2.5 DMA Block Size Register (DBSR) 141
7.2.6 DMA Mode Control Register (DMDR) 142
7.2.7 DMA Address Control Register (DACR) 151
7.2.8 DMA Module Request Select Register (DMRSR) 157
7.3 Transfer Modes 157
7.4 Operations 158
7.4.1 Address Modes 158
7.4.2 Transfer Modes 162
7.4.3 Activation Sources 166
7.4.4 Bus Access Modes 168
7.4.5 Extended Repeat Area Function 170
7.4.6 Address Update Function using Offset 172
7.4.7 Register during DMA Transfer 176
7.4.8 Priority of Channels 181
7.4.9 DMA Basic Bus Cycle 182
7.4.10 Bus Cycles in Dual Address Mode 183
7.4.11 Bus Cycles in Single Address Mode 191

7.5 DMA Transfer End 196
7.6 Relationship among DMAC and Other Bus Masters 198
7.6.1 CPU Priority Control Function Over DMAC 198
7.6.2 Bus Arbitration among DMAC and Other Bus Masters 199
7.7 Interrupt Sources 200
7.8 Notes on Usage 203

Section 8 I/O Ports 205
8.1 Register Descriptions 210
8.1.1 Data Direction Register (PnDDR) (n = 1 to 3, 6, A, D, H, J, and K) 212
8.1.2 Data Register (PnDR) (n = 1 to 3, 6, A, D, H, J, and K) 212
8.1.3 Port Register (PORTn) (n = 1 to 6, A, D, H, J, and K) 213
8.1.4 Input Buffer Control Register (PnICR) (n = 1 to 6, A, D, H, J, and K) 213
8.1.5 Pull-Up MOS Control Register (PnPCR) (n = D, H, J, and K) 214
8.1.6 Open-Drain Control Register (PnODR) (n = 2) 215
8.1.7 Port H Realtime Input Data Register (PHRTIDR) 215
8.2 Output Buffer Control 216
8.2.1 Port 1 216
8.2.2 Port 2 219
8.2.3 Port 3 221
8.2.4 Port 6 225
8.2.5 Port A 227
8.2.6 Port D 230
8.2.7 Port H 233
8.2.8 Port J 233
8.2.9 Port K 236
8.3 Port Function Controller 243
8.3.1 Port Function Control Register 9 (PFCR9) 243
8.3.2 Port Function Control Register A (PFCRA) 245
8.3.3 Port Function Control Register B (PFCRB) 247
8.4 Usage Notes 249
8.4.1 Notes on Input Buffer Control Register (ICR) Setting 249
8.4.2 Notes on Port Function Control Register (PFCR) Settings 249

Section 9 16-Bit Timer Pulse Unit (TPU) 251
9.1 Features 251
9.2 Input/Output Pins 258
9.3 Register Descriptions 260
9.3.1 Timer Control Register (TCR) 265
9.3.2 Timer Mode Register (TMDR) 270

9.3.3 Timer I/O Control Register (TIOR) ... 272
9.3.4 Timer Interrupt Enable Register (TIER) ... 290
9.3.5 Timer Status Register (TSR) ... 292
9.3.6 Timer Counter (TCNT) ... 296
9.3.7 Timer General Register (TGR) ... 296
9.3.8 Timer Start Register (TSTR) ... 297
9.3.9 Timer Synchronous Register (TSYR) ... 298
9.4 Operation ... 299
9.4.1 Basic Functions ... 299
9.4.2 Synchronous Operation ... 305
9.4.3 Buffer Operation ... 307
9.4.4 Cascaded Operation ... 311
9.4.5 PWM Modes ... 313
9.4.6 Phase Counting Mode ... 318
9.5 Interrupt Sources ... 326
9.6 DMAC Activation ... 329
9.7 A/D Converter Activation ... 329
9.8 Operation Timing ... 330
9.8.1 Input/Output Timing ... 330
9.8.2 Interrupt Signal Timing ... 334
9.9 Usage Notes ... 337
9.9.1 Module Stop Mode Setting ... 337
9.9.2 Input Clock Restrictions ... 337
9.9.3 Caution on Cycle Setting ... 338
9.9.4 Conflict between TCNT Write and Clear Operations ... 338
9.9.5 Conflict between TCNT Write and Increment Operations ... 339
9.9.6 Conflict between TGR Write and Compare Match ... 339
9.9.7 Conflict between Buffer Register Write and Compare Match ... 340
9.9.8 Conflict between TGR Read and Input Capture ... 340
9.9.9 Conflict between TGR Write and Input Capture ... 341
9.9.10 Conflict between Buffer Register Write and Input Capture ... 341
9.9.11 Conflict between Overflow/Underflow and Counter Clearing ... 342
9.9.12 Conflict between TCNT Write and Overflow/Underflow ... 342
9.9.13 Multiplexing of I/O Pins ... 343
9.9.14 Interrupts and Module Stop Mode ... 343

Section 10 Programmable Pulse Generator (PPG) ... 345
10.1 Features ... 345
10.2 Input/Output Pins ... 346
10.3 Register Descriptions ... 347

10.3.1 Next Data Enable Registers H, L (NDERH, NDERL) 347
10.3.2 Output Data Registers H, L (PODRH, PODRL)..... 349
10.3.3 Next Data Registers H, L (NDRH, NDRL) 350
10.3.4 PPG Output Control Register (PCR) 353
10.3.5 PPG Output Mode Register (PMR) 354
10.4 Operation 356
10.4.1 Output Timing..... 356
10.4.2 Sample Setup Procedure for Normal Pulse Output..... 357
10.4.3 Example of Normal Pulse Output (Example of 5-Phase Pulse Output)..... 358
10.4.4 Non-Overlapping Pulse Output..... 359
10.4.5 Sample Setup Procedure for Non-Overlapping Pulse Output..... 361
10.4.6 Example of Non-Overlapping Pulse Output
(Example of 4-Phase Complementary Non-Overlapping Pulse Output) 362
10.4.7 Inverted Pulse Output 364
10.4.8 Pulse Output Triggered by Input Capture 365
10.5 Usage Notes 365
10.5.1 Module Stop Mode Setting 365
10.5.2 Operation of Pulse Output Pins..... 365

Section 11 Watchdog Timer (WDT).....367
11.1 Features..... 367
11.2 Register Descriptions 368
11.2.1 Timer Counter (TCNT)..... 368
11.2.2 Timer Control/Status Register (TCSR)..... 369
11.2.3 Reset Control/Status Register (RSTCSR)..... 371
11.3 Operation 372
11.3.1 Watchdog Timer Mode 372
11.3.2 Interval Timer Mode 373
11.4 Interrupt Source 373
11.5 Usage Notes 374
11.5.1 Notes on Register Access..... 374
11.5.2 Conflict between Timer Counter (TCNT) Write and Increment..... 375
11.5.3 Changing Values of Bits CKS2 to CKS0..... 375
11.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode..... 375
11.5.5 Transition to Watchdog Timer Mode or Software Standby Mode..... 376

Section 12 Serial Communication Interface (SCI) 377
12.1 Features..... 377
12.2 Input/Output Pins..... 379
12.3 Register Descriptions 380

12.3.1 Receive Shift Register (RSR) 381
12.3.2 Receive Data Register (RDR) 381
12.3.3 Transmit Data Register (TDR) 381
12.3.4 Transmit Shift Register (TSR) 382
12.3.5 Serial Mode Register (SMR) 382
12.3.6 Serial Control Register (SCR) 385
12.3.7 Serial Status Register (SSR) 390
12.3.8 Smart Card Mode Register (SCMR) 398
12.3.9 Bit Rate Register (BRR) 399
12.4 Operation in Asynchronous Mode 406
12.4.1 Data Transfer Format 407
12.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode 408
12.4.3 Clock 409
12.4.4 SCI Initialization (Asynchronous Mode) 410
12.4.5 Serial Data Transmission (Asynchronous Mode) 411
12.4.6 Serial Data Reception (Asynchronous Mode) 413
12.5 Multiprocessor Communication Function 417
12.5.1 Multiprocessor Serial Data Transmission 419
12.5.2 Multiprocessor Serial Data Reception 420
12.6 Operation in Clocked Synchronous Mode 423
12.6.1 Clock 423
12.6.2 SCI Initialization (Clocked Synchronous Mode) 424
12.6.3 Serial Data Transmission (Clocked Synchronous Mode) 425
12.6.4 Serial Data Reception (Clocked Synchronous Mode) 427
12.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode) 428
12.7 Operation in Smart Card Interface Mode 430
12.7.1 Sample Connection 430
12.7.2 Data Format (Except in Block Transfer Mode) 431
12.7.3 Block Transfer Mode 432
12.7.4 Receive Data Sampling Timing and Reception Margin 433
12.7.5 Initialization 434
12.7.6 Data Transmission (Except in Block Transfer Mode) 435
12.7.7 Serial Data Reception (Except in Block Transfer Mode) 438
12.7.8 Clock Output Control 439
12.8 Interrupt Sources 441
12.8.1 Interrupts in Normal Serial Communication Interface Mode 441
12.8.2 Interrupts in Smart Card Interface Mode 442
12.9 Usage Notes 443

12.9.1 Module Stop Mode Setting 443
12.9.2 Break Detection and Processing 443
12.9.3 Mark State and Break Detection 443
12.9.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only) 443
12.9.5 Relation between Writing to TDR and TDRE Flag 444
12.9.6 Restrictions on Using DMAC 444
12.9.7 SCI Operations during Mode Transitions 445

Section 13 Controller Area Network (RCAN-ET) 449
13.1 Summary 449
13.1.1 Overview 449
13.1.2 Scope 449
13.1.3 Audience 449
13.1.4 References 450
13.1.5 Features 450
13.2 Architecture 451
13.2.1 Block Diagram 451
13.2.2 Important 452
13.2.3 Input/Output Pins 453
13.2.4 Memory Map 454
13.3 Mailbox 455
13.3.1 Mailbox Structure 455
13.3.2 Message Control Field 457
13.3.3 Local Acceptance Filter Mask (LAFM) 461
13.3.4 Message Data Fields 462
13.4 RCAN-ET Control Registers 463
13.4.1 Master Control Register (MCR) 463
13.4.2 General Status Register (GSR) 469
13.4.3 Bit Configuration Register (BCR0, BCR1) 471
13.4.4 Interrupt Request Register (IRR) 476
13.4.5 Interrupt Mask Register (IMR) 483
13.4.6 Transmit Error Counter (TEC) and Receive Error Counter (REC) 484
13.5 RCAN-ET Mailbox Registers 485
13.5.1 Transmit Pending Register (TXPR0, TXPR1) 486
13.5.2 Transmit Cancel Register (TXCR0) 489
13.5.3 Transmit Acknowledge Register0 (TXACK0) 490
13.5.4 Abort Acknowledge Register0 (ABACK0) 491
13.5.5 Data Frame Receive Pending Register0 (RXPR0) 492
13.5.6 Remote Frame Receive Pending Register0 (RFPR0) 493

13.5.7 Mailbox Interrupt Mask Register0 (MBIMR0) 494
13.5.8 Unread Message Status Register0 (UMSR0) 495
13.5.9 RCAN Monitor Register (RCANMON) 496
13.6 Application Note 497
13.6.1 Configuration of RCAN-ET 497
13.6.2 Test Mode Settings 502
13.6.3 Message Transmission Sequence 504
13.6.4 Message Receive Sequence 506
13.6.5 Reconfiguration of Mailbox 508
13.7 Interrupt Sources 510
13.8 DMAC Interface 511
13.9 CAN Bus Interface 512
13.10 Usage Notes 513
13.10.1 Module Stop Mode 513
13.10.2 Reset 513
13.10.3 CAN Sleep Mode 513
13.10.4 Register Access 513
13.10.5 Interrupts 514

Section 14 Synchronous Serial Communication Unit (SSU) 515
14.1 Features 515
14.2 Input/Output Pins 517
14.3 Register Descriptions 518
14.3.1 SS Control Register H (SSCRH) 520
14.3.2 SS Control Register L (SSCRL) 522
14.3.3 SS Mode Register (SSMR) 523
14.3.4 SS Enable Register (SSER) 524
14.3.5 SS Status Register (SSSR) 525
14.3.6 SS Control Register 2 (SSCR2) 528
14.3.7 SS Transmit Data Registers 0 to 3 (SSTDR0 to SSTDR3) 530
14.3.8 SS Receive Data Registers 0 to 3 (SSRDR0 to SSRDR3) 531
14.3.9 SS Shift Register (SSTRSR) 533
14.4 Operation 534
14.4.1 Transfer Clock 534
14.4.2 Relationship of Clock Phase, Polarity, and Data 534
14.4.3 Relationship between Data Input/Output Pins and Shift Register 535
14.4.4 Communication Modes and Pin Functions 536
14.4.5 SSU Mode 538
14.4.6 $\overline{\text{SCS}}$ Pin Control and Conflict Error 549
14.4.7 Clock Synchronous Communication Mode 550

14.5 Interrupt Requests 557
14.6 Usage Note 558
14.6.1 Setting of Module Stop Mode 558

Section 15 A/D Converter 559
15.1 Features 559
15.2 Input/Output Pins 562
15.3 Register Descriptions 563
15.3.1 A/D Data Registers A to H (ADDRA to ADDRH) 564
15.3.2 A/D Control/Status Register (ADCSR) 565
15.3.3 A/D Control Register (ADCR) 567
15.4 Operation 568
15.4.1 Single Mode 568
15.4.2 Scan Mode 569
15.4.3 Input Sampling and A/D Conversion Time 571
15.4.4 External Trigger Input Timing 572
15.5 Interrupt Source 573
15.6 A/D Conversion Accuracy Definitions 573
15.7 Usage Notes 575
15.7.1 Module Stop Mode Setting 575
15.7.2 Permissible Signal Source Impedance 575
15.7.3 Influences on Absolute Accuracy 576
15.7.4 Setting Range of Analog Power Supply and Other Pins 576
15.7.5 Notes on Board Design 576
15.7.6 Notes on Noise Countermeasures 577
15.7.7 A/D Input Hold Function in Software Standby Mode 578

Section 16 RAM 579

Section 17 Flash Memory (0.18-μm F-ZTAT Version) 581
17.1 Features 581
17.2 Mode Transition Diagram 584
17.3 Memory MAT Configuration 586
17.4 Block Structure 587
17.5 Programming/Erasing Interface 588
17.6 Input/Output Pins 590
17.7 Register Descriptions 591
17.7.1 Programming/Erasing Interface Registers 592
17.7.2 Programming/Erasing Interface Parameters 599
17.7.3 RAM Emulation Register (RAMER) 612

17.8 On-Board Programming Mode 613
17.8.1 Boot Mode 613
17.8.2 User Program Mode 617
17.8.3 User Boot Mode 627
17.8.4 On-Chip Program and Storable Area for Program Data 631
17.9 Protection 637
17.9.1 Hardware Protection 637
17.9.2 Software Protection 638
17.9.3 Error Protection 638
17.10 Flash Memory Emulation Using RAM 640
17.11 Switching between User MAT and User Boot MAT 643
17.12 Programmer Mode 644
17.13 Standard Serial Communication Interface Specifications for Boot Mode 644
17.14 Usage Notes 672

Section 18 Clock Pulse Generator 677
18.1 Register Description 678
18.1.1 System Clock Control Register (SCKCR) 678
18.2 Oscillator 681
18.2.1 Connecting Crystal Resonator 681
18.2.2 External Clock Input 682
18.3 PLL Circuit 682
18.4 Frequency Divider 682
18.5 Usage Notes 683
18.5.1 Notes on Clock Pulse Generator 683
18.5.2 Notes on Resonator 684
18.5.3 Notes on Board Design 684
18.5.4 Notes on Input Clock Frequency 685

Section 19 Power-Down Modes 687
19.1 Features 687
19.2 Register Descriptions 689
19.2.1 Standby Control Register (SBYCR) 689
19.2.2 Module Stop Control Registers A and B (MSTPCRA and MSTPCRB) 691
19.2.3 Module Stop Control Register C (MSTPCRC) 694
19.3 Multi-Clock Function 695
19.4 Module Stop Mode 695
19.5 Sleep Mode 696
19.5.1 Transition to Sleep Mode 696
19.5.2 Clearing Sleep Mode 696

19.6 All-Module-Clock-Stop Mode 696
19.7 Software Standby Mode 697
19.7.1 Transition to Software Standby Mode 697
19.7.2 Clearing Software Standby Mode 697
19.7.3 Setting Oscillation Settling Time after Clearing Software Standby Mode 698
19.7.4 Software Standby Mode Application Example 700
19.8 Bϕ Clock Output Control 701
19.9 Usage Notes 702
19.9.1 I/O Port Status 702
19.9.2 Current Consumption during Oscillation Settling Standby Period 702
19.9.3 DMAC Module Stop 702
19.9.4 On-Chip Peripheral Module Interrupts 702
19.9.5 Writing to MSTPCRA, MSTPCRB, and MSTPCRC 702

Section 20 List of Registers 703
20.1 Register Addresses (Address Order) 704
20.2 Register Bits 723
20.3 Register States in Each Operating Mode 743

Section 21 Electrical Characteristics 757
21.1 Absolute Maximum Ratings 757
21.2 DC Characteristics 758
21.3 AC Characteristics 760
21.3.1 Clock Timing 761
21.3.2 Control Signal Timing 763
21.3.3 Timing of On-Chip Peripheral Modules 764
21.4 A/D Conversion Characteristics 772
21.5 Flash Memory Characteristics 773

Appendix 775
A. Port States in Each Pin State 775
B. Product Lineup 776
C. Package Dimensions 777

Index 779

Figures

Section 1 Overview
Figure 1.1 Block Diagram of H8SX/1527R 2
Figure 1.2 Block Diagram of H8SX/1525R 3
Figure 1.3 Pin Assignments of H8SX/1527R 4
Figure 1.4 Pin Assignments of H8SX/1525R 5

Section 2 CPU
Figure 2.1 CPU Operating Modes 21
Figure 2.2 Exception Vector Table (Normal Mode) 22
Figure 2.3 Stack Structure (Normal Mode) 22
Figure 2.4 Exception Vector Table (Middle and Advanced Modes) 24
Figure 2.5 Stack Structure (Middle and Advanced Modes) 25
Figure 2.6 Exception Vector Table (Maximum Modes) 26
Figure 2.7 Stack Structure (Maximum Mode) 26
Figure 2.8 Memory Map 27
Figure 2.9 CPU Registers 28
Figure 2.10 Usage of General Registers 29
Figure 2.11 Stack 30
Figure 2.12 General Register Data Formats 34
Figure 2.13 Memory Data Formats 35
Figure 2.14 Instruction Formats 53
Figure 2.15 Branch Address Specification in Memory Indirect Mode 60
Figure 2.16 State Transitions 64

Section 3 MCU Operating Modes
Figure 3.1 Address Map (Advanced Mode) 70

Section 4 Exception Handling
Figure 4.1 Reset Sequence (On-Chip ROM Enabled Advanced Mode) 75
Figure 4.2 Stack Status after Exception Handling 83
Figure 4.3 Operation when SP Value Is Odd 84

Section 5 Interrupt Controller
Figure 5.1 Block Diagram of Interrupt Controller 86
Figure 5.2 Block Diagram of Interrupts IRQn 102
Figure 5.3 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0 ... 111
Figure 5.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2 ... 113
Figure 5.5 Interrupt Exception Handling 114
Figure 5.6 Block Diagram of DMAC and Interrupt Controller 116

Figure 5.7 Conflict between Interrupt Generation and Disabling 121

Section 6 Bus Controller (BSC)
Figure 6.1 Block Diagram of Bus Controller 125
Figure 6.2 Internal Bus Configuration 127
Figure 6.3 Example of Timing when Write Data Buffer Function is Used 130

Section 7 DMA Controller (DMAC)
Figure 7.1 Block Diagram of DMAC 135
Figure 7.2 Example of Signal Timing in Dual Address Mode 159
Figure 7.3 Operations in Dual Address Mode 159
Figure 7.4 Data Flow in Single Address Mode 160
Figure 7.5 Example of Signal Timing in Single Address Mode 161
Figure 7.6 Operations in Single Address Mode 161
Figure 7.7 Example of Signal Timing in Normal Transfer Mode 162
Figure 7.8 Operations in Normal Transfer Mode 162
Figure 7.9 Operations in Repeat Transfer Mode 163
Figure 7.10 Operations in Block Transfer Mode 164
Figure 7.11 Operation in Single Address Mode in Block Transfer Mode (Block Area Specified) 165
Figure 7.12 Operation in Dual Address Mode in Block Transfer Mode (Block Area Not Specified) 165
Figure 7.13 Example of Timing in Cycle Stealing Mode 169
Figure 7.14 Example of Timing in Burst Mode 169
Figure 7.15 Example of Extended Repeat Area Operation 171
Figure 7.16 Example of Extended Repeat Area Function in Block Transfer Mode 171
Figure 7.17 Address Update Method 172
Figure 7.18 Operation of Offset Addition 173
Figure 7.19 XY Conversion Operation Using Offset Addition in Repeat Transfer Mode 174
Figure 7.20 XY Conversion Flowchart Using Offset Addition in Repeat Transfer Mode 175
Figure 7.21 Procedure for Changing Register Setting For Channel being Transferred 179
Figure 7.22 Example of Timing for Channel Priority 181
Figure 7.23 Example of Bus Timing of DMA Transfer 182
Figure 7.24 Example of Transfer in Normal Transfer Mode by Cycle Stealing 183
Figure 7.25 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Source DSAR = Odd Address and Source Address Increment) 184
Figure 7.26 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Destination DDAR = Odd Address and Destination Address Decrement) 184
Figure 7.27 Example of Transfer in Normal Transfer Mode by Burst Access 185
Figure 7.28 Example of Transfer in Block Transfer Mode 186

Figure 7.29 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Falling Edge 187
Figure 7.30 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level 188
Figure 7.31 Example of Transfer in Block Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level 189
Figure 7.32 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1 190
Figure 7.33 Example of Transfer in Single Address Mode (Byte Read) 191
Figure 7.34 Example of Transfer in Single Address Mode (Byte Write) 192
Figure 7.35 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Falling Edge 193
Figure 7.36 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level 194
Figure 7.37 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1 195
Figure 7.38 Interrupt and Interrupt Sources 202
Figure 7.39 Procedure Example of Resuming Transfer by Clearing Interrupt Source 202

Section 8 I/O Ports
Figure 8.1 Port Block Diagram 211

Section 9 16-Bit Timer Pulse Unit (TPU)
Figure 9.1 Block Diagram of TPU (Unit 0) 256
Figure 9.2 Block Diagram of TPU (Unit 1) 257
Figure 9.3 Example of Counter Operation Setting Procedure 299
Figure 9.4 Free-Running Counter Operation 300
Figure 9.5 Periodic Counter Operation 301
Figure 9.6 Example of Setting Procedure for Waveform Output by Compare Match 301
Figure 9.7 Example of 0-Output/1-Output Operation 302
Figure 9.8 Example of Toggle Output Operation 302
Figure 9.9 Example of Setting Procedure for Input Capture Operation 303
Figure 9.10 Example of Input Capture Operation 304
Figure 9.11 Example of Synchronous Operation Setting Procedure 305
Figure 9.12 Example of Synchronous Operation 306
Figure 9.13 Compare Match Buffer Operation 307
Figure 9.14 Input Capture Buffer Operation 308
Figure 9.15 Example of Buffer Operation Setting Procedure 308
Figure 9.16 Example of Buffer Operation (1) 309
Figure 9.17 Example of Buffer Operation (2) 310
Figure 9.18 Example of Cascaded Operation Setting Procedure 311

Figure 9.19 Example of Cascaded Operation (1) 312
Figure 9.20 Example of Cascaded Operation (2) 312
Figure 9.21 Example of PWM Mode Setting Procedure 315
Figure 9.22 Example of PWM Mode Operation (1) 316
Figure 9.23 Example of PWM Mode Operation (2) 316
Figure 9.24 Example of PWM Mode Operation (3) 317
Figure 9.25 Example of Phase Counting Mode Setting Procedure 319
Figure 9.26 Example of Phase Counting Mode 1 Operation 320
Figure 9.27 Example of Phase Counting Mode 2 Operation 321
Figure 9.28 Example of Phase Counting Mode 3 Operation 322
Figure 9.29 Example of Phase Counting Mode 4 Operation 323
Figure 9.30 Phase Counting Mode Application Example 325
Figure 9.31 Count Timing in Internal Clock Operation 330
Figure 9.32 Count Timing in External Clock Operation 330
Figure 9.33 Output Compare Output Timing 331
Figure 9.34 Input Capture Input Signal Timing 331
Figure 9.35 Counter Clear Timing (Compare Match) 332
Figure 9.36 Counter Clear Timing (Input Capture) 332
Figure 9.37 Buffer Operation Timing (Compare Match) 333
Figure 9.38 Buffer Operation Timing (Input Capture) 333
Figure 9.39 TGI Interrupt Timing (Compare Match) 334
Figure 9.40 TGI Interrupt Timing (Input Capture) 334
Figure 9.41 TCIV Interrupt Setting Timing 335
Figure 9.42 TCIU Interrupt Setting Timing 335
Figure 9.43 Timing for Status Flag Clearing by CPU 336
Figure 9.44 Timing for Status Flag Clearing by DMAC Activation (1) 336
Figure 9.45 Timing for Status Flag Clearing by DMAC Activation (2) 337
Figure 9.46 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode 337
Figure 9.47 Conflict between TCNT Write and Clear Operations 338
Figure 9.48 Conflict between TCNT Write and Increment Operations 339
Figure 9.49 Conflict between TGR Write and Compare Match 339
Figure 9.50 Conflict between Buffer Register Write and Compare Match 340
Figure 9.51 Conflict between TGR Read and Input Capture 340
Figure 9.52 Conflict between TGR Write and Input Capture 341
Figure 9.53 Conflict between Buffer Register Write and Input Capture 341
Figure 9.54 Conflict between Overflow and Counter Clearing 342
Figure 9.55 Conflict between TCNT Write and Overflow 342

Section 10 Programmable Pulse Generator (PPG)

Figure 10.1 Block Diagram of PPG 345
Figure 10.2 Schematic Diagram of PPG 356

Figure 10.3 Timing of Transfer and Output of NDR Contents (Example) 356
Figure 10.4 Setup Procedure for Normal Pulse Output (Example) 357
Figure 10.5 Normal Pulse Output Example (5-Phase Pulse Output) 358
Figure 10.6 Non-Overlapping Pulse Output 359
Figure 10.7 Non-Overlapping Operation and NDR Write Timing 360
Figure 10.8 Setup Procedure for Non-Overlapping Pulse Output (Example) 361
Figure 10.9 Non-Overlapping Pulse Output Example (4-Phase Complementary) 362
Figure 10.10 Inverted Pulse Output (Example) 364
Figure 10.11 Pulse Output Triggered by Input Capture (Example) 365

Section 11 Watchdog Timer (WDT)
Figure 11.1 Block Diagram of WDT 367
Figure 11.2 Operation in Watchdog Timer Mode 372
Figure 11.3 Operation in Interval Timer Mode 373
Figure 11.4 Writing to TCNT, TCSR, and RSTCSR 374
Figure 11.5 Conflict between TCNT Write and Increment 375

Section 12 Serial Communication Interface (SCI)
Figure 12.1 Block Diagram of SCI 378
Figure 12.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits) 406
Figure 12.3 Receive Data Sampling Timing in Asynchronous Mode 408
Figure 12.4 Phase Relation between Output Clock and Transmit Data (Asynchronous Mode) 409
Figure 12.5 Sample SCI Initialization Flowchart 410
Figure 12.6 Example of Operation for Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit) 411
Figure 12.7 Sample Serial Transmission Flowchart 412
Figure 12.8 Example of SCI Operation for Reception (Example with 8-Bit Data, Parity, One Stop Bit) 413
Figure 12.9 Sample Serial Reception Flowchart (1) 415
Figure 12.9 Sample Serial Reception Flowchart (2) 416
Figure 12.10 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A) 418
Figure 12.11 Sample Multiprocessor Serial Transmission Flowchart 419
Figure 12.12 Example of SCI Operation for Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit) 420
Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (1) 421
Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (2) 422
Figure 12.14 Data Format in Clocked Synchronous Communication (LSB-First) 423
Figure 12.15 Sample SCI Initialization Flowchart 424

Figure 12.16 Example of Operation for Transmission in Clocked Synchronous Mode 426
Figure 12.17 Sample Serial Transmission Flowchart 426
Figure 12.18 Example of Operation for Reception in Clocked Synchronous Mode 427
Figure 12.19 Sample Serial Reception Flowchart 428
Figure 12.20 Sample Flowchart of Simultaneous Serial Transmission and Reception 429
Figure 12.21 Pin Connection for Smart Card Interface 430
Figure 12.22 Data Formats in Normal Smart Card Interface Mode 431
Figure 12.23 Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0) 431
Figure 12.24 Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1) 432
Figure 12.25 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate) 433
Figure 12.26 Data Re-Transfer Operation in SCI Transmission Mode 436
Figure 12.27 TEND Flag Set Timing during Transmission 436
Figure 12.28 Sample Transmission Flowchart 437
Figure 12.29 Data Re-Transfer Operation in SCI Reception Mode 438
Figure 12.30 Sample Reception Flowchart 439
Figure 12.31 Clock Output Fixing Timing 439
Figure 12.32 Clock Stop and Restart Procedure 440
Figure 12.33 Sample Transmission using DMAC in Clocked Synchronous Mode 444
Figure 12.34 Sample Flowchart for Mode Transition during Transmission 446
Figure 12.35 Port Pin States during Mode Transition (Internal Clock, Asynchronous Transmission) 447
Figure 12.36 Port Pin States during Mode Transition (Internal Clock, Clocked Synchronous Transmission) 447
Figure 12.37 Sample Flowchart for Mode Transition during Reception 448

Section 13 Controller Area Network (RCAN-ET)

Figure 13.1 RCAN-ET Architecture 451
Figure 13.2 RCAN-ET Memory Map 454
Figure 13.3 Mailbox-N Structure 456
Figure 13.4 Acceptance Filter 461
Figure 13.5 ID Reorder 464
Figure 13.6 Connection Diagram of Port Interface 497
Figure 13.7 Reset Sequence 498
Figure 13.8 Halt Mode/Sleep Mode 500
Figure 13.9 Halt Mode/Sleep Mode 501
Figure 13.10 Transmission Request 504
Figure 13.11 Internal Arbitration for transmission 505
Figure 13.12 Message receive sequence 506
Figure 13.13 Change ID of Receive Box or Change Receive Box to Transmit Box 509
Figure 13.14 DMAC Transfer Flowchart 511

Figure 13.15 High-Speed Interface Using HA13721 512

Section 14 Synchronous Serial Communication Unit (SSU)

Figure 14.1 Block Diagram of SSU 516
Figure 14.2 Relationship of Clock Phase, Polarity, and Data 534
Figure 14.3 Relationship between Data Input/Output Pins and the Shift Register 535
Figure 14.4 Example of Initial Settings in SSU Mode 538
Figure 14.5 Example of Transmission Operation (SSU Mode)
When 8-bit data length is selected (SSTDR0 is valid)
with CPOS = 0 and CPHS = 0 (1) 540
Figure 14.5 Example of Transmission Operation (SSU Mode)
When 16-bit data length is selected (SSTDR0 and SSTDR1 are valid)
with CPOS = 0 and CPHS = 0 (2) 540
Figure 14.5 Example of Transmission Operation (SSU Mode)
When 24-bit data length is selected (SSTDR0, SSTDR1, and SSTDR2 are valid)
with CPOS = 0 and CPHS = 0 (3) 541
Figure 14.5 Example of Transmission Operation (SSU Mode)
When 32-bit data length is selected (SSTDR0, SSTDR1, SSTDR2
and SSTDR3 are valid) with CPOS = 0 and CPHS = 0 (4) 541
Figure 14.6 Flowchart Example of Data Transmission (SSU Mode) 542
Figure 14.7 Example of Reception Operation (SSU Mode)
When 8-bit data length is selected (SSRDR0 is valid)
with CPOS = 0 and CPHS = 0 (1) 543
Figure 14.7 Example of Reception Operation (SSU Mode)
When 16-bit data length is selected (SSRDR0 and SSRDR1 are valid)
with CPOS = 0 and CPHS = 0 (2) 544
Figure 14.7 Example of Reception Operation (SSU Mode)
When 24-bit data length is selected (SSRDR0, SSRDR1, and SSRDR2 are valid)
with CPOS = 0 and CPHS = 0 (3) 544
Figure 14.7 Example of Reception Operation (SSU Mode)
When 32-bit data length is selected (SSRDR0, SSRDR1, SSRDR2
and SSRDR3 are valid) with CPOS = 0 and CPHS = 0 (4) 545
Figure 14.8 Flowchart Example of Data Reception (SSU Mode) 546
Figure 14.9 Flowchart Example of Simultaneous Transmission/Reception (SSU Mode) 548
Figure 14.10 Conflict Error Detection Timing (Before Transfer) 549
Figure 14.11 Conflict Error Detection Timing (After Transfer End) 549
Figure 14.12 Example of Initial Settings in Clock Synchronous Communication Mode 550
Figure 14.13 Example of Transmission Operation
(Clock Synchronous Communication Mode) 551
Figure 14.14 Flowchart Example of Transmission Operation
(Clock Synchronous Communication Mode) 552

Figure 14.15 Example of Reception Operation (Clock Synchronous Communication Mode)... 553
Figure 14.16 Flowchart Example of Data Reception
(Clock Synchronous Communication Mode) ... 554
Figure 14.17 Flowchart Example of Simultaneous Transmission/Reception
(Clock Synchronous Communication Mode) ... 556

Section 15 A/D Converter

Figure 15.1 Block Diagram of A/D Converter (Unit 0/AD_0) ... 560
Figure 15.2 Block Diagram of A/D Converter (Unit 1/AD_1) ... 561
Figure 15.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected) ... 569
Figure 15.4 Example of A/D Conversion
(Scan Mode, Three Channels (AN0 to AN2) Selected) ... 570
Figure 15.5 A/D Conversion Timing ... 571
Figure 15.6 External Trigger Input Timing ... 572
Figure 15.7 A/D Conversion Accuracy Definitions ... 574
Figure 15.8 A/D Conversion Accuracy Definitions ... 574
Figure 15.9 Example of Analog Input Circuit ... 575
Figure 15.10 Example of Analog Input Protection Circuit ... 577
Figure 15.11 Analog Input Pin Equivalent Circuit ... 578

Section 17 Flash Memory (0.18-mm F-ZTAT Version)

Figure 17.1 Block Diagram of Flash Memory ... 583
Figure 17.2 Mode Transition of Flash Memory ... 584
Figure 17.3 Memory MAT Configuration ... 586
Figure 17.4 Block Structure of User MAT ... 587
Figure 17.5 Procedure for Creating Procedure Program ... 588
Figure 17.6 System Configuration in Boot Mode ... 613
Figure 17.7 Automatic-Bit-Rate Adjustment Operation ... 614
Figure 17.8 Boot Mode State Transition Diagram ... 615
Figure 17.9 Programming/Erasing Flow ... 617
Figure 17.10 RAM Map when Programming/Erasing is Executed ... 618
Figure 17.11 Programming Procedure in User Program Mode ... 619
Figure 17.12 Erasing Procedure in User Program Mode ... 624
Figure 17.13 Repeating Procedure of Erasing, Programming,
and RAM Emulation in User Program Mode ... 626
Figure 17.14 Procedure for Programming User MAT in User Boot Mode ... 628
Figure 17.15 Procedure for Erasing User MAT in User Boot Mode ... 630
Figure 17.16 Transitions to Error Protection State ... 639
Figure 17.17 RAM Emulation Flow ... 640
Figure 17.18 Address Map of Overlaid RAM Area ... 641
Figure 17.19 Programming Tuned Data ... 642

Figure 17.20 Switching between User MAT and User Boot MAT 643
Figure 17.21 Boot Program States 645
Figure 17.22 Bit-Rate-Adjustment Sequence 646
Figure 17.23 Communication Protocol Format 647
Figure 17.24 New Bit-Rate Selection Sequence 658
Figure 17.25 Programming Sequence 662
Figure 17.26 Erasure Sequence 663

Section 18 Clock Pulse Generator
Figure 18.1 Block Diagram of Clock Pulse Generator 677
Figure 18.2 Connection of Crystal Resonator (Example) 681
Figure 18.3 Crystal Resonator Equivalent Circuit 681
Figure 18.4 External Clock Input (Examples) 682
Figure 18.5 Clock Modification Timing 684
Figure 18.6 Note on Board Design for Oscillation Circuit 684
Figure 18.7 Connection Example of Bypass Capacitor 685

Section 19 Power-Down Modes
Figure 19.1 Mode Transitions 688
Figure 19.2 Software Standby Mode Application Example 700

Section 21 Electrical Characteristics
Figure 21.1 Output Load Circuit 760
Figure 21.2 System Bus Clock Timing 761
Figure 21.3 Oscillation Settling Timing after Software Standby Mode 762
Figure 21.4 Oscillation Settling Timing 762
Figure 21.5 External Input Clock Timing 762
Figure 21.6 Reset Input Timing 763
Figure 21.7 Interrupt Input Timing 764
Figure 21.8 I/O Port Input/Output Timing 767
Figure 21.9 Data Input Timing for Realtime Input Port 767
Figure 21.10 TPU Input/Output Timing 768
Figure 21.11 TPU Clock Input Timing 768
Figure 21.12 PPG Output Timing 768
Figure 21.13 SCK Clock Input/Output Timing 768
Figure 21.14 SCI Input/Output Timing: Clocked Synchronous Mode 769
Figure 21.15 A/D Converter External Trigger Input Timing 769
Figure 21.16 RCAN-ET Input/Output Timing 769
Figure 21.17 SSU Timing (Master, CPHS = 1) 770
Figure 21.18 SSU Timing (Master, CPHS = 0) 770
Figure 21.19 SSU Timing (Slave, CPHS = 1) 771
Figure 21.20 SSU Timing (Slave, CPHS = 0) 771

Appendix

Figure C.1 Package Dimensions (FP-100M) 777

Tables

Section 1 Overview
Table 1.1 Pin Configuration in Each Operating Mode 6
Table 1.2 Pin Functions 10

Section 2 CPU
Table 2.1 Instruction Classification 36
Table 2.2 Combinations of Instructions and Addressing Modes (1) 38
Table 2.2 Combinations of Instructions and Addressing Modes (2) 41
Table 2.3 Operation Notation 42
Table 2.4 Data Transfer Instructions 43
Table 2.5 Block Transfer Instructions 44
Table 2.6 Arithmetic Operation Instructions 45
Table 2.7 Logic Operation Instructions 47
Table 2.8 Shift Operation Instructions 48
Table 2.9 Bit Manipulation Instructions 49
Table 2.10 Branch Instructions 51
Table 2.11 System Control Instructions 52
Table 2.12 Addressing Modes 54
Table 2.13 Absolute Address Access Ranges 58
Table 2.14 Effective Address Calculation for Transfer and Operation Instructions 61
Table 2.15 Effective Address Calculation for Branch Instructions 62

Section 3 MCU Operating Modes
Table 3.1 MCU Operating Mode Settings 65
Table 3.2 Settings of Bits MSD3 to MSD0 67

Section 4 Exception Handling
Table 4.1 Exception Types and Priority 71
Table 4.2 Exception Handling Vector Table 72
Table 4.3 Calculation Method of Exception Handling Vector Table Address 74
Table 4.4 Status of CCR and EXR after Trace Exception Handling 76
Table 4.5 Bus Cycle and Address Error 77
Table 4.6 States of CCR and EXR after Address Error Exception Handling 79
Table 4.7 Interrupt Sources 79
Table 4.8 Status of CCR and EXR after Trap Instruction Exception Handling 81
Table 4.9 Status of CCR and EXR after Illegal Instruction Exception Handling 82

Section 5 Interrupt Controller
Table 5.1 Pin Configuration 86

Table 5.2 Interrupt Sources, Vector Address Offsets, and Interrupt Priority 103
Table 5.3 Interrupt Control Modes 110
Table 5.4 Interrupt Response Times 115
Table 5.5 Number of Execution States in Interrupt Handling Routine 116
Table 5.6 Interrupt Source Selection and Clear Control 117
Table 5.7 CPU Priority Control 119
Table 5.8 Example of Priority Control Function Setting and Control State 120

Section 6 Bus Controller (BSC)
Table 6.1 Synchronization Clocks and Their Corresponding Functions 128
Table 6.2 Number of Access Cycles for On-Chip Memory Spaces 129
Table 6.3 Number of Access Cycles for Registers of On-Chip Peripheral Modules 129

Section 7 DMA Controller (DMAC)
Table 7.1 Data Access Size, Valid Bits, and Settable Size 142
Table 7.2 Settings and Areas of Extended Repeat Area 156
Table 7.3 Transfer Modes 157
Table 7.4 List of On-chip module interrupts to DMAC 167
Table 7.5 Priority among DMAC Channels 181
Table 7.6 Interrupt Sources and Priority 200

Section 8 I/O Ports
Table 8.1 Port Functions 205
Table 8.2 Register Configuration in Each Port 210
Table 8.3 Input Pull-Up MOS State 214
Table 8.4 Available Output Signals and Settings in Each Port 239

Section 9 16-Bit Timer Pulse Unit (TPU)
Table 9.1 Unit Configuration for Each Product 252
Table 9.2 TPU Functions (Unit 0) 252
Table 9.3 TPU Functions (Unit 1) 254
Table 9.4 Pin Configuration 258
Table 9.5 CCLR2 to CCLR0 (Channels 0 and 3) 266
Table 9.6 CCLR2 to CCLR0 (Channels 1, 2, 4, and 5) 266
Table 9.7 Input Clock Edge Selection 267
Table 9.8 TPSC2 to TPSC0 (Channel 0) 267
Table 9.9 TPSC2 to TPSC0 (Channel 1) 267
Table 9.10 TPSC2 to TPSC0 (Channel 2) 268
Table 9.11 TPSC2 to TPSC0 (Channel 3) 268
Table 9.12 TPSC2 to TPSC0 (Channel 4) 269
Table 9.13 TPSC2 to TPSC0 (Channel 5) 269
Table 9.14 MD3 to MD0 271

Table 9.15 TIORH_0 274
Table 9.16 TIORL_0 275
Table 9.17 TIOR_1 276
Table 9.18 TIOR_2 277
Table 9.19 TIORH_3 278
Table 9.20 TIORL_3 279
Table 9.21 TIOR_4 280
Table 9.22 TIOR_5 281
Table 9.23 TIORH_0 282
Table 9.24 TIORL_0 283
Table 9.25 TIOR_1 284
Table 9.26 TIOR_2 285
Table 9.27 TIORH_3 286
Table 9.28 TIORL_3 287
Table 9.29 TIOR_4 288
Table 9.30 TIOR_5 289
Table 9.31 Register Combinations in Buffer Operation 307
Table 9.32 Cascaded Combinations 311
Table 9.33 PWM Output Registers and Output Pins 314
Table 9.34 Clock Input Pins in Phase Counting Mode 318
Table 9.35 Up/Down-Count Conditions in Phase Counting Mode 1 320
Table 9.36 Up/Down-Count Conditions in Phase Counting Mode 2 321
Table 9.37 Up/Down-Count Conditions in Phase Counting Mode 3 322
Table 9.38 Up/Down-Count Conditions in Phase Counting Mode 4 323
Table 9.39 TPU Interrupts 326

Section 10 Programmable Pulse Generator (PPG)
Table 10.1 Pin Configuration 346

Section 11 Watchdog Timer (WDT)
Table 11.1 WDT Interrupt Source 373

Section 12 Serial Communication Interface (SCI)
Table 12.1 Pin Configuration 379
Table 12.2 Relationships between N Setting in BRR and Bit Rate B 399
Table 12.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1) 400
Table 12.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2) 401
Table 12.4 Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode) 402
Table 12.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode) 402
Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode) 403
Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode) 404

Table 12.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, S = 372) 405
Table 12.9 Maximum Bit Rate for Each Operating Frequency (Smart Card Interface Mode, S = 372) 405
Table 12.10 Serial Transfer Formats (Asynchronous Mode) 407
Table 12.11 SSR Status Flags and Receive Data Handling 414
Table 12.12 SCI Interrupt Sources 441
Table 12.13 SCI Interrupt Sources 442

Section 13 Synchronous Serial Communication Unit (SSU)
Table 13.1 Address map for Each Meailbox 455
Table 13.2 Roles of Mailboxes 456
Table 13.3 Mailbox Function Setting 460
Table 13.4 RCAN-ET Control Registers Configuration 463
Table 13.5 TSG and TSEG Setting 475
Table 13.6 RCAN-ET Mailbox Registers 485
Table 13.7 Conditions to access registers 502
Table.13.8 Test Mode Settings 502
Table 13.9 RCAN-ET Interrupt Sources 510

Section 14 Synchronous Serial Communication Unit (SSU)
Table 14.1 Pin Configuration 517
Table 14.2 Correspondence Between DATS Bit Setting and SSTDR 531
Table 14.3 Correspondence Between DATS Bit Setting and SSRDR 532
Table 14.4 Communication Modes and Pin States of SSI and SSO Pins 536
Table 14.5 Communication Modes and Pin States of SSCK Pin 537
Table 14.6 Communication Modes and Pin States of $\overline{\text{SCS}}$ Pin 537
Table 14.7 Interrupt Sources 557

Section 15 A/D Converter
Table 15.1 Pin Configuration 562
Table 15.2 Analog Input Channels and Corresponding ADDR Registers 564
Table 15.3 A/D Conversion Characteristics (Single Mode) 572
Table 15.4 A/D Conversion Characteristics (Scan Mode) 572
Table 15.5 A/D Converter Interrupt Source 573
Table 15.6 Analog Pin Specifications 577

Section 17 Flash Memory (0.18-mm F-ZTAT Version)
Table 17.1 Differences between Boot Mode, User Program Mode, User Boot Mode, and Programmer Mode 585
Table 17.2 Pin Configuration 590
Table 17.3 Registers/Parameters and Target Modes 592

Table 17.4 Parameters and Target Modes 599
Table 17.5 On-Board Programming Mode Setting 613
Table 17.6 System Clock Frequency for Automatic-Bit-Rate Adjustment 614
Table 17.7 Executable Memory MAT 632
Table 17.8 Usable Area for Programming in User Program Mode 633
Table 17.9 Usable Area for Erasure in User Program Mode 634
Table 17.10 Usable Area for Programming in User Boot Mode 635
Table 17.11 Usable Area for Erasure in User Boot Mode 636
Table 17.12 Hardware Protection 637
Table 17.13 Software Protection 638
Table 17.14 Device Types Supported in Programmer Mode 644
Table 17.15 Inquiry and Selection Commands 648
Table 17.16 Programming/Erasing Commands 661
Table 17.17 Status Code 671
Table 17.18 Error Code 671
Table 17.19 Initiation Intervals of User Branch Processing 673

Section 18 Clock Pulse Generator
Table 18.1 Damping Resistance Value 681
Table 18.2 Crystal Resonator Characteristics 682

Section 19 Power-Down Modes
Table 19.1 Operating States 688
Table 19.2 Oscillation Settling Time Settings 698
Table 19.3 Bϕ Pin (PA7) State in Each Processing State 701

Section 21 Electrical Characteristics
Table 21.1 Absolute Maximum Ratings 757
Table 21.2 DC Characteristics (1) 758
Table 21.2 DC Characteristics (2) 759
Table 21.3 Permissible Output Currents 760
Table 21.4 Clock Timing 761
Table 21.5 Control Signal Timing 763
Table 21.6 Timing of On-Chip Peripheral Modules (1) 764
Table 21.6 Timing of On-Chip Peripheral Modules (2) 766
Table 21.7 A/D Conversion Characteristics 772
Table 21.8 Flash Memory Characteristics 773

Appendix
Table A.1 Port States in Each Pin State 775

Section 1 Overview

1.1 Features

- 32-bit high-speed H8SX CPU
  Upward compatible with the H8/300 CPU, H8/300H CPU, and H8S CPU
  Sixteen 16-bit general registers
  87 basic instructions
- Extensive peripheral functions
  DMA controller (DMAC)
  16-bit timer pulse unit (TPU)
  Programmable pulse generator (PPG)*
  Watch dog timer (WDT)
  Serial communication interface (SCI) can be used in asynchronous and clocked synchronous mode
  Controller area network (RCAN-ET)
  Synchronous serial communication unit (SSU)
  10-bit A/D converter
  Clock pulse generator

Note: * Supported only by the H8SX/1527R.

- On-chip memory

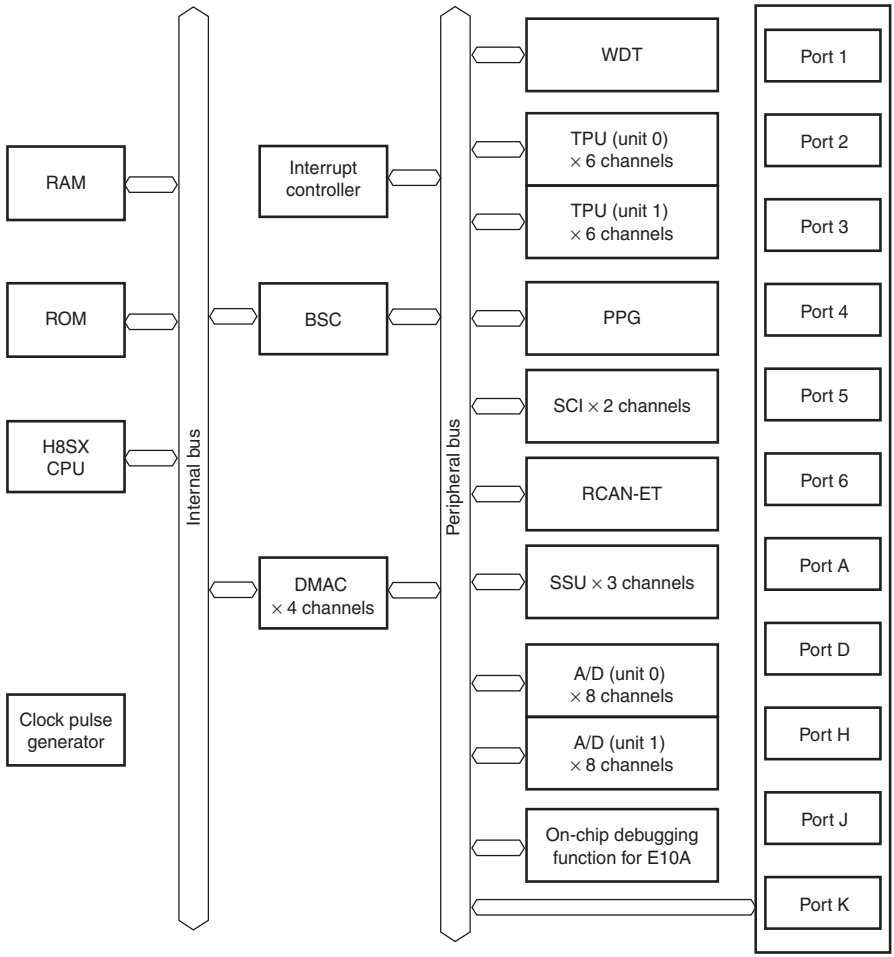
Product Classification		Product Model	ROM	RAM
Flash memory version	H8SX/1527R	R5F61527R	256 kbytes	12 kbytes
	H8SX/1525R	R5F61525R	256 kbytes	12 kbytes

- General I/O port
  65 input/output ports
  17 input ports
- Supports power-down modes

- Small package

Package	Code	Body Size	Pin Pitch
QFP-100	PRQP0100KB-A (FP-100M)	14.0 × 14.0 mm	0.50 mm

1.2 Block Diagram



[Legend]
CPU: Central processing unit
DMAC: DMA controller
BSC: Bus controller
WDT: Watchdog timer
TPU: 16-bit timer pulse unit
PPG: Programmable pulse generator
SCI: Serial communication interface
RCAN-ET: Controller area network
SSU: Synchronous communication unit

Figure 1.1 Block Diagram of H8SX/1527R

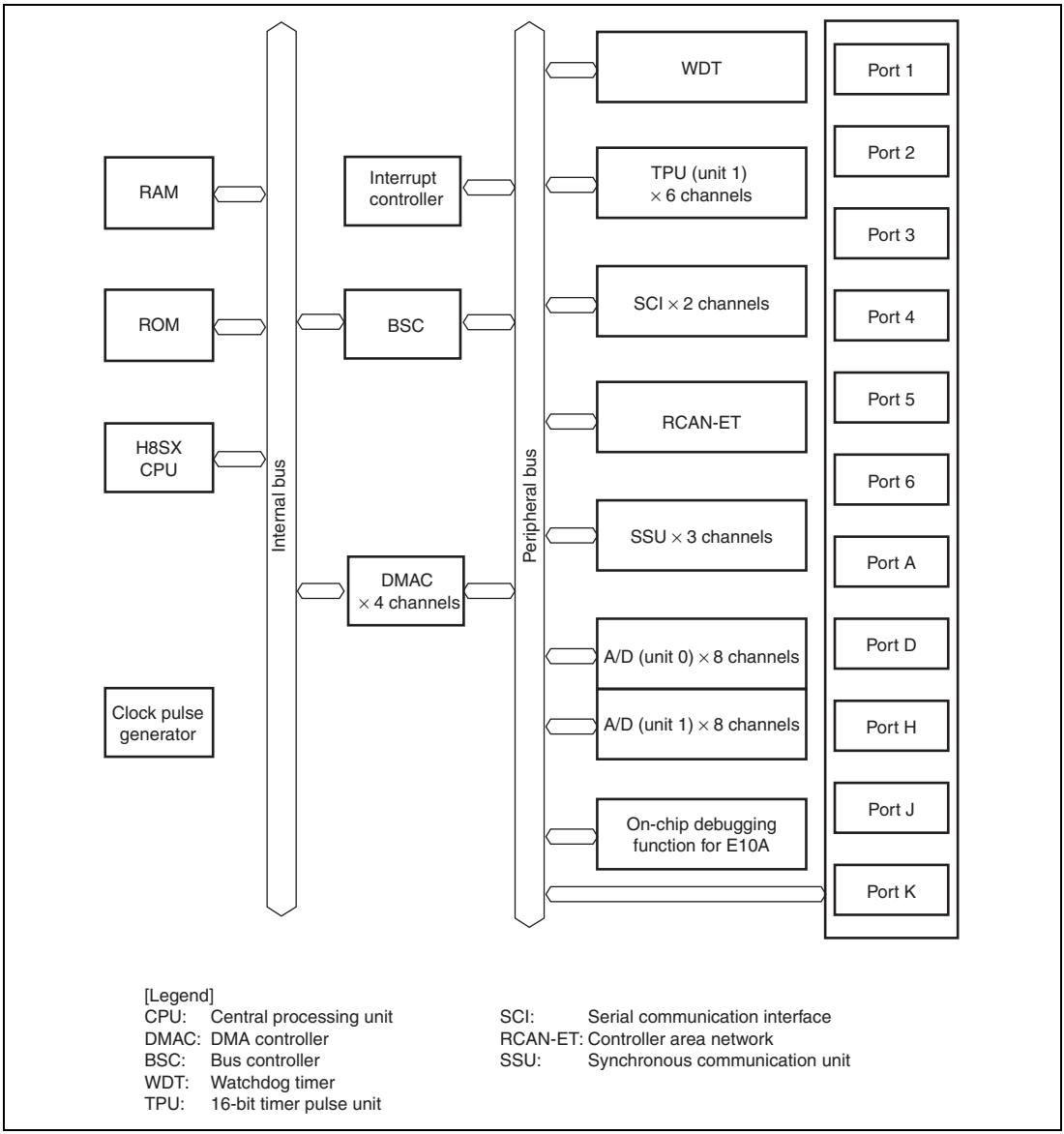
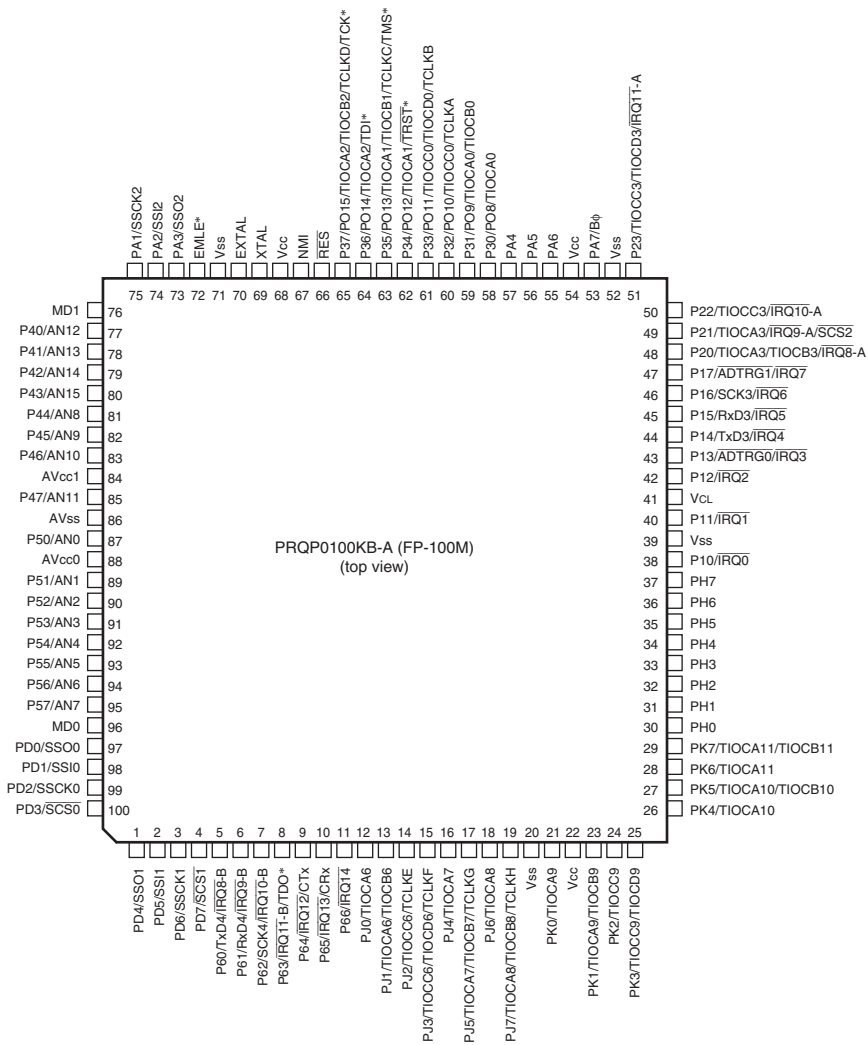


Figure 1.2 Block Diagram of H8SX/1525R

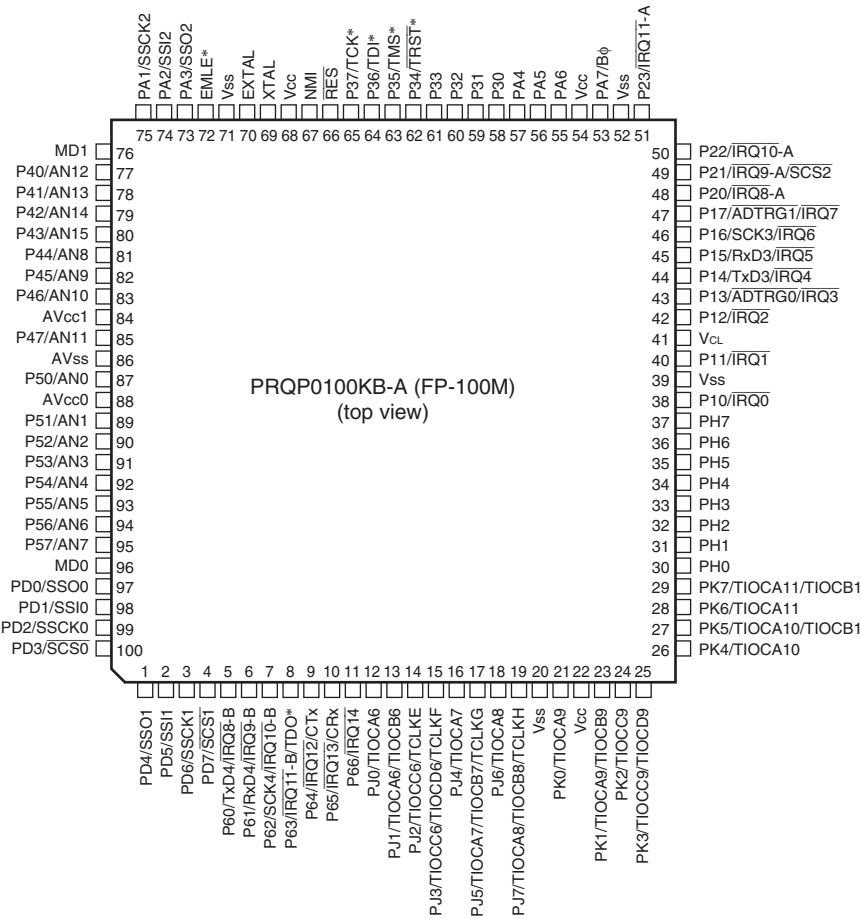
1.3 Pin Assignments

1.3.1 Pin Assignments



Note: * The EMLE (emulator enable) pin enables/disables the on-chip debugging functions. When the EMLE pin is driven high, the TDO, TDI, TCK, TMS, and TRST pins are used specific for the E10A. In this case, other pin functions are disabled.

Figure 1.3 Pin Assignments of H8SX/1527R



Note: * The EMLE (emulator enable) pin enables/disables the on-chip debugging functions. When the EMLE pin is driven high, the TDO, TDI, TCK, TMS, and TRST pins are used specific for the E10A. In this case, other pin functions are disabled.

Figure 1.4 Pin Assignments of H8SX/1525R

1.3.2 Pin Configuration in Each Operating Mode

Table 1.1 Pin Configuration in Each Operating Mode

Pin No.	Abbreviation in Mode 1, Mode 2, and Mode 3
1	PD4/SSO1
2	PD5/SSI1
3	PD6/SSCK1
4	PD7/$\overline{\text{SCS1}}$
5	P60/TxD4/$\overline{\text{IRQ8}}$-B
6	P61/RxD4/$\overline{\text{IRQ9}}$-B
7	P62/SCK4/$\overline{\text{IRQ10}}$-B
8	P63/$\overline{\text{IRQ11}}$-B/TDO*[2]
9	P64/$\overline{\text{IRQ12}}$/CTx
10	P65/$\overline{\text{IRQ13}}$/CRx
11	P66/$\overline{\text{IRQ14}}$
12	PJ0/TIOCA6
13	PJ1/TIOCA6/TIOCB6
14	PJ2/TIOCC6/TCLKE
15	PJ3/TIOCC6/TIOCD6/TCLKF
16	PJ4/TIOCA7
17	PJ5/TIOCA7/TIOCB7/TCLKG
18	PJ6/TIOCA8
19	PJ7/TIOCA8/TIOCB8/TCLKH
20	Vss
21	PK0/TIOCA9
22	Vcc
23	PK1/TIOCA9/TIOCB9
24	PK2/TIOCC9
25	PK3/TIOCC9/TIOCD9
26	PK4/TIOCA10
27	PK5/TIOCA10/TIOCB10
28	PK6/TIOCA11
29	PK7/TIOCA11/TIOCB11
30	PH0

Pin No.	Abbreviation in Mode 1, Mode 2, and Mode 3
31	PH1
32	PH2
33	PH3
34	PH4
35	PH5
36	PH6
37	PH7
38	P10/$\overline{\text{IRQ0}}$
39	Vss
40	P11/$\overline{\text{IRQ1}}$
41	VCL
42	P12/$\overline{\text{IRQ2}}$
43	P13/$\overline{\text{ADTRG0}}$/$\overline{\text{IRQ3}}$
44	P14/TxD3/$\overline{\text{IRQ4}}$
45	P15/RxD3/$\overline{\text{IRQ5}}$
46	P16/SCK3/$\overline{\text{IRQ6}}$
47	P17/$\overline{\text{ADTRG1}}$/$\overline{\text{IRQ7}}$
48	P20/(TIOCA3/TIOCB3)*[1]/$\overline{\text{IRQ8}}$-A
49	P21/(TIOCA3)*[1]/$\overline{\text{IRQ9}}$-A/SCS2
50	P22/(TIOCC3)*[1]/$\overline{\text{IRQ10}}$-A
51	P23/(TIOCC3/TIOCD3)*[1]/$\overline{\text{IRQ11}}$-A
52	Vss
53	PA7/Bϕ
54	Vcc
55	PA6
56	PA5
57	PA4
58	P30/(PO8/TIOCA0)*[1]
59	P31/(PO9/TIOCA0/TIOCB0)*[1]
60	P32/(PO10/TIOCC0/TCLKA)*[1]

Pin No.	Abbreviation in Mode 1, Mode 2, and Mode 3
61	P33/(PO11/TIOCC0/TIOCD0/TCLKB)*[1]
62	P34/(PO12/TIOCA1)*[1]/$\overline{\text{TRST}}$*[2]
63	P35/(PO13/TIOCA1/TIOCB1/TCLKC)*[1]/TMS*[2]
64	P36/(PO14/TIOCA2)*[1]/TDI*[2]
65	P37/(PO15/TIOCA2/TIOCB2/TCLKD)*[1]/TCK*[2]
66	$\overline{\text{RES}}$
67	NMI
68	Vcc
69	XTAL
70	EXTAL
71	Vss
72	EMLE*[2]
73	PA3/SSO2
74	PA2/SSI2
75	PA1/SSCK2
76	MD1
77	P40/AN12
78	P41/AN13
79	P42/AN14
80	P43/AN15
81	P44/AN8
82	P45/AN9
83	P46/AN10
84	AVcc1
85	P47/AN11
86	AVss
87	P50/AN0
88	AVcc0
89	P51/AN1
90	P52/AN2

Pin No.	Abbreviation in Mode 1, Mode 2, and Mode 3
91	P53/AN3
92	P54/AN4
93	P55/AN5
94	P56/AN6
95	P57/AN7
96	MD0
97	PD0/SSO0
98	PD1/SSI0
99	PD2/SSCK0
100	PD3/$\overline{\text{SCS0}}$

Notes: 1. Not supported by the H8SX/1525R.

2. The EMLE (emulator enable) pin enables/disables the on-chip debugging functions. When the EMLE pin is driven high, the TDO, TDI, TCK, TMS, and TRST pins are used specific for the E10A. In this case, other pin functions are disabled.

1.3.3 Pin Functions

Table 1.2 Pin Functions

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
Power supply	V_{CC}	22, 54, 68	22, 54, 68	Input	Power supply pins. Connect to the system power supply.
	V_{CL}	41	41	Input	Connect to VSS via a 0.1-uF capacitor (place the capacitor close to this pin).
	V_{SS}	20, 39, 52, 71	20, 39, 52, 71	Input	Ground pins. Connect to the system power supply (0 V).
Clock	XTAL	69	69	Input	Pins for a crystal resonator. External clock can be input to the EXTAL pin. For a connection example, see section 18, Clock Pulse Generator.
	EXTAL	70	70	Input	
	Bϕ	53	53	Output	Supplies the system clock to external devices.
Operating mode control	MD1	76	76	Input	Pins for setting the operating mode. The signal levels of these pins must not be changed during operation.
	MD0	96	96		
System control	$\overline{RES}$	66	66	Input	Reset signal input pin. This LSI enters the reset state when this signal goes low.
	EMLE	72	72	Input	Input pin for on-chip emulator enable signal. Normally the signal level should be fixed low.

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
Interrupts	NMI	67	67	Input	Non-maskable interrupt request signal. When this pin is not in use, this signal must be fixed high.
	$\overline{\text{IRQ14}}$	11	11	Input	Maskable interrupt request signal.
	$\overline{\text{IRQ13}}$	10	10		
	$\overline{\text{IRQ12}}$	9	9		
	$\overline{\text{IRQ11}}$-A/$\overline{\text{IRQ11}}$-B	51/8	51/8		
	$\overline{\text{IRQ10}}$-A/$\overline{\text{IRQ10}}$-B	50/7	50/7		
	$\overline{\text{IRQ9}}$-A/$\overline{\text{IRQ9}}$-B	49/6	49/6		
	$\overline{\text{IRQ8}}$-A/$\overline{\text{IRQ8}}$-B	48/5	48/5		
	$\overline{\text{IRQ7}}$	47	47		
	$\overline{\text{IRQ6}}$	46	46		
	$\overline{\text{IRQ5}}$	45	45		
	$\overline{\text{IRQ4}}$	44	44		
	$\overline{\text{IRQ3}}$	43	43		
	$\overline{\text{IRQ2}}$	42	42		
	$\overline{\text{IRQ1}}$	40	40		
	$\overline{\text{IRQ0}}$	38	38		
Debugging interface	$\overline{\text{TRST}}$	62	62	Input	Interface pins for debugging by the on-chip emulator.
	TMS	63	63	Input	
	TDO	8	8	Output	
	TDI	64	64	Input	
	TCK	65	65	Input	
16-bit timer pulse unit (TPU) (unit 0)*	TCLKA	60	—	Input	Input pins for the external clocks.
	TCLKB	61	—		
	TCLKC	63	—		
	TCLKD	65	—		
	TIOCA0	58, 59	—	I/O	Signals for TGRA_0 to TGRD_0. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB0	59	—		
	TIOCC0	60, 61	—		
	TIOCD0	61	—		
	TIOCA1	62, 63	—	I/O	Signals for TGRA_1 and TGRB_1. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB1	63	—		

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
16-bit timer pulse unit (TPU) (unit 0)*	TIOCA2	64, 65	—	I/O	Signals for TGRA_2 and TGRB_2. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB2	65	—		
	TIOCA3	48, 49	—	I/O	Signals for TGRA_3 toTGRD_3. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB3	48	—		
	TIOCC3	50, 51	—		
	TIOCD3	51	—		
16-bit timer pulse unit (TPU) (unit 1)	TCLKE	14	14	Input	Input pins for the external clocks.
	TCLKF	15	15		
	TCLKG	17	17		
	TCLKH	19	18		
	TIOCA6	12, 13	12, 13	I/O	Signals for TGRA_6 toTGRD_6. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB6	13	13		
	TIOCC6	14, 15	14, 15		
	TIOCD6	15	14		
	TIOCA7	16, 17	16, 17	I/O	Signals for TGRA_7 toTGRB_7. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB7	17	17		
	TIOCA8	18, 19	18, 19	I/O	Signals for TGRA_8 toTGRB_8. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB8	19	19		
	TIOCA9	21, 23	21, 23	I/O	Signals for TGRA_9 toTGRD_9. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB9	23	23		
	TIOCC9	24, 25	24, 25		
	TIOCD9	25	25		
	TIOCA10	26, 27	26, 27	I/O	Signals for TGRA_10 toTGRB_10. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB10	27	27		
	TIOCA11	28, 29	28, 29	I/O	Signals for TGRA_11 toTGRB_11. These are used for the input capture inputs/output compare outputs/PWM outputs.
	TIOCB11	29	29		

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
Programmable pulse generator (PPG)*	PO15	65	—	Output	Output pins for the pulse signals.
	PO14	64	—		
	PO13	63	—		
	PO12	62	—		
	PO11	61	—		
	PO10	60	—		
	PO9	59	—		
	PO8	58	—		
Serial communication interface (SCI)	TxD3	44	44	Output	Output pins for transmit data.
	TxD4	5	5		
	RxD3	45	45	Input	Input pins for receive data.
	RxD4	6	6		
	SCK3	46	46	I/O	Input/output pins for clock signals.
	SCK4	7	7		
Controller area network (RCAN-ET)	CTx	9	9	Output	Output pin for CAN bus transmission.
	CRx	10	10	Input	Input pin for CAN bus reception.
Synchronous serial communication unit (SSU)	$\overline{\text{SSO2}}$	73	73	I/O	Input/output pins for data.
	$\overline{\text{SSO1}}$	1	1		
	$\overline{\text{SSO0}}$	97	97		
	SSI2	74	74	I/O	Input/output pins for data.
	SSI1	2	2		
	SSI0	98	98		
	SSCK2	75	75	I/O	Input/output pins for clock.
	SSCK1	3	3		
	SSCK0	99	99		
	SCS2	49	49	I/O	Input/output pins for chip select.
	SCS1	4	4		
	SCS0	100	100		

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
A/D converter	AN15	80	80	Input	Input pins for the analog signals for the A/D converter.
	AN14	79	79		
	AN13	78	78		
	AN12	77	77		
	AN11	85	85		
	AN10	83	83		
	AN9	82	82		
	AN8	81	81		
	AN7	95	95		
	AN6	94	94		
	AN5	93	93		
	AN4	92	92		
	AN3	91	91		
	AN2	90	90		
	AN1	89	89		
	AN0	87	87		
	$\overline{\text{ADTRG0}}$	43	43	Input	Input pins for the external trigger signal to start A/D conversion.
	$\overline{\text{ADTRG1}}$	47	47		
	$AV_{CC}0$	88	88	Input	Analog power supply and reference power supply pins for the A/D converter. When the A/D converter is not in use, connect to the system power supply.
	$AV_{CC}1$	84	84		
	AV_{SS}	86	86	Input	Ground pin for the A/D and D/A converters. Connect to the system power supply (0 V).

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
I/O port	P17	47	47	I/O	8-bit input/output pins.
	P16	46	46		
	P15	45	45		
	P14	44	44		
	P13	43	43		
	P12	42	42		
	P11	40	40		
	P10	38	38		
	P23	51	51	I/O	4-bit input/output pins.
	P22	50	50		
	P21	49	49		
	P20	48	48		
	P37	65	65	I/O	8-bit input/output pins.
	P36	64	64		
	P35	63	63		
	P34	62	62		
	P33	61	61		
	P32	60	60		
	P31	59	59		
	P30	58	58		
	P47	85	85	Input	8-bit input pins.
	P46	83	83		
	P45	82	82		
	P44	81	81		
	P43	80	80		
	P42	79	79		
	P41	78	78		
	P40	77	77		

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
I/O port	P57	95	95	Input	8-bit input pins.
	P56	94	94		
	P55	93	93		
	P54	92	92		
	P53	91	91		
	P52	90	90		
	P51	89	89		
	P50	87	87		
	P66	11	11	I/O	7-bit input/output pins.
	P65	10	10		
	P64	9	9		
	P63	8	8		
	P62	7	7		
	P61	6	6		
	P60	5	5		
	PA7	53	53	Input	1-bit input pin.
	PA6	55	55	I/O	6-bit input/output pins.
	PA5	56	56		
	PA4	57	57		
	PA3	73	73		
	PA2	74	74		
	PA1	75	75		
	PD7	4	4	I/O	8-bit input/output pins.
	PD6	3	3		
	PD5	2	2		
	PD4	1	1		
	PD3	100	100		
	PD2	99	99		
	PD1	98	98		
	PD0	97	97		

Classification	Abbreviation	Pin Number H8SX/1527R	Pin Number H8SX/1525R	I/O	Description
I/O port	PH7	37	37	I/O	8-bit input/output pins.
	PH6	36	36		
	PH5	35	35		
	PH4	34	34		
	PH3	33	33		
	PH2	32	32		
	PH1	31	31		
	PH0	30	30		
	PJ7	19	19	I/O	8-bit input/output pins.
	PJ6	18	18		
	PJ5	17	17		
	PJ4	16	16		
	PJ3	15	15		
	PJ2	14	14		
	PJ1	13	13		
	PJ0	12	12		
	PK7	29	29	I/O	8-bit input/output pins.
	PK6	28	28		
	PK5	27	27		
	PK4	26	26		
	PK3	25	25		
	PK2	24	24		
	PK1	23	23		
	PK0	21	21		

Note: * Supported only by the H8SX/1527R.

Section 2 CPU

The H8SX CPU is a high-speed CPU with an internal 32-bit architecture that is upward-compatible with the H8/300, H8/300H, and H8S CPUs.

The H8SX CPU has sixteen 16-bit general registers, can handle a 4-Gbyte linear address space, and is ideal for a realtime control system.

2.1 Features

- Upward-compatible with H8/300, H8/300H, and H8S CPUs
  - Can execute H8/300, H8/300H, and H8S/2000 object programs
- Sixteen 16-bit general registers
  - Also usable as sixteen 8-bit registers or eight 32-bit registers
- 87 basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Bit field transfer instructions
  - Powerful bit-manipulation instructions
  - Bit condition branch instructions
  - Multiply-and-accumulate instruction
- Eleven addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:2,ERn), @(d:16,ERn), or @(d:32,ERn)]
  - Index register indirect with displacement [@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)]
  - Register indirect with pre-/post-increment or pre-/post-decrement [@+ERn, @ERn+, @−ERn, or @ERn−]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:3, #xx:4, #xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Program-counter relative with index register [@(RnL.B,PC), @(Rn.W,PC), or @(ERn.L,PC)]
  - Memory indirect [@@aa:8]
  - Extended memory indirect [@@vec:7]

- Two base registers
  - Vector base register
  - Short address base register
- 4-Gbyte address space
  - Program: 4 Gbytes
  - Data: 4 Gbytes
- High-speed operation
  - All frequently-used instructions executed in one or two states
  - 8/16/32-bit register-register add/subtract: 1 state
  - 8 × 8-bit register-register multiply: 1 state
  - 16 ÷ 8-bit register-register divide: 10 states
  - 16 × 16-bit register-register multiply: 1 state
  - 32 ÷ 16-bit register-register divide: 18 states
  - 32 × 32-bit register-register multiply: 5 states
  - 32 ÷ 32-bit register-register divide: 18 states
- Four CPU operating modes
  - Normal mode
  - Middle mode
  - Advanced mode
  - Maximum mode
- Power-down modes
  - Transition is made by execution of SLEEP instruction
  - Choice of CPU operating clocks

Notes: 1. Advanced mode is only supported as the CPU operating mode of the H8SX/1520R Group. Normal, middle, and maximum modes are not supported.
2. The multiplier and divider are supported by the H8SX/1520R Group.
3. In the H8SX/1520R Group, an instruction is fetched in 32-bit mode.

2.2 CPU Operating Modes

The H8SX CPU has four operating modes: normal, middle, advanced and maximum modes. For details on mode settings, see section 3.1, Operating Mode Selection.

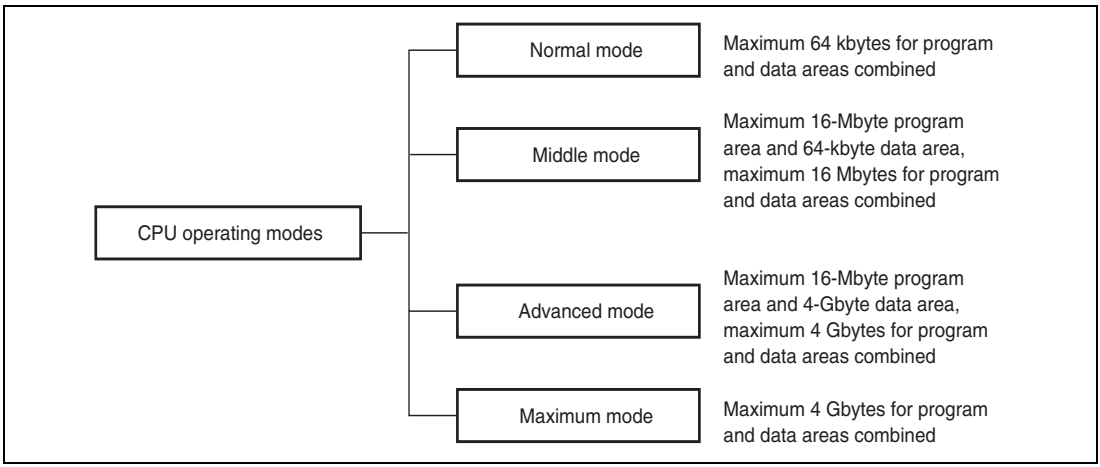


Figure 2.1 CPU Operating Modes

2.2.1 Normal Mode

The exception vector table and stack have the same structure as in the H8/300 CPU.

Note: Normal mode is not supported in this LSI.

- Address Space
  The maximum address space of 64 kbytes can be accessed.
- Extended Registers (En)
  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register it can contain any value, even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)
- Instruction Set
  All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid.

- Exception Vector Table and Memory Indirect Branch Addresses

  In normal mode, the top area starting at H'0000 is allocated to the exception vector table. One branch address is stored per 16 bits. The structure of the exception vector table is shown in figure 2.2.

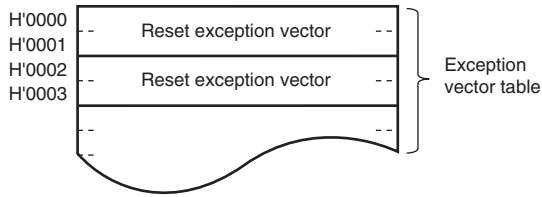
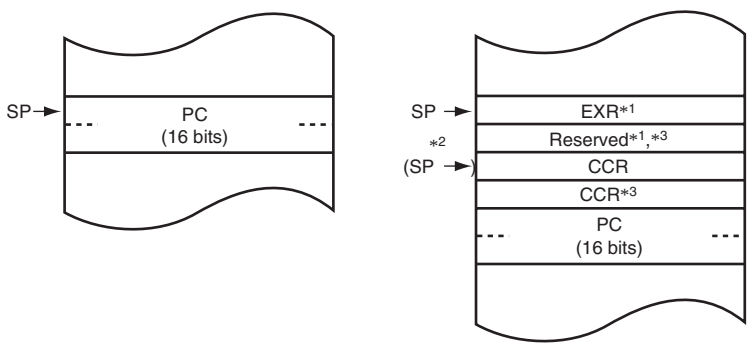


Figure 2.2 Exception Vector Table (Normal Mode)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.3. The PC contents are saved or restored in 16-bit units.



(a) Subroutine Branch (b) Exception Handling

Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored on return.

Figure 2.3 Stack Structure (Normal Mode)

2.2.2 Middle Mode

The program area in middle mode is extended to 16 Mbytes as compared with that in normal mode.

- Address Space

  The maximum address space of 16 Mbytes can be accessed as a total of the program and data areas. For individual areas, up to 16 Mbytes of the program area or up to 64 kbytes of the data area can be allocated.

- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers. When the extended register En is used as a 16-bit register (in other than the JMP and JSR instructions), it can contain any value even when the corresponding general register Rn is used as an address register. (If the general register Rn is referenced in the register indirect addressing mode with pre-/post-increment or pre-/post-decrement and a carry or borrow occurs, however, the value in the corresponding extended register En will be affected.)

- Instruction Set

  All instructions and addressing modes can be used. Only the lower 16 bits of effective addresses (EA) are valid and the upper eight bits are sign-extended.

- Exception Vector Table and Memory Indirect Branch Addresses

  In middle mode, the top area starting at H'000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

  The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

  In middle mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.

2.2.3 Advanced Mode

The data area is extended to 4 Gbytes as compared with that in middle mode.

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed. For individual areas, up to 16 Mbytes of the program area and up to 4 Gbytes of the data area can be allocated.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers, or as the upper 16-bit segments of 32-bit registers or address registers.
- Instruction Set

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

  In advanced mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The upper eight bits are ignored and the lower 24 bits are stored. The structure of the exception vector table is shown in figure 2.4.

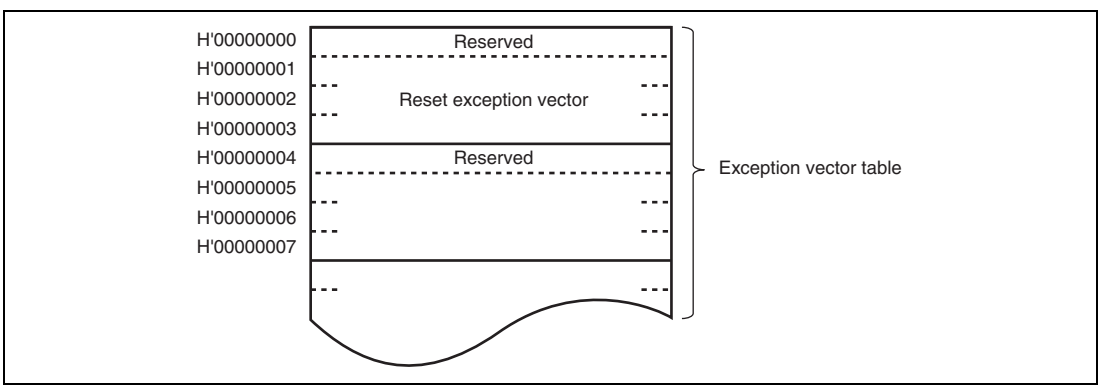


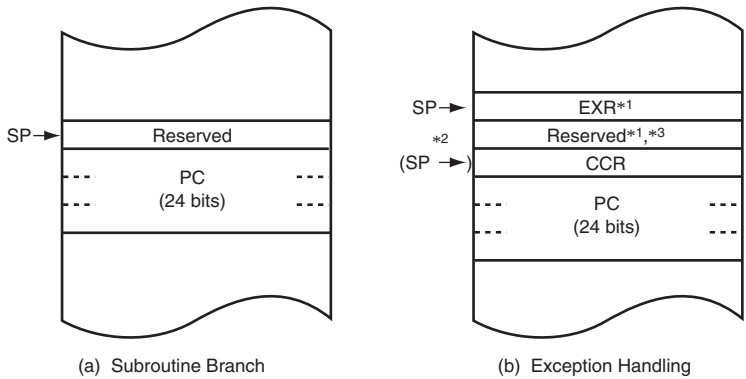
Figure 2.4 Exception Vector Table (Middle and Advanced Modes)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

In advanced mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address. The upper eight bits are reserved and assumed to be H'00.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.5. The PC contents are saved or restored in 24-bit units.



Notes: 1. When EXR is not used it is not stored on the stack.
2. SP when EXR is not used.
3. Ignored on return.

Figure 2.5 Stack Structure (Middle and Advanced Modes)

2.2.4 Maximum Mode

The program area is extended to 4 Gbytes as compared with that in advanced mode.

- Address Space

  The maximum address space of 4 Gbytes can be linearly accessed.
- Extended Registers (En)

  The extended registers (E0 to E7) can be used as 16-bit registers or as the upper 16-bit segments of 32-bit registers or address registers.
- Instruction Set

  All instructions and addressing modes can be used.
- Exception Vector Table and Memory Indirect Branch Addresses

  In maximum mode, the top area starting at H'00000000 is allocated to the exception vector table. One branch address is stored per 32 bits. The structure of the exception vector table is shown in figure 2.6.

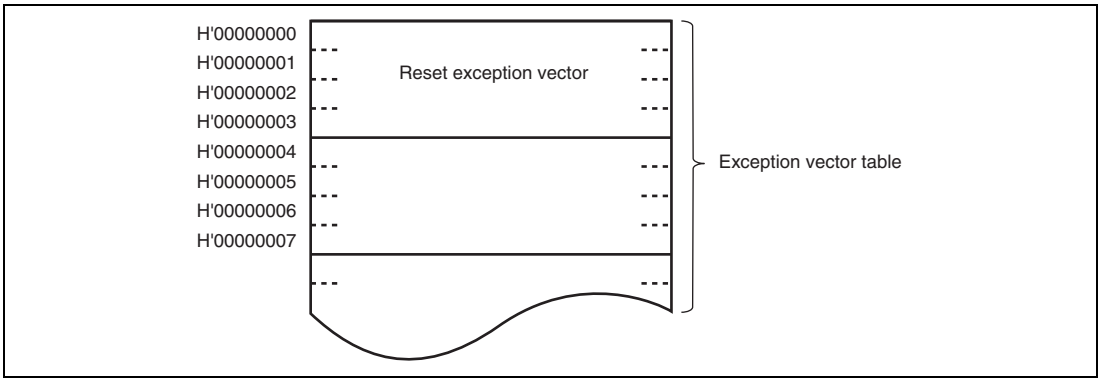


Figure 2.6 Exception Vector Table (Maximum Modes)

The memory indirect (@@aa:8) and extended memory indirect (@@vec:7) addressing modes are used in the JMP and JSR instructions. An 8-bit absolute address included in the instruction code specifies a memory location. Execution branches to the contents of the memory location.

In maximum mode, an operand is a 32-bit (longword) operand, providing a 32-bit branch address.

- Stack Structure

  The stack structure of PC at a subroutine branch and that of PC and CCR at an exception handling are shown in figure 2.7. The PC contents are saved or restored in 32-bit units. The EXR contents are saved or restored regardless of whether or not EXR is in use.

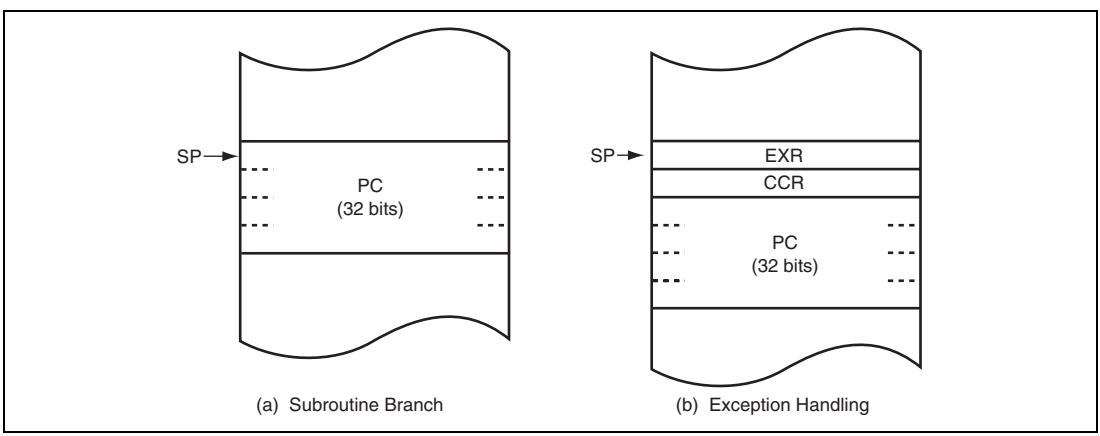


Figure 2.7 Stack Structure (Maximum Mode)

2.3 Instruction Fetch

The H8SX CPU has two modes for instruction fetch: 16-bit and 32-bit modes. It is recommended that the mode be set according to the bus width of the memory in which a program is stored. The instruction-fetch mode setting does not affect operation other than instruction fetch such as data accesses.

Note: In the H8SX/1520R Group, an instruction is fetched in 32-bit mode.

2.4 Address Space

Figure 2.8 shows a memory map of the H8SX CPU. The address space differs depending on the CPU operating mode.

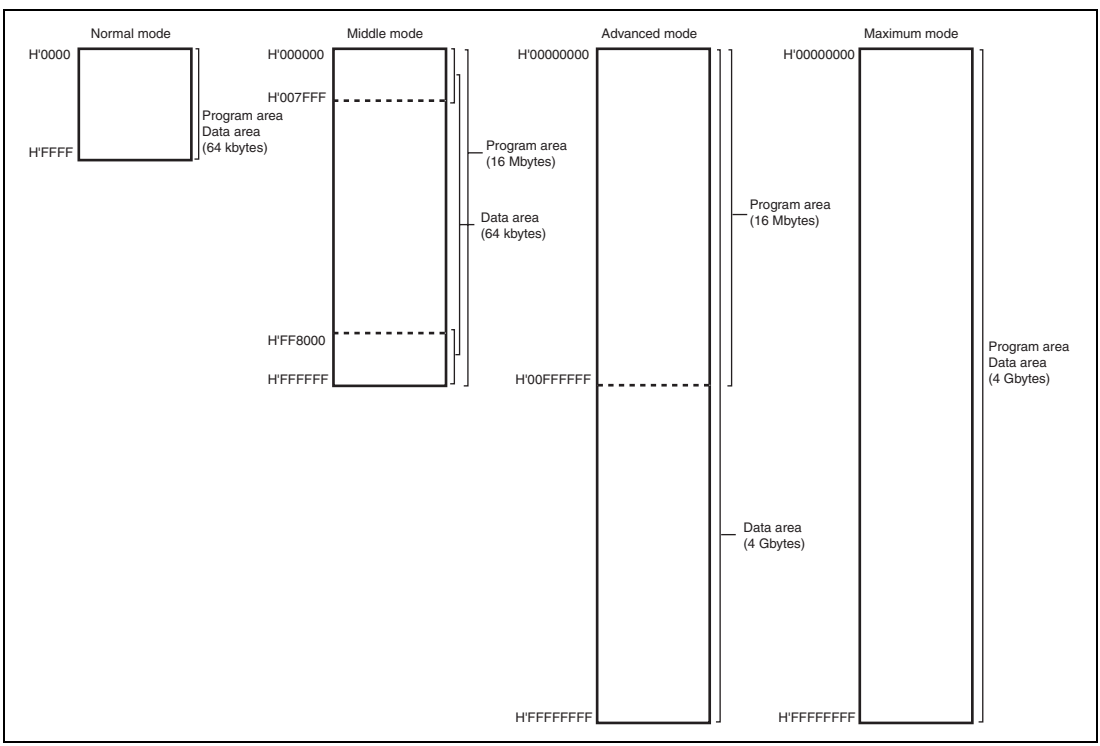


Figure 2.8 Memory Map

2.5 Registers

The H8SX CPU has the internal registers shown in figure 2.9. There are two types of registers: general registers and control registers. The control registers are the 32-bit program counter (PC), 8-bit extended control register (EXR), 8-bit condition-code register (CCR), 32-bit vector base register (VBR), 32-bit short address base register (SBR), and 64-bit multiply-accumulate register (MAC).

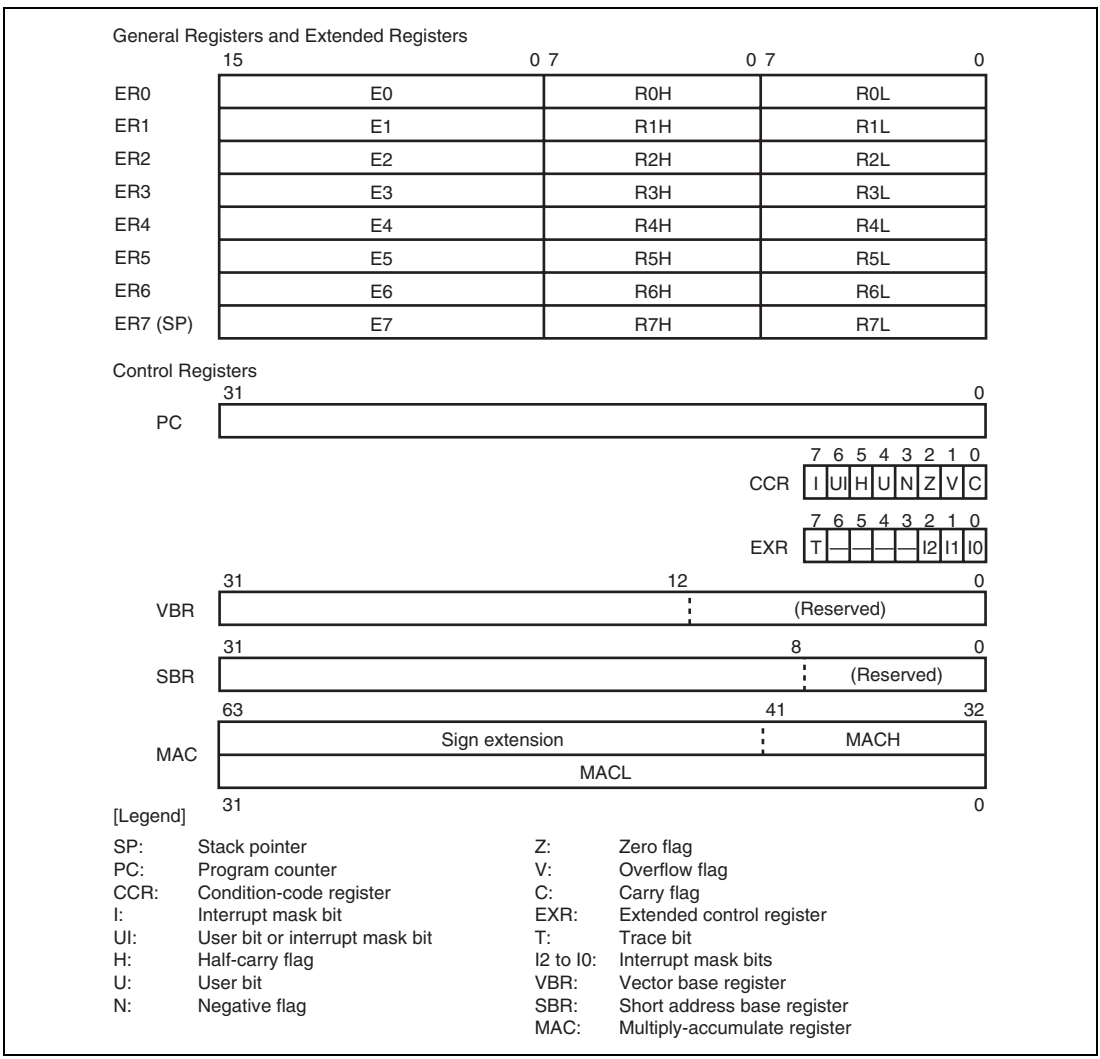


Figure 2.9 CPU Registers

2.5.1 General Registers

The H8SX CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. Figure 2.10 illustrates the usage of the general registers.

When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

When the general registers are used as 16-bit registers, the ER registers are divided into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

When the general registers are used as 8-bit registers, the R registers are divided into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

The general registers ER (ER0 to ER7), R (R0 to R7), and RL (R0L to R7L) are also used as index registers. The size in the operand field determines which register is selected.

The usage of each register can be selected independently.

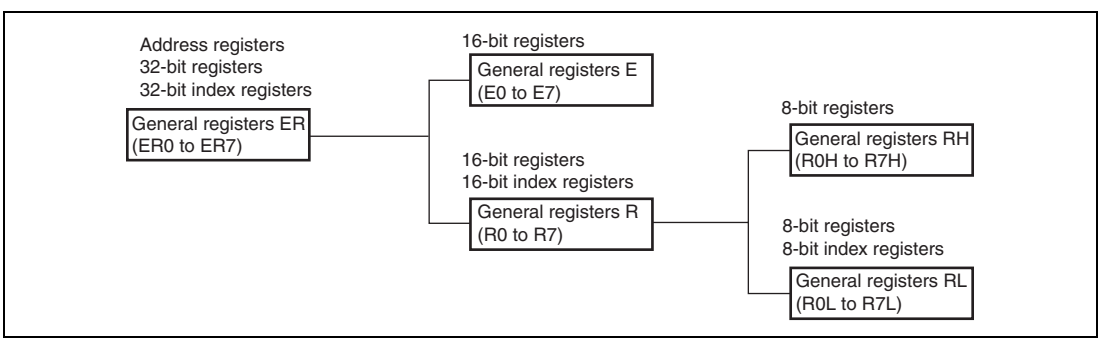


Figure 2.10 Usage of General Registers

General register ER7 has the function of stack pointer (SP) in addition to its general-register function, and is used implicitly in exception handling and subroutine branches. Figure 2.11 shows the stack.

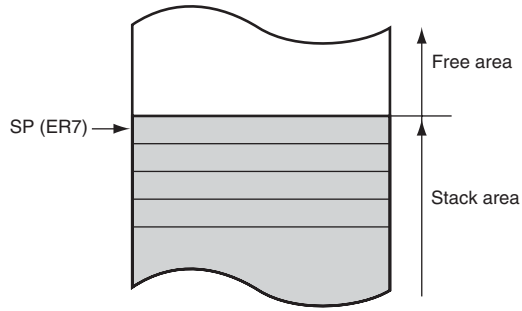


Figure 2.11 Stack

2.5.2 Program Counter (PC)

PC is a 32-bit counter that indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 16 bits (one word) or a multiple of 16 bits, so the least significant bit is ignored. (When the instruction code is fetched, the least significant bit is regarded as 0.

2.5.3 Condition-Code Register (CCR)

CCR is an 8-bit register that contains internal CPU status information, including an interrupt mask (I) and user (UI, U) bits and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

Operations can be performed on the CCR bits by the LDC, STC, ANDC, ORC, and XORC instructions. The N, Z, V, and C flags are used as branch conditions for conditional branch (Bcc) instructions.

Bit	Bit Name	Initial Value	R/W	Description
7	I	1	R/W	Interrupt Mask Bit Masks interrupts when set to 1. This bit is set to 1 at the start of an exception handling.
6	UI	Undefined	R/W	User Bit or Interrupt Mask Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions. This bit can also be used as an interrupt mask bit.

Bit	Bit Name	Initial Value	R/W	Description
5	H	Undefined	R/W	Half-Carry Flag When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.
4	U	Undefined	R/W	User Bit Can be written to and read from by software using the LDC, STC, ANDC, ORC, and XORC instructions.
3	N	Undefined	R/W	Negative Flag Stores the value of the most significant bit (regarded as sign bit) of data.
2	Z	Undefined	R/W	Zero Flag Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.
1	V	Undefined	R/W	Overflow Flag Set to 1 when an arithmetic overflow occurs, and cleared to 0 otherwise.
0	C	Undefined	R/W	Carry Flag Set to 1 when a carry occurs, and cleared to 0 otherwise. A carry has the following types: • Carry from the result of addition • Borrow from the result of subtraction • Carry from the result of shift or rotation The carry flag is also used as a bit accumulator by bit manipulation instructions.

2.5.4 Extended Control Register (EXR)

EXR is an 8-bit register that contains the trace bit (T) and three interrupt mask bits (I2 to I0).

Operations can be performed on the EXR bits by the LDC, STC, ANDC, ORC, and XORC instructions.

For details, see the hardware manual for the corresponding product.

Bit	Bit Name	Initial Value	R/W	Description
7	T	0	R/W	Trace Bit When this bit is set to 1, a trace exception is generated each time an instruction is executed. When this bit is cleared to 0, instructions are executed in sequence.
6 to 3	—	All 1	R/W	Reserved These bits are always read as 1.
2	I2	1	R/W	Interrupt Mask Bits These bits designate the interrupt mask level (0 to 7).
1	I1	1	R/W	
0	I0	1	R/W	

2.5.5 Vector Base Register (VBR)

VBR is a 32-bit register in which the upper 20 bits are valid. The lower 12 bits of this register are read as 0s. This register is a base address of the vector area for exception handlings other than a reset and a CPU address error (extended memory indirect is also out of the target). The initial value is H'00000000. The VBR contents are changed with the LDC and STC instructions.

2.5.6 Short Address Base Register (SBR)

SBR is a 32-bit register in which the upper 24 bits are valid. The lower eight bits are read as 0s. In 8-bit absolute address addressing mode (@aa:8), this register is used as the upper address. The initial value is H'FFFFFF00. The SBR contents are changed with the LDC and STC instructions.

2.5.7 Multiply-Accumulate Register (MAC)

MAC is a 64-bit register that stores the results of multiply-and-accumulate operations. It consists of two 32-bit registers denoted MACH and MACL. The lower 10 bits of MACH are valid; the upper bits are sign extended. The MAC contents are changed with the MAC, CLRMAC, LDMAC, and STMAC instructions.

2.5.8 Initial Values of CPU Registers

Reset exception handling loads the start address from the vector table into the PC, clears the T bit in EXR to 0, and sets the I bits in CCR and EXR to 1. The general registers, MAC, and the other bits in CCR are not initialized. In particular, the initial value of the stack pointer (ER7) is undefined. The SP should therefore be initialized using an MOV.L instruction executed immediately after a reset.

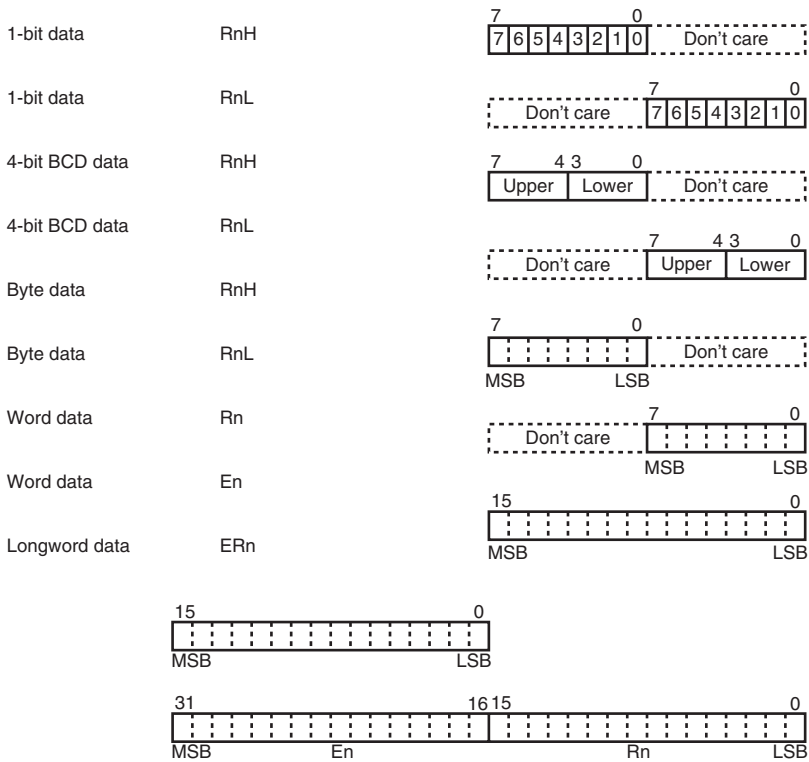
2.6 2Data Formats

The H8SX CPU can process 1-bit, 4-bit BCD, 8-bit (byte), 16-bit (word), and 32-bit (longword) data.

Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

2.6.1 General Register Data Formats

Figure 2.12 shows the data formats in general registers.



[Legend]

ERn: General register ER
En: General register E
Rn: General register R
RnH: General register RH
RnL: General register RL
MSB: Most significant bit
LSB: Least significant bit

Figure 2.12 General Register Data Formats

2.6.2 Memory Data Formats

Figure 2.13 shows the data formats in memory.

The H8SX CPU can access word data and longword data which are stored at any addresses in memory. When word data begins at an odd address or longword data begins at an address other than a multiple of 4, a bus cycle is divided into two or more accesses. For example, when longword data begins at an odd address, the bus cycle is divided into byte, word, and byte accesses. In this case, these accesses are assumed to be individual bus cycles.

However, instructions to be fetched, word and longword data to be accessed during execution of the stack manipulation, branch table manipulation, block transfer instructions, and MAC instruction should be located to even addresses.

When SP (ER7) is used as an address register to access the stack, the operand size should be word size or longword size.

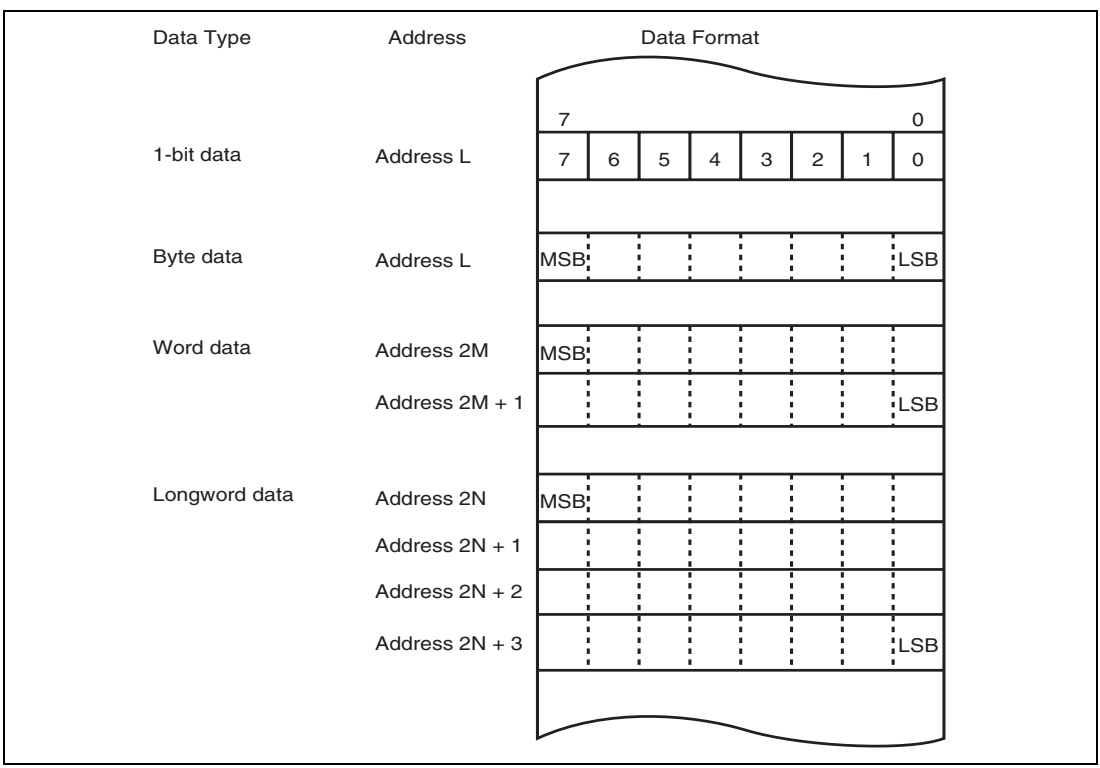


Figure 2.13 Memory Data Formats

2.7 Instruction Set

The H8SX CPU has 87 types of instructions. The instructions are classified by function as shown in table 2.1. The arithmetic operation, logic operation, shift, and bit manipulation instructions are called operation instruction in this manual.

Table 2.1 Instruction Classification

Function	Instructions	Size	Types
Data transfer	MOV	B/W/L	6
	MOVFPE*[6], MOVTPE*[6]	B	
	POP, PUSH*[1]	W/L	
	LDM, STM	L	
	MOVA	B/W*[2]	
Block transfer	EEPMOV	B	3
	MOVMD	B/W/L	
	MOVSD	B	
Arithmetic operations	ADD, ADDX, SUB, SUBX, CMP, NEG, INC, DEC	B/W/L	27
	DAA, DAS	B	
	ADDS, SUBS	L	
	MULXU, DIVXU, MULXS, DIVXS	B/W	
	MULU, DIVU, MULS, DIVS	W/L	
	MULU/U, MULS/U	L	
	EXTU, EXTS	W/L	
	TAS	B	
	MAC	—	
	LDMAC, STMAC	—	
	CLRMAC	—	
Logic operations	AND, OR, XOR, NOT	B/W/L	4
Shift	SHLL, SHLR, SHAL, SHAR, ROTL, ROTR, ROTXL, ROTXR	B/W/L	8
Bit manipulation	BSET, BCLR, BNOT, BTST, BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST	B	20
	BSET/EQ, BSET/NE, BCLR/EQ, BCLR/NE, BSTZ, BISTZ	B	
	BFLD, BFST	B	

Function	Instructions	Size	Types
Branch	BRA/BS, BRA/BC, BSR/BS, BSR/BC	B*[3]	9
	Bcc*[5], JMP, BSR, JSR, RTS	—	
	RTS/L	L*[5]	
	BRA/S	—	
System control	TRAPA, RTE, SLEEP, NOP	—	10
	RTE/L	L*[5]	
	LDC, STC, ANDC, ORC, XORC	B/W/L	
		Total	87

[Legend]

B: Byte size

W: Word size

L: Longword size

Notes:
1. POP.W Rn and PUSH.W Rn are identical to MOV.W @SP+, Rn and MOV.W Rn, @−SP.

   POP.L ERn and PUSH.L ERn are identical to MOV.L @SP+, ERn and MOV.L ERn, @−SP.
2. Size of data to be added with a displacement
3. Size of data to specify a branch condition
4. Bcc is the generic designation of a conditional branch instruction.
5. Size of general register to be restored
6. Not available in this LSI.

2.7.1 Instructions and Addressing Modes

Table 2.2 indicates the combinations of instructions and addressing modes that the H8SX CPU can use.

Table 2.2 Combinations of Instructions and Addressing Modes (1)

			Addressing Mode								
Classification	**Instruction**	**Size**	**#xx**	**Rn**	**@ERn**	**@(d,ERn)**	**@(d, RnL.B/ Rn.W/ ERn.L)**	**@–ERn/ @ERn+/ @ERn–/ @+ERn**	**@aa:8**	**@aa:16/ @aa:32**	**—**
Data transfer	MOV	B/W/L	S	SD	SD	SD	SD	SD		SD	
		B		S/D					S/D		
	MOVFPE, MOVTPE*[12]	B		S/D						S/D*[1]	
	POP, PUSH	W/L		S/D				S/D*[2]			
	LDM, STM	L		S/D				S/D*[2]			
	MOVA*[4]	B/W		S	S	S	S	S		S	
Block transfer	EEPMOV	B									SD*[3]
	MOVMD	B/W/L									SD*[3]
	MOVSD	B									SD*[3]
Arithmetic operations	ADD, CMP	B	S	D	D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	SUB	B	S		D	D	D	D	D	D	
		B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	ADDX, SUBX	B/W/L	S	SD							
		B/W/L	S		SD						
		B/W/L	S					SD*[5]			
	INC, DEC	B/W/L		D							
	ADDS, SUBS	L		D							
	DAA, DAS	B		D							

			Addressing Mode								
Classification	**Instruction**	**Size**	**#xx**	**Rn**	**@ERn**	**@(d,ERn)**	**@(d, RnL.B/ Rn.W/ ERn.L)**	**@–ERn/ @ERn+/ @ERn–/ @+ERn**	**@aa:8**	**@aa:16/ @aa:32**	**—**
Arithmetic operations	MULXU, DIVXU	B/W	S:4	SD							
	MULU, DIVU	W/L	S:4	SD							
	MULXS, DIVXS	B/W	S:4	SD							
	MULS, DIVS	W/L	S:4	SD							
	NEG	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
	EXTU, EXTS	W/L		D	D	D	D	D		D	
	TAS	B			D						
	MAC	—									
	CLRMAC	—									O
	LDMAC	—		S							
	STMAC	—		D							
Logic operations	AND, OR, XOR	B		S	D	D	D	D	D	D	
		B		D	S	S	S	S	S	S	
		B			SD	SD	SD	SD		SD	
		W/L	S	SD	SD	SD	SD	SD		SD	
	NOT	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Shift	SHLL, SHLR	B		D	D	D	D	D	D	D	
		B/W/L*[6]		D	D	D	D	D		D	
		B/W/L*[7]		D							
	SHAL, SHAR ROTL, ROTR ROTXL, ROTXR	B		D	D	D	D	D	D	D	
		W/L		D	D	D	D	D		D	
Bit manipulation	BSET, BCLR, BNOT, BTST, BSET/cc, BCLR/cc	B		D	D				D	D	
	BAND, BIAND, BOR, BIOR, BXOR, BIXOR, BLD, BILD, BST, BIST, BSTZ, BISTZ	B		D	D				D	D	

			Addressing Mode								
Classification	**Instruction**	**Size**	**#xx**	**Rn**	**@ERn**	**@(d,ERn)**	**@(d, RnL.B/ Rn.W/ ERn.L)**	**@–ERn/ @ERn+/ @ERn–/ @+ERn**	**@aa:8**	**@aa:16/ @aa:32**	**—**
Bit manipulation	BFLD	B		D	S				S	S	
	BFST	B		S	D				D	D	
Branch	BRA/BS, BRA/BC*[8]	B			S				S	S	
	BSR/BS, BSR/BC*[8]	B			S				S	S	
System control	LDC (CCR, EXR)	B/W*[9]	S	S	S	S		S*[10]		S	
	LDC (VBR, SBR)	L		S							
	STC (CCR, EXR)	B/W*[9]		D	D	D		D*[11]		D	
	STC (VBR, SBR)	L		D							
	ANDC, ORC, XORC	B	S								
	SLEEP	—									O
	NOP	—									O

[Legend]

d: d:16 or d:32

S: Can be specified as a source operand.

D: Can be specified as a destination operand.

SD: Can be specified as either a source or destination operand or both.

S/D: Can be specified as either a source or destination operand.

S:4: 4-bit immediate data can be specified as a source operand.

Notes:
1. Only @aa:16 is available.
2. @ERn+ as a source operand and @–ERn as a destination operand
3. Specified by ER5 as a source address and ER6 as a destination address for data transfer.
4. Size of data to be added with a displacement
5. Only @ERn– is available
6. When the number of bits to be shifted is 1, 2, 4, 8, or 16
7. When the number of bits to be shifted is specified by 5-bit immediate data or a general register
8. Size of data to specify a branch condition
9. Byte when immediate or register direct, otherwise, word
10. Only @ERn+ is available
11. Only @–ERn is available
12. Not available in this LSI.

Table 2.2 Combinations of Instructions and Addressing Modes (2)

			Addressing Mode							
Classification	**Instruction**	**Size**	**@ERn**	**@(d,PC)**	**@(RnL.B/Rn.W/ERn.L, PC)**	**@aa:24**	**@ aa:32**	**@@ aa:8**	**@@vec:7**	**—**
Branch	BRA/BS, BRA/BC	—		O						
	BSR/BS, BSR/BC	—		O						
	Bcc	—		O						
	BRA	—		O	O					
	BRA/S	—		O*						
	JMP	—	O			O	O	O	O	
	BSR	—		O						
	JSR	—	O			O	O	O	O	
	RTS, RTS/L	—								O
System control	TRAPA	—								O
	RTE, RTE/L	—								O

[Legend]

d: d:8 or d:16

Note: * Only @(d:8, PC) is available.

2.7.2 Table of Instructions Classified by Function

Tables 2.4 to 2.11 summarize the instructions in each functional category. The notation used in these tables is defined in table 2.3.

Table 2.3 Operation Notation

Operation Notation	Description
Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extended control register
CCR	Condition-code register
VBR	Vector base register
SBR	Short address base register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Addition
−	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Move
~	Logical not (logical complement)
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

Table 2.4 Data Transfer Instructions

Instruction	Size	Function
MOV	B/W/L	#IMM → (EAd), (EAs) → (EAd) Transfers data between immediate data, general registers, and memory.
MOVFPE*	B	(EAs) → Rd
MOVTPE*	B	Rs → (EAs)
POP	W/L	@SP+ → Rn Restores the data from the stack to a general register.
PUSH	W/L	Rn → @−SP Saves general register contents on the stack.
LDM	L	@SP+ → Rn (register list) Restores the data from the stack to multiple general registers. Two, three, or four general registers which have serial register numbers can be specified.
STM	L	Rn (register list) → @−SP Saves the contents of multiple general registers on the stack. Two, three, or four general registers which have serial register numbers can be specified.
MOVA	B/W	EA → Rd Zero-extends and shifts the contents of a specified general register or memory data and adds them with a displacement. The result is stored in a general register.

Note: * Not available in this LSI.

Table 2.5 Block Transfer Instructions

Instruction	Size	Function
EEPMOV.B EEPMOV.W	B	Transfers a data block. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4 or R4L.
MOVMD.B	B	Transfers a data block. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4.
MOVMD.W	W	Transfers a data block. Transfers word data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of word data to be transferred is specified by R4.
MOVMD.L	L	Transfers a data block. Transfers longword data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of longword data to be transferred is specified by R4.
MOVSD.B	B	Transfers a data block with zero data detection. Transfers byte data which begins at a memory location specified by ER5 to a memory location specified by ER6. The number of byte data to be transferred is specified by R4. When zero data is detected during transfer, the transfer stops and execution branches to a specified address.

Table 2.6 Arithmetic Operation Instructions

Instruction	Size	Function
ADD SUB	B/W/L	(EAd) ± #IMM → (EAd), (EAd) ± (EAs) → (EAd) Performs addition or subtraction on data between immediate data, general registers, and memory. Immediate byte data cannot be subtracted from byte data in a general register.
ADDX SUBX	B/W/L	(EAd) ± #IMM ± C → (EAd), (EAd) ± (EAs) ± C → (EAd) Performs addition or subtraction with carry on data between immediate data, general registers, and memory. The addressing mode which specifies a memory location can be specified as register indirect with post-decrement or register indirect.
INC DEC	B/W/L	Rd ± 1 → Rd, Rd ± 2 → Rd Increments or decrements a general register by 1 or 2. (Byte operands can be incremented or decremented by 1 only.)
ADDS SUBS	L	Rd ± 1 → Rd, Rd ± 2 → Rd, Rd ± 4 → Rd Adds or subtracts the value 1, 2, or 4 to or from data in a general register.
DAA DAS	B	Rd (decimal adjust) → Rd Decimal-adjusts an addition or subtraction result in a general register by referring to the CCR to produce 2-digit 4-bit BCD data.
MULXU	B/W	Rd × Rs → Rd Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULU	W/L	Rd × Rs → Rd Performs unsigned multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULU/U	L	Rd × Rs → Rd Performs unsigned multiplication on data in two general registers (32 bits × 32 bits → upper 32 bits).
MULXS	B/W	Rd × Rs → Rd Performs signed multiplication on data in two general registers: either 8 bits × 8 bits → 16 bits, or 16 bits × 16 bits → 32 bits.
MULS	W/L	Rd × Rs → Rd Performs signed multiplication on data in two general registers: either 16 bits × 16 bits → 16 bits, or 32 bits × 32 bits → 32 bits.
MULS/U	L	Rd × Rs → Rd Performs signed multiplication on data in two general registers (32 bits × 32 bits → upper 32 bits).

Instruction	Size	Function
DIVXU	B/W	Rd ÷ Rs → Rd Performs unsigned division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder, or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.
DIVU	W/L	Rd ÷ Rs → Rd Performs unsigned division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
DIVXS	B/W	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 8 bits → 8-bit quotient and 8-bit remainder, or 32 bits ÷ 16 bits → 16-bit quotient and 16-bit remainder.
DIVS	W/L	Rd ÷ Rs → Rd Performs signed division on data in two general registers: either 16 bits ÷ 16 bits → 16-bit quotient, or 32 bits ÷ 32 bits → 32-bit quotient.
CMP	B/W/L	(EAd) – #IMM, (EAd) – (EAs) Compares data between immediate data, general registers, and memory and stores the result in CCR.
NEG	B/W/L	0 – (EAd) → (EAd) Takes the two's complement (arithmetic complement) of data in a general register or the contents of a memory location.
EXTU	W/L	(EAd) (zero extension) → (EAd) Performs zero-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be zero-extended.
EXTS	W/L	(EAd) (sign extension) → (EAd) Performs sign-extension on the lower 8 or 16 bits of data in a general register or memory to word or longword size. The lower 8 bits to word or longword, or the lower 16 bits to longword can be sign-extended.
TAS	B	@ERd – 0, 1 → (<bit 7> of @EAd) Tests memory contents, and sets the most significant bit (bit 7) to 1.
MAC	—	(EAs) × (EAd) + MAC → MAC Performs signed multiplication on memory contents and adds the result to MAC.
CLRMAC	—	0 → MAC Clears MAC to zero.

Instruction	Size	Function
LDMAC	—	Rs → MAC Loads data from a general register to MAC.
STMAC	—	MAC → Rd Stores data from MAC to a general register.

Table 2.7 Logic Operation Instructions

Instruction	Size	Function
AND	B/W/L	(EAd) ∧ #IMM → (EAd), (EAd) ∧ (EAs) → (EAd) Performs a logical AND operation on data between immediate data, general registers, and memory.
OR	B/W/L	(EAd) ∨ #IMM → (EAd), (EAd) ∨ (EAs) → (EAd) Performs a logical OR operation on data between immediate data, general registers, and memory.
XOR	B/W/L	(EAd) ⊕ #IMM → (EAd), (EAd) ⊕ (EAs) → (EAd) Performs a logical exclusive OR operation on data between immediate data, general registers, and memory.
NOT	B/W/L	~ (EAd) → (EAd) Takes the one's complement of the contents of a general register or a memory location.

Table 2.8 Shift Operation Instructions

Instruction	Size	Function
SHLL SHLR	B/W/L	(EAd) (shift) → (EAd) Performs a logical shift on the contents of a general register or a memory location. The contents of a general register or a memory location can be shifted by 1, 2, 4, 8, or 16 bits. The contents of a general register can be shifted by any bits. In this case, the number of bits is specified by 5-bit immediate data or the lower 5 bits of the contents of a general register.
SHAL SHAR	B/W/L	(EAd) (shift) → (EAd) Performs an arithmetic shift on the contents of a general register or a memory location. 1-bit or 2-bit shift is possible.
ROTL ROTR	B/W/L	(EAd) (rotate) → (EAd) Rotates the contents of a general register or a memory location. 1-bit or 2-bit rotation is possible.
ROTXL ROTXR	B/W/L	(EAd) (rotate) → (EAd) Rotates the contents of a general register or a memory location with the carry bit. 1-bit or 2-bit rotation is possible.

Table 2.9 Bit Manipulation Instructions

Instruction	Size	Function
BSET	B	1 → (<bit-No.> of <EAd>) Sets a specified bit in the contents of a general register or a memory location to 1. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BSET/cc	B	if cc, 1 → (<bit-No.> of <EAd>) If the specified condition is satisfied, this instruction sets a specified bit in a memory location to 1. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BCLR	B	0 → (<bit-No.> of <EAd>) Clears a specified bit in the contents of a general register or a memory location to 0. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BCLR/cc	B	if cc, 0 → (<bit-No.> of <EAd>) If the specified condition is satisfied, this instruction clears a specified bit in a memory location to 0. The bit number can be specified by 3-bit immediate data, or by the lower three bits of a general register. The Z flag status can be specified as a condition.
BNOT	B	~ (<bit-No.> of <EAd>) → (<bit-No.> of <EAd>) Inverts a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BTST	B	~ (<bit-No.> of <EAd>) → Z Tests a specified bit in the contents of a general register or a memory location and sets or clears the Z flag accordingly. The bit number is specified by 3-bit immediate data or the lower three bits of a general register.
BAND	B	C ∧ (<bit-No.> of <EAd>) → C ANDs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIAND	B	C ∧ [~ (<bit-No.> of <EAd>)] → C ANDs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BOR	B	C ∨ (<bit-No.> of <EAd>) → C ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BIOR	B	C ∨ [~ (<bit-No.> of <EAd>)] → C ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BXOR	B	C ⊕ (<bit-No.> of <EAd>) → C Exclusive-ORs the carry flag with a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BIXOR	B	C ⊕ [~ (<bit-No.> of <EAd>)] → C Exclusive-ORs the carry flag with the inverse of a specified bit in the contents of a general register or a memory location and stores the result in the carry flag. The bit number is specified by 3-bit immediate data.
BLD	B	(<bit-No.> of <EAd>) → C Transfers a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BILD	B	~ (<bit-No.> of <EAd>) → C Transfers the inverse of a specified bit in the contents of a general register or a memory location to the carry flag. The bit number is specified by 3-bit immediate data.
BST	B	C → (<bit-No.> of <EAd>) Transfers the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.
BSTZ	B	Z → (<bit-No.> of <EAd>) Transfers the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BIST	B	~ C → (<bit-No.> of <EAd>) Transfers the inverse of the carry flag value to a specified bit in the contents of a general register or a memory location. The bit number is specified by 3-bit immediate data.

Instruction	Size	Function
BISTZ	B	~ Z → (<bit-No.> of <EAd>) Transfers the inverse of the zero flag value to a specified bit in the contents of a memory location. The bit number is specified by 3-bit immediate data.
BFLD	B	(EAs) (bit field) → Rd Transfers a specified bit field in memory location contents to the lower bits of a specified general register.
BFST	B	Rs → (EAd) (bit field) Transfers the lower bits of a specified general register to a specified bit field in memory location contents.

Table 2.10 Branch Instructions

Instruction	Size	Function
BRA/BS BRA/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a specified address.
BSR/BS BSR/BC	B	Tests a specified bit in memory location contents. If the specified condition is satisfied, execution branches to a subroutine at a specified address.
Bcc	—	Branches to a specified address if the specified condition is satisfied.
BRA/S	—	Branches unconditionally to a specified address after executing the next instruction. The next instruction should be a 1-word instruction except for the block transfer and branch instructions.
JMP	—	Branches unconditionally to a specified address.
BSR	—	Branches to a subroutine at a specified address.
JSR	—	Branches to a subroutine at a specified address.
RTS	—	Returns from a subroutine.
RTS/L	—	Returns from a subroutine, restoring data from the stack to multiple general registers.

Table 2.11 System Control Instructions

Instruction	Size	Function
TRAPA	—	Starts trap-instruction exception handling.
RTE	—	Returns from an exception-handling routine.
RTE/L	—	Returns from an exception-handling routine, restoring data from the stack to multiple general registers.
SLEEP	—	Causes a transition to a power-down state.
LDC	B/W	#IMM → CCR, (EAs) → CCR, #IMM → EXR, (EAs) → EXR Loads immediate data or the contents of a general register or a memory location to CCR or EXR. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	Rs → VBR, Rs → SBR Transfers the general register contents to VBR or SBR.
STC	B/W	CCR → (EAd), EXR → (EAd) Transfers the contents of CCR or EXR to a general register or memory. Although CCR and EXR are 8-bit registers, word-size transfers are performed between them and memory. The upper 8 bits are valid.
	L	VBR → Rd, SBR → Rd Transfers the contents of VBR or SBR to a general register.
ANDC	B	CCR ∧ #IMM → CCR, EXR ∧ #IMM → EXR Logically ANDs the CCR or EXR contents with immediate data.
ORC	B	CCR ∨ #IMM → CCR, EXR ∨ #IMM → EXR Logically ORs the CCR or EXR contents with immediate data.
XORC	B	CCR ⊕ #IMM → CCR, EXR ⊕ #IMM → EXR Logically exclusive-ORs the CCR or EXR contents with immediate data.
NOP	—	PC + 2 → PC Only increments the program counter.

2.7.3 Basic Instruction Formats

The H8SX CPU instructions consist of 2-byte (1-word) units. An instruction consists of an operation field (op field), a register field (r field), an effective address extension (EA field), and a condition field (cc).

Figure 2.14 shows examples of instruction formats.

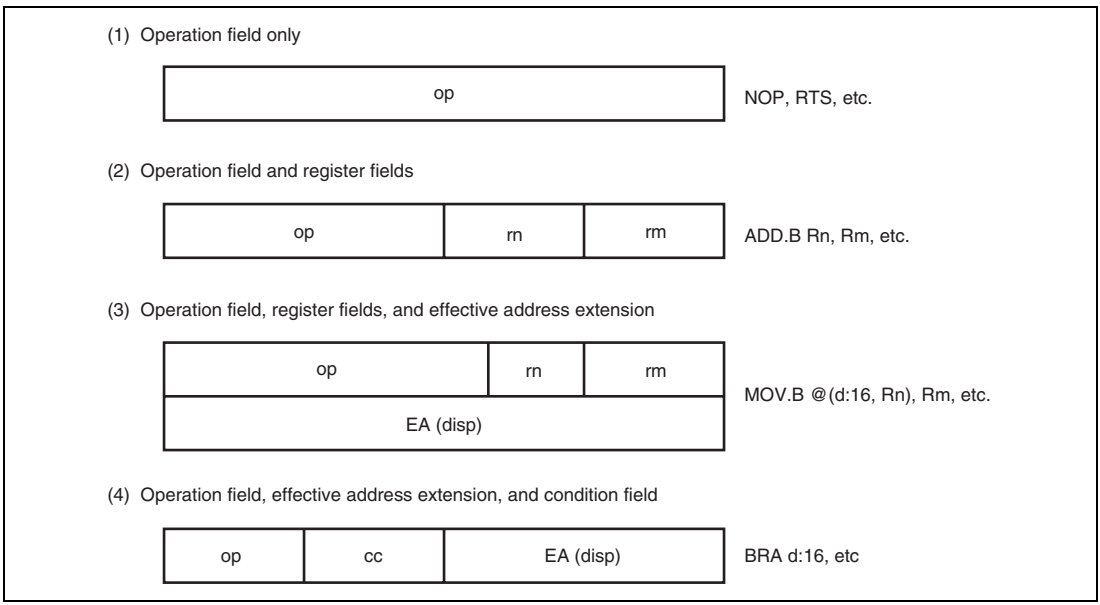


Figure 2.14 Instruction Formats

- Operation Field

  Indicates the function of the instruction, and specifies the addressing mode and operation to be carried out on the operand. The operation field always includes the first four bits of the instruction. Some instructions have two operation fields.
- Register Field

  Specifies a general register. Address registers are specified by 3 bits, data registers by 3 bits or 4 bits. Some instructions have two register fields. Some have no register field.
- Effective Address Extension

  8, 16, or 32 bits specifying immediate data, an absolute address, or a displacement.
- Condition Field

  Specifies the branch condition of Bcc instructions.

2.8 Addressing Modes and Effective Address Calculation

The H8SX CPU supports the 11 addressing modes listed in table 2.12. Each instruction uses a subset of these addressing modes.

Bit manipulation instructions use register direct, register indirect, or absolute addressing mode to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

Table 2.12 Addressing Modes

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:2,ERn)/@(d:16,ERn)/@(d:32,ERn)
4	Index register indirect with displacement	@(d:16, RnL.B)/@(d:16,Rn.W)/@(d:16,ERn.L) @(d:32, RnL.B)/@(d:32,Rn.W)/@(d:32,ERn.L)
5	Register indirect with post-increment	@ERn+
	Register indirect with pre-decrement	@–ERn
	Register indirect with pre-increment	@+ERn
	Register indirect with post-decrement	@ERn–
6	Absolute address	@aa:8/@aa:16/@aa:24/@aa:32
7	Immediate	#xx:3/#xx:4/#xx:8/#xx:16/#xx:32
8	Program-counter relative	@(d:8,PC)/@(d:16,PC)
9	Program-counter relative with index register	@(RnL.B,PC)/@(Rn.W,PC)/@(ERn.L,PC)
10	Memory indirect	@@aa:8
11	Extended memory indirect	@@vec:7

2.8.1 Register Direct—Rn

The operand value is the contents of an 8-, 16-, or 32-bit general register which is specified by the register field in the instruction code.

R0H to R7H and R0L to R7L can be specified as 8-bit registers.

R0 to R7 and E0 to E7 can be specified as 16-bit registers.

ER0 to ER7 can be specified as 32-bit registers.

2.8.2 Register Indirect—@ERn

The operand value is the contents of the memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code.

In advanced mode, if this addressing mode is used in a branch instruction, the lower 24 bits are valid and the upper 8 bits are all assumed to be 0 (H'00).

2.8.3 Register Indirect with Displacement—@(d:2, ERn), @(d:16, ERn), or @(d:32, ERn)

The operand value is the contents of a memory location which is pointed to by the sum of the contents of an address register (ERn) and a 16- or 32-bit displacement. ERn is specified by the register field of the instruction code. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn.

This addressing mode has a short format (@(d:2, ERn)). The short format can be used when the displacement is 1, 2, or 3 and the operand is byte data, when the displacement is 2, 4, or 6 and the operand is word data, or when the displacement is 4, 8, or 12 and the operand is longword data.

2.8.4 Index Register Indirect with Displacement—@(d:16,RnL.B), @(d:32,RnL.B), @(d:16,Rn.W), @(d:32,Rn.W), @(d:16,ERn.L), or @(d:32,ERn.L)

The operand value is the contents of a memory location which is pointed to by the sum of the following operation result and a 16- or 32-bit displacement: a specified bits of the contents of an address register (RnL, Rn, ERn) specified by the register field in the instruction code are zero-extended to 32-bit data and multiplied by 1, 2, or 4. The displacement is included in the instruction code and the 16-bit displacement is sign-extended when added to ERn. If the operand is byte data, ERn is multiplied by 1. If the operand is word or longword data, ERn is multiplied by 2 or 4, respectively.

2.8.5 Register Indirect with Post-Increment, Pre-Decrement, Pre-Increment, or Post-Decrement—@ERn+, @−ERn, @+ERn, or @ERn−

(1) Register indirect with post-increment—@ERn+

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After the memory location is accessed, 1, 2, or 4 is added to the address register contents and the sum is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.

(2) Register indirect with pre-decrement—@−ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is subtracted from the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.

(3) Register indirect with pre-increment—@+ERn

The operand value is the contents of a memory location which is pointed to by the following operation result: the value 1, 2, or 4 is added to the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After that, the operand value is stored in the address register. The value added is 1 for byte access, 2 for word access, or 4 for longword access.

(4) Register indirect with post-decrement—@ERn−

The operand value is the contents of a memory location which is pointed to by the contents of an address register (ERn). ERn is specified by the register field of the instruction code. After the memory location is accessed, 1, 2, or 4 is subtracted from the address register contents and the remainder is stored in the address register. The value subtracted is 1 for byte access, 2 for word access, or 4 for longword access.

If the contents of a general register which is also used as an address register is written to memory using this addressing mode, data to be written is the contents of the address register after calculating an effective address.

2.8.6 Absolute Address—@aa:8, @aa:16, @aa:24, or @aa:32

The operand value is the contents of a memory location which is pointed to by an absolute address included in the instruction code.

There are 8-bit (@aa:8), 16-bit (@aa:16), 24-bit (@aa:24), and 32-bit (@aa:32) absolute addresses.

To access the data area, the absolute address of 8 bits (@aa:8), 16 bits (@aa:16), or 32 bits (@aa:32) is used. For an 8-bit absolute address, the upper 24 bits are specified by SBR. For a 16-bit absolute address, the upper 16 bits are sign-extended. A 32-bit absolute address can access the entire address space.

To access the program area, the absolute address of 24 bits (@aa:24) or 32 bits (@aa:32) is used. For a 24-bit absolute address, the upper 8 bits are all assumed to be 0 (H'00).

Table 2.13 shows the accessible absolute address ranges.

Table 2.13 Absolute Address Access Ranges

Absolute Address		Normal Mode	Middle Mode	Advanced Mode	Maximum Mode
Data area	8 bits (@aa:8)	A consecutive 256-byte area (the upper address is set in SBR)			
	16 bits (@aa:16)	H'0000 to H'FFFF	H'000000 to H'007FFF,	H'00000000 to H'00007FFF, H'FFFF8000 to H'FFFFFFFF	
	32 bits (@aa:32)		H'FF8000 to H'FFFFFF	H'00000000 to H'FFFFFFFF	
Program area	24 bits (@aa:24)		H'000000 to H'FFFFFF	H'00000000 to H'00FFFFFF	
	32 bits (@aa:32)			H'00000000 to H'00FFFFFF	H'00000000 to H'FFFFFFFF

2.8.7 Immediate—#xx

The operand value is 8-bit (#xx:8), 16-bit (#xx:16), or 32-bit (#xx:32) data included in the instruction code.

This addressing mode has short formats in which 3- or 4-bit immediate data can be used.

When the size of immediate data is less than that of the destination operand value (byte, word, or longword) the immediate data is zero-extended.

The ADDS, SUBS, INC, and DEC instructions contain immediate data implicitly. Some bit manipulation instructions contain 3-bit immediate data in the instruction code, for specifying a bit number. The BFLD and BFST instructions contain 8-bit immediate data in the instruction code, for specifying a bit field. The TRAPA instruction contains 2-bit immediate data in the instruction code, for specifying a vector address.

2.8.8 Program-Counter Relative—@(d:8, PC) or @(d:16, PC)

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of an 8- or 16-bit displacement in the instruction code and the 32-bit address of the PC contents. The 8-bit or 16-bit displacement is sign-extended to 32 bits when added to the PC contents. The PC contents to which the displacement is added is the address of the first byte of the next instruction, so the possible branching range is –126 to +128 bytes (–63 to +64 words) or –32766 to +32768 bytes (–16383 to +16384 words) from the branch instruction. The resulting value should be an even number. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

2.8.9 Program-Counter Relative with Index Register—@(RnL.B, PC), @(Rn.W, PC), or @(ERn.L, PC)

This mode is used in the Bcc and BSR instructions. The operand value is a 32-bit branch address, which is the sum of the following operation result and the 32-bit address of the PC contents: the contents of an address register specified by the register field in the instruction code (RnL, Rn, or ERn) is zero-extended and multiplied by 2. The PC contents to which the displacement is added is the address of the first byte of the next instruction. In advanced mode, only the lower 24 bits of this branch address are valid; the upper 8 bits are all assumed to be 0 (H'00).

2.8.10 Memory Indirect—@@aa:8

This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by an 8-bit absolute address in the instruction code.

The upper bits of an 8-bit absolute address are all assumed to be 0, so the address range is 0 to 255 (H'0000 to H'00FF in normal mode, H'000000 to H'0000FF in other modes).

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

Note that the top part of the address range is also used as the exception handling vector area. A vector address of an exception handling other than a reset or a CPU address error can be changed by VBR.

Figure 2.15 shows an example of specification of a branch address using this addressing mode.

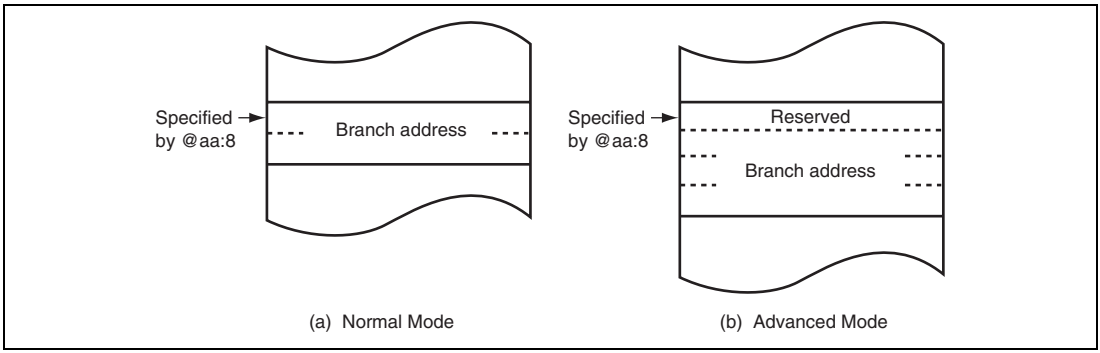


Figure 2.15 Branch Address Specification in Memory Indirect Mode

2.8.11 Extended Memory Indirect—@@vec:7

This mode can be used by the JMP and JSR instructions. The operand value is a branch address, which is the contents of a memory location pointed to by the following operation result: the sum of 7-bit data in the instruction code and the value of H'80 is multiplied by 2 or 4.

The address range to store a branch address is H'0100 to H'01FF in normal mode and H'000200 to H'0003FF in other modes. In assembler notation, an address to store a branch address is specified.

In normal mode, the memory location is pointed to by word-size data and the branch address is 16 bits long. In other modes, the memory location is pointed to by longword-size data. In middle or advanced mode, the first byte of the longword-size data is assumed to be all 0 (H'00).

2.8.12 Effective Address Calculation

Tables 2.14 and 2.15 show how effective addresses are calculated in each addressing mode. The lower bits of the effective address are valid and the upper bits are ignored (zero extended or sign extended) according to the CPU operating mode.

The valid bits in middle mode are as follows:

- The lower 16 bits of the effective address are valid and the upper 16 bits are sign-extended for the transfer and operation instructions.
- The lower 24 bits of the effective address are valid and the upper eight bits are zero-extended for the branch instructions.

Table 2.14 Effective Address Calculation for Transfer and Operation Instructions

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Immediate op | IMM		
2	Register direct op | rm | rn		
3	Register indirect op | r	31 … 0 General register contents	31 … 0
4	Register indirect with 16-bit displacement op | r disp	31 … 0 General register contents 31 … 15 … 0 Sign extension | disp	31 … 0
	Register indirect with 32-bit displacement op | r disp	31 … 0 General register contents disp	31 … 0
5	Index register indirect with 16-bit displacement op | r disp	31 … 0 Zero extension Contents of general register (RL, R, or ER) 1, 2, or 31 … 15 … 0 Sign extension | disp	31 … 0
	Index register indirect with 32-bit displacement op | r disp	31 … 0 Zero extension Contents of general register (RL, R, or ER) 1, 2, or 31 … 0 disp	31 … 0
6	Register indirect with post-increment or post-decrement op | r	31 … 0 General register contents 1, 2, or 4	31 … 0
	Register indirect with pre-increment or pre-decrement op | r	31 … 0 General register contents 1, 2, or 4	31 … 0
7	8-bit absolute address op | aa	31 … 7 … 0 SBR | aa	31 … 0
	16-bit absolute address op aa	31 … 15 … 0 Sign extension | aa	31 … 0
	32-bit absolute address op aa	31 … 0 aa	31 … 0

Table 2.15 Effective Address Calculation for Branch Instructions

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register indirect op r	31 0 General register contents	31 0
2	Program-counter relative with 8-bit displacement op disp	31 0 PC contents 31 7 0 Sign extension disp	31 0
	Program-counter relative with 16-bit displacement op disp	31 0 PC contents 31 15 0 Sign extension disp	31 0
3	Program-counter relative with index register op r	31 0 Zero extension Contents of general register (RL, R, or ER) 2 31 0 PC contents	31 0
4	24-bit absolute address op aa	Zero 31 extension 23 0 aa	31 0
	32-bit absolute address op aa	31 0 aa	31 0
5	Memory indirect op aa	31 7 0 Zero extension aa 31 0 Memory contents	31 0
6	Extended memory indirect op vec	31 7 0 Zero extension 1 vec 2 or 4 31 0 31 0 Memory contents	31 0

2.8.13 MOVA Instruction

The MOVA instruction stores the effective address in a general register.

1. Firstly, data is obtained by the addressing mode shown in item 2of table 2.14.
2. Next, the effective address is calculated using the obtained data as the index by the addressing mode shown in item 5 of table 2.14. The obtained data is used instead of the general register. The result is stored in a general register. For details, see H8SX Family Software Manual.

2.9 Processing States

The H8SX CPU has five main processing states: the reset state, exception-handling state, program execution state, bus-released state, and program stop state. Figure 2.16 indicates the state transitions.

- Reset state

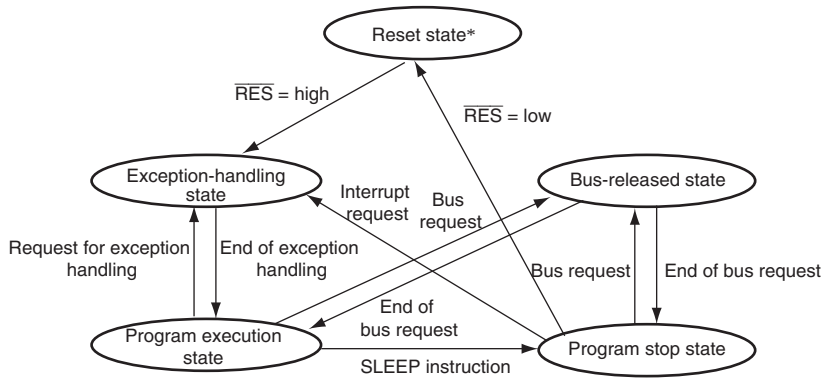
  In this state the CPU and internal peripheral modules are all initialized and stopped. When the $\overline{\text{RES}}$ input goes low, all current processing stops and the CPU enters the reset state. All interrupts are masked in the reset state. Reset exception handling starts when the $\overline{\text{RES}}$ signal changes from low to high. The reset state can also be entered by a watchdog timer overflow. For details, refer to section 4, Exception Handling.
- Exception-handling state

  The exception-handling state is a transient state that occurs when the CPU alters the normal processing flow due to activation of an exception source, such as, a reset, trace, interrupt, or trap instruction. The CPU fetches a start address (vector) from the exception handling vector table and branches to that address. For further details, refer to section 4, Exception Handling.
- Program execution state

  In this state the CPU executes program instructions in sequence.
- Bus-released state

  In this state, the bus has been released in response to a bus request from the DMA controller (DMAC). While the bus is released, the CPU halts operations.
- Program stop state

  This is a power-down state in which the CPU stops operating. The program stop state occurs when a SLEEP instruction is executed or the CPU enters software standby mode. For details, refer to section 19, Power-Down Modes.



Note: * A transition to the reset state occurs whenever the $\overline{\text{RES}}$ signal goes low. A transition can also be made to the reset state when the watchdog timer overflows.

Figure 2.16 State Transitions

Section 3 MCU Operating Modes

3.1 Operating Mode Selection

This LSI has three operating modes (modes 1 to 3). The operating mode is selected by the setting of mode pins (MD1 and MD0). Table 3.1 lists MCU operating mode settings.

In this LSI, advanced mode for the CPU operating mode and 16-Mbyte address space are available. LSI initiation mode can be selected from boot mode and user boot mode for programming/erasing the flash memory and single chip initiation mode.

Table 3.1 MCU Operating Mode Settings

MCU Operating Mode	MD1	MD0	CPU Operating Mode	Address Space	Description	On-Chip ROM
1	0	1	Advanced	16 Mbytes	User boot mode	Enabled
2	1	0			Boot mode	Enabled
3	1	1			Single chip initiation mode	Enabled

In mode 1 and mode 2, which are user boot mode and boot mode, the flash memory can be programmed and erased. For details on user boot mode and boot mode, see section 17, Flash Memory (0.18-μm F-ZTAT Version).

In mode 3, this LSI operates in single chip mode.

3.2 Register Descriptions

The following registers are related to the operating mode setting.

- Mode control register (MDCR)
- System control register (SYSCR)

3.2.1 Mode Control Register (MDCR)

MDCR indicates the current operating mode. When MDCR is read, the states of signals input on pins MD1 and MD0 are latched. The latch is released by a reset.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	—	—	MDS3	MDS2	MDS1	MDS0
Initial Value	0	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	0	1	0	1	Undefined*	Undefined*	Undefined*	Undefined*
R/W	R	R	R	R	R	R	R	R

Note: * Determined by pins MD1 and MD0.

Bit	Bit Name	Initial Value	R/W	Descriptions
15	—	0	R	Reserved
14	—	1	R	These are read-only bits and cannot be modified.
13	—	0	R	
12	—	1	R	
11	MDS3	Undefined*	R	Mode Select 3 to 0
10	MDS2	Undefined*	R	These bits indicate the operating mode selected by the mode pins (MD1 and MD0) (see table 3.2).
9	MDS1	Undefined*	R	
8	MDS0	Undefined*	R	

Bit	Bit Name	Initial Value	R/W	Descriptions
7	—	0	R	Reserved
6	—	1	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	—	1	R	
3	—	Undefined*	R	
2	—	Undefined*	R	
1	—	Undefined*	R	
0	—	Undefined*	R	

Note: * Determined by pins MD1 and MD0.

Table 3.2 Settings of Bits MSD3 to MSD0

			MDCR			
MCU Operating Mode	MD1	MD0	MDS3	MDS2	MDS1	MDS0
1	0	1	1	1	0	1
2	1	0	1	1	0	0
3	1	1	0	1	0	0

3.2.2 System Control Register (SYSCR)

SYSCR controls MAC saturation operation and selects enables/disables the on-chip RAM and the flash memory control registers.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	MACS	—	—	—	—	RAME
Initial Value	1	1	0	1	0	1	0	1
R/W	R	R	R/W	R	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	FLSHE	—	—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Descriptions
15, 14	—	All 1	R	Reserved These are read-only bits and cannot be modified.
13	MACS	0	R/W	MAC Saturation Operation Control Selects either saturation operation or non-saturation operation for the MAC instruction. 0: MAC instruction is non-saturation operation 1: MAC instruction is saturation operation
12	—	1	R	Reserved This is a read-only bit and cannot be modified.
11	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
10	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
9	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
8	RAME	1	R/W	RAM Enable Enables or disables the on-chip RAM. This bit is initialized when the reset state is released. Do not write 0 during access to the on-chip RAM. 0: On-chip RAM disabled 1: On-chip RAM enabled
7	FLSHE	0	R/W	Flash Memory Control Register Enable Controls accesses to the flash memory control registers. Setting this bit to 1 enables to read from and write to the flash memory control registers. Clearing this bit to 0 disables the flash memory control registers. At this time, the contents of the flash memory control registers are retained. The write value should be 0 when the LSI is not the flash memory version. 0: Disables the flash memory control registers 1: Enables the flash memory control registers
6 to 2	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
1, 0	—	All 1	R/W	Reserved This bit is always read as 1. The write value should always be 1.

3.3 Operating Mode Descriptions

3.3.1 Mode 1

Mode 1 is the user boot mode for the flash memory. The operations are the same as that in mode 3 other than programming/erasing the flash memory.

3.3.2 Mode 2

Mode 2 is the boot mode for the flash memory. The operations are the same as that in mode 3 other than programming/erasing the flash memory.

3.3.3 Mode 3

Mode 3 is advanced mode in which the address space is 16 Mbytes, and single-chip mode with the on-chip ROM enabled.

3.4 Address Map

3.4.1 Address Map (Advanced Mode)

Figure 3.1 shows the address map.

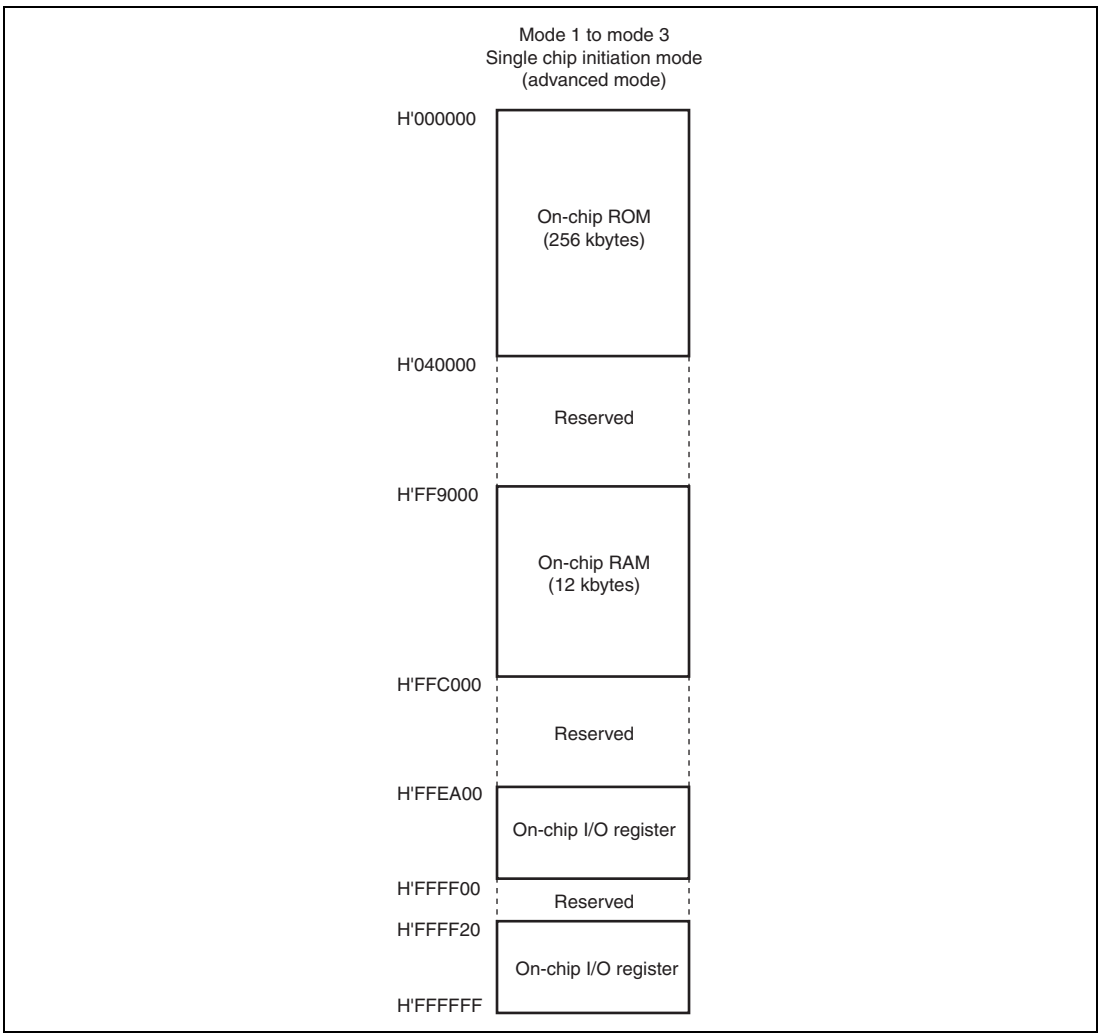


Figure 3.1 Address Map (Advanced Mode)

Section 4 Exception Handling

4.1 Exception Handling Types and Priority

As table 4.1 indicates, exception handling is caused by a reset, a trace, an address error, an interrupt, a trap instruction, and illegal instructions (general illegal instruction and slot illegal instruction). Exception handling is prioritized as shown in table 4.1. If two or more exceptions occur simultaneously, they are accepted and processed in order of priority. Exception sources, the stack structure, and operation of the CPU vary depending on the interrupt control mode. For details on the interrupt control mode, see section 5, Interrupt Controller.

Table 4.1 Exception Types and Priority

Priority	Exception Type	Exception Handling Start Timing
High ↑	Reset	Exception handling starts at the timing of level change from low to high on the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows. The CPU enters the reset state when the $\overline{\text{RES}}$ pin is low.
	Illegal instruction	Exception handling starts when an undefined code is executed.
	Trace*[1]	Exception handling starts after execution of the current instruction or exception handling, if the trace (T) bit in EXR is set to 1.
	Address error	After an address error occurs, the exception handling starts on completion of the current instruction execution.
	Interrupt	Exception handling starts after execution of the current instruction or exception handling, if an interrupt request has occurred.*[2]
Low	Trap instruction*[3]	Exception handling starts by execution of a trap instruction (TRAPA).

Notes: 1. Traces are enabled only in interrupt control mode 2. Trace exception handling is not executed after execution of an RTE instruction.

2. Interrupt detection is not performed on completion of ANDC, ORC, XORC, or LDC instruction execution, or on completion of reset exception handling.

3. Trap instruction exception handling requests are accepted at all times in the program execution state.

4.2 Exception Sources and Exception Handling Vector Table

Different vector table address offsets are assigned to different exception sources. The vector table addresses are calculated from the contents of the vector base register (VBR) and vector table address offset of the vector number. The start address of the exception service routine is fetched from the exception handling vector table indicated by this vector table address.

Table 4.2 shows the correspondence between the exception sources and vector table address offsets. Table 4.3 shows the calculation method of exception handling vector table addresses.

Since the usable modes differ depending on the product, for details on the available modes, see section 3, MCU Operating Modes.

Table 4.2 Exception Handling Vector Table

			Vector Table Address Offset*[1]	
Exception Source		**Vector Number**	**Normal Mode*[2]**	**Advanced, Middle, Maximum Modes**
Reset		0	H'0000 to H'0001	H'0000 to H'0003
Reserved for system use		1	H'0002 to H'0003	H'0004 to H'0007
		2	H'0004 to H'0005	H'0008 to H'000B
		3	H'0006 to H'0007	H'000C to H'000F
Illegal instruction		4	H'0008 to H'0009	H'0010 to H'0013
Trace		5	H'000A to H'000B	H'0014 to H'0017
Reserved for system use		6	H'000C to H'000D	H'0018 to H'001B
Interrupt (NMI)		7	H'000E to H'000F	H'001C to H'001F
Trap instruction	(#0)	8	H'0010 to H'0011	H'0020 to H'0023
	(#1)	9	H'0012 to H'0013	H'0024 to H'0027
	(#2)	10	H'0014 to H'0015	H'0028 to H'002B
	(#3)	11	H'0016 to H'0017	H'002C to H'002F
CPU address error		12	H'0018 to H'0019	H'0030 to H'0033
DMA address error*[3]		13	H'001A to H'001B	H'0034 to H'0037
Reserved for system use		14	H'001C to H'001D	H'0038 to H'003B
		|	|	|
		63	H'007E to H'007F	H'00FC to H'00FF

Exception Source		Vector Number	Vector Table Address Offset*[1] Normal Mode*[2]	Vector Table Address Offset*[1] Advanced, Middle, Maximum Modes
External interrupt	IRQ0	64	H'0080 to H'0081	H'0100 to H'0103
	IRQ1	65	H'0082 to H'0083	H'0104 to H'0107
	IRQ2	66	H'0084 to H'0085	H'0108 to H'010B
	IRQ3	67	H'0086 to H'0087	H'010C to H'010F
	IRQ4	68	H'0088 to H'0089	H'0110 to H'0113
	IRQ5	69	H'008A to H'008B	H'0114 to H'0117
	IRQ6	70	H'008C to H'008D	H'0118 to H'011B
	IRQ7	71	H'008E to H'008F	H'011C to H'011F
	IRQ8	72	H'0090 to H'0091	H'0120 to H'0123
	IRQ9	73	H'0092 to H'0093	H'0124 to H'0127
	IRQ10	74	H'0094 to H'0095	H'0128 to H'012B
	IRQ11	75	H'0096 to H'0097	H'012C to H'012F
	IRQ12	76	H'0098 to H'0099	H'0130 to H'0133
	IRQ13	77	H'009A to H'009B	H'0134 to H'0137
	IRQ14	78	H'009C to H'009D	H'0138 to H'013B
Reserved for system use		79	H'009E to H'009F	H'013C to H'013F
		80	H'00A0 to H'00A1	H'0140 to H'0143
Internal interrupt*[4]		81	H'00A2 to H'00A3	H'0144 to H'0147
		⏐	⏐	⏐
		255	H'01FE to H'01FF	H'03FC to H'03FF

Notes: 1. Lower 16 bits of the address.
2. Not available in this LSI.
3. A DMA address error is generated within the DMAC.
4. For details on the interrupt vector table, see section 5.5, Interrupt Exception Handling Vector Table.

Table 4.3 Calculation Method of Exception Handling Vector Table Address

Exception Source	Calculation Method of Vector Table Address
Reset, CPU address error	Vector table address = (vector table address offset)
Other than above	Vector table address = VBR + (vector table address offset)

[Legend]
VBR: Vector base register
Vector table address offset: See table 4.2.

4.3 Reset

A reset has priority over any other exception. When the $\overline{\text{RES}}$ pin goes low, all processing halts and this LSI enters the reset state. To ensure that this LSI is reset, hold the $\overline{\text{RES}}$ pin low for at least 20 ms when the power is turned on. When operation is in progress, hold the $\overline{\text{RES}}$ pin low for at least 20 cycles.

The chip can also be reset by overflow of the watchdog timer. For details, see section 11, Watchdog Timer (WDT).

A reset initializes the internal state of the CPU and the registers of the on-chip peripheral modules. The interrupt control mode is 0 immediately after a reset.

4.3.1 Reset Exception Handling

When the $\overline{\text{RES}}$ pin goes high after being held low for the necessary time, this LSI starts reset exception handling as follows:

1. The internal state of the CPU and the registers of the on-chip peripheral modules are initialized, VBR is cleared to H'00000000, the T bit is cleared to 0 in EXR, and the I bits are set to 1 in EXR and CCR.
2. The reset exception handling vector address is read and transferred to the PC, and program execution starts from the address indicated by the PC.

Figure 4.1 shows an example of the reset sequence.

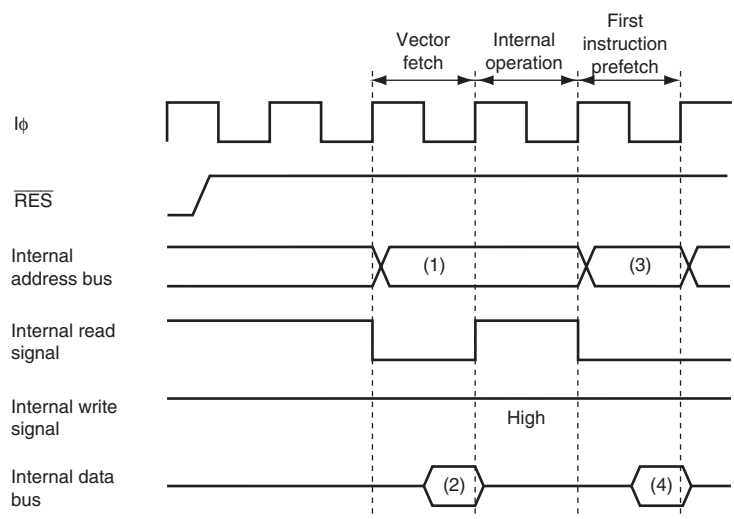
4.3.2 Interrupts after Reset

If an interrupt is accepted after a reset but before the stack pointer (SP) is initialized, the PC and CCR will not be saved correctly, leading to a program crash. To prevent this, all interrupt requests, including NMI, are disabled immediately after a reset. Since the first instruction of a program is always executed immediately after the reset state ends, make sure that this instruction initializes the stack pointer (example: MOV.L #xx: 32, SP).

4.3.3 On-Chip Peripheral Functions after Reset Release

After the reset state is released, MSTPCRA, MSTPCRB, and MSTPCRC are initialized to H'0FFF, H'FFFF, and H'FF00 respectively, and all modules except the DMAC enter module stop mode.

Consequently, on-chip peripheral module registers cannot be read from or written to. Register reading and writing is enabled when module stop mode is canceled.



(1) Reset exception handling vector address (when reset, (1) = H'000000)
(2) Start address (contents of reset exception handling vector address)
(3) Start address ((3) = (2))
(4) First instruction in the exception handling routine

Figure 4.1 Reset Sequence (On-Chip ROM Enabled Advanced Mode)

4.4 Traces

Traces are enabled in interrupt control mode 2. Trace mode is not activated in interrupt control mode 0, irrespective of the state of the T bit. Before changing interrupt control modes, the T bit must be cleared to 0. For details on interrupt control modes, see section 5, Interrupt Controller.

If the T bit in EXR is set to 1, trace mode is activated. In trace mode, a trace exception occurs on completion of each instruction. Trace mode is not affected by interrupt masking by CCR. Table 4.4 shows the state of CCR and EXR after execution of trace exception handling. Trace mode is canceled by clearing the T bit in EXR to 0 during the trace exception handling. However, the T bit saved on the stack retains its value of 1, and when control is returned from the trace exception handling routine by the RTE instruction, trace mode resumes. Trace exception handling is not carried out after execution of the RTE instruction.

Interrupts are accepted even within the trace exception handling routine.

Table 4.4 Status of CCR and EXR after Trace Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**I2 to I0**	**T**
0	Trace exception handling cannot be used.			
2	1	—	—	0

[Legend]
1: Set to 1
0: Cleared to 0
—: Retains the previous value.

4.5 Address Error

4.5.1 Address Error Source

Instruction fetch, stack operation, data read/write, and single-address transfer shown in table 4.5 may cause an address error.

Table 4.5 Bus Cycle and Address Error

Bus Cycle			
Type	**Bus Master**	**Description**	**Address Error**
Instruction fetch	CPU	Fetches instructions from even addresses	No (normal)
		Fetches instructions from odd addresses	Occurs
		Fetches instructions from areas other than on-chip peripheral module space*[1]	No (normal)
		Fetches instructions from on-chip peripheral module space*[1]	Occurs
		Fetches instructions from external memory space in single-chip mode	Occurs
		Fetches instructions from access reserved area.*[2]	Occurs
Stack operation	CPU	Accesses stack when the stack pointer value is even address	No (normal)
		Accesses stack when the stack pointer value is odd	Occurs
Data read/write	CPU	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to reserved area*[2]	Occurs
Data read/write	DMAC	Accesses word data from even addresses	No (normal)
		Accesses word data from odd addresses	No (normal)
		Accesses external memory space in single-chip mode	Occurs
		Accesses to reserved area*[2]	Occurs

Bus Cycle			
Type	**Bus Master**	**Description**	**Address Error**
Single address transfer	DMAC	In single address transfer, the device to be accessed with an address is in the external memory space	No (normal)
		In single address transfer, the device to be accessed with an address is not in the external memory space	Occurs

Notes: 1. For on-chip peripheral module space, see section 6, Bus Controller (BSC).
2. For the access reserved area, refer to figure 3.1 in section 3.4, Address Map. An address error will not occur when the reserved area from H'FF8000 to H'FF8FFF is accessed.

4.5.2 Address Error Exception Handling

When an address error occurs, address error exception handling starts after the bus cycle causing the address error ends and current instruction execution completes. The address error exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the address error is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Even though an address error occurs during a transition to an address error exception handling, the address error is not accepted. This prevents an address error from occurring due to stacking for exception handling, thereby preventing infinitive stacking.

If the SP contents are not a multiple of 2 when an address error exception handling occurs, the stacked values (PC, CCR, and EXR) are undefined.

When an address error occurs, the following is performed to halt the DMAC.

- ERRF bit in DMDR_0 of the DMAC is set to 1
- DTE bits for all the channels of the DMAC are cleared to 0 and the DMAC is forced to halt

Table 4.6 shows the states of CCR and EXR after the address error exception handling.

Table 4.6 States of CCR and EXR after Address Error Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	7

[Legend]
1: Set to 1.
0: Cleared to 0.
—: Retains the previous value.

4.6 Interrupts

4.6.1 Interrupt Sources

Interrupt sources are NMI, IRQ0 to IRQ14, and on-chip peripheral modules, as shown in table 4.7.

Table 4.7 Interrupt Sources

Type	Source	Number of Sources
NMI	NMI pin (external input)	1
IRQ0 to IRQ14	Pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ11}}$ (external input)	15
On-chip peripheral module	Watchdog timer (WDT)	1
	A/D converter	2
	16-bit timer pulse unit (TPU)	52*[1]/26*[2]
	DMA controller (DMAC)	8
	Serial communications interface (SCI)	8
	Synchronous serial communication unit (SSU)	9
	Controller area network (RCAN-ET)	4

Notes: 1. The number of interrupts for the H8SX/1527R
2. The number of interrupts for the H8SX/1525R

Different vector numbers and vector table offsets are assigned to different interrupt sources. For vector number and vector table offset, refer to table 5.2 in section 5.5, Interrupt Exception Handling Vector Table.

4.6.2 Interrupt Exception Handling

Interrupts are controlled by the interrupt controller. The interrupt controller has two interrupt control modes and can assign interrupts other than NMI to eight priority/mask levels to enable multiple-interrupt control. The source to start interrupt exception handling and the vector address differ depending on the product. For details, refer to section 5, Interrupt Controller.

The interrupt exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the interrupt source is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

4.7 Instruction Exception Handling

There are two types of instructions that cause exception handling: trap instruction and illegal instructions.

4.7.1 Trap Instruction

Trap instruction exception handling starts when a TRAPA instruction is executed. Trap instruction exception handling can be executed at all times in the program execution state. The trap instruction exception handling is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the vector number specified in the TRAPA instruction is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

A start address is read from the vector table corresponding to a vector number from 0 to 3, as specified in the instruction code.

Table 4.8 shows the state of CCR and EXR after execution of trap instruction exception handling.

Table 4.8 Status of CCR and EXR after Trap Instruction Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	—

[Legend]
1: Set to 1
0: Cleared to 0
—: Retains the previous value.

4.7.2 Exception Handling by Illegal Instruction

There are two illegal instructions: general illegal instruction and slot illegal instruction.

The exception handling by the general illegal instruction starts when an undefined code is decoded.

The exception handling by the slot illegal instruction starts when the following instruction which is placed in a delay slot (immediately after a delayed branch instruction) is executed: an instruction which consists of two words or more or which changes the contents of PC.

The general illegal and slot illegal instructions are always executable in the program execution state.

The exception handling for the general illegal and a slot illegal instructions is as follows:

1. The contents of PC, CCR, and EXR are saved in the stack.
2. The interrupt mask bit is updated and the T bit is cleared to 0.
3. An exception handling vector table address corresponding to the occurred exception is generated, the start address of the exception service routine is loaded from the vector table to PC, and program execution starts from that address.

Table 4.9 shows the state of CCR and EXR after execution of illegal instruction exception handling.

Table 4.9 Status of CCR and EXR after Illegal Instruction Exception Handling

	CCR		EXR	
Interrupt Control Mode	**I**	**UI**	**T**	**I2 to I0**
0	1	—	—	—
2	1	—	0	—

[Legend]
1: Set to 1
0: Cleared to 0
—: Retains the previous value.

4.8 Stack Status after Exception Handling

Figure 4.2 shows the stack after completion of exception handling.

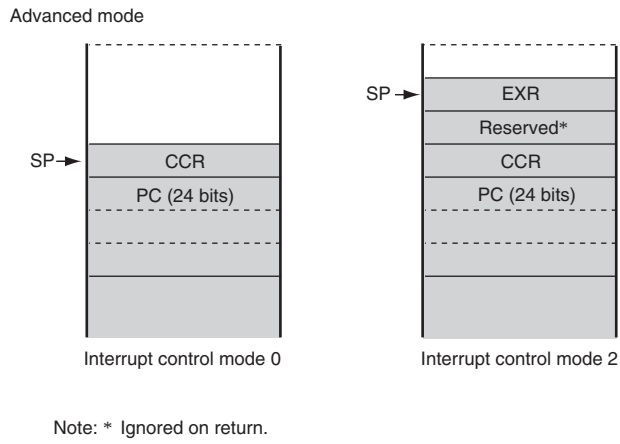


Figure 4.2 Stack Status after Exception Handling

4.9 Usage Note

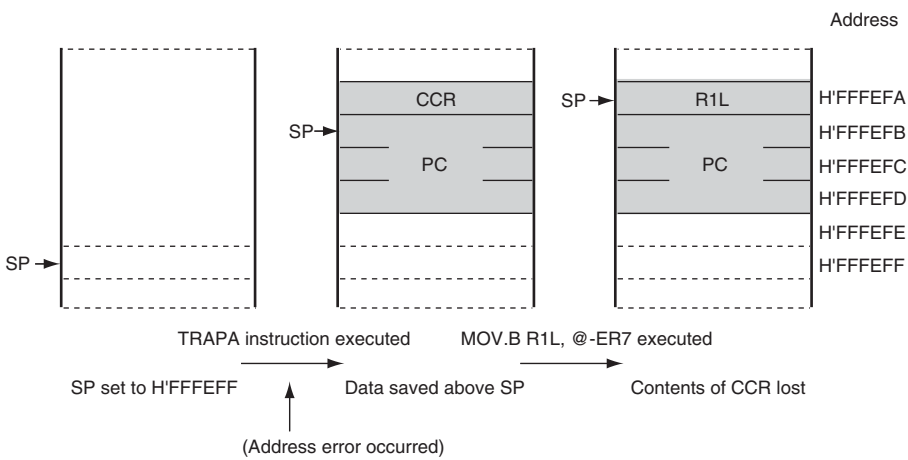
When performing stack-manipulating access, this LSI assumes that the lowest address bit is 0. The stack should always be accessed by a word transfer instruction or a longword transfer instruction, and the value of the stack pointer (SP: ER7) should always be kept even. Use the following instructions to save registers:

- PUSH.W Rn (or MOV.W Rn, @-SP)
- PUSH.L ERn (or MOV.L ERn, @-SP)

Use the following instructions to restore registers:

- POP.W Rn (or MOV.W @SP+, Rn)
- POP.L ERn (or MOV.L @SP+, ERn)

Performing stack manipulation while SP is set to an odd value leads to an address error. Figure 4.3 shows an example of operation when the SP value is odd.



[Legend]
CCR : Condition code register
PC : Program counter
R1L : General register R1L
SP : Stack pointer

Note: This diagram illustrates an example in which the interrupt control mode is 0, in advanced mode.

Figure 4.3 Operation when SP Value Is Odd

Section 5 Interrupt Controller

5.1 Features

- Two interrupt control modes

  Any of two interrupt control modes can be set by means of bits INTM1 and INTM0 in the interrupt control register (INTCR).
- Priority can be assigned by the interrupt priority register (IPR)

  IPR provides for setting interrupt priory. Eight levels can be set for each module for all interrupts except for the interrupt requests listed below. The following six interrupt requests are given priority of 8, therefore they are accepted at all times.

  — NMI

  — General illegal instructions

  — Trace

  — Trap instructions

  — CPU address error

  — DMA address error*
- Independent vector addresses

  All interrupt sources are assigned independent vector addresses, making it unnecessary for the source to be identified in the interrupt handling routine.
- 16 external interrupts

  NMI is the highest-priority interrupt, and is accepted at all times. Rising edge or falling edge detection can be selected for NMI. Falling edge, rising edge, or both edge detection, or level sensing, can be selected for $\overline{\text{IRQ14}}$ to $\overline{\text{IRQ0}}$.
- DMAC control

  DMAC can be activated by means of interrupts.
- CPU priority control function

  The priority levels can be assigned to the CPU and DMAC. The priority level of the CPU can be automatically assigned on an exception generation. Priority can be given to the CPU interrupt exception handling over that of the DMAC transfer.

Note: * A DMA address error is generated within the DMAC.

A block diagram of the interrupt controller is shown in figure 5.1.

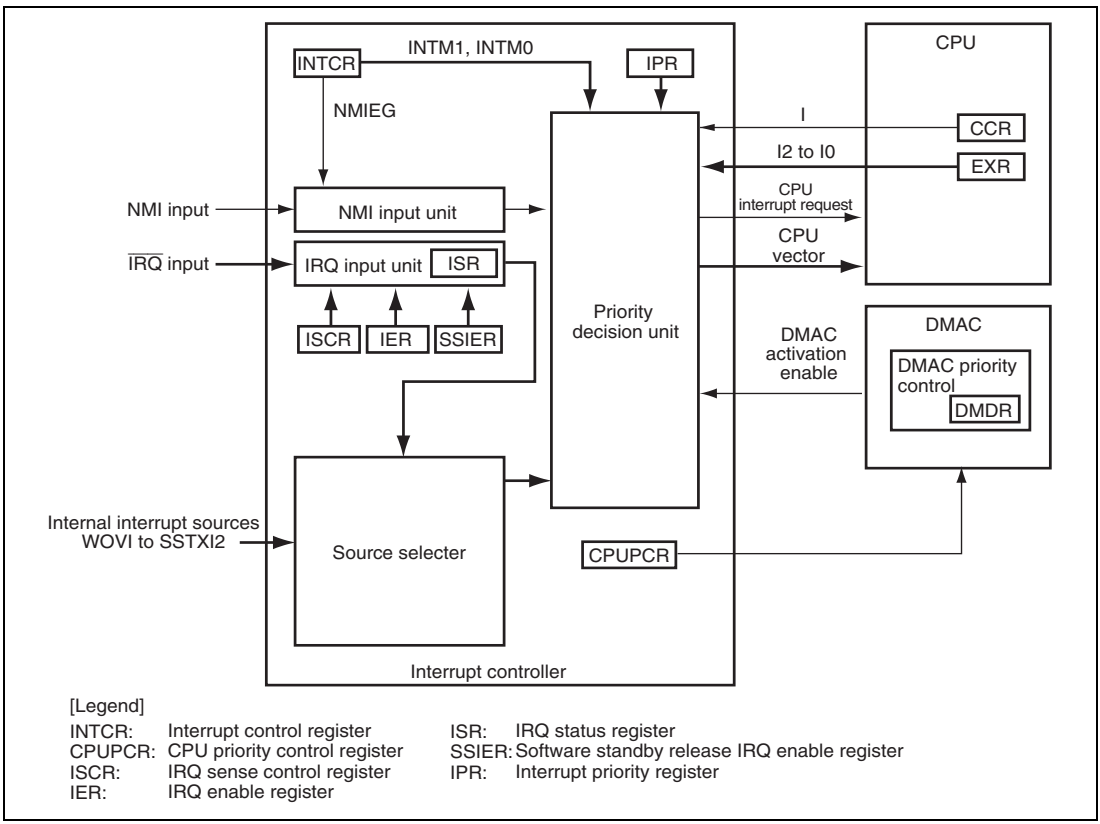


Figure 5.1 Block Diagram of Interrupt Controller

5.2 Input/Output Pins

Table 5.1 shows the pin configuration of the interrupt controller.

Table 5.1 Pin Configuration

Name	I/O	Function
NMI	Input	Nonmaskable External Interrupt Rising or falling edge can be selected.
$\overline{\text{IRQ14}}$ to $\overline{\text{IRQ0}}$	Input	Maskable External Interrupts Rising, falling, or both edges, or level sensing, can be selected.

5.3 Register Descriptions

The interrupt controller has the following registers.

- Interrupt control register (INTCR)
- CPU priority control register (CPUPCR)
- Interrupt priority registers A to G, I, K to O, Q, and R (IPRA to IPRG, IPRI, IPRK to IPRO, IPRQ, and IPRR)
- IRQ enable register (IER)
- IRQ sense control registers H and L (ISCRH, ISCRL)
- IRQ status register (ISR)
- Software standby release IRQ enable register (SSIER)

5.3.1 Interrupt Control Register (INTCR)

INTCR selects the interrupt control mode, and the detected edge for NMI.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	INTM1	INTM0	NMIEG	—	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R	R	R

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	INTM1	0	R/W	Interrupt Control Select Mode 1 and 0
4	INTM0	0	R/W	These bits select either of two interrupt control modes for the interrupt controller. 00: Interrupt control mode 0 Interrupts are controlled by I bit in CCR. 01: Setting prohibited. 10: Interrupt control mode 2 Interrupts are controlled by bits I2 to I0 in EXR, and IPR. 11: Setting prohibited.

Bit	Bit Name	Initial Value	R/W	Description
3	NMIEG	0	R/W	NMI Edge Select Selects the input edge for the NMI pin. 0: Interrupt request generated at falling edge of NMI input 1: Interrupt request generated at rising edge of NMI input
2 to 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

5.3.2 CPU Priority Control Register (CPUPCR)

CPUPCR sets whether or not the CPU has priority over the DMAC. The interrupt exception handling by the CPU can be given priority over that of the DMAC transfer. The priority level of the DMAC for each channel is set by the DMAC control register.

Bit	7	6	5	4	3	2	1	0
Bit Name	CPUPCE	—	—	—	IPSETE	CPUP2	CPUP1	CPUP0
Initial Value	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)*

Note: * When the IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7	CPUPCE	0	R/W	CPU Priority Control Enable Controls the CPU priority control function. Setting this bit to 1 enables the CPU priority control over the DMAC. 0: CPU always has the lowest priority 1: CPU priority control enabled
6 to 4	—	All 0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
3	IPSETE	0	R/W	Interrupt Priority Set Enable Controls the function which automatically assigns the interrupt priority level of the CPU. Setting this bit to 1 automatically sets bits CPUP2 to CPUP0 by the CPU interrupt mask bit (I bit in CCR or bits I2 to I0 in EXR). 0: Bits CPUP2 to CPUP0 are not updated automatically 1: The interrupt mask bit value is reflected in bits CPUP2 to CPUP0
2	CPUP2	0	R/(W)*	CPU Priority Level 2 to 0 These bits set the CPU priority level. When the CPUPCE is set to 1, the CPU priority control function over the DMAC becomes valid and the priority of CPU processing is assigned in accordance with the settings of bits CPUP2 to CPUP0. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
1	CPUP1	0	R/(W)*	
0	CPUP0	0	R/(W)*	

Note: * When the IPSETE bit is set to 1, the CPU priority is automatically updated, so these bits cannot be modified.

5.3.3 Interrupt Priority Registers A to G, I, K to O, Q, and R (IPRA to IPRG, IPRI, IPRK to IPRO, IPRQ, and IPRR)

IPR sets priory (levels 7 to 0) for interrupts other than NMI.

Setting a value in the range from B'000 to B'111 in the 3-bit groups of bits 14 to 12, 10 to 8, 6 to 4, and 2 to 0 assigns a priority level to the corresponding interrupt. For the correspondence between the interrupt sources and the IPR settings, see table 5.2.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IPR14	IPR13	IPR12	—	IPR10	IPR9	IPR8
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	IPR6	IPR5	IPR4	—	IPR2	IPR1	IPR0
Initial Value	0	1	1	1	0	1	1	1
R/W	R	R/W	R/W	R/W	R	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R	Reserved This is a read-only bit and cannot be modified.
14 13 12	IPR14 IPR13 IPR12	1 1 1	R/W R/W R/W	Sets the priority level of the corresponding interrupt source. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
11	—	0	R	Reserved This is a read-only bit and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
10 9 8	IPR10 IPR9 IPR8	1 1 1	R/W R/W R/W	Sets the priority level of the corresponding interrupt source. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
7	—	0	R	Reserved This is a read-only bit and cannot be modified.
6 5 4	IPR6 IPR5 IPR4	1 1 1	R/W R/W R/W	Sets the priority level of the corresponding interrupt source. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)
3	—	0	R	Reserved This is a read-only bit and cannot be modified.
2 1 0	IPR2 IPR1 IPR0	1 1 1	R/W R/W R/W	Sets the priority level of the corresponding interrupt source. 000: Priority level 0 (lowest) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (highest)

5.3.4 IRQ Enable Register (IER)

IER enables or disables interrupt requests IRQ14 to IRQ0.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
14	IRQ14E	0	R/W	IRQ14 Enable The IRQ14 interrupt request is enabled when this bit is 1.
13	IRQ13E	0	R/W	IRQ13 Enable The IRQ13 interrupt request is enabled when this bit is 1.
12	IRQ12E	0	R/W	IRQ12 Enable The IRQ12 interrupt request is enabled when this bit is 1.
11	IRQ11E	0	R/W	IRQ11 Enable The IRQ11 interrupt request is enabled when this bit is 1.
10	IRQ10E	0	R/W	IRQ10 Enable The IRQ10 interrupt request is enabled when this bit is 1.
9	IRQ9E	0	R/W	IRQ9 Enable The IRQ9 interrupt request is enabled when this bit is 1.

Bit	Bit Name	Initial Value	R/W	Description
8	IRQ8E	0	R/W	IRQ8 Enable The IRQ8 interrupt request is enabled when this bit is 1.
7	IRQ7E	0	R/W	IRQ7 Enable The IRQ7 interrupt request is enabled when this bit is 1.
6	IRQ6E	0	R/W	IRQ6 Enable The IRQ6 interrupt request is enabled when this bit is 1.
5	IRQ5E	0	R/W	IRQ5 Enable The IRQ5 interrupt request is enabled when this bit is 1.
4	IRQ4E	0	R/W	IRQ4 Enable The IRQ4 interrupt request is enabled when this bit is 1.
3	IRQ3E	0	R/W	IRQ3 Enable The IRQ3 interrupt request is enabled when this bit is 1.
2	IRQ2E	0	R/W	IRQ2 Enable The IRQ2 interrupt request is enabled when this bit is 1.
1	IRQ1E	0	R/W	IRQ1 Enable The IRQ1 interrupt request is enabled when this bit is 1.
0	IRQ0E	0	R/W	IRQ0 Enable The IRQ0 interrupt request is enabled when this bit is 1.

5.3.5 IRQ Sense Control Registers H and L (ISCRH and ISCRL)

ISCRH and ISCRL select the source that generates an interrupt request on pins $\overline{\text{IRQ14}}$ to $\overline{\text{IRQ0}}$.

Upon changing the setting of ISCR, IRQnF (n = 14 to 0) in ISR is often set to 1 accidentally through an internal operation. In this case, an interrupt exception handling is executed if an IRQn interrupt request is enabled. In order to prevent such an accidental interrupt from occurring, the setting of ISCR should be changed while the IRQn interrupt is disabled, and then the IRQnF in ISR should be cleared to 0.

- ISCRH

Bit	15	14	13	12	11	10	9	8
Bit Name	—	—	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF
Initial Value	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

- ISCRL

Bit	15	14	13	12	11	10	9	8
Bit Name	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF
Initial Value	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

- ISCRH

Bit	Bit Name	Initial Value	R/W	Description
15, 14	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
13	IRQ14SR	0	R/W	IRQ14 Sense Control Rise
12	IRQ14SF	0	R/W	IRQ14 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ14}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ14}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ14}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ14}}$
11	IRQ13SR	0	R/W	IRQ13 Sense Control Rise
10	IRQ13SF	0	R/W	IRQ13 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ13}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ13}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ13}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ13}}$
9	IRQ12SR	0	R/W	IRQ12 Sense Control Rise
8	IRQ12SF	0	R/W	IRQ12 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ12}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ12}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ12}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ12}}$
7	IRQ11SR	0	R/W	IRQ11 Sense Control Rise
6	IRQ11SF	0	R/W	IRQ11 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ11}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ11}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ11}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ11}}$

Bit	Bit Name	Initial Value	R/W	Description
5	IRQ10SR	0	R/W	IRQ10 Sense Control Rise
4	IRQ10SF	0	R/W	IRQ10 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ10}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ10}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ10}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ10}}$
3	IRQ9SR	0	R/W	IRQ9 Sense Control Rise
2	IRQ9SF	0	R/W	IRQ9 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ9}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ9}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ9}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ9}}$
1	IRQ8SR	0	R/W	IRQ8 Sense Control Rise
0	IRQ8SF	0	R/W	IRQ8 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ8}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ8}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ8}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ8}}$

- ISCRL

Bit	Bit Name	Initial Value	R/W	Description
15	IRQ7SR	0	R/W	IRQ7 Sense Control Rise
14	IRQ7SF	0	R/W	IRQ7 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ7}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ7}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ7}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ7}}$
13	IRQ6SR	0	R/W	IRQ6 Sense Control Rise
12	IRQ6SF	0	R/W	IRQ6 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ6}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ6}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ6}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ6}}$
11	IRQ5SR	0	R/W	IRQ5 Sense Control Rise
10	IRQ5SF	0	R/W	IRQ5 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ5}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ5}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ5}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ5}}$
9	IRQ4SR	0	R/W	IRQ4 Sense Control Rise
8	IRQ4SF	0	R/W	IRQ4 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ4}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ4}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ4}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ4}}$

Bit	Bit Name	Initial Value	R/W	Description
7	IRQ3SR	0	R/W	IRQ3 Sense Control Rise
6	IRQ3SF	0	R/W	IRQ3 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ3}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ3}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ3}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ3}}$
5	IRQ2SR	0	R/W	IRQ2 Sense Control Rise
4	IRQ2SF	0	R/W	IRQ2 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ2}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ2}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ2}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ2}}$
3	IRQ1SR	0	R/W	IRQ1 Sense Control Rise
2	IRQ1SF	0	R/W	IRQ1 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ1}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ1}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ1}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ1}}$
1	IRQ0SR	0	R/W	IRQ0 Sense Control Rise
0	IRQ0SF	0	R/W	IRQ0 Sense Control Fall 00: Interrupt request generated by low level of $\overline{\text{IRQ0}}$ 01: Interrupt request generated at falling edge of $\overline{\text{IRQ0}}$ 10: Interrupt request generated at rising edge of $\overline{\text{IRQ0}}$ 11: Interrupt request generated at both falling and rising edges of $\overline{\text{IRQ0}}$

5.3.6 IRQ Status Register (ISR)

ISR is an IRQ14 to IRQ0 interrupt request register.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F
Initial Value	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)*

Bit	7	6	5	4	3	2	1	0
Bit Name	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F
Initial Value	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)*

Note: * Only 0 can be written, to clear the flag. The bit manipulation instructions or memory operation instructions should be used to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R/(W)*	Reserved This bit is always read as 0. The write value should always be 0.
14	IRQ14F	0	R/(W)*	[Setting condition] • When the interrupt selected by ISCR occurs [Clearing conditions] • Writing 0 after reading IRQnF = 1 • When interrupt exception handling is executed when low-level sensing is selected and $\overline{\text{IRQn}}$ input is high • When IRQn interrupt exception handling is executed when falling-, rising-, or both-edge sensing is selected
13	IRQ13F	0	R/(W)*	
12	IRQ12F	0	R/(W)*	
11	IRQ11F	0	R/(W)*	
10	IRQ10F	0	R/(W)*	
9	IRQ9F	0	R/(W)*	
8	IRQ8F	0	R/(W)*	
7	IRQ7F	0	R/(W)*	
6	IRQ6F	0	R/(W)*	
5	IRQ5F	0	R/(W)*	
4	IRQ4F	0	R/(W)*	
3	IRQ3F	0	R/(W)*	
2	IRQ2F	0	R/(W)*	
1	IRQ1F	0	R/(W)*	
0	IRQ0F	0	R/(W)*	

Note: * Only 0 can be written, to clear the flag. The bit manipulation instructions or memory operation instructions should be used to clear the flag.

5.3.7 Software Standby Release IRQ Enable Register (SSIER)

SSIER selects pins used to leave software standby mode from pins $\overline{\text{IRQ14}}$ to $\overline{\text{IRQ0}}$.

Bit	15	14	13	12	11	10	9	8
Bit Name	—	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
15	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
14	SSI14	0	R/W	Software Standby Release IRQ Setting These bits select the $\overline{\text{IRQn}}$ pins used to leave software standby mode (n = 14 to 0). 0: IRQn requests are not sampled in software standby mode 1: When an IRQn request occurs in software standby mode, this LSI leaves software standby mode after the oscillation settling time has elapsed
13	SSI13	0	R/W	
12	SSI12	0	R/W	
11	SSI11	0	R/W	
10	SSI10	0	R/W	
9	SSI9	0	R/W	
8	SSI8	0	R/W	
7	SSI7	0	R/W	
6	SSI6	0	R/W	
5	SSI5	0	R/W	
4	SSI4	0	R/W	
3	SSI3	0	R/W	
2	SSI2	0	R/W	
1	SSI1	0	R/W	
0	SSI0	0	R/W	

5.4 Interrupt Sources

5.4.1 External Interrupts

There are sixteen external interrupts: NMI and IRQ14 to IRQ0. These interrupts can be used to leave software standby mode.

(1) NMI Interrupts:

Nonmaskable interrupt request (NMI) is the highest-priority interrupt, and is always accepted by the CPU regardless of the interrupt control mode or the settings of the CPU interrupt mask bits. The NMIEG bit in INTCR selects whether an interrupt is requested at the rising or falling edge on the NMI pin.

When an NMI interrupt is generated, the interrupt controller determines that an error has occurred, and performs the following procedure.

- Sets the ERRF bit in DMDR_0 to 1.
- Clears the DTE bits for all the channels of the DMAC and forcibly halts transfer.

(2) IRQn Interrupts:

An IRQn interrupt is requested by a signal input on pins $\overline{\text{IRQ14}}$ to $\overline{\text{IRQ0}}$. $\overline{\text{IRQn}}$ (n = 14 to 0) have the following features:

- Using ISCR, it is possible to select whether an interrupt is generated by a low level, falling edge, rising edge, or both edges, on pins $\overline{\text{IRQn}}$.
- Enabling or disabling of interrupt requests IRQn can be selected by IER.
- The interrupt priority can be set by IPR.
- The status of interrupt requests IRQn is indicated in ISR. ISR flags can be cleared to 0 by software. The bit manipulation instructions or memory operation instructions should be used to clear the flag in ISR.

Detection of IRQn interrupts is enabled through the P1ICR, P2ICR, P5ICR, and P6ICR register settings, and does not change regardless of the output setting. However, when a pin is used as an external interrupt input pin, the pin must not be used as an I/O pin for another function by clearing the corresponding DDR bit to 0.

A block diagram of interrupts IRQn is shown in figure 5.2.

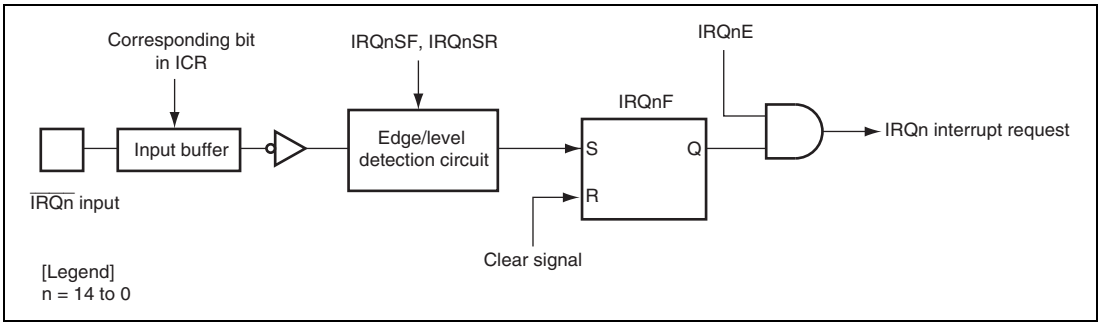


Figure 5.2 Block Diagram of Interrupts IRQn

When the IRQ sensing control in ISCR is set to a low level of signal $\overline{\text{IRQn}}$, the level of $\overline{\text{IRQn}}$ should be held low until an interrupt handling starts. Then set the corresponding input signal $\overline{\text{IRQn}}$ to high in the interrupt handling routine and clear the IRQnF to 0. Interrupts may not be executed when the corresponding input signal $\overline{\text{IRQn}}$ is set to high before the interrupt handling begins.

5.4.2 Internal Interrupts

The sources for internal interrupts from on-chip peripheral modules have the following features:

- For each on-chip peripheral module there are flags that indicate the interrupt request status, and enable bits that enable or disable these interrupts. They can be controlled independently. When the enable bit is set to 1, an interrupt request is issued to the interrupt controller.
- The interrupt priority can be set by means of IPR.
- The DMAC can be activated by a TPU, SCI, RCAN-ET, SSU, or other interrupt request.
- DMAC activation can be controlled by the CPU priority control function over the DMAC.

5.5 Interrupt Exception Handling Vector Table

Table 5.2 lists interrupt exception handling sources, vector address offsets, and interrupt priority.

In the default priority order, a lower vector number corresponds to a higher priority. When interrupt control mode 2 is set, priority levels can be changed by setting the IPR contents. The priority for interrupt sources allocated to the same level in IPR follows the default priority, that is, they are fixed.

Table 5.2 Interrupt Sources, Vector Address Offsets, and Interrupt Priority

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
External pin	NMI	7	H'001C	—	High	—
	IRQ0	64	H'0100	IPRA14 to IPRA12	↑	—
	IRQ1	65	H'0104	IPRA10 to IPRA8		—
	IRQ2	66	H'0108	IPRA6 to IPRA4		—
	IRQ3	67	H'010C	IPRA2 to IPRA0		—
	IRQ4	68	H'0110	IPRB14 to IPRB12		—
	IRQ5	69	H'0114	IPRB10 to IPRB8		—
	IRQ6	70	H'0118	IPRB6 to IPRB4		—
	IRQ7	71	H'011C	IPRB2 to IPRB0		—
	IRQ8	72	H'0120	IPRC14 to IPRC12		—
	IRQ9	73	H'0124	IPRC10 to IPRC8		—
	IRQ10	74	H'0128	IPRC6 to IPRC4		—
	IRQ11	75	H'012C	IPRC2 to IPRC0		—
	IRQ12	76	H'0130	IPRD14 to IPRE12		—
	IRQ13	77	H'0134	IPRD10 to IPRD8		—
	IRQ14	78	H'0138	IPRD6 to IPRD4		—
—	Reserved for system use	79	H'013C	—		—
		80	H'0140			—
WDT	WOVI	81	H'0144	IPRE10 to IPRE8		—
—	Reserved for system use	82	H'0148	—		—
		83	H'014C			—
		84	H'0150			—
		85	H'0154		Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
A/D_0	ADI0	86	H'0158	IPRF10 to IPRF8	High	O
A/D_1	ADI1	87	H'015C		↑	O
TPU_0*[2]	TGI0A	88	H'0160	IPRF6 to IPRF4		O
	TGI0B	89	H'0164			—
	TGI0C	90	H'0168			—
	TGI0D	91	H'016C			—
	TCI0V	92	H'0170			—
TPU_1*[2]	TGI1A	93	H'0174	IPRF2 to IPRF0		O
	TGI1B	94	H'0178			—
	TCI1V	95	H'017C			—
	TCI1U	96	H'0180			—
TPU_2*[2]	TGI2A	97	H'0184	IPRG14 to IPRG12		O
	TGI2B	98	H'0188			—
	TCI2V	99	H'018C			—
	TCI2U	100	H'0190			—
TPU_3*[2]	TGI3A	101	H'0194	IPRG10 to IPRG8		O
	TGI3B	102	H'0198			—
	TGI3C	103	H'019C			—
	TGI3D	104	H'01A0			—
	TCI3V	105	H'01A4			—
TPU_4*[2]	TGI4A	106	H'01A8	IPRG6 to IPRG4		—
	TGI4B	107	H'01AC			—
	TCI4V	108	H'01B0			—
	TCI4U	109	H'01B4			—
TPU_5*[2]	TGI5A	110	H'01B8	IPRG2 to IPRG0		—
	TGI5B	111	H'01BC			—
	TCI5V	112	H'01C0			—
	TCI5U	113	H'01C4			—
—	Reserved for system use	114	H'01C8			—
		115	H'01CC			—
		116	H'01D0			—
		117	H'01D4			—
		118	H'01D8		Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
—	Reserved for system use	119	H'01DC		High	—
		120	H'01E0			—
		121	H'01E4			—
		122	H'01E8			—
		123	H'01EC			—
		124	H'01F0			—
		125	H'01F4			—
		126	H'01F8			—
		127	H'01FC			—
DMAC	DMTEND0	128	H'0200	IPRI14 to IPRI12		—
	DMTEND1	129	H'0204	IPRI10 to IPRI8		—
	DNTEND2	130	H'0208	IPRI6 to IPRI4		—
	DMTEND3	131	H'020C	IPRI2 to IPRI0		—
—	Reserved for system use	132	H'0210	—		—
		133	H'0214			—
		134	H'0218			—
		135	H'021C			—
DMAC	DMEEND0	136	H'0220	IPRK14 to IPRK12		—
	DMEEND1	137	H'0224			—
	DNEEND2	138	H'0228			—
	DMEEND3	139	H'022C			—
—	Reserved for system use	140	H'0230	—		—
		141	H'0234			—
		142	H'0238			—
		143	H'023C			—
		144	H'0240			—
		145	H'0244			—
		146	H'0248			—
		147	H'024C			—
		148	H'0250			—
		149	H'0254			—
		150	H'0258			—
		151	H'025C		Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
—	Reserved for system use	152	H'0260	—	High	—
		153	H'0264			—
		154	H'0268			—
		155	H'026C			—
SCI_3	ERI3	156	H'0270	IPRL10 to IPRL8		—
	RXI3	157	H'0274			O
	TXI3	158	H'0278			O
	TEI3	159	H'027C			—
SCI_4	ERI4	160	H'0280	IPRL6 to IPRL4		—
	RXI4	161	H'0284			O
	TXI4	162	H'0288			O
	TEI4	163	H'028C			—
TPU_6	TGI6A	164	H'0290	IPRL2 to IPRL0		O
	TGI6B	165	H'0294			—
	TGI6C	166	H'0298			—
	TGI6D	167	H'029C			—
	TCI6V	168	H'02A0	IPRM14 to IPRM12		—
TPU_7	TGI7A	169	H'02A4	IPRM10 to IPRM8		O
	TGI7B	170	H'02A8			—
	TCI7V	171	H'02AC	IPRM6 to IPRM4		—
	TCI7U	172	H'02B0			—
TPU_8	TGI8A	173	H'02B4	IPRM2 to IPRM0		O
	TGI8B	174	H'02B8			—
	TCI8V	175	H'02BC	IPRN14 to IPRN12		—
	TCI8U	176	H'02C0			—
TPU_9	TGI9A	177	H'02C4	IPRN10 to IPRN8		O
	TGI9B	178	H'02C8			—
	TGI9C	179	H'02CC			—
	TGI9D	180	H'02D0			—
	TCI9V	181	H'02D4	IPRN6 to IPRN4	Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
TPU_10	TGI10A	182	H'02D8	IPRN2 to IPRN0	High	O
	TGI10B	183	H'02DC			—
	Reserved for system use	184	H'02E0			—
	Reserved for system use	185	H'02E4			—
	TCI10V	186	H'02E8	IPRO14 to IPRO12		—
	TCI10U	187	H'02EC			—
TPU_11	TGI11A	188	H'02F0	IPRO10 to IPRO8		O
	TGI11B	189	H'02F4			—
	TCI11V	190	H'02F8	IPRO6 to IPRO4		—
	TCI11U	191	H'02FC			—
—	Reserved for system use	192	H'0300	—		—
		193	H'0304			—
		194	H'0308			—
		195	H'030C			—
		196	H'0310			—
		197	H'0314			—
		198	H'0318			—
		199	H'031C			—
		200	H'0320			—
		201	H'0324			—
		202	H'0328			—
		203	H'032C			—
		204	H'0330			—
		205	H'0334			—
		206	H'0338			—
		207	H'033C			—
		208	H'0340			—
		209	H'0344			—
		210	H'0348			—
		211	H'034C			—
		212	H'0350			—
		213	H'0354		Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
—	Reserved for system use	214	H'0358	—	High	—
		215	H'035C			—
		216	H'0360			—
		217	H'0364			—
		218	H'0368			—
		219	H'036C			—
RCAN-ET	ERS_0/OVR_0	220	H'0370	IPRQ2 to IPRQ0		—
	RM0_0	221	H'0374			O
	RM1_0	222	H'0378			—
	SLE_0	223	H'037C			—
SSU_0	Reserved for system use	224	H'0380	IPRR14 to IPRR12		—
		225	H'0384			—
		226	H'0388			—
	SSERI0	227	H'038C			—
	SSRXI0	228	H'0390	IPRR10 to IPRR8		O
	SSTXI0	229	H'0394			O
SSU_1	Reserved for system use	230	H'0398			—
	SSERI1	231	H'039C			—
	SSRXI1	232	H'03A0	IPRR6 to IPRR4		O
	SSTXI1	233	H'03A4			O
SSU_2	Reserved for system use	234	H'03A8			—
	SSERI2	235	H'03AC			—
	SSRXI2	236	H'03B0	IPRR2 to IPRR0		O
	SSTXI2	237	H'03B4			O
	Reserved for system use	238	H'03B8			—
		239	H'03BC			—
—	Reserved for system use	240	H'03C0	—		—
		241	H'03C4			—
		242	H'03C8			—
		243	H'03CC			—
		244	H'03D0			—
		245	H'03D4		Low	—

Classification	Interrupt Source	Vector Number	Vector Address Offset*[1]	IPR	Priority	DMAC Activation
—	Reserved for system use	246	H'03D8	—	High	—
		247	H'03DC		↑	—
		248	H'03E0			—
		249	H'03E4			—
		250	H'03E8			—
		251	H'03EC			—
		252	H'03F0			—
		253	H'03F4			—
		254	H'03F8			—
		255	H'03FC		Low	—

Note: 1. Lower 16 bits of the start address in advanced mode.
2. Supported only by the H8SX/1527R.

5.6 Interrupt Control Modes and Interrupt Operation

The interrupt controller has two interrupt control modes: interrupt control mode 0 and interrupt control mode 2. Interrupt operations differ depending on the interrupt control mode. The interrupt control mode is selected by INTCR. Table 5.3 shows the differences between interrupt control mode 0 and interrupt control mode 2.

Table 5.3 Interrupt Control Modes

Interrupt Control Mode	Priority Setting Register	Interrupt Mask Bit	Description
0	Default	I	The priority levels of the interrupt sources are fixed default settings. The interrupts except for NMI is masked by the I bit.
2	IPR	I2 to I0	Eight priority levels can be set for interrupt sources except for NMI with IPR. 8-level interrupt mask control is performed by bits I2 to I0.

5.6.1 Interrupt Control Mode 0

In interrupt control mode 0, interrupt requests except for NMI are masked by the I bit in CCR of the CPU. Figure 5.3 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, the interrupt request is sent to the interrupt controller.
2. If the I bit in CCR is set to 1, only an NMI interrupt is accepted, and other interrupt requests are held pending. If the I bit is cleared to 0, an interrupt request is accepted.
3. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority, sends the request to the CPU, and holds other interrupt requests pending.
4. When the CPU accepts the interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC and CCR contents are saved to the stack area during the interrupt exception handling. The PC contents saved on the stack is the address of the first instruction to be executed after returning from the interrupt handling routine.
6. Next, the I bit in CCR is set to 1. This masks all interrupts except NMI.

7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

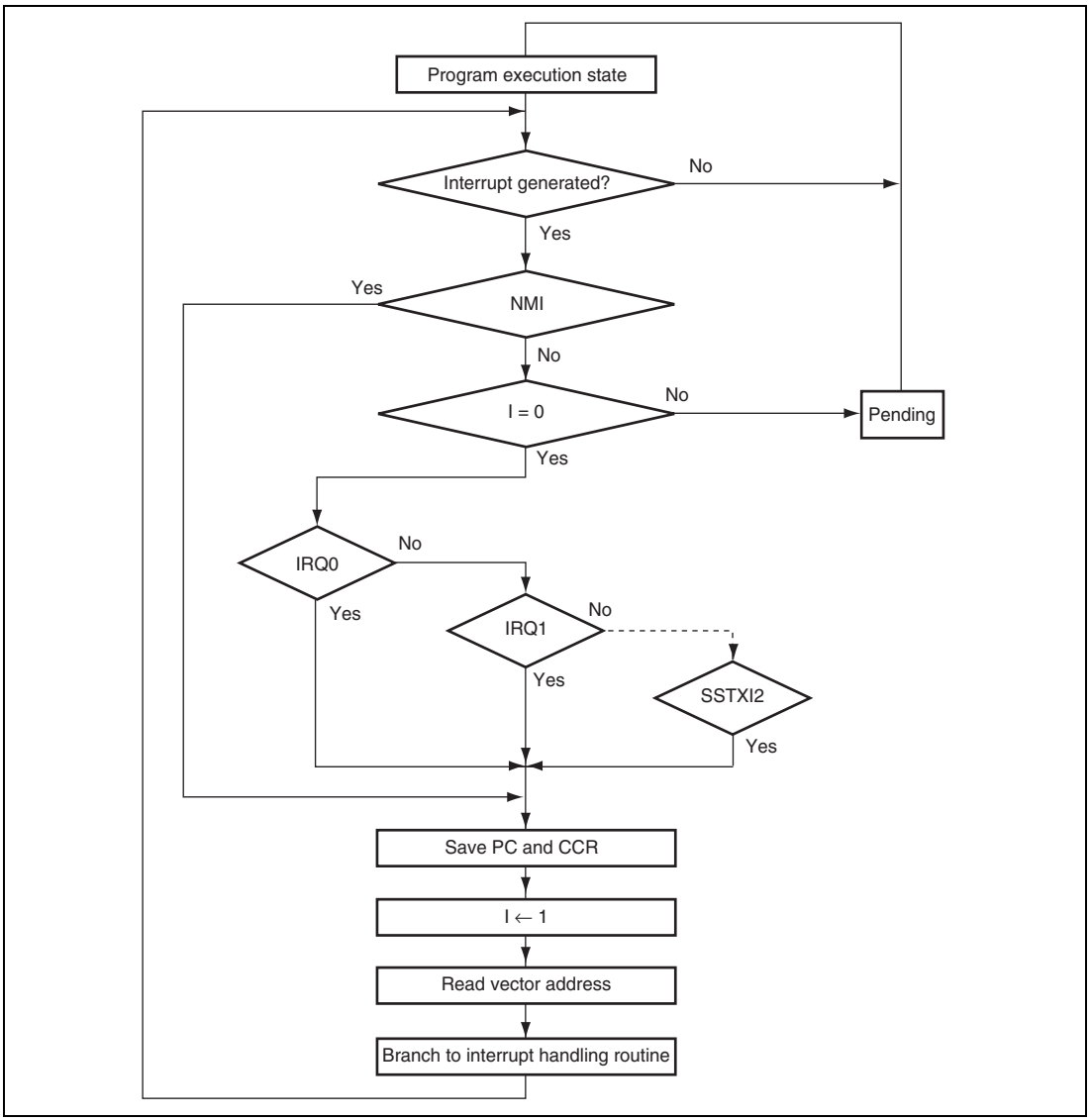


Figure 5.3 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 0

5.6.2 Interrupt Control Mode 2

In interrupt control mode 2, interrupt requests except for NMI are masked by comparing the interrupt mask level (I2 to I0 bits) in EXR of the CPU and the IPR setting. There are eight levels in mask control. Figure 5.4 shows a flowchart of the interrupt acceptance operation in this case.

1. If an interrupt request occurs when the corresponding interrupt enable bit is set to 1, an interrupt request is sent to the interrupt controller.
2. For multiple interrupt requests, the interrupt controller selects the interrupt request with the highest priority according to the IPR setting, and holds other interrupt requests pending. If multiple interrupt requests has the same priority, an interrupt request is selected according to the default setting shown in table 5.2.
3. Next, the priority of the selected interrupt request is compared with the interrupt mask level set in EXR. When the interrupt request does not have priority over the mask level set, it is held pending, and only an interrupt request with a priority over the interrupt mask level is accepted.
4. When the CPU accepts an interrupt request, it starts interrupt exception handling after execution of the current instruction has been completed.
5. The PC, CCR, and EXR contents are saved to the stack area during interrupt exception handling. The PC saved on the stack is the address of the first instruction to be executed after returning from the interrupt handling routine.
6. The T bit in EXR is cleared to 0. The interrupt mask level is rewritten with the priority of the accepted interrupt. If the accepted interrupt is NMI, the interrupt mask level is set to H'7.
7. The CPU generates a vector address for the accepted interrupt and starts execution of the interrupt handling routine at the address indicated by the contents of the vector address in the vector table.

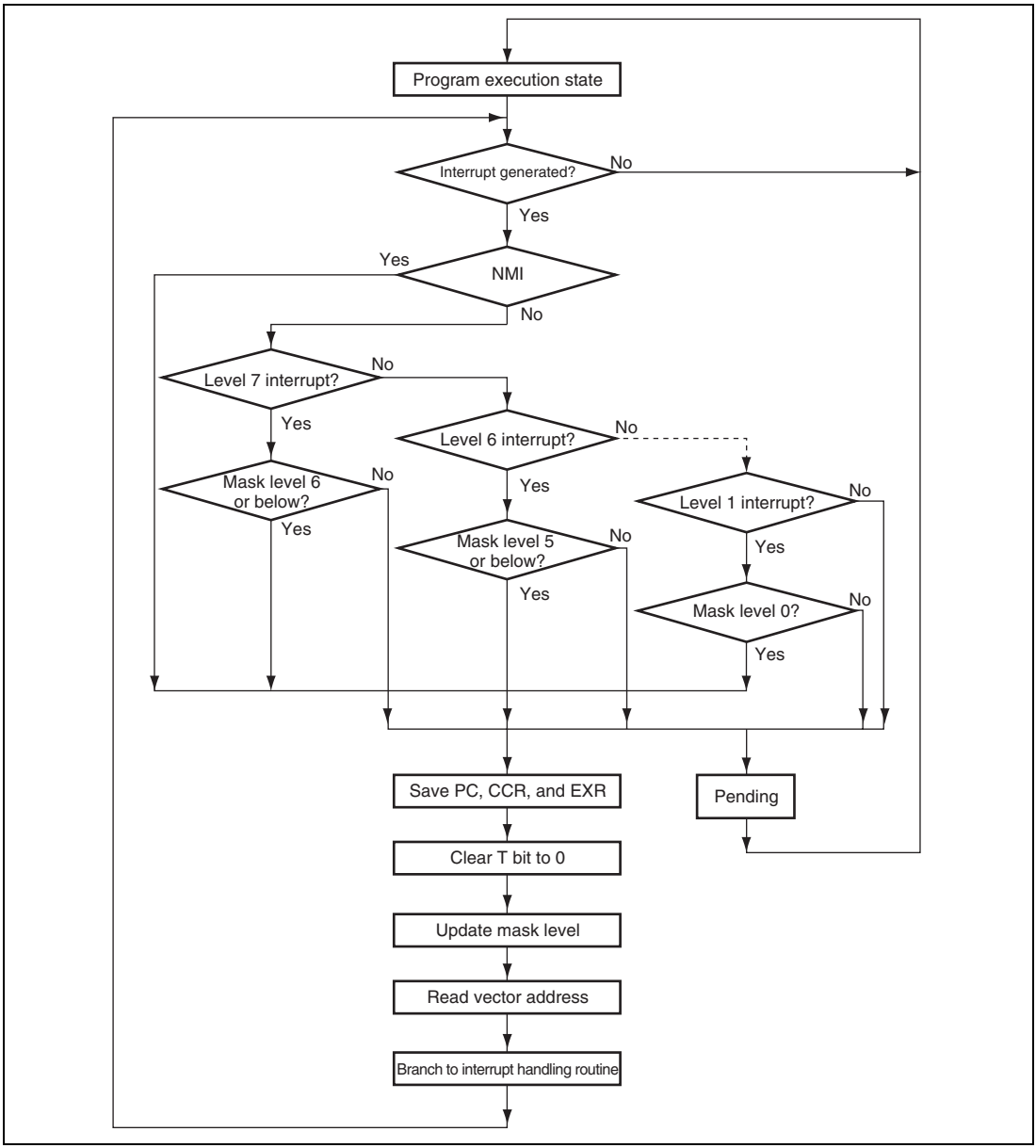


Figure 5.4 Flowchart of Procedure Up to Interrupt Acceptance in Interrupt Control Mode 2

5.6.3 Interrupt Exception Handling Sequence

Figure 5.5 shows the interrupt exception handling sequence. The example is for the case where interrupt control mode 0 is set in maximum mode, and the program area and stack area are in on-chip memory.

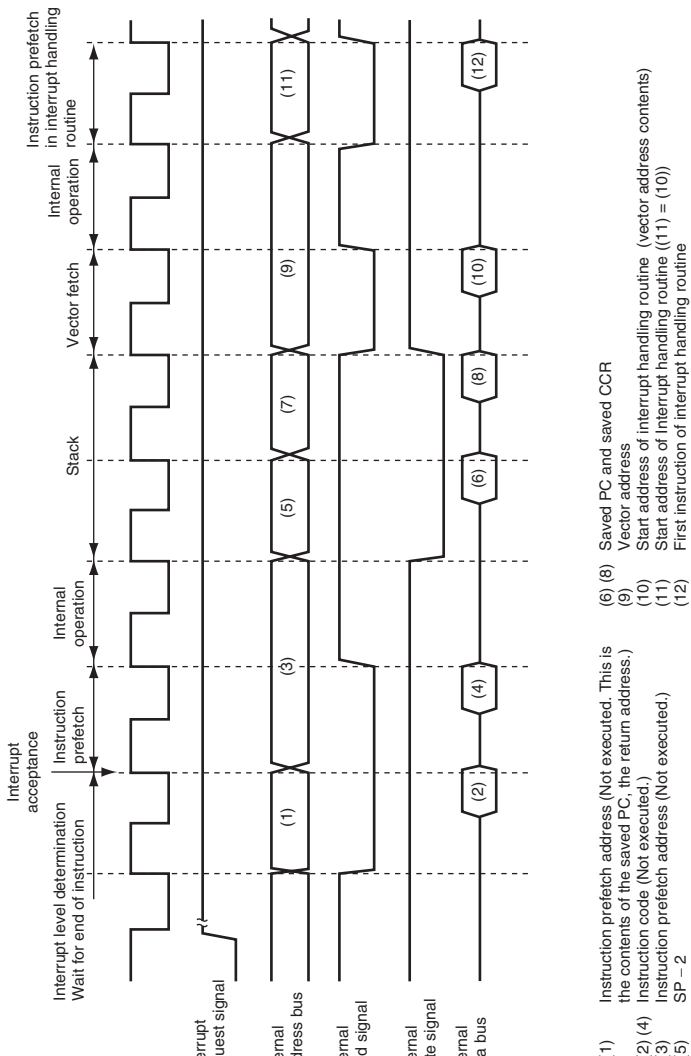


Figure 5.5 Interrupt Exception Handling

5.6.4 Interrupt Response Times

Table 5.4 shows interrupt response times – the interval between generation of an interrupt request and execution of the first instruction in the interrupt handling routine. The symbols for execution states used in table 5.4 are explained in table 5.5.

This LSI is capable of fast word transfer to on-chip memory, so allocating the program area in on-chip ROM and the stack area in on-chip RAM enables high-speed processing.

Table 5.4 Interrupt Response Times

	Normal Mode*[5]		Advanced Mode		Maximum Mode[5]	
Execution State	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**	**Interrupt Control Mode 0**	**Interrupt Control Mode 2**
Interrupt priority decision*[1]	3					
Number of states until executing instruction ends*[2]	$1 \text{ to } 19 + 2 \cdot S_I$					
PC, CCR, EXR stacking	S_K to $2 \cdot S_K$*[6]	$2 \cdot S_K$	S_K to $2 \cdot S_K$*[6]	$2 \cdot S_K$	$2 \cdot S_K$	$2 \cdot S_K$
Vector fetch	S_h					
Instruction fetch*[3]	$2 \cdot S_I$					
Internal processing*[4]	2					
Total (using on-chip memory)	10 to 31	11 to 31	10 to 31	11 to 31	11 to 31	11 to 31

Notes:
1. Two states for an internal interrupt.
2. In the case of the MULXS or DIVXS instruction
3. Prefetch after interrupt acceptance or for an instruction in the interrupt handling routine.
4. Internal operation after interrupt acceptance or after vector fetch
5. Not available in this LSI.
6. When setting the SP value to 4n, the interrupt response time is S_K; when setting to 4n + 2, the interrupt response time is $2 \cdot S_K$.

Table 5.5 Number of Execution States in Interrupt Handling Routine

	Object of Access						
		External Device					
		8-Bit Bus		16-Bit Bus		32-Bit Bus	
Symbol	**On-Chip Memory**	**2-State Access**	**3-State Access**	**2-State Access**	**3-State Access**	**2-State Access**	**3-State Access**
Vector fetch S_h	1	8	12 + 4m	4	6 + 2m	2	3 + m
Instruction fetch S_I	1	4	6 + 2m	2	3 + m	2	3 + m
Stack manipulation S_K	2	8	12 + 4m	4	6 + 2m	2	3 + m

[Legend]
m: Number of wait cycles in an external device access.

5.6.5 DMAC Activation by Interrupt

The DMAC can be activated by an interrupt. In this case, the following options are available:

- Interrupt request to the CPU
- Activation request to the DMAC
- Combination of the above

For details on interrupt requests that can be used to activate the DMAC, see table 5.2 and section 7, DMA Controller (DMAC).

Figure 5.6 shows a block diagram of the DMAC and interrupt controller.

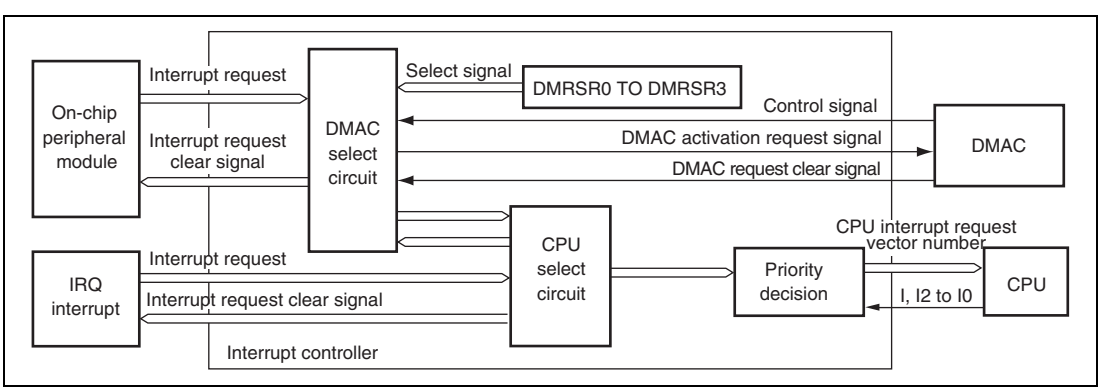


Figure 5.6 Block Diagram of DMAC and Interrupt Controller

(1) Selection of Interrupt Sources:

A DMAC activation request source for each channel is selected by DMRSR. The selected request is passed to the DMAC via a selector. When transfer by an on-chip module interrupt is enabled (DTF1 = 1, DTF0 = 0, and DTE = 1 in DMDR) and the DTA bit in DMDR is set to 1, the interrupt source selected for the DMAC activation source is controlled by the DMAC and cannot be used as a CPU interrupt source.

Interrupt sources that are not controlled by the DMAC are set for CPU interrupt sources.

When the same interrupt source is set as both the DMAC activation source and CPU interrupt source, the DMAC must be given priority over the CPU. If the IPSETE bit in CPUPCR is set to 1, the priority is determined according to the IPR setting. Therefore, the CPUP setting or the IPR setting corresponding to the interrupt source must be set to lower than or equal to the DMAP setting. If the CPU is given priority, the DMAC may not be activated, and the data transfer may not be performed.

(2) Operation Order

If the same interrupt is selected as both the DMAC activation source and CPU interrupt source, the respective operations are performed independently.

Table 5.6 lists the selection of interrupt sources and interrupt source clear control by means of the setting of the DTA bit in DMDR of the DMAC.

Table 5.6 Interrupt Source Selection and Clear Control

Setting		
DMAC	**Interrupt Source Selection/Clear Control**	
DTA	**DMAC**	**CPU**
0	O	√
1	√	X

[Legend]

√: The corresponding interrupt is used. The interrupt source is cleared.
(The interrupt source flag must be cleared in the CPU interrupt handling routine.)

O: The corresponding interrupt is used. The interrupt source is not cleared.

X: The corresponding interrupt is not available.

(3) Usage Note

The interrupt sources of the SCI, A/D converter, RCAN-ET and SSU are cleared according to the setting shown in table 5.6, when the DMAC reads/writes the prescribed register.

5.7 CPU Priority Control Function Over DMAC

The interrupt controller has a function to control the priority among the DMAC and the CPU by assigning priority levels to the DMAC and CPU. Since the priority level can automatically be assigned to the CPU on an interrupt occurrence, it is possible to execute the CPU interrupt exception handling prior to the DMAC transfer.

The priority level of the CPU is assigned by bits CPUP2 to CPUP0 in CPUPCR. The priority level of the DMAC is assigned to each channel by bits DMAP2 to DMAP0 in the DMA mode control registers 0 to 3 (DMDR_0 to DMDR_3).

The priority control function over the DMAC is enabled by setting the CPUPCE bit in CPUPCR to 1. When the CPUPCE bit is 1, the DMAC activation source is controlled according to the respective priority level.

The priority level of the DMAC can be specified for each channel. The DMAC activation source is controlled according to the priority level of the CPU and the priority level of the DMAC indicated by bits DMAP2 to DMAP0. If the CPU has priority, the DMAC activation source is held. The DMAC is activated when the condition by which the activation source is held is cancelled (CPUCPCE = 1 and value of bits CPUP2 to CPUP0 is greater than that of bits DMAP2 to DMAP0). When the different priority levels of the DMAC are assigned for the channels, the channel having higher priority continues to transfer while the channel having lower priority than the CPU is held.

There are two methods for assigning the priority level to the CPU by the IPSETE bit in CPUPCR. Setting the IPSETE bit to 1 enables a function to automatically assign the value of the interrupt mask bit of the CPU to the CPU priority level. Clearing the IPSETE bit to 0 disables the function to automatically assign the priority level. Therefore, the priority level is assigned directly by software rewriting bits CPUP2 to CPUP0. Even if the IPSETE bit is 1, the priority level of the CPU is software assignable by rewriting the interrupt mask bit of the CPU (I bit in CCR or I2 to I0 bits in EXR).

The priority level which is automatically assigned when the IPSETE bit is 1 differs according to the interrupt control mode.

In interrupt control mode 0, the I bit in CCR of the CPU is reflected in bit CPUP2. Bits CPUP1 and CPUP0 are fixed 0. In interrupt control mode 2, the values of bits I2 to I0 in EXR of the CPU are reflected in bits CPUP2 to CPUP0.

Table 5.7 shows the CPU priority control.

Table 5.7 CPU Priority Control

Interrupt Control Mode	Interrupt Priority	Interrupt Mask Bit	IPSETE in CPUPCR	Control Status	
				CPUP2 to CPUP0	Updating of CPUP2 to CPUP0
0	Default	I = any	0	B'111 to B'000	Enabled
		I = 0	1	B'000	Disabled
		I = 1		B'100	
2	IPR setting	I2 to I0	0	B'111 to B'000	Enabled
			1	I2 to I0	Disabled

Table 5.8 shows an setting example of the priority control function over the DMAC and the transfer request control state. Although the DMAC priority levels can be assigned for each channel, table 5.8 gives a single channel description. Thus, transfer for each channel can be performed independently by assigning the different priority levels.

Table 5.8 Example of Priority Control Function Setting and Control State

Interrupt Control Mode	CPUPCE in CPUPCR	CPUP2 to CPUP0	DMAP2 to DMAP0	Transfer Request Control State
				DMAC
0	0	Any	Any	Enabled
	1	B'000	B'000	Enabled
		B'100	B'000	Masked
		B'100	B'011	Masked
		B'100	B'101	Enabled
		B'000	B'101	Enabled
2	0	Any	Any	Enabled
	1	B'000	B'000	Enabled
		B'000	B'101	Enabled
		B'011	B'101	Enabled
		B'100	B'101	Enabled
		B'101	B'101	Enabled
		B'110	B'101	Masked
		B'111	B'101	Masked
		B'101	B'101	Enabled
		B'101	B'101	Enabled

5.8 Usage Notes

5.8.1 Conflict between Interrupt Generation and Disabling

When an interrupt enable bit is cleared to 0 to mask the interrupt, the masking becomes effective after execution of the instruction.

When an interrupt enable bit is cleared to 0 by an instruction such as BCLR or MOV, if an interrupt is generated during execution of the instruction, the interrupt concerned will still be enabled on completion of the instruction, and so interrupt exception handling for that interrupt will be executed on completion of the instruction. However, if there is an interrupt request with priority over that interrupt, interrupt exception handling will be executed for the interrupt with priority, and another interrupt will be ignored. The same also applies when an interrupt source flag is cleared to 0. Figure 5.7 shows an example in which the TCIEV bit in TIER of the TPU is cleared to 0. The above conflict will not occur if an enable bit or interrupt source flag is cleared to 0 while the interrupt is masked.

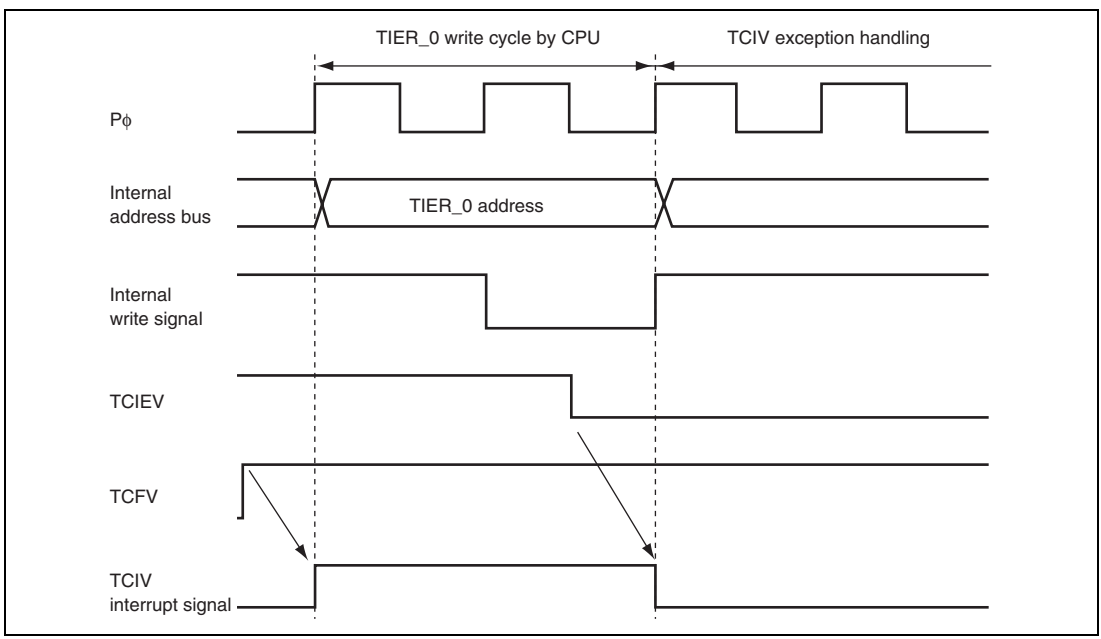


Figure 5.7 Conflict between Interrupt Generation and Disabling

5.8.2 Instructions that Disable Interrupts

Instructions that disable interrupts immediately after execution are LDC, ANDC, ORC, and XORC. After any of these instructions is executed, all interrupts including NMI are disabled and the next instruction is always executed. When the I bit is set by one of these instructions, the new value becomes valid two states after execution of the instruction ends.

5.8.3 Times when Interrupts are Disabled

There are times when interrupt acceptance is disabled by the interrupt controller.

The interrupt controller disables interrupt acceptance for a 3-state period after the CPU has updated the mask level with an LDC, ANDC, ORC, or XORC instruction, and for a period of writing to the registers of the interrupt controller.

5.8.4 Interrupts during Execution of EEPMOV Instruction

Interrupt operation differs between the EEPMOV.B and the EEPMOV.W instructions.

With the EEPMOV.B instruction, an interrupt request (including NMI) issued during the transfer is not accepted until the transfer is completed.

With the EEPMOV.W instruction, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the next instruction. Therefore, if an interrupt is generated during execution of an EEPMOV.W instruction, the following coding should be used.

```
L1:    EEPMOV.W
       MOV.W  R4,R4
       BNE    L1
```

5.8.5 Interrupts during Execution of MOVMD and MOVSD Instructions

With the MOVMD and MOVSD instructions, if an interrupt request is issued during the transfer, interrupt exception handling starts at the end of the individual transfer cycle. The PC value saved on the stack in this case is the address of the MOVMD or MOVSD instruction. The transfer of the remaining data is resumed after returning from the interrupt handling routine.

5.8.6 Interrupt Flags of Peripheral Modules

To clear an interrupt request flag of a peripheral module by the CPU, the flag must be read from after being cleared within the interrupt handling routine even if the peripheral module clock is not generated by dividing the system clock. This makes the request signal synchronized with the system clock.

Section 6 Bus Controller (BSC)

This LSI has an on-chip bus controller (BSC) that has a bus arbitration function and controls the operation of the internal bus masters; CPU and DMAC.

6.1 Features

- Write data buffer function

  Write access to an on-chip peripheral module and access to the on-chip memory can be performed in parallel.
- Bus arbitration function

  Includes a bus arbiter that arbitrates bus mastership between the CPU and DMAC.

  Bus mastership can be shared between the CPU and DMAC when a conflict occurs.
- Multi-clock function

  On-chip peripheral functions can be synchronized with the on-chip peripheral module clock (Pϕ).

A block diagram of the bus controller is shown in figure 6.1.

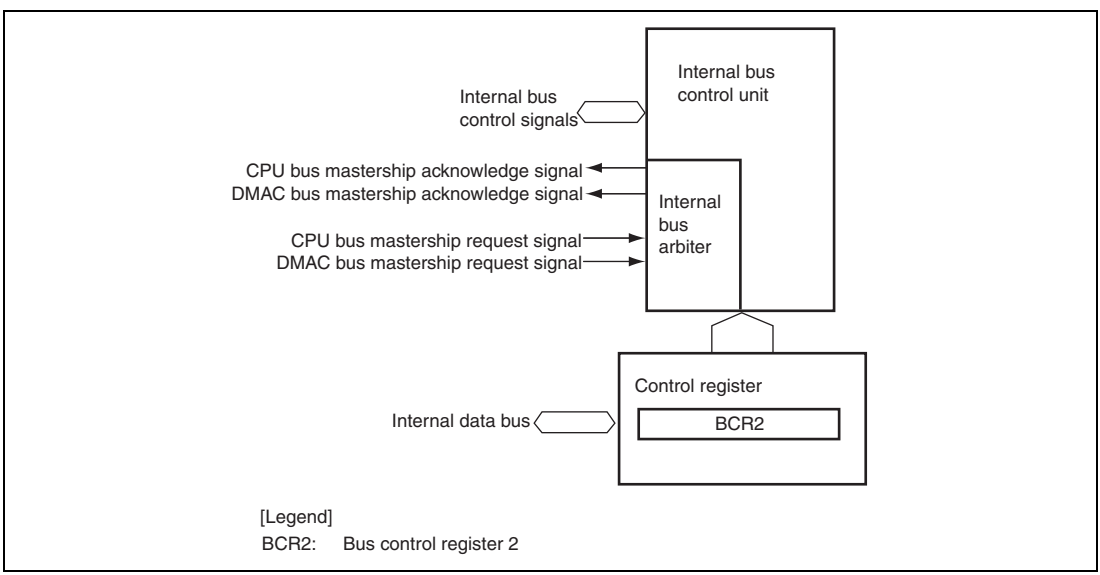


Figure 6.1 Block Diagram of Bus Controller

6.2 Register Descriptions

The bus controller has the following registers.

- Bus control register 2 (BCR2)

6.2.1 Bus Control Register 2 (BCR2)

BCR2 is used for bus arbitration control of the CPU and DMAC, and enabling/disabling of the write data buffer function to the peripheral device.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	IBCCS	—	—	—	PWDBE
Initial Value	0	0	0	0	0	0	1	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R	Reserved These are read-only bits and cannot be modified.
5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
4	IBCCS	0	R/W	Internal Bus Cycle Control Select Selects the internal bus arbiter function. 0: Releases the bus mastership according to the priority 1: Executes the bus cycles alternatively when a CPU bus mastership request conflicts with a DMAC bus mastership request
3, 2	—	All 0	R	Reserved These are read-only bits and cannot be modified.
1	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
0	PWDBE	0	R/W	Peripheral Module Write Data Buffer Enable Specifies whether or not to use the write data buffer function for the peripheral module write cycles. 0: Write data buffer function not used 1: Write data buffer function used

6.3 Bus Configuration

Figure 6.2 shows the internal bus configuration of this LSI. The internal bus of this LSI consists of the following two types.

- Internal system bus 1

  A bus that connects the CPU, DMAC, on-chip ROM, on-chip RAM, and internal peripheral bus.
- Internal peripheral bus

  A bus that accesses registers in the DMAC, bus controller and interrupt controller and registers of peripheral modules such as SCI and timer.

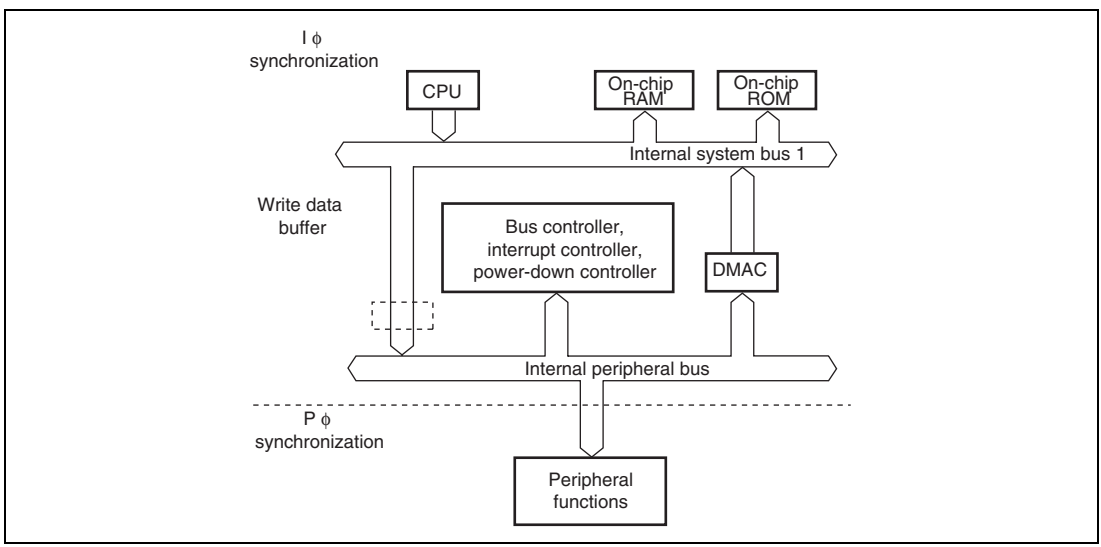


Figure 6.2 Internal Bus Configuration

6.4 Multi-Clock Function

The internal functions of this LSI operate synchronously with the system clock (Iϕ) or the peripheral module clock (Pϕ). Table 6.1 shows the synchronization clock and their corresponding functions.

Table 6.1 Synchronization Clocks and Their Corresponding Functions

Synchronization Clock	Function Name
Iϕ	MCU operating mode Interrupt controller Bus controller CPU DMAC Internal memory Clock pulse generator Power down control
Pϕ	I/O ports TPU PPG WDT SCI RCAN-ET SSU A/D

The frequency of each synchronization clock (Iϕ and Pϕ) is specified by the system clock control register (SCKCR) independently. For further details, see section 18, Clock Pulse Generator.

6.5 Internal Bus

6.5.1 Access to Internal Address Space

The internal address spaces of this LSI are the on-chip ROM space, on-chip RAM space and register space for the on-chip peripheral modules. The number of cycles necessary for access differs according the space.

Table 6.2 shows the number of access cycles for each on-chip memory space.

Table 6.2 Number of Access Cycles for On-Chip Memory Spaces

Access Space	Access	Number of Access Cycles
On-chip ROM space	Read	One Iϕ cycle
On-chip RAM space	Read	One Iϕ cycle
	Write	Two Iϕ cycles

In access to the registers for on-chip peripheral modules, the number of access cycles differs according to the register to be accessed. When the dividing ratio of the operating clock of a bus master and that of a peripheral module is 1 : n, synchronization cycles using a clock divided by 0 to n-1 are inserted for register access.

Table 6.3 Number of Access Cycles for Registers of On-Chip Peripheral Modules

	Number of Cycles		
Module to be Accessed	Read	Write	Write Data Buffer Function
DMAC registers	2Iϕ		Disabled
MCU operating mode, clock pulse generator, power-down control, interrupt controller, and bus controller registers	2Iϕ	3Iϕ	Disabled
I/O port PFCR registers and WDT registers	2Pϕ	3Pϕ	Disabled
TPU, PPG, SCI, and A/D registers and I/O port registers other than PFCR	2Pϕ		Enabled
RCAN-ET registers	4Pϕ		Enabled
SSU registers	3Pϕ		Enabled

6.6 Write Data Buffer Function

6.6.1 Write Data Buffer Function for Peripheral Module

This LSI has a write data buffer function for the peripheral module. Using the write data buffer function enables external writes and on-chip memory accesses in parallel. The write data buffer function is made available by setting the PWDBE bit in BCR2 to 1.

Figure 6.3 shows an example of the timing when the write data buffer function is used. When this function is used, if a peripheral module write continues for two cycles or longer, and there is an internal access next, only the peripheral module write is executed in the first two cycles. However, from the next cycle onward, on-chip memory accesses are executed in parallel, without waiting for the peripheral module write to be complete.

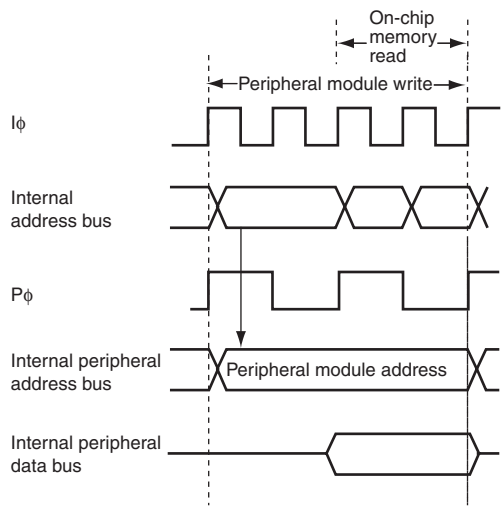


Figure 6.3 Example of Timing when Write Data Buffer Function is Used

6.7 Bus Arbitration

This LSI has bus arbiters that arbitrate bus mastership operations (bus arbitration). The internal bus arbiter handles the CPU and DMAC accesses.

The bus arbiters decide priority at the prescribed timing, and permit use of the bus by means of the bus request acknowledge signal.

6.7.1 Operation

The bus arbiter detects the bus masters' bus request signals, and if the bus is requested, sends a bus request acknowledge signal to the bus master. If there are bus requests from more than one bus master, the bus request acknowledge signal is sent to the one with the highest priority. When a bus master receives the bus request acknowledge signal, it takes possession of the bus until that signal is canceled.

The priority of the internal bus arbitration:

(High) DMAC > CPU (Low)

If the DMAC accesses continue, the CPU can be given priority over the DMAC to execute the bus cycles alternatively between them by setting the IBCCS bit in BCR2.

6.7.2 Bus Transfer Timing

Even if a bus request is received from a bus master with a higher priority over that of the bus master that has taken control of the bus and is currently operating, the bus is not necessarily transferred immediately. There are specific timings at which each bus master can release the bus.

(1) CPU

The CPU is the lowest-priority bus master, and if a bus request is received from the DMAC, the bus arbiter transfers the bus to the bus master that issued the request.

The timing for transfer of the bus is at the end of the bus cycle. In sleep mode, the bus is transferred synchronously with the clock.

Note, however, that the bus cannot be transferred in the following cases.

- The word or longword access is performed in some divisions.
- Stack handling is performed in multiple bus cycles.

- Transfer data read or write by memory transfer instructions, block transfer instructions, or TAS instruction.

  (In the block transfer instructions, the bus can be transferred in the write cycle and the following transfer data read cycle.)
- From the target read to write in the bit manipulation instructions or memory operation instructions.

  (In an instruction that performs no write operation according to the instruction condition, up to a cycle corresponding the write cycle)

(2) DMAC

The DMAC sends the internal bus arbiter a request for the bus when an activation request is generated.

Once the DMAC takes control of the bus, it continues the transfer processing cycles, or releases the bus every transfer cycle.

The bus cannot be transferred in the following cases.

- Between a read cycle and the corresponding write cycle in dual address mode

While the IBCCS bit in BCR2 is cleared to 0, the bus cannot be transferred in the following cases.

- During 1-block data transfer in block transfer mode
- During burst access transfer

The DMAC releases the bus when the consecutive transfer cycles completed except the above cycles.

6.8 Bus Controller Operation in Reset

In a reset, this LSI, including the bus controller, enters the reset state immediately, and any executing bus cycle is aborted.

6.9 Usage Notes

All-Module-Clock-Stop Mode: In this LSI, if the ACSE bit in MSTPCR is set to 1 with the setting for all peripheral module clocks to be stopped (MSTPCR = H'FFFFFFFF), a transition is made to the all-module-clock-stop mode. For details, see section 19, Power-Down Modes.

Section 7 DMA Controller (DMAC)

This LSI includes a 4-channel DMA controller (DMAC).

7.1 Features

- Maximum of 4-G byte address space can be accessed
- Byte, word, or longword can be set as data transfer unit
- Maximum of 4-G bytes (4,294,967,295 bytes) can be set as total transfer size
  Supports free-running mode in which total transfer size setting is not needed
- DMAC activation methods are auto-request, on-chip module interrupt, and external request.
  Auto request: CPU activates (cycle stealing or burst access can be selected)
  On-chip module interrupt: Interrupt requests from on-chip peripheral modules can be selected as an activation source
  External request*: Low level or falling edge detection of the $\overline{\text{DREQ}}$ signal can be selected (external request is available for all four channels)
- Dual or single address mode can be selected as address mode
  Dual address mode: Both source and destination are specified by addresses
  Single address mode*: Either source or destination is specified by the $\overline{\text{DREQ}}$ signal and the other is specified by address
- Normal, repeat, or block transfer can be selected as transfer mode
  Normal transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat transfer mode: One byte, one word, or one longword data is transferred at a single transfer request
  Repeat size of data is transferred and then a transfer address returns to the transfer start address
  Up to 65536 transfers (65,536 bytes/words/longwords) can be set as repeat size
  Block transfer mode: One block data is transferred at a single transfer request
  Up to 65,536 bytes/words/longwords can be set as block size
- Extended repeat area function which repeats the addressees within a specified area using the transfer address with the fixed upper bits (ring buffer transfer can be performed, as an example) is available
  One bit (two bytes) to 27 bits (128 Mbytes) for transfer source and destination can be set as extended repeat areas

- Address update can be selected from fixed address, offset addition, and increment or decrement by 1, 2, or 4

  Address update by offset addition enables to transfer data at addresses which are not placed continuously
- Word or longword data can be transferred to an address which is not aligned with the respective boundary

  Data is divided according to its address (byte or word) when it is transferred
- Two types of interrupts can be requested to the CPU

  A transfer end interrupt is generated after the number of data specified by the transfer counter is transferred. A transfer escape end interrupt is generated when the remaining total transfer size is less than the transfer data size at a single transfer request, when the repeat size of data transfer is completed, or when the extended repeat area overflows.

Note: * An external request and single address mode are not supported by the H8SX/1520R Group.

A block diagram of the DMAC is shown in figure 7.1.

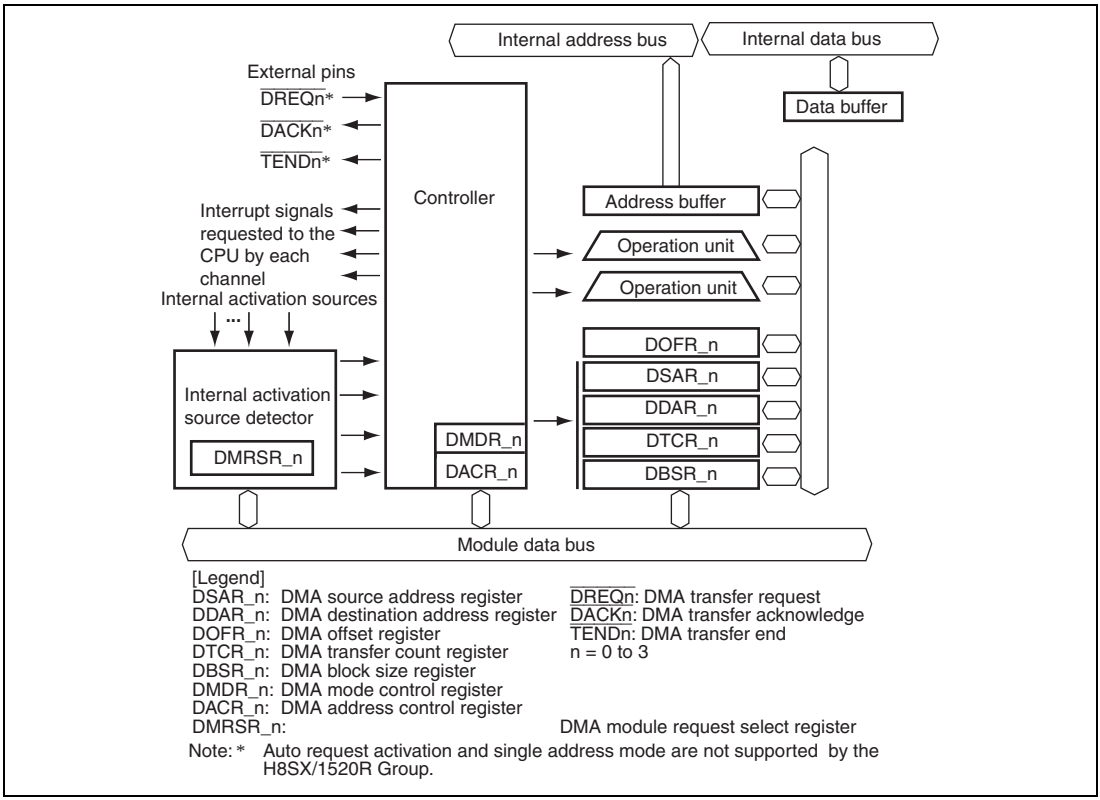


Figure 7.1 Block Diagram of DMAC

7.2 Register Descriptions

The DMAC has the following registers.

Channel 0

- DMA source address register_0 (DSAR_0)
- DMA destination address register_0 (DDAR_0)
- DMA offset register_0 (DOFR_0)
- DMA transfer count register_0 (DTCR_0)
- DMA block size register_0 (DBSR_0)
- DMA mode control register_0 (DMDR_0)
- DMA address control register_0 (DACR_0)
- DMA module request select register_0 (DMRSR_0)

Channel 1

- DMA source address register_1 (DSAR_1)
- DMA destination address register_1 (DDAR_1)
- DMA offset register_1 (DOFR_1)
- DMA transfer count register_1 (DTCR_1)
- DMA block size register_1 (DBSR_1)
- DMA mode control register_1 (DMDR_1)
- DMA address control register_1 (DACR_1)
- DMA module request select register_1 (DMRSR_1)

Channel 2

- DMA source address register_2 (DSAR_2)
- DMA destination address register_2 (DDAR_2)
- DMA offset register_2 (DOFR_2)
- DMA transfer count register_2 (DTCR_2)
- DMA block size register_2 (DBSR_2)
- DMA mode control register_2 (DMDR_2)
- DMA address control register_2 (DACR_2)
- DMA module request select register_2 (DMRSR_2)

Channel 3

- DMA source address register_3 (DSAR_3)
- DMA destination address register_3 (DDAR_3)
- DMA offset register_3 (DOFR_3)
- DMA transfer count register_3 (DTCR_3)
- DMA block size register_3 (DBSR_3)
- DMA mode control register_3 (DMDR_3)
- DMA address control register_3 (DACR_3)
- DMA module request select register_3 (DMRSR_3)

7.2.1 DMA Source Address Register (DSAR)

DSAR is a 32-bit readable/writable register that specifies the transfer source address. DSAR updates the transfer source address every time data is transferred. When DDAR is specified as the destination address (the DIRS bit in DACR is 1) in single address mode, DSAR is ignored.

Although DSAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	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

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	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

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

7.2.2 DMA Destination Address Register (DDAR)

DDAR is a 32-bit readable/writable register that specifies the transfer destination address. DDAR updates the transfer destination address every time data is transferred. When DSAR is specified as the source address (the DIRS bit in DACR is 0) in single address mode, DDAR is ignored.

Although DDAR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	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

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	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

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

7.2.3 DMA Offset Register (DOFR)

DOFR is a 32-bit readable/writable register that specifies the offset to update the source and destination addresses. Although different values are specified for individual channels, the same values must be specified for the source and destination sides of a single channel.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	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

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	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

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

7.2.4 DMA Transfer Count Register (DTCR)

DTCR is a 32-bit readable/writable register that specifies the size of data to be transferred (total transfer size).

To transfer 1-byte data in total, set H'00000001 in DTCR. When H'00000000 is set in this register, it means that the total transfer size is not specified and data is transferred with the transfer counter stopped (free running mode). When H'FFFFFFFF is set, the total transfer size is 4 Gbytes (4,294,967,295), which is the maximum size. While data is being transferred, this register indicates the remaining transfer size. The value corresponding to its data access size is subtracted every time data is transferred (byte: –1, word: –2, and longword: –4).

Although DTCR can always be read from by the CPU, it must be read from in longwords and must not be written to while data for the channel is being transferred.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	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

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	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

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

7.2.5 DMA Block Size Register (DBSR)

DBSR specifies the repeat size or block size. DBSR is enabled in repeat transfer mode and block transfer mode and is disabled in normal transfer mode.

Bit	31	30	29	28	27	26	25	24
Bit Name	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24
Initial Value	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
Bit	23	22	21	20	19	18	17	16
Bit Name	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16
Initial Value	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
Bit	15	14	13	12	11	10	9	8
Bit Name	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8
Initial Value	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
Bit	7	6	5	4	3	2	1	0
Bit Name	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	BKSZH31 to BKSZH16	Undefined	R/W	Specify the repeat size or block size. When H'0001 is set, the repeat or block size is one byte, one word, or one longword. When H'0000 is set, it means the maximum value (refer to table 7.1). While the DMA is in operation, the setting is fixed.
15 to 0	BKSZ15 to BKSZ0	Undefined	R/W	Indicate the remaining repeat or block size while the DMA is in operation. The value is decremented by 1 every time data is transferred. When the remaining size becomes 0, the value of the BKSZH bits is loaded. Set the same value as the BKSZH bits.

Table 7.1 Data Access Size, Valid Bits, and Settable Size

Mode	Data Access Size	BKSZH Valid Bits	BKSZ Valid Bits	Settable Size (Byte)
Repeat transfer and block transfer	Byte	31 to 16	15 to 0	1 to 65,536
	Word			2 to 131,072
	Longword			4 to 262,144

7.2.6 DMA Mode Control Register (DMDR)

DMDR controls the DMAC operation.

- DMDR_0

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	ERRF	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/(W)*	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: * Only 0 can be written to this bit after having been read as 1, to clear the flag.

- DMDR_1 to DMDR_3

Bit	31	30	29	28	27	26	25	24
Bit Name	DTE	DACKE	TENDE	—	DREQS	NRD	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R
Bit	23	22	21	20	19	18	17	16
Bit Name	ACT	—	—	—	—	—	ESIF	DTIF
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R/(W)*	R/(W)*
Bit	15	14	13	12	11	10	9	8
Bit Name	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
Bit	7	6	5	4	3	2	1	0
Bit Name	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Note: * Only 0 can be written to this bit after having been read as 1, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
31	DTE	0	R/W	Data Transfer Enable Enables/disables a data transfer for the corresponding channel. When this bit is set to 1, it indicates that the DMAC is in operation. Setting this bit to 1 starts a transfer when the auto-request is selected. When the on-chip module interrupt or external request is selected, a transfer request after setting this bit to 1 starts the transfer. While data is being transferred, clearing this bit to 0 stops the transfer. In block transfer mode, if writing 0 to this bit while data is being transferred, this bit is cleared to 0 after the current 1-block size data transfer. If an event which stops (sustains) a transfer occurs externally, this bit is automatically cleared to 0 to stop the transfer. Operating modes and transfer methods must not be changed while this bit is set to 1. 0: Disables a data transfer 1: Enables a data transfer (DMA is in operation) [Clearing conditions] • When the specified total transfer size of transfers is completed • When a transfer is stopped by an overflow interrupt by a repeat size end • When a transfer is stopped by an overflow interrupt by an extended repeat size end • When a transfer is stopped by a transfer size error interrupt • When clearing this bit to 0 to stop a transfer In block transfer mode, this bit changes after the current block transfer. • When an address error or an NMI interrupt is requested • In the reset state or hardware standby mode

Bit	Bit Name	Initial Value	R/W	Description
30	DACKE	0	R/W	$\overline{\text{DACK}}$ Signal Output Enable Enables/disables the $\overline{\text{DACK}}$ signal output in single address mode. This bit is ignored in dual address mode. 0: Enables $\overline{\text{DACK}}$ signal output 1: Disables $\overline{\text{DACK}}$ signal output
29	TENDE	0	R/W	$\overline{\text{TEND}}$ Signal Output Enable Enables/disables the $\overline{\text{TEND}}$ signal output. 0: Enables $\overline{\text{TEND}}$ signal output 1: Disables $\overline{\text{TEND}}$ signal output
28	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
27	DREQS	0	R/W	$\overline{\text{DREQ}}$ Select Selects whether a low level or the falling edge of the $\overline{\text{DREQ}}$ signal used in external request mode is detected. When a block transfer is performed in external request mode, clear this bit to 0 to select the low level detection. 0: Low level detection 1: Falling edge detection (the first transfer after a transfer enabled is detected on a low level)
26	NRD	0	R/W	Next Request Delay Selects the accepting timing of the next transfer request. 0: Starts accepting the next transfer request after completion of the current transfer 1: Starts accepting the next transfer request one cycle after completion of the current transfer
25, 24	—	All 0	R	Reserved These are read-only bits and cannot be modified.
23	ACT	0	R	Active State Indicates the operating state for the channel. 0: Waiting for a transfer request or a transfer disabled state by clearing the DTE bit to 0 1: Active state
22 to 20	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
19	ERRF	0	R/(W)*	System Error Flag Indicates that an address error or an NMI interrupt has been generated. This bit is available only in DMDR_0. Setting this bit to 1 prohibits writing to the DTE bit for all the channels. This bit is reserved in DMDR_1 to DMDR_3. It is always read as 0 and cannot be modified. 0: An address error or an NMI interrupt has not been generated 1: An address error or an NMI interrupt has been generated [Clearing condition] • When clearing to 0 after reading ERRF = 1 [Setting condition] • When an address error or an NMI interrupt has been generated However, when an address error or an NMI interrupt has been generated in module stop mode, this bit is not set.
18	—	0	R	Reserved This is a read-only bit and cannot be modified.
17	ESIF	0	R/(W)*	Transfer Escape Interrupt Flag Indicates that a transfer escape end interrupt has been requested. A transfer escape end means that a transfer is terminated before the transfer counter reaches 0. 0: A transfer escape end interrupt has not been requested 1: A transfer escape end interrupt has been requested [Clearing conditions] • When setting the DTE bit to 1 • When clearing to 0 before reading ESIF = 1 [Setting conditions] • When a transfer size error interrupt is requested • When a repeat size end interrupt is requested • When a transfer end interrupt by an extended repeat area overflow is requested

Bit	Bit Name	Initial Value	R/W	Description
16	DTIF	0	R/(W)*	Data Transfer Interrupt Flag Indicates that a transfer end interrupt by the transfer counter has been requested. 0: A transfer end interrupt by the transfer counter has not been requested 1: A transfer end interrupt by the transfer counter has been requested [Clearing conditions] • When setting the DTE bit to 1 • When clearing to 0 after reading DTIF = 1 [Setting condition] • When DTCR reaches 0 and the transfer is completed
15	DTSZ1	0	R/W	Data Access Size 1 and 0
14	DTSZ0	0	R/W	Select the data access size for a transfer. 00: Byte size (eight bits) 01: Word size (16 bits) 10: Longword size (32 bits) 11: Setting prohibited
13	MDS1	0	R/W	Transfer Mode Select 1 and 0
12	MDS0	0	R/W	Select the transfer mode. 00: Normal transfer mode 01: Block transfer mode 10: Repeat transfer mode 11: Setting prohibited

Bit	Bit Name	Initial Value	R/W	Description
11	TSEIE	0	R/W	Transfer Size Error Interrupt Enable Enables/disables a transfer size error interrupt. When the next transfer is requested while this bit is set to 1 and the contents of the transfer counter is less than the size of data to be transferred at a single transfer request, the DTE bit is cleared to 0. At this time, the ESIF bit is set to 1 to indicate that a transfer size error interrupt has been requested. The sources of a transfer size error are as follows: • In normal or repeat transfer mode, the total transfer size set in DTCR is less than the data access size • In block transfer mode, the total transfer size set in DTCR is less than the block size 0: Disables a transfer size error interrupt request 1: Enables a transfer size error interrupt request
10	—	0	R	Reserved This is a read-only bit and cannot be modified.
9	ESIE	0	R/W	Transfer Escape Interrupt Enable Enables/disables a transfer escape end interrupt request. When the ESIF bit is set to 1 with this bit set to 1, a transfer escape end interrupt is requested to the CPU. The transfer end interrupt request is cleared by clearing this bit or the ESIF bit to 0. 0: Disables a transfer escape end interrupt 1: Enables a transfer escape end interrupt
8	DTIE	0	R/W	Data Transfer End Interrupt Enable Enables/disables a transfer end interrupt request by the transfer counter. When the DTIF bit is set to 1 with this bit set to 1, a transfer end interrupt is requested to the CPU. The transfer end interrupt request is cleared by clearing this bit or the DTIF bit to 0. 0: Disables a transfer end interrupt 1: Enables a transfer end interrupt

Bit	Bit Name	Initial Value	R/W	Description
7	DTF1	0	R/W	Data Transfer Factor 1 and 0
6	DTF0	0	R/W	Select a DMAC activation source. When the on-chip peripheral module setting is selected, the interrupt source should be selected by DMRSR. When the external request setting is selected, the sampling method should be selected by the DREQS bit. 00: Auto request (cycle stealing) 01: Auto request (burst access) 10: On-chip module interrupt 11: External request
5	DTA	0	R/W	Data Transfer Acknowledge This bit is valid while the DMA transfer is performed by the on-chip module interrupt. This bit decides whether the source flag selected by DMRSR is cleared or not. 0: The source flag is not cleared while the DMA transfer is performed by the on-chip module interrupt. Since the source flag is not cleared by the DMA transfer, it should be cleared by the CPU. 1: The source flag is cleared while the DMA transfer is performed by the on-chip module interrupt. Since the source flag is cleared by the DMA transfer, there is no need to request an interrupt to the CPU.
4, 3	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
2	DMAP2	0	R/W	DMA Priority Level 2 to 0
1	DMAP1	0	R/W	Select the priority level of the DMAC. When the CPU has priority over the DMAC, the DMAC masks a transfer request and waits for the timing when the CPU priority becomes lower than the DMAC priority. The priority levels can be set to the individual channels. This bit is valid when the CPUPCE bit in CPUPCR is set to 1. 000: Priority level 0 (low) 001: Priority level 1 010: Priority level 2 011: Priority level 3 100: Priority level 4 101: Priority level 5 110: Priority level 6 111: Priority level 7 (high)
0	DMAP0	0	R/W	

Note: * Only 0 can be written to, to clear the flag.

7.2.7 DMA Address Control Register (DACR)

DACR specifies the operating mode and transfer method.

Bit	31	30	29	28	27	26	25	24
Bit Name	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R	R	R	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	SAT1	SAT0	—	—	DAT1	DAT0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R	R	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
31	AMS	0	R/W	Address Mode Select Selects address mode from single or dual address mode. In single address mode, the $\overline{\text{DACK}}$ pin is enabled according to the DACKE bit. 0: Dual address mode 1: Single address mode
30	DIRS	0	R/W	Single Address Direction Select Specifies the data transfer direction in single address mode. This bit s ignored in dual address mode. 0: Specifies DSAR as source address 1: Specifies DDAR as destination address
29 to 27	—	0	R/W	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
26	RPTIE	0	R/W	Repeat Size End Interrupt Enable Enables/disables a repeat size end interrupt request. In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat-size data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. Even when the repeat area is not specified (ARS1 = 1 and ARS0 = 0), a repeat size end interrupt after a 1-block data transfer can be requested. In addition, in block transfer mode, when the next transfer is requested after 1-block data transfer while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate that a repeat size end interrupt is requested. 0: Disables a repeat size end interrupt 1: Enables a repeat size end interrupt
25	ARS1	0	R/W	Area Select 1 and 0
24	ARS0	0	R/W	Specify the block area or repeat area in block or repeat transfer mode. 00: Specify the block area or repeat area on the source address 01: Specify the block area or repeat area on the destination address 10: Do not specify the block area or repeat area 11: Setting prohibited
23, 22	—	All 0	R	Reserved These are read-only bits and cannot be modified.
21	SAT1	0	R/W	Source Address Update Mode 1 and 0
20	SAT0	0	R/W	Select the update method of the source address (DSAR). When DSAR is not specified as the transfer source in single address mode, this bit is ignored. 00: Source address is fixed 01: Source address is updated by adding the offset 10: Source address is updated by adding 1, 2, or 4 according to the data access size 11: Source address is updated by subtracting 1, 2, or 4 according to the data access size

Bit	Bit Name	Initial Value	R/W	Description
19, 18	—	All 0	R	Reserved These are read-only bits and cannot be modified.
17	DAT1	0	R/W	Destination Address Update Mode 1 and 0
16	DAT0	0	R/W	Select the update method of the destination address (DDAR). When DDAR is not specified as the transfer destination in single address mode, this bit is ignored. 00: Destination address is fixed 01: Destination address is updated by adding the offset 10: Destination address is updated by adding 1, 2, or 4 according to the data access size 11: Destination address is updated by subtracting 1, 2, or 4 according to the data access size
15	SARIE	0	R/W	Interrupt Enable for Source Address Extended Area Overflow Enables/disables an interrupt request for an extended area overflow on the source address. When an extended repeat area overflow on the source address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the source address is requested. When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which a transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped. When the extended repeat area is not specified, this bit is ignored. 0: Disables an interrupt request for an extended area overflow on the source address 1: Enables an interrupt request for an extended area overflow on the source address
14, 13	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
12	SARA4	0	R/W	Source Address Extended Repeat Area Specify the extended repeat area on the source address (DSAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2. When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively. When an overflow in the extended repeat area occurs with the SARIE bit set to 1, an interrupt can be requested. Table 7.2 shows the settings and areas of the extended repeat area.
11	SARA3	0	R/W	
10	SARA2	0	R/W	
9	SARA1	0	R/W	
8	SARA0	0	R/W	
7	DARIE	0	R/W	Destination Address Extended Repeat Area Overflow Interrupt Enable Enables/disables an interrupt request for an extended area overflow on the destination address. When an extended repeat area overflow on the destination address occurs while this bit is set to 1, the DTE bit in DMDR is cleared to 0. At this time, the ESIF bit in DMDR is set to 1 to indicate an interrupt by an extended repeat area overflow on the destination address is requested. When block transfer mode is used with the extended repeat area function, an interrupt is requested after completion of a 1-block size transfer. When setting the DTE bit in DMDR of the channel for which the transfer has been stopped to 1, the transfer is resumed from the state when the transfer is stopped. When the extended repeat area is not specified, this bit is ignored. 0: Disables an interrupt request for an extended area overflow on the destination address 1: Enables an interrupt request for an extended area overflow on the destination address
6, 5	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
4	DARA4	0	R/W	Destination Address Extended Repeat Area
3	DARA3	0	R/W	Specify the extended repeat area on the destination address (DDAR). With the extended repeat area, the specified lower address bits are updated and the remaining upper address bits are fixed. The extended repeat area size is specified from four bytes to 128 Mbytes in units of byte and a power of 2. When the lower address is overflowed from the extended repeat area by address update, the address becomes the start address and the end address of the area for address addition and subtraction, respectively. When an overflow in the extended repeat area occurs with the DARIE bit set to 1, an interrupt can be requested. Table 7.2 shows the settings and areas of the extended repeat area.
2	DARA2	0	R/W	
1	DARA1	0	R/W	
0	DARA0	0	R/W	

Table 7.2 Settings and Areas of Extended Repeat Area

SARA4 to SARA0 or DARA4 to DARA0	Extended Repeat Area
00000	Not specified
00001	2 bytes specified as extended repeat area by the lower 1 bit of the address
00010	4 bytes specified as extended repeat area by the lower 2 bits of the address
00011	8 bytes specified as extended repeat area by the lower 3 bits of the address
00100	16 bytes specified as extended repeat area by the lower 4 bits of the address
00101	32 bytes specified as extended repeat area by the lower 5 bits of the address
00110	64 bytes specified as extended repeat area by the lower 6 bits of the address
00111	128 bytes specified as extended repeat area by the lower 7 bits of the address
01000	256 bytes specified as extended repeat area by the lower 8 bits of the address
01001	512 bytes specified as extended repeat area by the lower 9 bits of the address
01010	1 kbyte specified as extended repeat area by the lower 10 bits of the address
01011	2 kbytes specified as extended repeat area by the lower 11 bits of the address
01100	4 kbytes specified as extended repeat area by the lower 12 bits of the address
01101	8 kbytes specified as extended repeat area by the lower 13 bits of the address
01110	16 kbytes specified as extended repeat area by the lower 14 bits of the address
01111	32 kbytes specified as extended repeat area by the lower 15 bits of the address
10000	64 kbytes specified as extended repeat area by the lower 16 bits of the address
10001	128 kbytes specified as extended repeat area by the lower 17 bits of the address
10010	256 kbytes specified as extended repeat area by the lower 18 bits of the address
10011	512 kbytes specified as extended repeat area by the lower 19 bits of the address
10100	1 Mbyte specified as extended repeat area by the lower 20 bits of the address
10101	2 Mbytes specified as extended repeat area by the lower 21 bits of the address
10110	4 Mbytes specified as extended repeat area by the lower 22 bits of the address
10111	8 Mbytes specified as extended repeat area by the lower 23 bits of the address
11000	16 Mbytes specified as extended repeat area by the lower 24 bits of the address
11001	32 Mbytes specified as extended repeat area by the lower 25 bits of the address
11010	64 Mbytes specified as extended repeat area by the lower 26 bits of the address
11011	128 Mbytes specified as extended repeat area by the lower 27 bits of the address
111××	Setting prohibited

[Legend]

×: Don't care

7.2.8 DMA Module Request Select Register (DMRSR)

DMRSR is an 8-bit readable/writable register that specifies the on-chip module interrupt source. The vector number of the interrupt source is specified in eight bits. However, 0 is regarded as no interrupt source. For the vector numbers of the interrupt sources, refer to table 7.4.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

7.3 Transfer Modes

Table 7.3 shows the DMAC transfer modes. The transfer modes can be specified to the individual channels.

Table 7.3 Transfer Modes

Address Mode	Transfer mode	Activation Source	Common Function	Address Register: Source	Address Register: Destination
Dual address	• Normal transfer • Repeat transfer • Block transfer Repeat or block size = 1 to 65,536 bytes, 1 to 65,536 words, or 1 to 65,536 longwords	• Auto request (activated by CPU) • On-chip module interrupt • External request	• Total transfer size: 1 to 4 Gbytes or not specified • Offset addition • Extended repeat area function	DSAR	DDAR
Single address	• Instead of specifying the source or destination address registers, data is directly transferred from/to the external device using the $\overline{\text{DACK}}$ pin • The same settings as above are available other than address register setting (e.g., above transfer modes can be specified) • One transfer can be performed in one bus cycle (the types of transfer modes are the same as those of dual address modes)			DSAR/ $\overline{\text{DACK}}$	$\overline{\text{DACK}}$/ DDAR

When the auto request setting is selected as the activation source, the cycle stealing or burst access can be selected. When the total transfer size is not specified (DTCR = H'00000000), the transfer counter is stopped and the transfer is continued without the limitation of the transfer count.

7.4 Operations

7.4.1 Address Modes

(1) Dual Address Mode

In dual address mode, the transfer source address is specified in DSAR and the transfer destination address is specified in DDAR. A transfer at a time is performed in two bus cycles (when the data bus width is less than the data access size or the access address is not aligned with the boundary of the data access size, the number of bus cycles are needed more than two because one bus cycle is divided into multiple bus cycles).

In the first bus cycle, data at the transfer source address is read and in the next cycle, the read data is written to the transfer destination address.

The read and write cycles are not separated. Other bus cycles (bus cycle by other bus masters, refresh cycle, and external bus release cycle) are not generated between read and write cycles.

The $\overline{\text{TEND}}$ signal output is enabled or disabled by the TENDE bit in DMDR. The $\overline{\text{TEND}}$ signal is output in two bus cycles. When an idle cycle is inserted before the bus cycle, the $\overline{\text{TEND}}$ signal is also output in the idle cycle. The $\overline{\text{DACK}}$ signal is not output.

Figure 7.2 shows an example of the signal timing in dual address mode and figure 7.3 shows the operation in dual address mode.

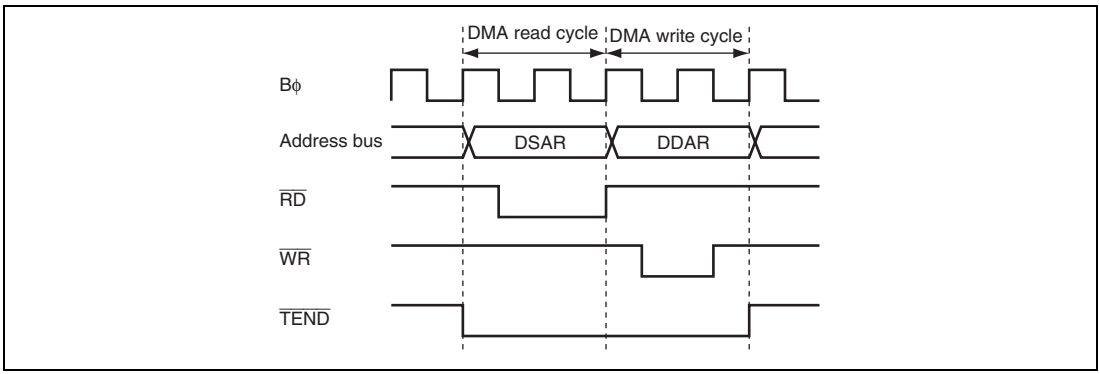


Figure 7.2 Example of Signal Timing in Dual Address Mode

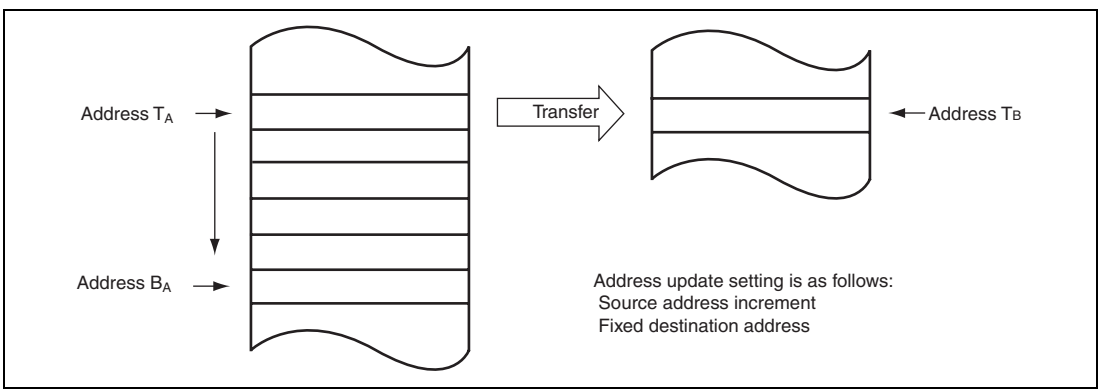


Figure 7.3 Operations in Dual Address Mode

(2) Single Address Mode

In single address mode, data between an external device and an external memory is directly transferred using the $\overline{\text{DACK}}$ pin instead of DSAR or DDAR. A transfer at a time is performed in one bus cycle. In this mode, the data bus width must be the same as the data access size. For details on the data bus width, see section 6, Bus Controller (BSC).

The DMAC accesses an external device as the transfer source or destination by outputting the strobe signal to the external device ($\overline{\text{DACK}}$) and accesses the other transfer target by outputting the address. Accordingly, the DMA transfer is performed in one bus cycle. Figure 7.4 shows an example of a transfer between an external memory and an external device with the $\overline{\text{DACK}}$ pin. In this example, the external device outputs data on the data bus and the data is written to the external memory in the same bus cycle.

The transfer direction is decided by the DIRS bit in DACR which specifies an external device with the $\overline{\text{DACK}}$ pin as the transfer source or destination. When DIRS = 0, data is transferred from an external memory (DSAR) to an external device with the $\overline{\text{DACK}}$ pin. When DIRS = 1, data is transferred from an external device with the $\overline{\text{DACK}}$ pin to an external memory (DDAR). The settings of registers which are not used as the transfer source or destination are ignored.

The $\overline{\text{DACK}}$ signal output is enabled in single address mode by the DACKE bit in DMDR. The $\overline{\text{DACK}}$ signal is low active.

The $\overline{\text{TEND}}$ signal output is enabled or disabled by the TENDE bit in DMDR. The $\overline{\text{TEND}}$ signal is output in one bus cycle. When an idle cycle is inserted before the bus cycle, the $\overline{\text{TEND}}$ signal is also output in the idle cycle.

Figure 7.5 shows an example of timing charts in single address mode and figure 7.6 shows an example of operation in single address mode.

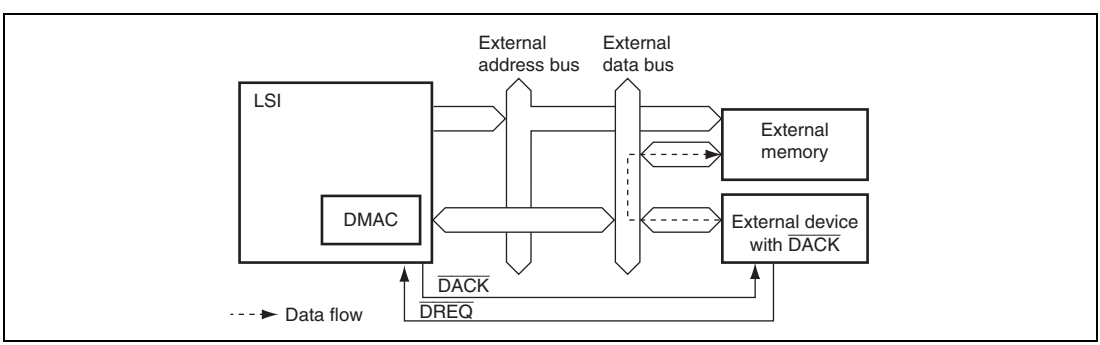


Figure 7.4 Data Flow in Single Address Mode

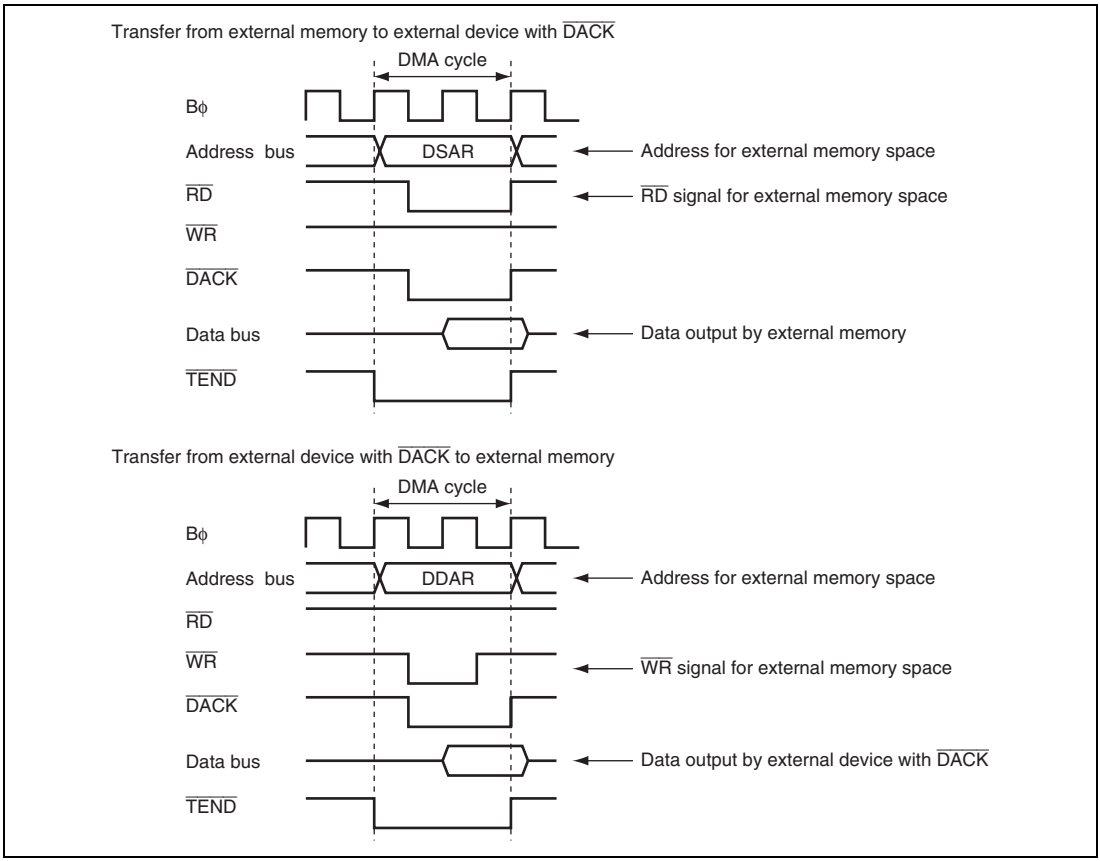


Figure 7.5 Example of Signal Timing in Single Address Mode

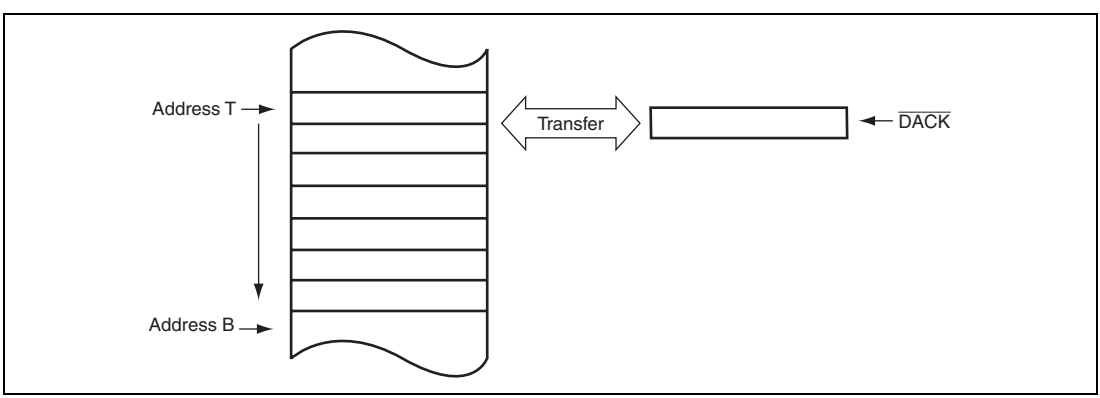


Figure 7.6 Operations in Single Address Mode

7.4.2 Transfer Modes

(1) Normal Transfer Mode

In normal transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. DBSR is ignored in normal transfer mode.

The $\overline{\text{TEND}}$ signal is output only in the last DMA transfer. The $\overline{\text{DACK}}$ signal is output every time a transfer request is received and a transfer starts.

Figure 7.7 shows an example of the signal timing in normal transfer mode and figure 7.8 shows the operation in normal transfer mode.

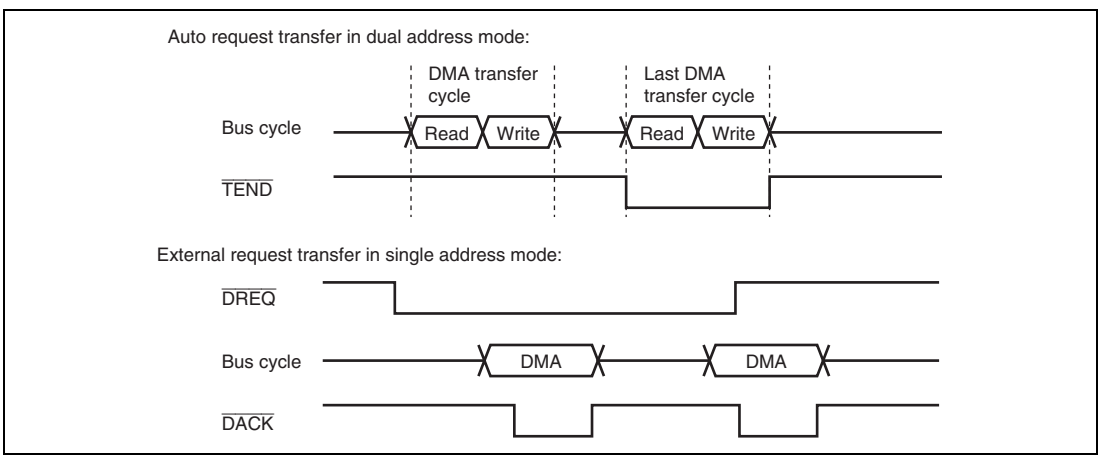


Figure 7.7 Example of Signal Timing in Normal Transfer Mode

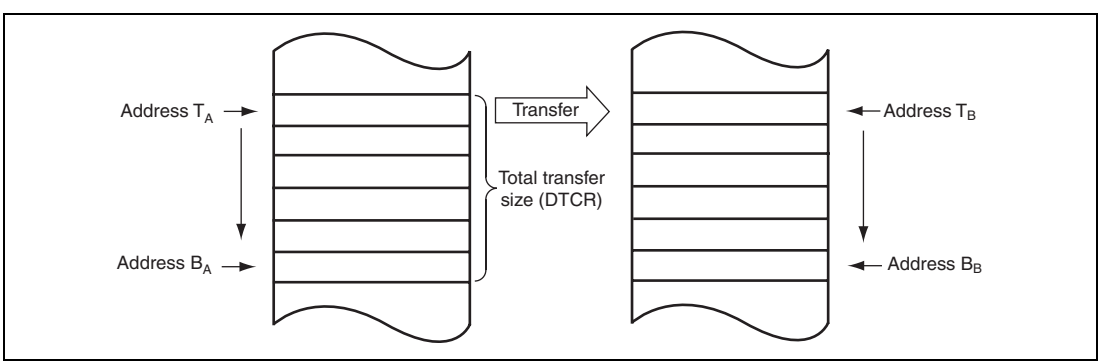


Figure 7.8 Operations in Normal Transfer Mode

(2) Repeat Transfer Mode

In repeat transfer mode, one data access size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as a total transfer size by DTCR. The repeat size can be specified in DBSR up to 65536 × data access size.

The repeat area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the repeat area returns to the transfer start address when the repeat size of transfers is completed. This operation is repeated until the total transfer size specified in DTCR is completed. When H'00000000 is specified in DTCR, it is regarded as the free running mode and repeat transfer is continued until the DTE bit in DMDR is cleared to 0.

In addition, a DMA transfer can be stopped and a repeat size end interrupt can be requested to the CPU when the repeat size of transfers is completed. When the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit is set to 1, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1 to complete the transfer. At this time, an interrupt is requested to the CPU when the ESIE bit in DMDR is set to 1.

The timings of the $\overline{\text{TEND}}$ and $\overline{\text{DACK}}$ signals are the same as in normal transfer mode.

Figure 7.9 shows the operation in repeat transfer mode while dual address mode is set.

When the repeat area is specified as neither source nor destination address side, the operation is the same as the normal transfer mode operation shown in figure 7.8. In this case, a repeat size end interrupt can also be requested to the CPU when the repeat size of transfers is completed.

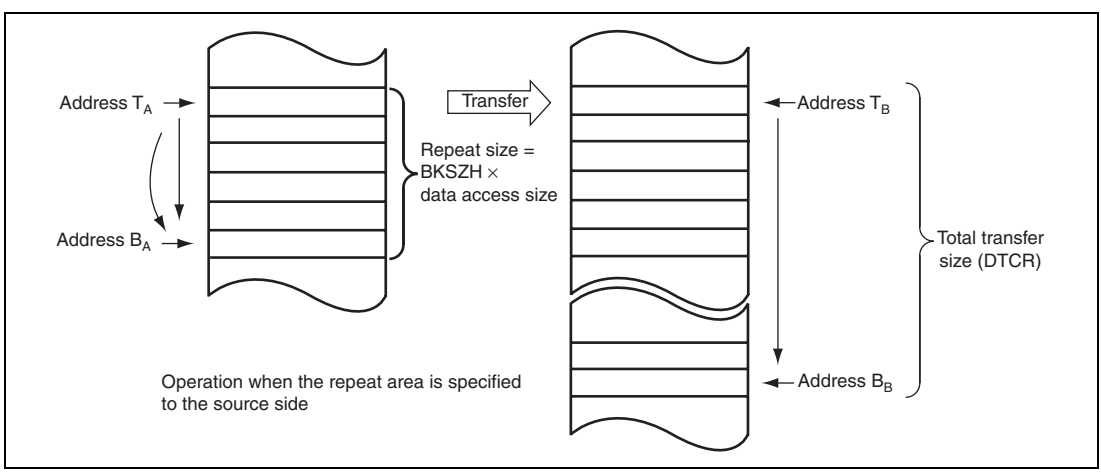


Figure 7.9 Operations in Repeat Transfer Mode

(3) Block Transfer Mode

In block transfer mode, one block size of data is transferred at a single transfer request. Up to 4 Gbytes can be specified as total transfer size by DTCR. The block size can be specified in DBSR up to 65536 × data access size.

While one block of data is being transferred, transfer requests from other channels are suspended. When the transfer is completed, the bus is released to the other bus master.

The block area can be specified for the source or destination address side by bits ARS1 and ARS0 in DACR. The address specified as the block area returns to the transfer start address when the block size of data is completed. When the block area is specified as neither source nor destination address side, the operation continues without returning the address to the transfer start address. A repeat size end interrupt can be requested.

The $\overline{\text{TEND}}$ signal is output every time 1-block data is transferred in the last DMA transfer cycle. When the external request is selected as an activation source, the low level detection of the $\overline{\text{DREQ}}$ signal (DREQS = 0) should be selected.

When an interrupt request by an extended repeat area overflow is used in block transfer mode, settings should be selected carefully. For details, see section 7.4.5, Extended Repeat Area Function.

Figure 7.10 shows an example of the DMA transfer timing in block transfer mode. The transfer conditions are as follows:

- Address mode: single address mode
- Data access size: byte
- 1-block size: three bytes

The block transfer mode operations in single address mode and in dual address mode are shown in figures 7.11 and 7.12, respectively.

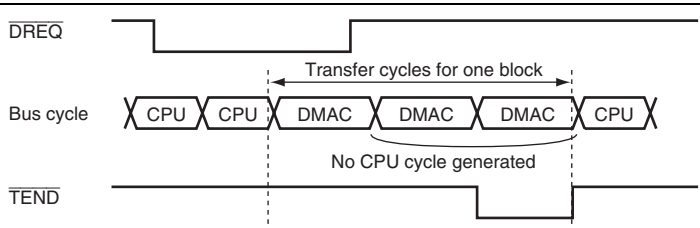


Figure 7.10 Operations in Block Transfer Mode

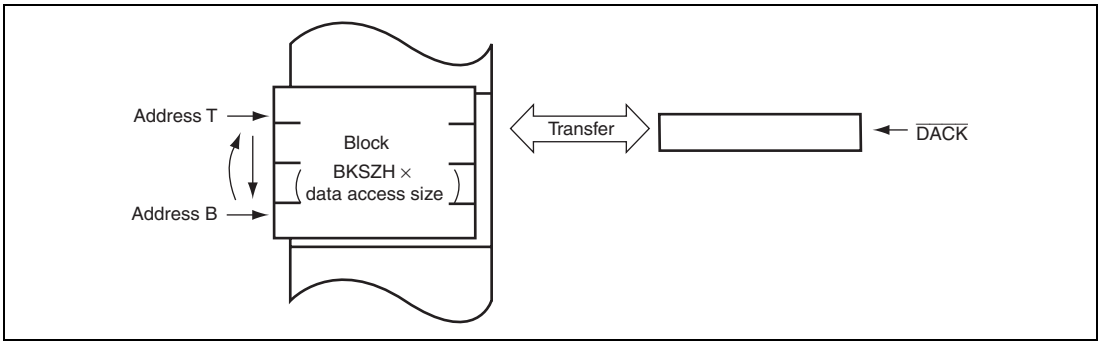


Figure 7.11 Operation in Single Address Mode in Block Transfer Mode (Block Area Specified)

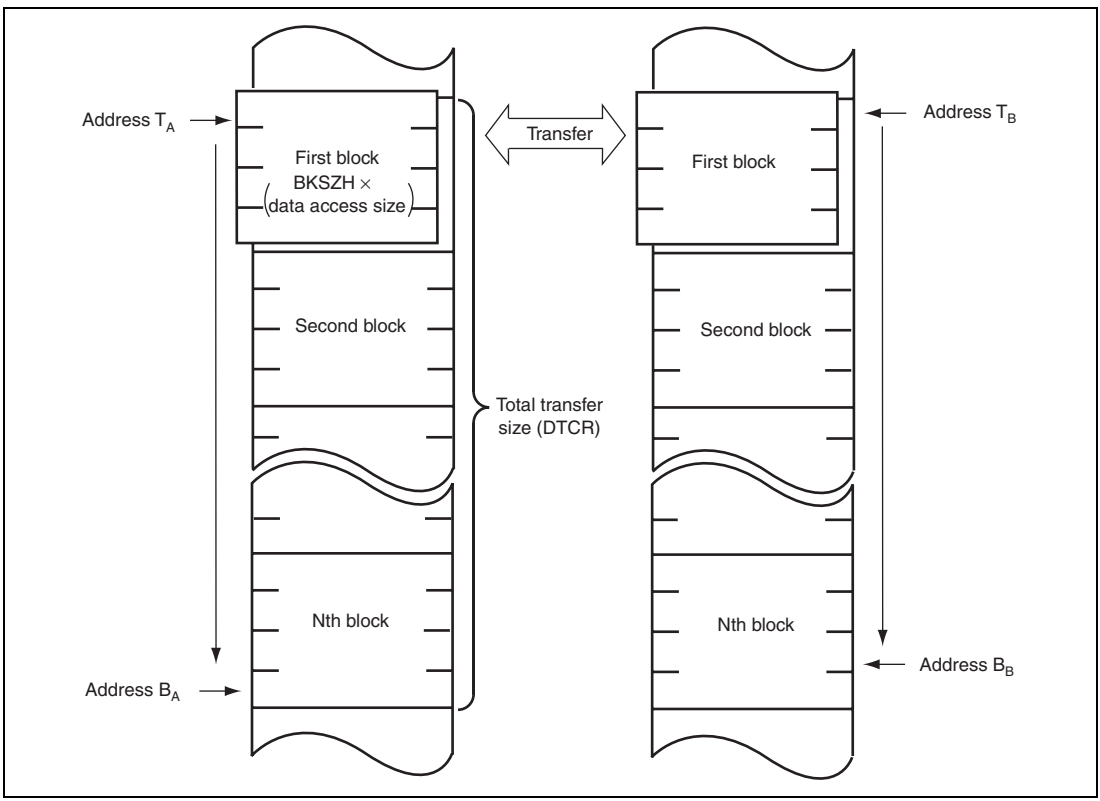


Figure 7.12 Operation in Dual Address Mode in Block Transfer Mode (Block Area Not Specified)

7.4.3 Activation Sources

The DMAC is activated by an auto request, an on-chip module interrupt, and an external request. The activation source is specified by bits DTF1 and DTF0 in DMDR.

(1) Activation by Auto Request

The auto request activation is used when a transfer request from an external device or an on-chip peripheral module is not generated such as a transfer between memory and memory or between memory and an on-chip peripheral module which does not request a transfer. A transfer request is automatically generated inside the DMAC. In auto request activation, setting the DTE bit in DMDR starts a transfer. The bus mode can be selected from cycle stealing and burst modes.

(2) Activation by On-Chip Module Interrupt

An interrupt request from an on-chip peripheral module (on-chip peripheral module interrupt) is used as a transfer request. When a DMA transfer is enabled (DTE = 1), the DMA transfer is started by an on-chip module interrupt.

The activation source of the on-chip module interrupt is selected by the DMA module request select register (DMRSR). The activation sources are specified to the individual channels. Table 7.4 is a list of on-chip module interrupts for the DMAC.

The interrupt request selected as an activation source can simultaneously generate interrupt requests to the CPU. For details, see section 5, Interrupt Controller.

The DMAC receives interrupt requests by on-chip peripheral modules independent of the interrupt controller. Therefore, the DMAC is not affected by priority given in the interrupt controller.

When the DMAC is activated with DTA = 1, the interrupt request flag is automatically cleared by a DMA transfer. If multiple channels use a single transfer request as an activation source, when the channel having priority is activated, the interrupt request flag is cleared. In this case, other channels may not be activated because the transfer request is not held in the DMAC.

When the DMAC is activated with DTA = 0, the interrupt request flag is not cleared by the DMAC. It should be cleared by the CPU.

When an activation source is selected while DTE = 0, the activation source does not request a transfer to the DMAC. It requests an interrupt to the CPU.

In addition, make sure that an interrupt request flag as an on-chip module interrupt source is cleared to 0 before writing 1 to the DTE bit.

Table 7.4 List of On-chip module interrupts to DMAC

On-Chip Module Interrupt Source	On-Chip Module	DMRSR (Vector Number)
ADI0 (A/D conversion end interrupt)	A/D_0	86
ADI1 (A/D conversion end interrupt)	A/D_1	87
TGI0A (TGI0A input capture/compare match)	TPU_0	88
TGI1A (TGI1A input capture/compare match)	TPU_1	93
TGI2A (TGI2A input capture/compare match)	TPU_2	97
TGI3A (TGI3A input capture/compare match)	TPU_3	101
RXI3 (receive data full interrupt for SCI channel 3)	SCI_3	157
TXI3 (transmit data empty interrupt for SCI channel 3)	SCI_3	158
RXI4 (receive data full interrupt for SCI channel 4)	SCI_4	161
TXI4 (transmit data empty interrupt for SCI channel 4)	SCI_4	162
TGI6A (TGI6A input capture/compare match)	TPU_6	164
TGI7A (TGI7A input capture/compare match)	TPU_7	169
TGI8A (TGI8A input capture/compare match)	TPU_8	173
TGI9A (TGI9A input capture/compare match)	TPU_9	177
TGI10A (TGI10A input capture/compare match)	TPU_10	182
TGI11A (TGI11A input capture/compare match)	TPU_11	188
RM0 (message reception in Mailbox 0)	RCAN-ET	221
SSRXI0 (receive data full interrupt for SSU channel 0)	SSU_0	228
SSTXI0 (transmit data empty interrupt or transmit end for SSU channel 0)	SSU_0	229
SSRXI1 (receive data full interrupt for SSU channel 1)	SSU_1	232
SSTXI1 (transmit data empty interrupt or transmit end for SSU channel 1)	SSU_1	233
SSRXI2 (receive data full interrupt for SSU channel 2)	SSU_2	236
SSTXI2 (transmit data empty interrupt or transmit end for SSU channel 2)	SSU_2	237

(3) Activation by External Request

A transfer is started by a transfer request signal ($\overline{\text{DREQ}}$) from an external device. When a DMA transfer is enabled (DTE = 1), the DMA transfer is started by the $\overline{\text{DREQ}}$ assertion.

A transfer request signal is input to the $\overline{\text{DREQ}}$ pin. The $\overline{\text{DREQ}}$ signal is detected on the falling edge or low level. Whether the falling edge or low level detection is used is selected by the DREQS bit in DMDR. To perform a block transfer, select the low level detection.

When an external request is selected as an activation source, clear the DDR bit to 0 and set the ICR bit to 1 for the corresponding pin. For details, see section 8, I/O Ports.

When a DMA transfer between on-chip peripheral modules is performed, select an activation source form the auto request and on-chip module interrupt (the external request cannot be used).

7.4.4 Bus Access Modes

There are two types of bus access modes: cycle stealing and burst.

When an activation source is the auto request, the cycle stealing or burst mode is selected by bit DTF0 in DMDR. When an activation source is the on-chip module interrupt or external request, the cycle stealing mode is selected.

(1) Cycle Stealing Mode

In cycle stealing mode, the DMAC releases the bus every time one unit of transfers (byte, word, longword, or 1-block size) is completed. After that, when a transfer is requested, the DMAC obtains the bus to transfer 1-unit data and then releases the bus on completion of the transfer. This operation is continued until the transfer end condition is satisfied.

When a transfer is requested to another channel during a DMA transfer, the DMAC releases the bus and then transfers data for the requested channel. For details on operations when a transfer is requested to multiple channels, see section 7.4.8, Priority of Channels.

Figure 7.13 shows an example of timing in cycle stealing mode. The transfer conditions are as follows:

- Address mode: Single address mode
- Sampling method of the $\overline{\text{DREQ}}$ signal: Low level detection

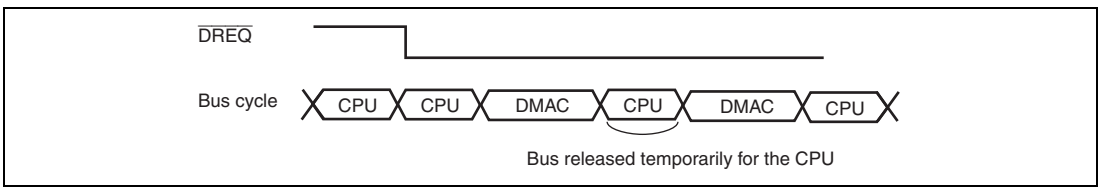


Figure 7.13 Example of Timing in Cycle Stealing Mode

(2) Burst Mode

In burst mode, once it takes the bus, the DMAC continues a transfer without releasing the bus until the transfer end condition is satisfied. Even if a transfer is requested from another channel having priority, the transfer is not stopped once it is started. The DMAC releases the bus in the next cycle after the transfer for the channel in burst mode is completed. This is similarly to operation in cycle stealing mode. However, setting the IBCCS bit in IBCR of the bus controller makes the DMAC release the bus to pass the bus to another bus master.

In block transfer mode, the burst mode setting is ignored (operation is the same as that in burst mode during one block of transfers). The DMAC is always operated in cycle stealing mode.

Clearing the DTE bit in DMDR stops a DMA transfer. A transfer requested before the DTE bit is cleared to 0 by the DMAC is executed. When an interrupt by a transfer size error, a repeat size end, or an extended repeat area overflow occurs, the DTE bit is cleared to 0 and the transfer ends.

Figure 7.14 shows an example of timing in burst mode.

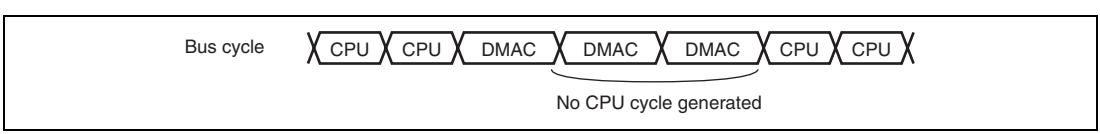


Figure 7.14 Example of Timing in Burst Mode

7.4.5 Extended Repeat Area Function

The source and destination address sides can be specified as the extended repeat area. The contents of the address register repeat addresses within the area specified as the extended repeat area. For example, to use a ring buffer as the transfer target, the contents of the address register should return to the start address of the buffer every time the contents reach the end address of the buffer (overflow on the ring buffer address). This operation can automatically be performed using the extended repeat area function of the DMAC.

The extended repeat areas can be specified independently to the source address register (DSAR) and destination address register (DDAR).

The extended repeat area on the source address is specified by bits SARA4 to SARA0 in DACR. The extended repeat area on the destination address is specified by bits DARA4 to DARA0 in DACR. The extended repeat area sizes for each side can be specified independently.

A DMA transfer is stopped and an interrupt by an extended repeat area overflow can be requested to the CPU when the contents of the address register reach the end address of the extended repeat area. When an overflow on the extended repeat area set in DSAR occurs while the SARIE bit in DACR is set to 1, the ESIF bit in DMDR is set to 1 and the DTE bit in DMDR is cleared to 0 to stop the transfer. At this time, if the ESIE bit in DMDR is set to 1, an interrupt by an extended repeat area overflow is requested to the CPU. When the DARIE bit in DACR is set to 1, an overflow on the extended repeat area set in DDAR occurs, meaning that the destination side is a target. During the interrupt handling, setting the DTE bit in DMDR resumes the transfer.

Figure 7.15 shows an example of the extended repeat area operation.

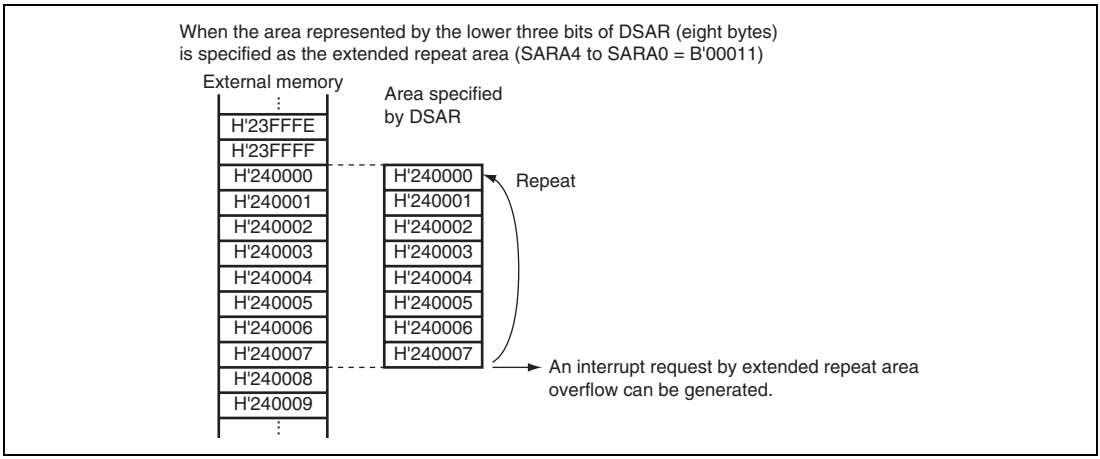


Figure 7.15 Example of Extended Repeat Area Operation

When an interrupt by an extended repeat area overflow is used in block transfer mode, the following should be taken into consideration.

When a transfer is stopped by an interrupt by an extended repeat area overflow, the address register must be set so that the block size is a power of 2 or the block size boundary is aligned with the extended repeat area boundary. When an overflow on the extended repeat area occurs during a transfer of one block, the interrupt by the overflow is suspended and the transfer overruns.

Figure 7.16 shows examples when the extended repeat area function is used in block transfer mode.

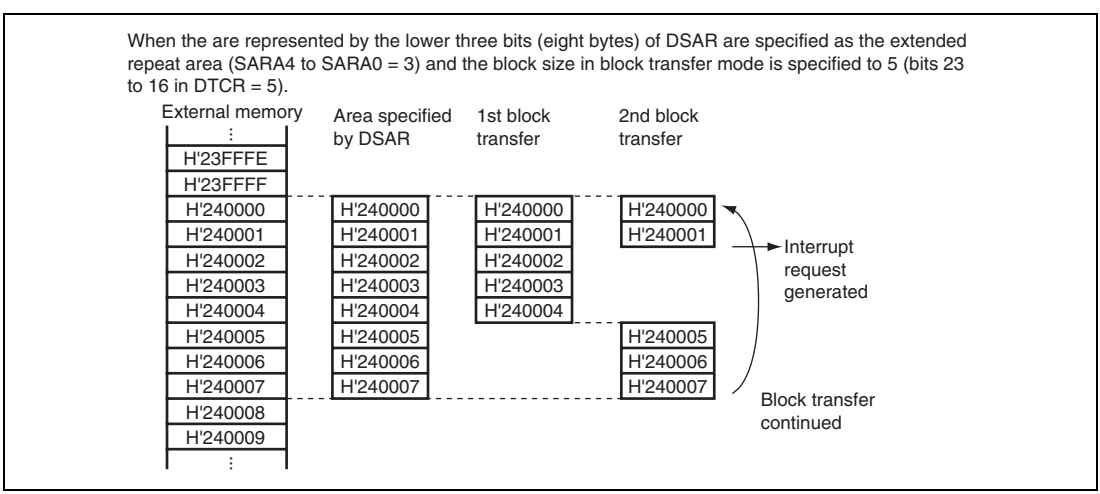


Figure 7.16 Example of Extended Repeat Area Function in Block Transfer Mode

7.4.6 Address Update Function using Offset

The source and destination addresses are updated by fixing, increment/decrement by 1, 2, or 4, or offset addition. When the offset addition is selected, the offset specified by the offset register (DOFR) is added to the address every time the DMAC transfers the data access size of data. This function realizes a data transfer where addresses are allocated to separated areas.

Figure 7.17 shows the address update method.

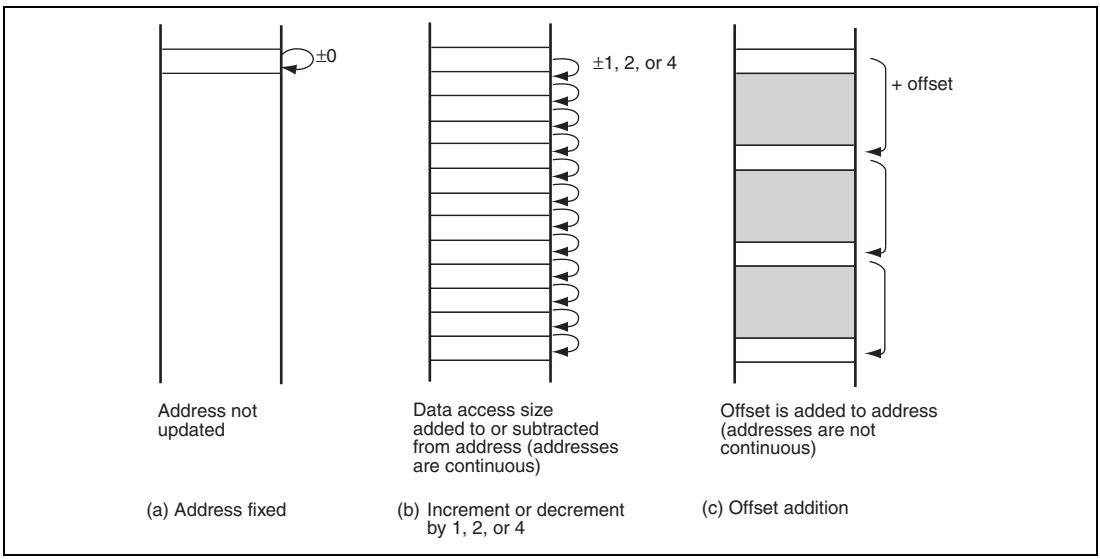


Figure 7.17 Address Update Method

In item (a), Address fixed, the transfer source or destination address is not updated indicating the same address.

In item (b), Increment or decrement by 1, 2, or 4, the transfer source or destination address is incremented or decremented by the value according to the data access size at each transfer. Byte, word, or longword can be specified as the data access size. The value of 1 for byte, 2 for word, and 4 for longword is used for updating the address. This operation realizes the data transfer placed in consecutive areas.

In item (c), Offset addition, the address update does not depend on the data access size. The offset specified by DOFR is added to the address every time the DMAC transfers data of the data access size.

The address is calculated by the offset set in DOFR and the contents of DSAR and DDAR. Although the DMAC calculates only addition, an offset subtraction can be realized by setting the negative value in DOFR. In this case, the negative value must be 2's complement.

(1) Basic Transfer Using Offset

Figure 7.18 shows a basic operation of a transfer using the offset addition.

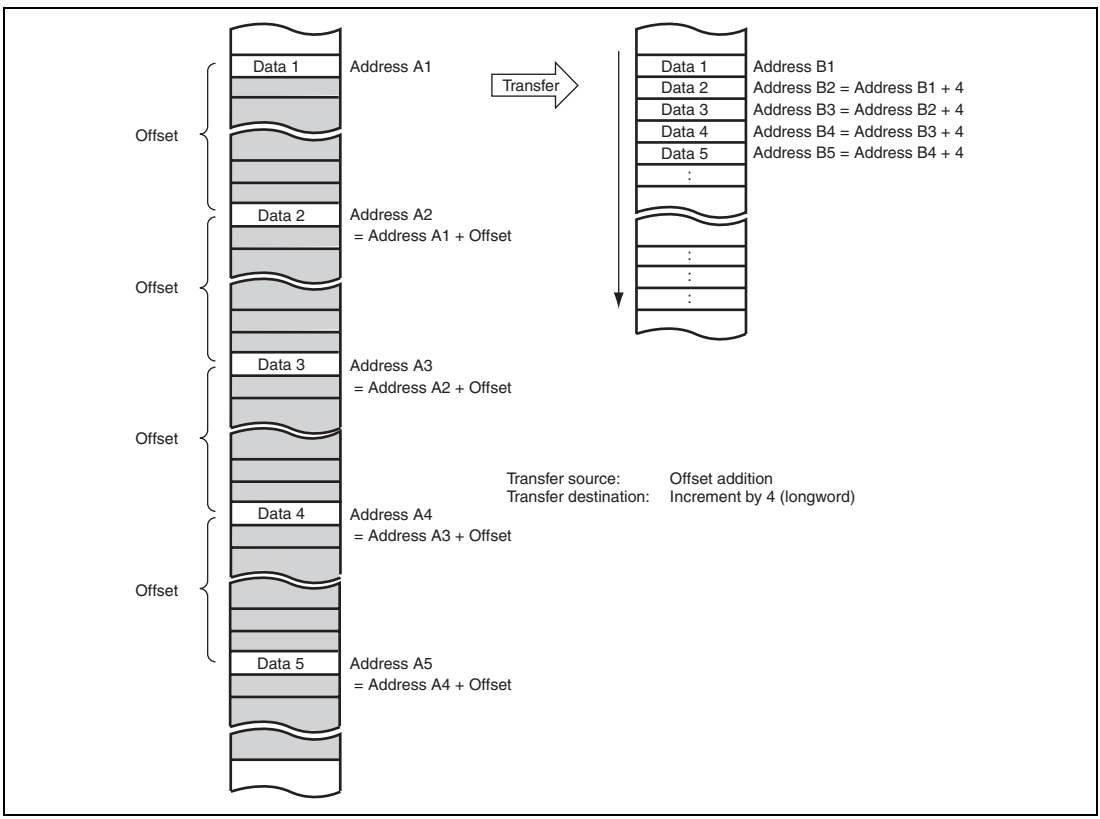


Figure 7.18 Operation of Offset Addition

In figure 7.18, the offset addition is selected as the transfer source address update and increment or decrement by 1, 2, or 4 is selected as the transfer destination address. The address update means that data at the address which is away from the previous transfer source address by the offset is read from. The data read from the address away from the previous address is written to the consecutive area in the destination side.

(2) XY Conversion Using Offset

Figure 7.19 shows the XY conversion using the offset addition in repeat transfer mode.

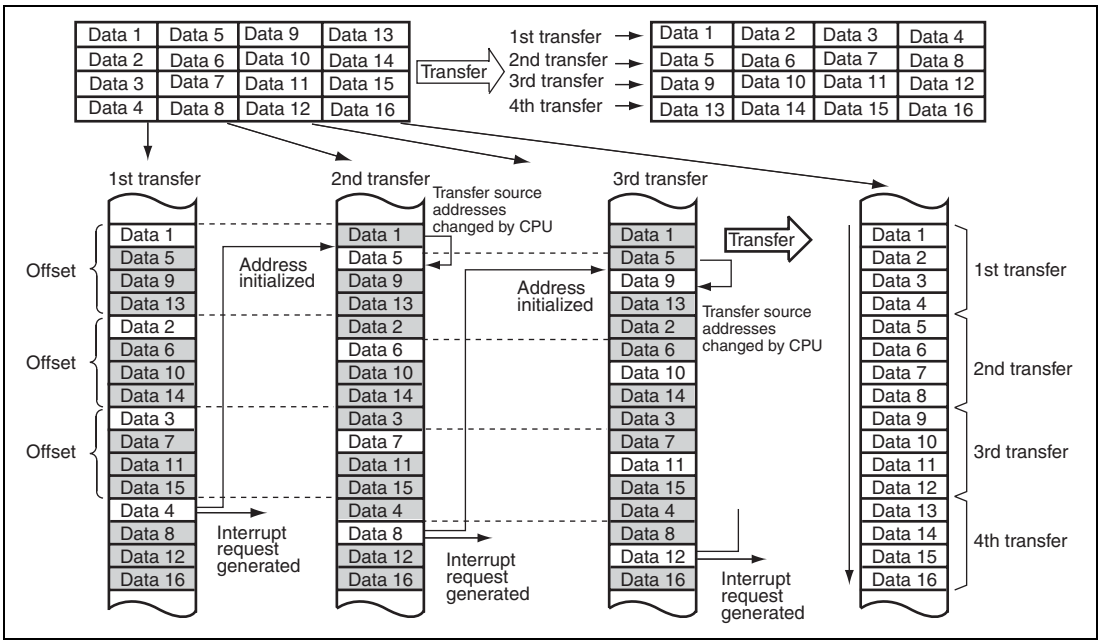


Figure 7.19 XY Conversion Operation Using Offset Addition in Repeat Transfer Mode

In figure 7.19, the source address side is specified to the repeat area by DACR and the offset addition is selected. The offset value is set to 4 × data access size (when the data access size is longword, H'00000010 is set in DOFR, as an example). The repeat size is set to 4 × data access size (when the data access size is longword, the repeat size is set to 4 × 4 = 16 bytes, as an example). The increment or decrement by 1, 2, or 4 is specified as the transfer destination address. A repeat size end interrupt is requested when the repeat size of transfers is completed.

When a transfer starts, the transfer source address is added to the offset every time data is transferred. The transfer data is written to the destination continuous addresses. When data 4 is transferred meaning that the repeat size of transfers is completed, the transfer source address returns to the transfer start address (address of data 1 on the transfer source) and a repeat size end interrupt is requested. While this interrupt stops the transfer temporarily, the contents of DSAR are written to the address of data 5 by the CPU (when the data access size is longword, write the data 1 address + 4). When the DTE bit in DMDR is set to 1, the transfer is resumed from the state when the transfer is stopped. Accordingly, operations are repeated and the transfer source data is transposed to the destination area (XY conversion).

Figure 7.29 shows a flowchart of the XY conversion.

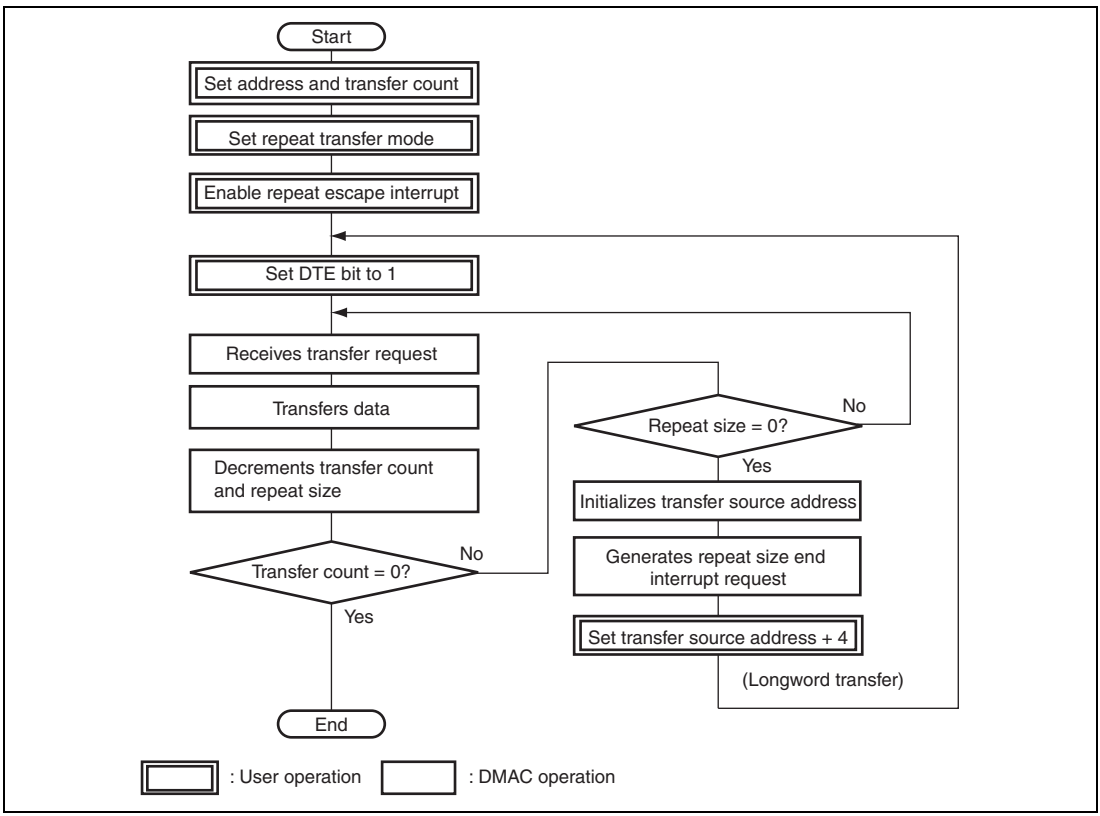


Figure 7.20 XY Conversion Flowchart Using Offset Addition in Repeat Transfer Mode

(3) Offset Subtraction

When setting the negative value in DOFR, the offset value must be 2's complement. The 2's complement is obtained by the following formula.

2's complement of offset = 1 + ~offset (~: bit inversion)

Example: 2's complement of H'0001FFFF
= H'FFFE0000 + H'00000001
= H'FFFE0001

The value of 2's complement can be obtained by the NEG.L instruction.

7.4.7 Register during DMA Transfer

The DMAC registers are updated by a DMA transfer. The value to be updated differs according to the other settings and transfer state. The registers to be updated are DSAR, DDAR, DTCR, bits BKSZH and BKSZ in DBSR, and the DTE, ACT, ERRF, ESIF, and DTIF bits in DMDR.

(1) DMA Source Address Register

When the transfer source address set in DSAR is accessed, the contents of DSAR are output and then are updated to the next address.

The increment or decrement can be specified by bits SAT1 and SAT0 in DACR. When SAT1 and SAT0 = B'00, the address is fixed. When SAT1 and SAT0 = B'01, the address is added with the offset. When SAT1 and SAT0 = B'10, the address is incremented. When SAT1 and SAT0 = B'11, the address is decremented. The size of increment or decrement depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the source address is word or longword, when the source address is not aligned with the word or longword boundary, the read bus cycle is divided into byte or word cycles. While data of one word or one longword is being read, the size of increment or decrement is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After one word or one longword of data is read, the address when the read cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the source address side, the source address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the source address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DSAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DSAR during the transfer may be updated regardless of the access by the CPU. Moreover, DSAR for the channel being transferred must not be written to.

(2) DMA Destination Address Register

When the transfer destination address set in DDAR is accessed, the contents of DDAR are output and then are updated to the next address.

The increment or decrement can be specified by bits DAT1 and DAT0 in DACR. When DAT1 and DAT0 = B'00, the address is fixed. When DAT1 and DAT0 = B'01, the address is added with the offset. When DAT1 and DAT0 = B'10, the address is incremented. When DAT1 and DAT0 = B'11, the address is decremented. The incrementing or decrementing size depends on the data access size.

The data access size is specified by bits DTSZ1 and DTSZ0 in DMDR. When DTSZ1 and DTSZ0 = B'00, the data access size is byte and the address is incremented or decremented by 1. When DTSZ1 and DTSZ0 = B'01, the data access size is word and the address is incremented or decremented by 2. When DTSZ1 and DTSZ0 = B'10, the data access size is longword and the address is incremented or decremented by 4. Even if the access data size of the destination address is word or longword, when the destination address is not aligned with the word or longword boundary, the write bus cycle is divided into byte and word cycles. While one word or one longword of data is being written, the incrementing or decrementing size is changing according to the actual data access size, for example, +1 or +2 for byte or word data. After the one word or one longword of data is written, the address when the write cycle is started is incremented or decremented by the value according to bits SAT1 and SAT0.

In block or repeat transfer mode, when the block or repeat size of data transfers is completed while the block or repeat area is specified to the destination address side, the destination address returns to the transfer start address and is not affected by the address update.

When the extended repeat area is specified to the destination address side, operation follows the setting. The upper address bits are fixed and is not affected by the address update.

While data is being transferred, DDAR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DDAR during the transfer may be updated regardless of the access by the CPU. Moreover, DDAR for the channel being transferred must not be written to.

(3) DMA Transfer Count Register (DTCR)

A DMA transfer decrements the contents of DTCR by the transferred bytes. When byte data is transferred, DTCR is decremented by 1. When word data is transferred, DTCR is decremented by 2. When longword data is transferred, DTCR is decremented by 4. However, when DTCR = 0, the contents of DTCR are not changed since the number of transfers is not counted.

While data is being transferred, all the bits of DTCR may be changed. DTCR must be accessed in longwords. If the upper word and lower word are read separately, incorrect data may be read from since the contents of DTCR during the transfer may be updated regardless of the access by the CPU. Moreover, DTCR for the channel being transferred must not be written to.

When a conflict occurs between the address update by DMA transfer and write access by the CPU, the CPU has priority. When a conflict occurs between change from 1, 2, or 4 to 0 in DTCR and write access by the CPU (other than 0), the CPU has priority in writing to DTCR. However, the transfer is stopped.

(4) DMA Block Size Register (DBSR)

DBSR is enabled in block or repeat transfer mode. Bits 31 to 16 in DBSR function as BKSZH and bits 15 to 0 in DBSR function as BKSZ. The BKSZH bits (16 bits) store the block size and repeat size and its value is not changed. The BKSZ bits (16 bits) function as a counter for the block size and repeat size and its value is decremented every transfer by 1. When the BKSZ value is to change from 1 to 0 by a DMA transfer, 0 is not stored but the BKSZH value is loaded into the BKSZ bits.

Since the upper 16 bits of DBSR are not updated, DBSR can be accessed in words.

DBSR for the channel being transferred must not be written to.

(5) DTE Bit in DMDR

Although the DTE bit in DMDR enables or disables data transfer by the CPU write access, it is automatically cleared to 0 according to the DMA transfer state by the DMAC.

The conditions for clearing the DTE bit by the DMAC are as follows:

- When the total size of transfers is completed
- When a transfer is completed by a transfer size error interrupt
- When a transfer is completed by a repeat size end interrupt
- When a transfer is completed by an extended repeat area overflow interrupt
- When a transfer is stopped by an NMI interrupt
- When a transfer is stopped by and address error
- Reset state
- Hardware standby mode
- When a transfer is stopped by writing 0 to the DTE bit

Writing to the registers for the channels when the corresponding DTE bit is set to 1 is prohibited (except for the DTE bit). When changing the register settings after writing 0 to the DTE bit, confirm that the DTE bit has been cleared to 0.

Figure 7.21 show the procedure for changing the register settings for the channel being transferred.

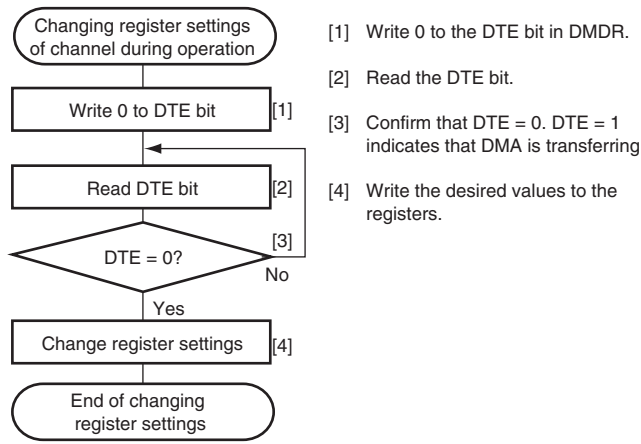


Figure 7.21 Procedure for Changing Register Setting For Channel being Transferred

(6) ACT Bit in DMDR

The ACT bit in DMDR indicates whether the DMAC is in the idle or active state. When DTE = 0 or DTE = 1 and the DMAC is waiting for a transfer request, the ACT bit is 0. Otherwise (the DMAC is in the active state), the ACT bit is 1. When individual transfers are stopped by writing 0 and the transfer is not completed, the ACT bit retains 1.

In block transfer mode, even if individual transfers are stopped by writing 0 to the DTE bit, the 1-block size of transfers is not stopped. The ACT bit retains 1 from writing 0 to the DTE bit to completion of a 1-block size transfer.

In burst mode, up to three times of DMA transfer are performed from the cycle in which the DTE bit is written to 0. The ACT bit retains 1 from writing 0 to the DTE bit to completion of DMA transfer.

(7) ERRF Bit in DMDR

When an address error or an NMI interrupt occur, the DMAC clears the DTE bits for all the channels to stop a transfer. In addition, it sets the ERRF bit in DMDR_0 to 1 to indicate that an address error or an NMI interrupt has occurred regardless of whether or not the DMAC is in operation.

(8) ESIF Bit in DMDR

When an interrupt by an transfer size error, a repeat size end, or an extended repeat area overflow is requested, the ESIF bit in DMDR is set to 1. When both the ESIF and ESIE bits are set to 1, a transfer escape interrupt is requested to the CPU.

The ESIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle of the interrupt source is completed.

The ESIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.7, Interrupt Sources.

(9) DTIF Bit in DMDR

The DTIF bit in DMDR is set to 1 after the total transfer size of transfers is completed. When both the DTIF and DTIE bits in DMDR are set to 1, a transfer end interrupt by the transfer counter is requested to the CPU.

The DTIF bit is set to 1 when the ACT bit in DMDR is cleared to 0 to stop a transfer after the bus cycle is completed.

The DTIF bit is automatically cleared to 0 and a transfer request is cleared if the transfer is resumed by setting the DTE bit to 1 during interrupt handling.

For details on interrupts, see section 7.7, Interrupt Sources.

7.4.8 Priority of Channels

The channels of the DMAC are given following priority levels: channel 0 > channel 1 > channel 2 > channel3. Table 7.5 shows the priority levels among the DMAC channels.

Table 7.5 Priority among DMAC Channels

Channel	Priority
Channel 0	High
Channel 1	↑
Channel 2	
Channel 3	Low

The channel having highest priority other than the channel being transferred is selected when a transfer is requested from other channels. The selected channel starts the transfer after the channel being transferred releases the bus. At this time, when a bus master other than the DMAC requests the bus, the cycle for the bus master is inserted.

In a burst transfer or a block transfer, channels are not switched.

Figure 7.22 shows a transfer example when multiple transfer requests from channels 0 to 2.

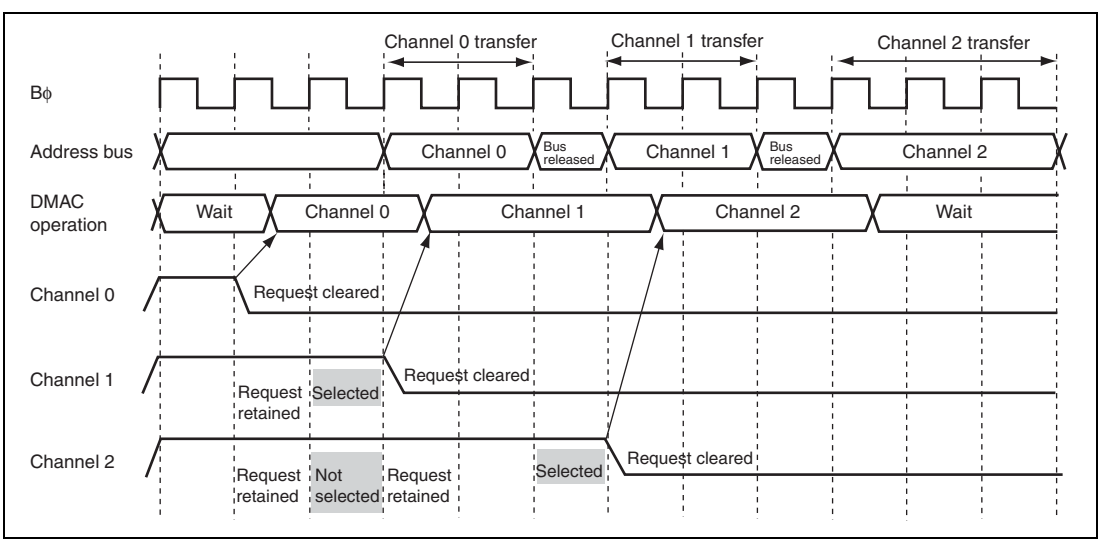


Figure 7.22 Example of Timing for Channel Priority

7.4.9 DMA Basic Bus Cycle

Figure 7.23 shows an examples of signal timing of a basic bus cycle. In figure 7.23, data is transferred in words from the 16-bit 2-state access space to the 8-bit 3-state access space. When the bus mastership is passed from the DMAC to the CPU, data is read from the source address and it is written to the destination address. The bus is not released between the read and write cycles by other bus requests. DMAC bus cycles follows the bus controller settings.

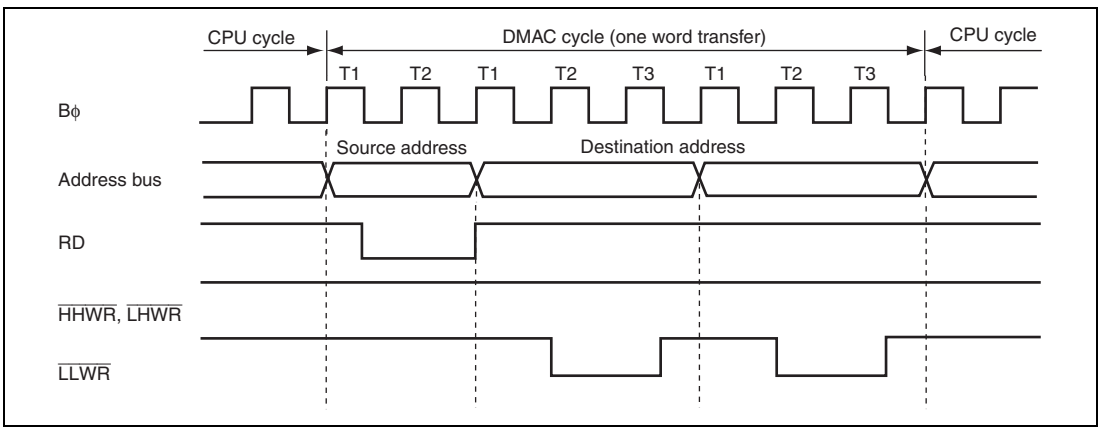


Figure 7.23 Example of Bus Timing of DMA Transfer

7.4.10 Bus Cycles in Dual Address Mode

(1) Normal Transfer Mode (Cycle Stealing Mode)

In cycle stealing mode, the bus is released every time one transfer size of data (one byte, one word, or one longword) is completed. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.24, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.

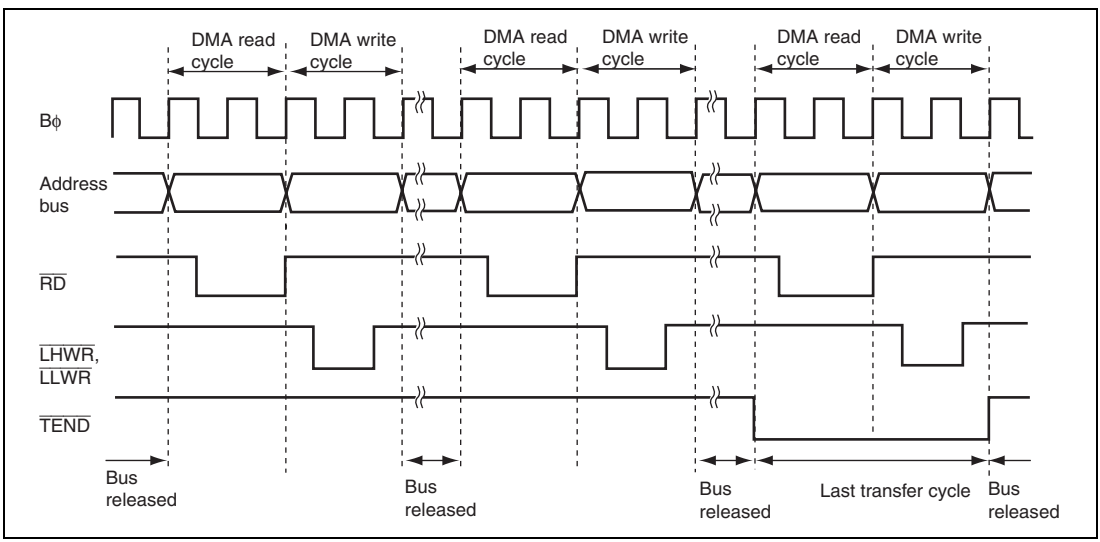


Figure 7.24 Example of Transfer in Normal Transfer Mode by Cycle Stealing

In figures 7.25 and 7.26, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in longwords from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by cycle stealing.

In figure 7.25, the transfer source (DSAR) is not aligned with a longword boundary and the transfer destination (DDAR) is aligned with a longword boundary.

In figure 7.26, the transfer source (DSAR) is aligned with a longword boundary and the transfer destination (DDAR) is not aligned with a longword boundary.

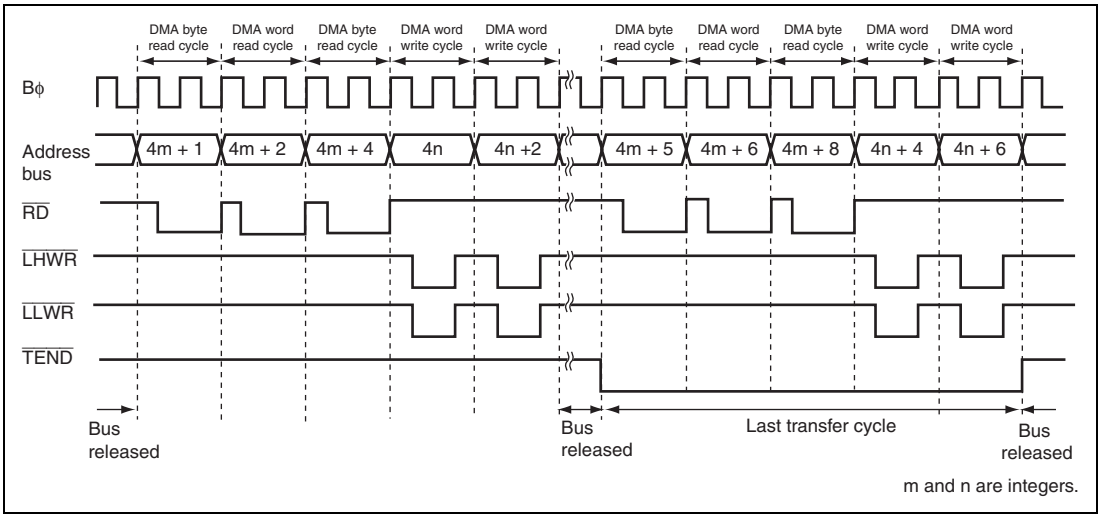


Figure 7.25 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Source DSAR = Odd Address and Source Address Increment)

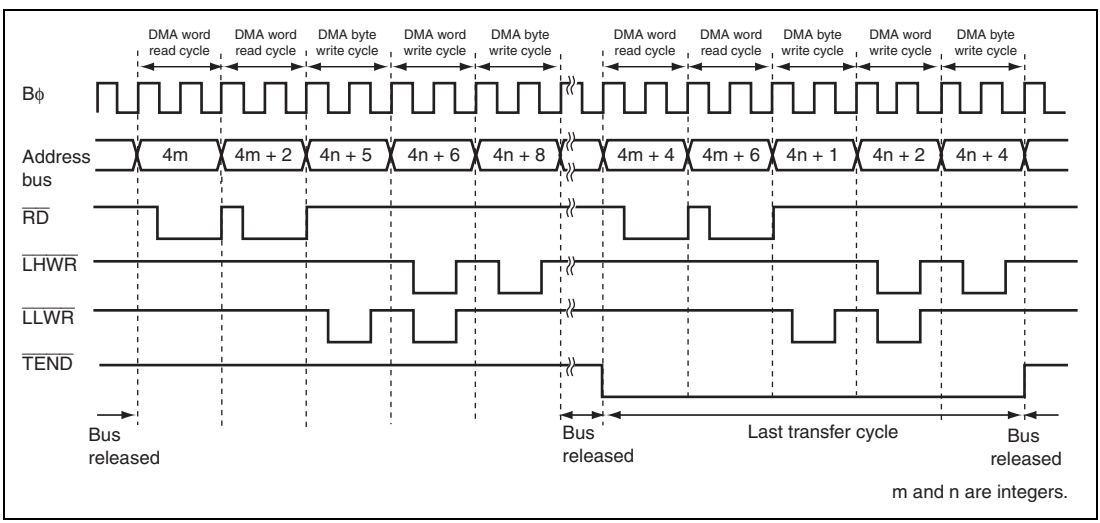


Figure 7.26 Example of Transfer in Normal Transfer Mode by Cycle Stealing (Transfer Destination DDAR = Odd Address and Destination Address Decrement)

(2) Normal Transfer Mode (Burst Mode)

In burst mode, one byte, one word, or one longword of data continues to be transferred until the transfer end condition is satisfied.

When a burst transfer starts, a transfer request from a channel having priority is suspended until the burst transfer is completed.

In figure 7.27, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in normal transfer mode by burst access.

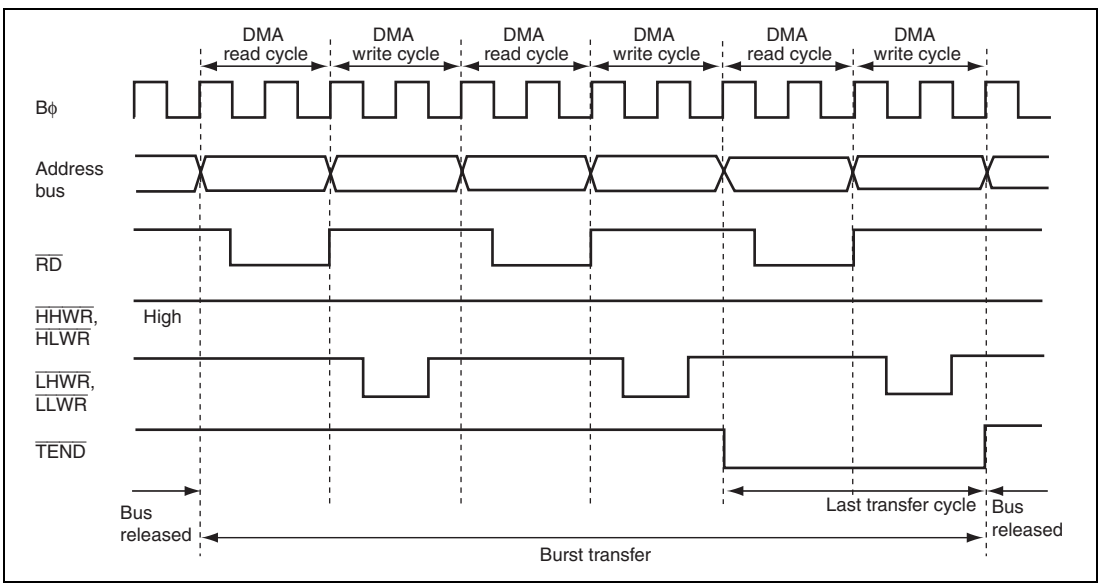


Figure 7.27 Example of Transfer in Normal Transfer Mode by Burst Access

(3) Block Transfer Mode

In block transfer mode, the bus is released every time a 1-block size of transfers at a single transfer request is completed.

In figure 7.28, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in words from the external 16-bit 2-state access space to the external 16-bit 2-state access space in block transfer mode.

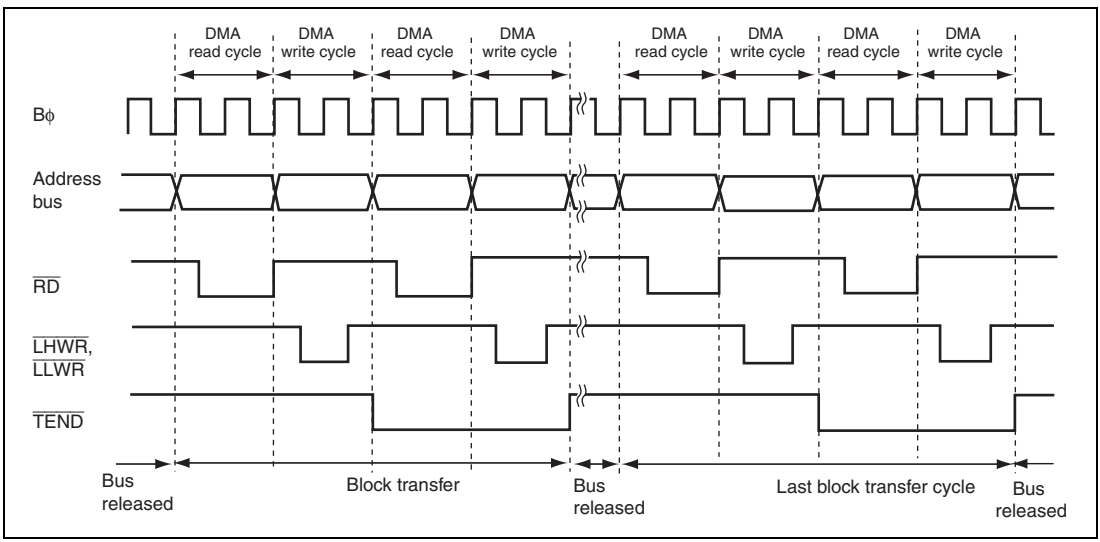


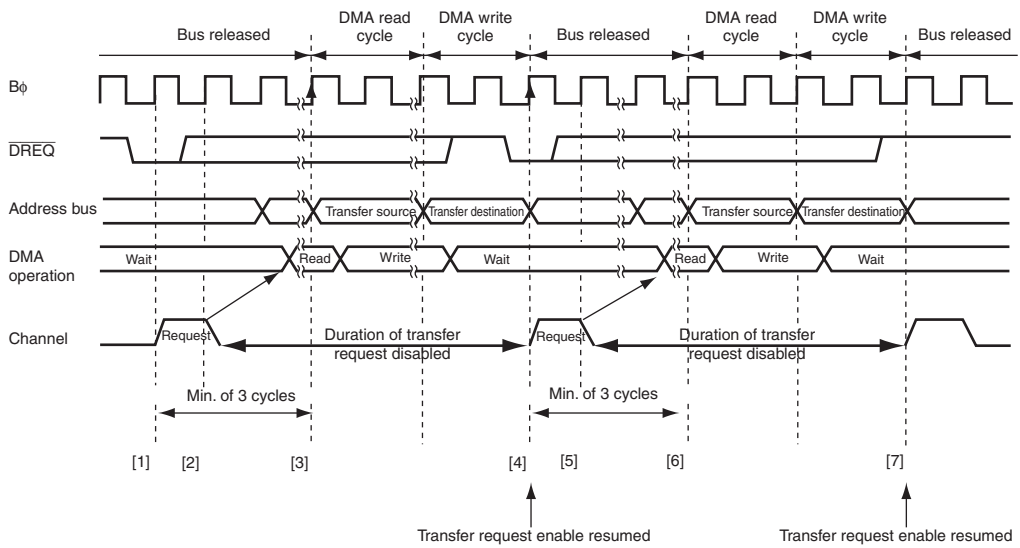
Figure 7.28 Example of Transfer in Block Transfer Mode

(4) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.29 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the DMA write cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7] When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer request enable is resumed after completion of the write cycle. (A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

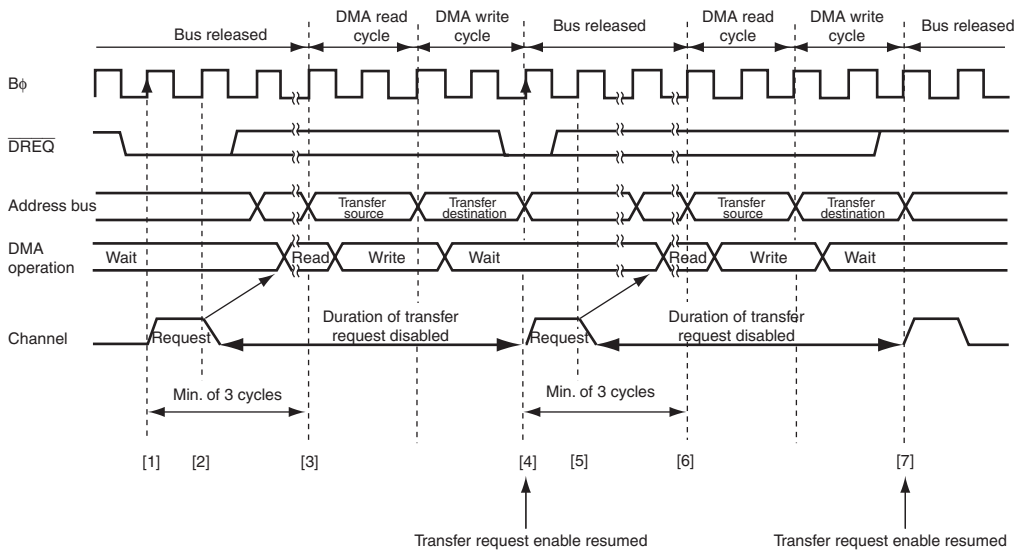
Figure 7.29 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Falling Edge

(5) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.30 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



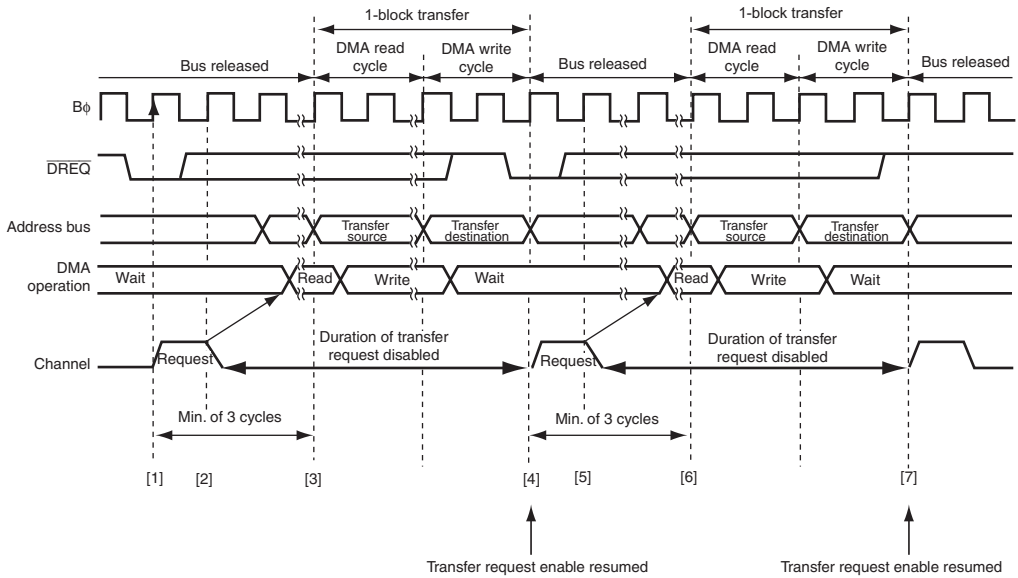
[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.30 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level

Figure 7.31 shows an example of block transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.31 Example of Transfer in Block Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level

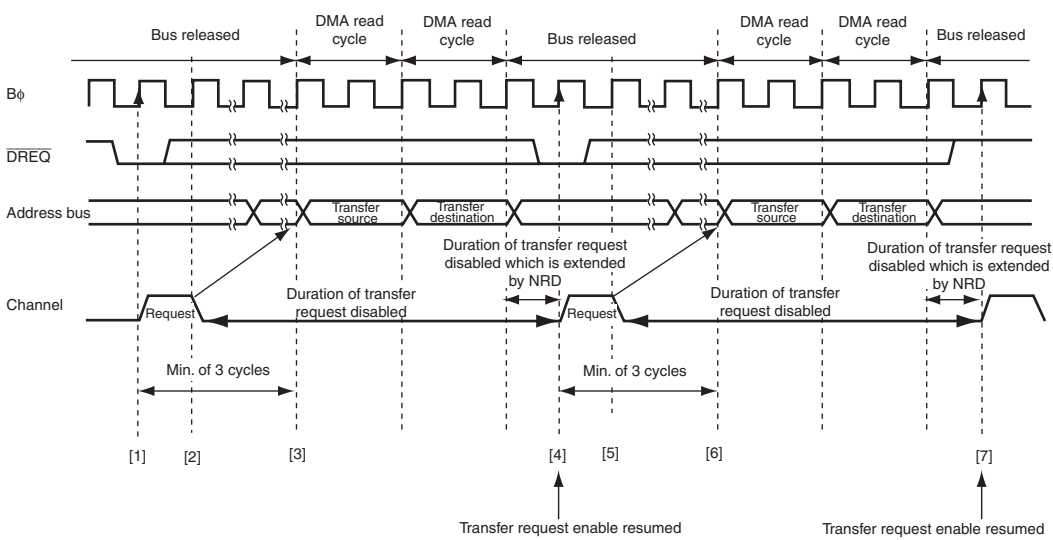
(6) Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.32 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the write cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed one cycle after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.32 Example of Transfer in Normal Transfer Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

7.4.11 Bus Cycles in Single Address Mode

(1) Single Address Mode (Read and Cycle Stealing)

In single address mode, one byte, one word, or one longword of data is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.33, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (read).

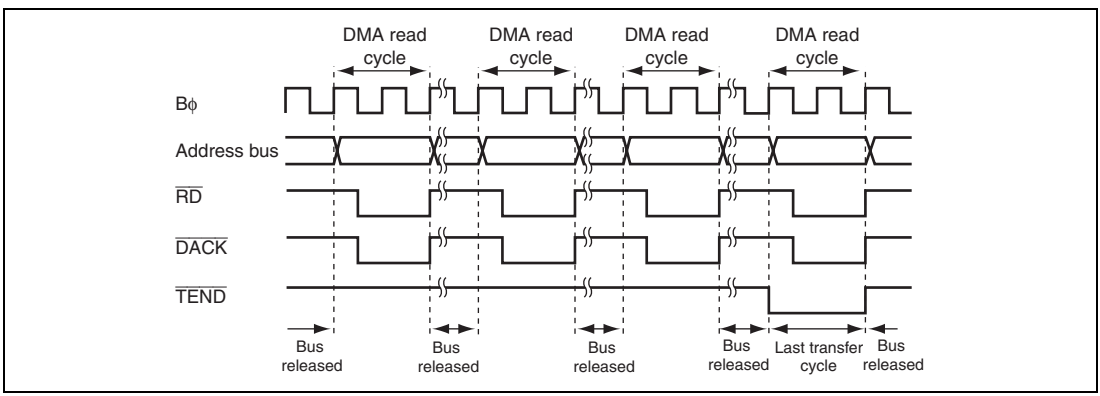


Figure 7.33 Example of Transfer in Single Address Mode (Byte Read)

(2) Single Address Mode (Write and Cycle Stealing)

In single address mode, data of one byte, one word, or one longword is transferred at a single transfer request and after the transfer the bus is released temporarily. One bus cycle or more by the CPU are executed in the bus released cycles.

In figure 7.34, the $\overline{\text{TEND}}$ signal output is enabled and data is transferred in bytes from the external 8-bit 2-state access space to the external device in single address mode (write).

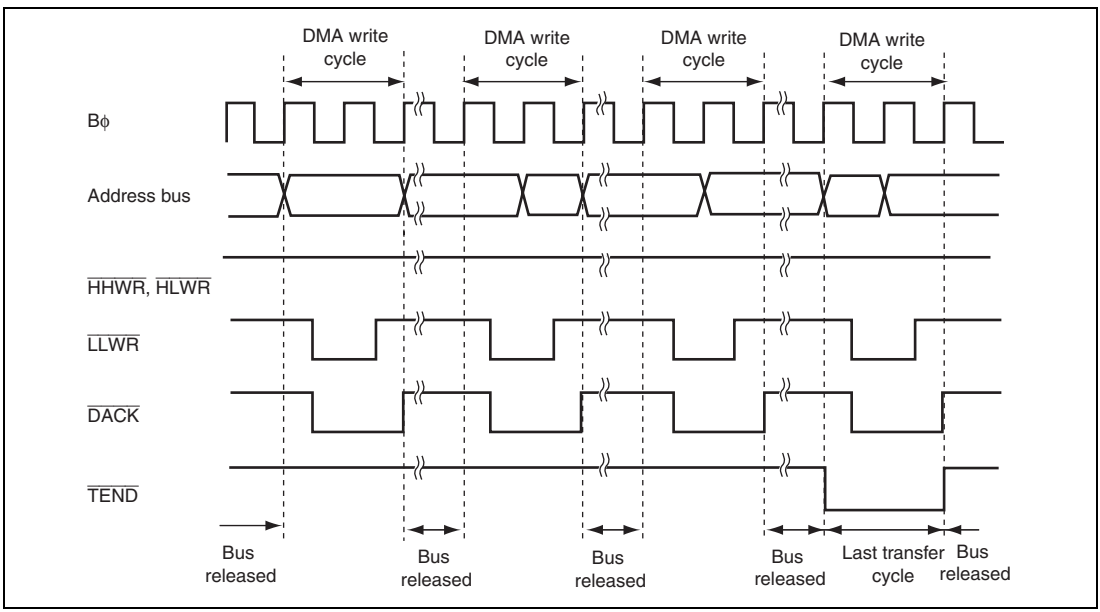


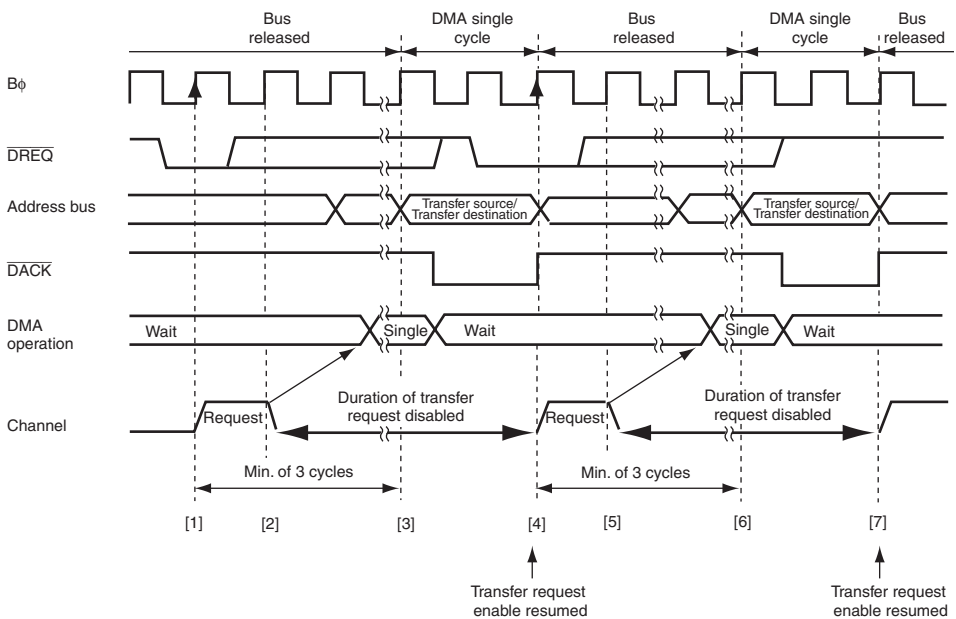
Figure 7.34 Example of Transfer in Single Address Mode (Byte Write)

(3) Activation Timing by $\overline{\text{DREQ}}$ Falling Edge

Figure 7.35 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal falling edge.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared and starts detecting a high level of the $\overline{\text{DREQ}}$ signal for falling edge detection. If a high level of the $\overline{\text{DREQ}}$ signal has been detected until completion of the single cycle, receiving the next transfer request resumes and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started and sampling the $\overline{\text{DREQ}}$ signal at the rising edge of the Bϕ signal is started to detect a high level of the $\overline{\text{DREQ}}$ signal.
[4][7] When a high level of the $\overline{\text{DREQ}}$ signal has been detected, transfer enable is resumed after completion of the write cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

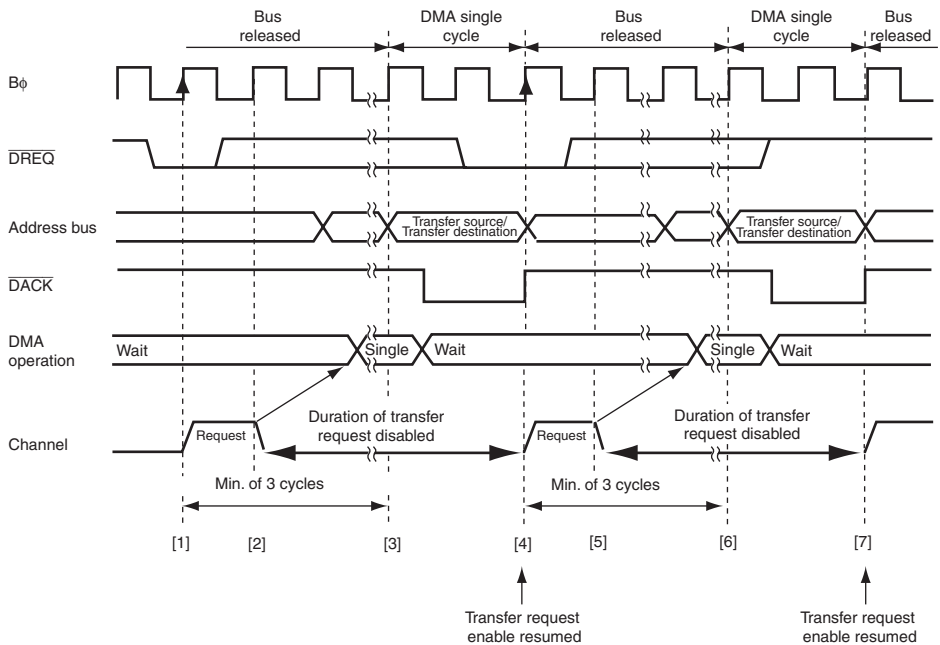
Figure 7.35 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Falling Edge

(4) Activation Timing by $\overline{\text{DREQ}}$ Low Level

Figure 7.36 shows an example of normal transfer mode activated by the $\overline{\text{DREQ}}$ signal low level.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed after completion of the single cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.36 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level

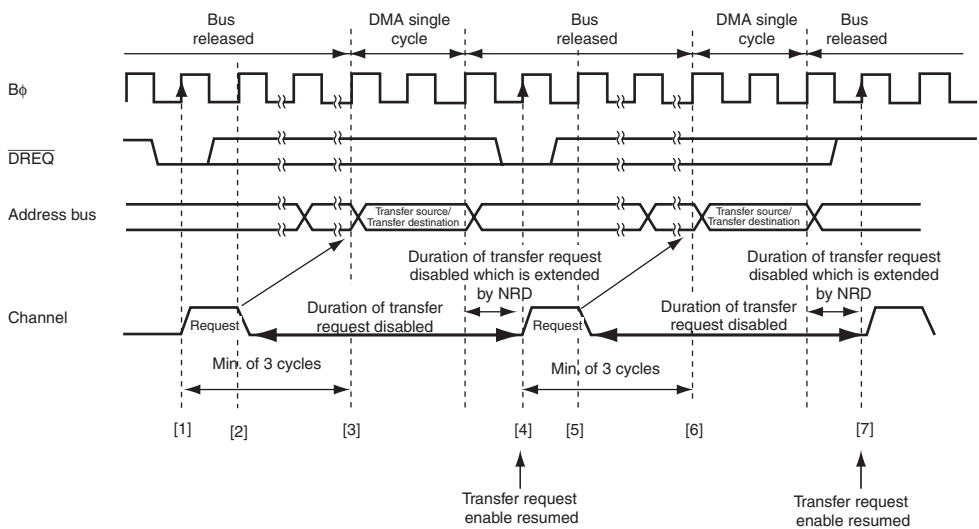
(5) Activation Timing by $\overline{\text{DREQ}}$ Low Level with NRD = 1

When the NRD bit in DMDR is set to 1, the timing of receiving the next transfer request is delayed for one cycle.

Figure 7.37 shows an example of single address mode activated by the $\overline{\text{DREQ}}$ signal low level with NRD = 1.

The $\overline{\text{DREQ}}$ signal is sampled every cycle from the next rising edge of the Bϕ signal immediately after the DTE bit write cycle.

When a low level of the $\overline{\text{DREQ}}$ signal is detected while a transfer request by the $\overline{\text{DREQ}}$ signal is enabled, a transfer request is held in the DMAC. When the DMAC is activated, the transfer request is cleared. Receiving the next transfer request resumes after one cycle of the transfer request duration inserted by NRD = 1 on completion of the single cycle and then a low level of the $\overline{\text{DREQ}}$ signal is detected. This operation is repeated until the transfer is completed.



[1] After DMA transfer request is enabled, a low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held.
[2][5] The DMAC is activated and the transfer request is cleared.
[3][6] A DMA cycle is started.
[4][7] Transfer request enable is resumed one cycle after completion of the single cycle.
(A low level of the $\overline{\text{DREQ}}$ signal is detected at the rising edge of the Bϕ signal and a transfer request is held. This is the same as [1].)

Figure 7.37 Example of Transfer in Single Address Mode Activated by $\overline{\text{DREQ}}$ Low Level with NRD = 1

7.5 DMA Transfer End

Operations on completion of a transfer differ according to the transfer end condition. DMA transfer completion is indicated that the DTE and ACT bits in DMDR are changed from 1 to 0.

(1) Transfer End by DTCR Change from 1, 2, or 4, to 0

When DTCR is changed from 1, 2, or 4 to 0, a DMA transfer for the channel is completed. The DTE bit in DMDR is cleared to 0 and the DTIF bit in DMDR is set to 1. At this time, when the DTIE bit in DMDR is set to 1, a transfer end interrupt by the transfer counter is requested. When the DTCR value is 0 before the transfer, the transfer is not stopped.

(2) Transfer End by Transfer Size Error Interrupt

When the following conditions are satisfied while the TSEIE bit in DMDR is set to 1, a transfer size error occurs and a DMA transfer is terminated. At this time, the DTE bit in DMR is cleared to 0 and the ESIF bit in DMDR is set to 1.

- In normal transfer mode and repeat transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the data access size
- In block transfer mode, when the next transfer is requested while a transfer is disabled due to the DTCR value less than the block size

When the TSEIE bit in DMDR is cleared to 0, data is transferred until the DTCR value reaches 0. A transfer size error is not generated. Operation in each transfer mode is shown below.

- In normal transfer mode and repeat transfer mode, when the DTCR value is less than the data access size, data is transferred in bytes
- In block transfer mode, when the DTCR value is less than the block size, the specified size of data in DTCR is transferred instead of transferring the block size of data. The transfer is performed in bytes.

(3) Transfer End by Repeat Size End Interrupt

In repeat transfer mode, when the next transfer is requested after completion of a 1-repeat size data transfer while the RPTIE bit in DACR is set to 1, a repeat size end interrupt is requested. When the interrupt is requested to complete DMA transfer, the DTE bit in DMDR is cleared to 0 and the ESIF bit in DMDR is set to 1. Under this condition, setting the DTE bit to 1 resumes the transfer.

In block transfer mode, when the next transfer is requested after completion of a 1-block size data transfer, a repeat size end interrupt can be requested.

(4) Transfer End by Interrupt on Extended Repeat Area Overflow

When an overflow on the extended repeat area occurs while the extended repeat area is specified and the SARIE or DARIE bit in DACR is set to 1, an interrupt by an extended repeat area overflow is requested. When the interrupt is requested, the DMA transfer is terminated, the DTE bit in DMDR is cleared to 0, and the ESIF bit in DMDR is set to 1.

In dual address mode, even if an interrupt by an extended repeat area overflow occurs during a read cycle, the following write cycle is performed.

In block transfer mode, even if an interrupt by an extended repeat area overflow occurs during a 1-block transfer, the remaining data is transferred. The transfer is not terminated by an extended repeat area overflow interrupt unless the current transfer is complete.

(5) Transfer End by Clearing DTE Bit in DMDR

When the DTE bit in DMDR is cleared to 0 by the CPU, a transfer is completed after the current DMA cycle and a DMA cycle in which the transfer request is accepted are completed.

In block transfer mode, a DMA transfer is completed after 1-block data is transferred.

(6) Transfer End by NMI Interrupt

When an NMI interrupt is requested, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an NMI interrupt is requested during a DMA transfer, the transfer is forced to stop. To perform DMA transfer after an NMI interrupt is requested, clear the ERRF bit to 0 and then set the DTE bits for the channels to 1.

The transfer end timings after an NMI interrupt is requested are shown below.

(a) Normal Transfer Mode and Repeat Transfer Mode

In dual address mode, a DMA transfer is completed after completion of the write cycle for one transfer unit.

In single address mode, a DMA transfer is completed after completion of the bus cycle for one transfer unit.

(b) Block Transfer Mode

A DMA transfer is forced to stop. Since a 1-block size of transfers is not completed, operation is not guaranteed.
In dual address mode, the write cycle corresponding to the read cycle is performed. This is similar to (a) in normal transfer mode.

(7) Transfer End by Address Error

When an address error occurs, the DTE bits for all the channels are cleared to 0 and the ERRF bit in DMDR_0 is set to 1. When an address error occurs during a DMA transfer, the transfer is forced to stop. To perform a DMA transfer after an address error occurs, clear the ERRF bit to 0 and then set the DTE bits for the channels.

The transfer end timing after an address error is the same as that after an NMI interrupt.

(8) Transfer End by Hardware Standby Mode or Reset

The DMAC is initialized by a reset and a transition to the hardware standby mode. A DMA transfer is not guaranteed.

7.6 Relationship among DMAC and Other Bus Masters

7.6.1 CPU Priority Control Function Over DMAC

The CPU priority control function over DMAC can be used according to the CPU priority control register (CPUPCR) setting. For details, see section 5.7, CPU Priority Control Function Over DMAC.

The priority level of the DMAC is specified by bits DMAP2 to DMAP0 and can be specified for each channel.

The priority level of the CPU is specified by bits CPUP2 to CPUP0. The value of bits CPUP2 to CPUP0 is updated according to the exception handling priority.

If the CPU priority control is enabled by the CPUPCE bit in CPUPCR, when the CPU has priority over the DMAC, a transfer request for the corresponding channel is masked and the transfer is not activated. When another channel has priority over or the same as the CPU, a transfer request is received regardless of the priority between channels and the transfer is activated.

If the priority level of the transfer request masked by the CPU priority control function is changed or the CPU priority is changed, the transfer request may be received and the transfer is started.

When the CPUPCE bit is cleared to 0, it is regarded as the lowest priority. Transfer requests masked are suspended. If a transfer request is suspended, it is cleared by clearing the DTE bit to 0.

7.6.2 Bus Arbitration among DMAC and Other Bus Masters

When DMA transfer cycles are consecutively performed, bus cycles of other bus masters may be inserted between the transfer cycles. The DMAC can release the bus temporarily to pass the bus to other bus masters.

The consecutive DMA transfer cycles may not be divided according to the transfer mode settings to achieve high-speed access.

The read and write cycles of a DMA transfer are not separated. Refreshing, external bus release, and on-chip bus master (CPU) cycles are not inserted between the read and write cycles of a DMA transfer.

In block transfer mode and an auto request transfer by burst access, bus cycles of the DMA transfer are consecutively performed. For this duration, since the DMAC has priority over the CPU, accesses to the external space is suspended (the IBCCS bit in the bus control register 2 (BCR2) is cleared to 0).

When the bus is passed to another channel or an auto request transfer by cycle stealing, bus cycles of the DMAC and on-chip bus master are performed alternatively.

When the arbitration function among the DMAC and on-chip bus masters is enabled by setting the IBCCS bit in BCR2, the bus is used alternatively except the bus cycles which are not separated. For details, see section 6, Bus Controller (BSC).

A conflict may occur between external space access of the DMAC and a refresh cycle or an external bus release cycle. Even if a burst or block transfer is performed by the DMAC, the transfer is stopped temporarily and a cycle of refresh or external bus release is inserted by the BSC (when the CPU external access does not have priority over a DMAC transfer, the transfer is not operated until the DMAC releases the bus).

In dual address mode, the DMAC releases the external bus after the external space write cycle. Since the read and write cycles are not separated, the bus is not released.

An internal space (on-chip memory and internal I/O registers) access of the DMAC and an external bus release cycle may be performed at the same time.

7.7 Interrupt Sources

The DMAC interrupt sources are a transfer end interrupt by the transfer counter and a transfer escape end interrupt which is generated when a transfer is terminated before the transfer counter reaches 0. Table 7.6 shows interrupt sources and priority.

Table 7.6 Interrupt Sources and Priority

Abbr.	Interrupt Sources	Priority
DMTEND0	Transfer end interrupt by channel 0 transfer counter	High
DMTEND1	Transfer end interrupt by channel 1 transfer counter	↑
DMTEND2	Transfer end interrupt by channel 2 transfer counter	
DMTEND3	Transfer end interrupt by channel 3 transfer counter	
DMEEND0	Interrupt by channel 0 transfer size error	
	Interrupt by channel 0 repeat size end	
	Interrupt by channel 0 extended repeat area overflow on source address	
	Interrupt by channel 0 extended repeat area overflow on destination address	
DMEEND1	Interrupt by channel 1 transfer size error	
	Interrupt by channel 1 repeat size end	
	Interrupt by channel 1 extended repeat area overflow on source address	
	Interrupt by channel 1 extended repeat area overflow on destination address	
DMEEND2	Interrupt by channel 2 transfer size error	
	Interrupt by channel 2 repeat size end	
	Interrupt by channel 2 extended repeat area overflow on source address	
	Interrupt by channel 2 extended repeat area overflow on destination address	
DMEEND3	Interrupt by channel 3 transfer size error	
	Interrupt by channel 3 repeat size end	
	Interrupt by channel 3 extended repeat area overflow on source address	
	Interrupt by channel 3 extended repeat area overflow on destination address	Low

Each interrupt is enabled or disabled by the DTIE and ESIE bits in DMDR for the corresponding channel. A DMTEND interrupt is generated by the combination of the DTIF and DTIE bits in DMDR. A DMEEND interrupt is generated by the combination of the ESIF and ESIE bits in DMDR. The DMEEND interrupt sources are not distinguished. The priority among channels are decided by the interrupt controller and it is shown in table 7.6. For details, see section 5, Interrupt Controller.

Each interrupt source is specified by the interrupt enable bit in the register for the corresponding channel. A transfer end interrupt by the transfer counter, a transfer size error interrupt, a repeat size end interrupt, an interrupt by an extended repeat area overflow on the source address, and an interrupt by an extended repeat area overflow on the destination address are enabled or disabled by the DTIE bit in DMDR, the TSEIE bit in DMDR, the RPTIE bit in DACR, SARIE bit in DACR, and the DARIE bit in DACR, respectively.

A transfer end interrupt by the transfer counter is generated when the DTIF bit in DMDR is set to 1. The DTIF bit is set to 1 when DTCR becomes 0 by a transfer while the DTIE bit in DMDR is set to 1.

An interrupt other than the transfer end interrupt by the transfer counter is generated when the ESIF bit in DMDR is set to 1. The ESIF bit is set to 1 when the conditions are satisfied by a transfer while the enable bit is set to 1.

A transfer size error interrupt is generated when the next transfer cannot be performed because the DTCR value is less than the data access size, meaning that the data access size of transfers cannot be performed. In block transfer mode, the block size is compared with the DTCR value for transfer error decision.

A repeat size end interrupt is generated when the next transfer is requested after completion of the repeat size of transfers in repeat transfer mode. Even when the repeat area is not specified in the address register, the transfer can be stopped periodically according to the repeat size. At this time, when a transfer end interrupt by the transfer counter is generated, the ESIF bit is set to 1.

An interrupt by an extended repeat area overflow on the source and destination addresses is generated when the address exceeds the extended repeat area (overflow). At this time, when a transfer end interrupt by the transfer counter, the ESIF bit is set to 1.

Figure 7.38 is a block diagram of interrupts and interrupt flags. To clear an interrupt, clear the DTIF or ESIF bit in DMDR to 0 in the interrupt handling routine or continue the transfer by setting the DTE bit in DMDR after setting the register. Figure 7.39 shows procedure to resume the transfer by clearing a interrupt.

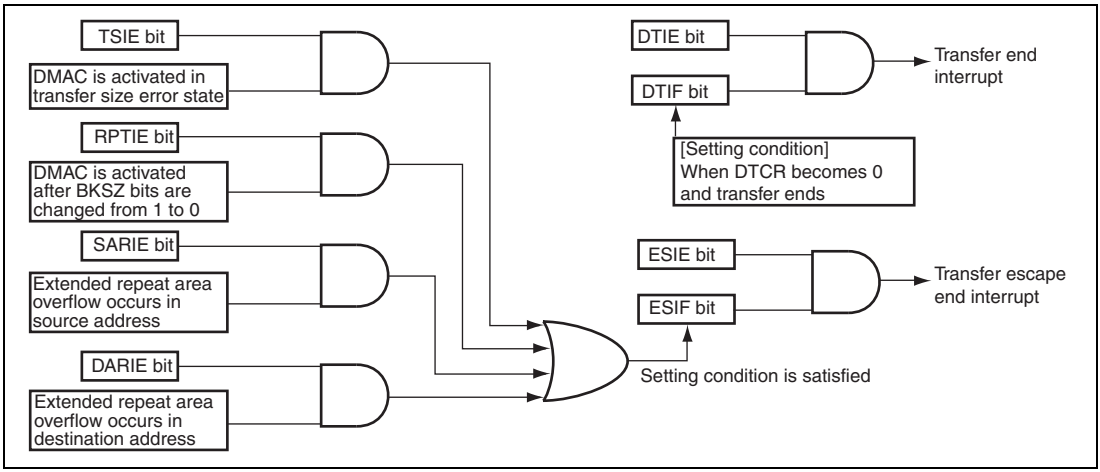
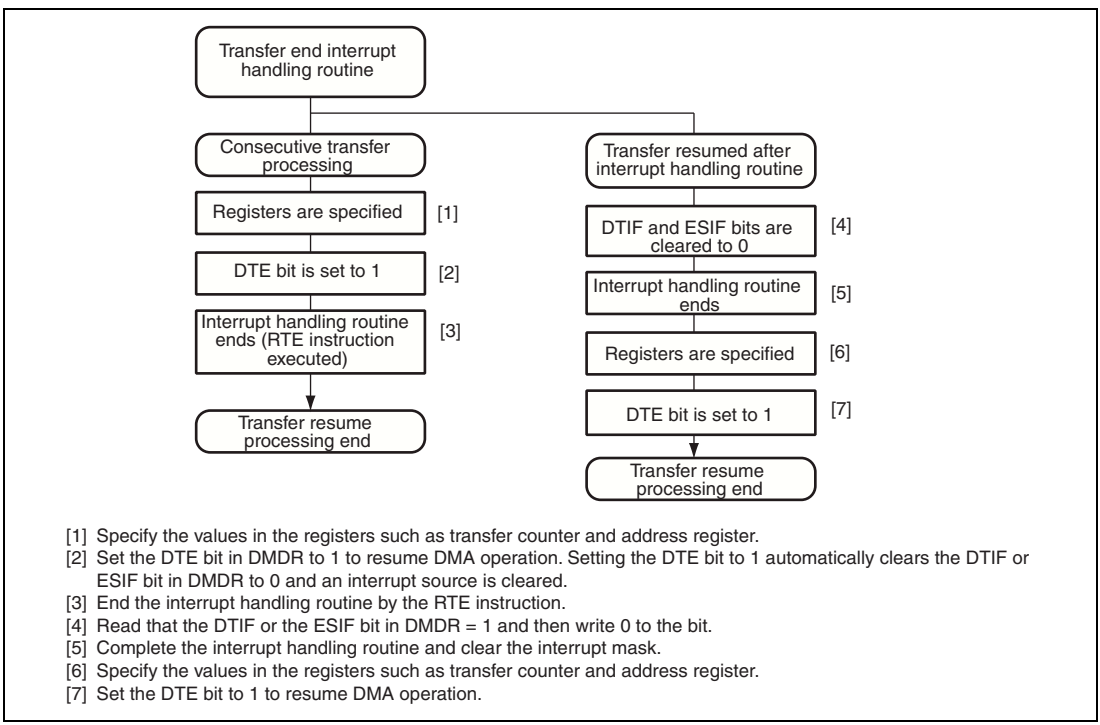


Figure 7.38 Interrupt and Interrupt Sources



[1] Specify the values in the registers such as transfer counter and address register.
[2] Set the DTE bit in DMDR to 1 to resume DMA operation. Setting the DTE bit to 1 automatically clears the DTIF or ESIF bit in DMDR to 0 and an interrupt source is cleared.
[3] End the interrupt handling routine by the RTE instruction.
[4] Read that the DTIF or the ESIF bit in DMDR = 1 and then write 0 to the bit.
[5] Complete the interrupt handling routine and clear the interrupt mask.
[6] Specify the values in the registers such as transfer counter and address register.
[7] Set the DTE bit to 1 to resume DMA operation.

Figure 7.39 Procedure Example of Resuming Transfer by Clearing Interrupt Source

7.8 Notes on Usage

1. DMAC Register Access During Operation

   Except for clearing the DTE bit in DMDR, the settings for channels being transferred (including waiting state) must not be changed. The register settings must be changed during the transfer prohibited state.

2. Settings of Module Stop Function

   The DMAC operation can be enabled or disabled by the module stop control register. The DMAC is enabled by the initial value.

   Setting bit MSTPA13 in MSTPCRA stops the clock supplied to the DMAC and the DMAC enters the module stop state. However, when a transfer for a channel is enabled or when an interrupt is being requested, bit MSTPA13 cannot be set to 1. Clear the DTE bit to 0, clear the DTIF or DTIE bit in DMDR to 0, and then set bit MSTPA13.

   When the clock is stopped, the DMAC registers cannot be accessed. However, the following register settings are valid in the module stop state. Disable them before entering the module stop state, if necessary.

   — TENDE bit in DMDR is 1 (the TEND signal output enabled)

   — DACKE bit in DMDR is 1 (the DACK signal output enabled)

3. Activation by $\overline{\text{DREQ}}$ Falling Edge

   The $\overline{\text{DREQ}}$ falling edge detection is synchronized with the DMAC internal operation.

   A. Activation request waiting state: Waiting for detecting the $\overline{\text{DREQ}}$ low level. A transition to 2. is made.

   B. Transfer waiting state: Waiting for a DMAC transfer. A transition to 3. is made.

   C. Transfer prohibited state: Waiting for detecting the $\overline{\text{DREQ}}$ high level. A transition to 1. is made.

   After a DMAC transfer enabled, a transition to 1. is made. Therefore, the $\overline{\text{DREQ}}$ signal is sampled by low level detection at the first activation after a DMAC transfer enabled.

4. Acceptation of Activation Source

   At the beginning of an activation source reception, a low level is detected regardless of the setting of $\overline{\text{DREQ}}$ falling edge or low level detection. Therefore, if the $\overline{\text{DREQ}}$ signal is driven low before setting DMDR, the low level is received as a transfer request.

   When the DMAC is activated, clear the $\overline{\text{DREQ}}$ signal of the previous transfer.

Section 8 I/O Ports

Table 8.1 summarizes the port functions. The pins of each port also have other functions such as input/output pins of on-chip peripheral modules or external interrupt input pins. Each I/O port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, a port register (PORT) used to read the pin states, and an input buffer control register (ICR) that controls input buffer on/off. Ports 4 and 5 do not have a DR or a DDR register.

Ports D, H, J and K have internal input pull-up MOSs and a pull-up MOS control register (PCR) that controls the on/off state of the input pull-up MOSs.

Port 2 includes an open-drain control register (ODR) that controls on/off of the output buffer PMOSs.

Ports 1, 2, 3, 6, A, D, H, J, and K can drive a single TTL load and capacitive loads up to 30 pF.

All of the I/O ports can drive Darlington transistors when functioning as output ports.

Schmitt-trigger inputs are enabled when a port is used as the $\overline{\text{IRQ}}$ and TPU inputs.

Table 8.1 Port Functions

Port	Description	Bit	Function: I/O	Function: Input	Function: Output	Schmitt-Trigger Input*[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port 1	General I/O port also functioning as interrupt inputs, SCI I/Os, and A/D converter inputs	7	P17	$\overline{\text{ADTRG1}}$/ $\overline{\text{IRQ7}}$	—	$\overline{\text{IRQ7}}$	—	—
		6	P16/SCK3	$\overline{\text{IRQ6}}$	—	$\overline{\text{IRQ6}}$		
		5	P15	RxD3/ $\overline{\text{IRQ5}}$	—	$\overline{\text{IRQ5}}$		
		4	P14	$\overline{\text{IRQ4}}$	TxD3	$\overline{\text{IRQ4}}$		
		3	P13	$\overline{\text{ADTRG0}}$/ $\overline{\text{IRQ3}}$	—	$\overline{\text{IRQ3}}$		
		2	P12	$\overline{\text{IRQ2}}$	—	$\overline{\text{IRQ2}}$		
		1	P11	$\overline{\text{IRQ1}}$	—	$\overline{\text{IRQ1}}$		
		0	P10	$\overline{\text{IRQ0}}$	—	$\overline{\text{IRQ0}}$		

Port	Description	Bit	Function: I/O	Function: Input	Function: Output	Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port 2	General I/O port also functioning as interrupt inputs, TPU I/Os*[2], and SSU I/Os	3	P23/ TIOCD3	TIOCC3/ $\overline{\text{IRQ11}}$-A	—	P23, TIOCC3, TIOCD3, $\overline{\text{IRQ11}}$-A	—	O
		2	P22/ TIOCC3	$\overline{\text{IRQ10}}$-A	—	P22, TIOCC3, $\overline{\text{IRQ10}}$-A		
		1	P21/ TIOCA3/ $\overline{\text{SCS2}}$	$\overline{\text{IRQ9}}$-A	—	P21, TIOCA3, $\overline{\text{IRQ9}}$-A		
		0	P20/ TIOCB3	TIOCA3/ $\overline{\text{IRQ8}}$-A	—	P20, TIOCA3, TIOCB3, $\overline{\text{IRQ8}}$-A		
Port 3	General I/O port also functioning as PPG outputs*[2] and TPU I/Os	7	P37/ TIOCB2	TIOCA2/ TCLKD	PO15	P37, TIOCA2, TIOCB2, TCLKD	—	—
		6	P36/ TIOCA2	—	PO14	P36, TIOCA2		
		5	P35/ TIOCB1	TIOCA1/ TCLKC	PO13	P35, TIOCA1, TIOCB1, TCLKC		
		4	P34/ TIOCA1	—	PO12	P34, TIOCA1		
		3	P33/ TIOCD0	TIOCC0/ TCLKB	PO11	P33, TIOCC0, TIOCD0, TCLKB		
		2	P32/ TIOCC0	TCLKA	PO10	P32, TIOCC0, TCLKA		
		1	P31/ TIOCB0	TIOCA0	PO9	P31, TIOCA0, TIOCB0		
		0	P30/ TIOCA0	—	PO8	P30, TIOCA0		

Port	Description	Bit	Function: I/O	Function: Input	Function: Output	Schmitt-Trigger Input *1	Input Pull-up MOS Function	Open-Drain Output Function
Port 4	General I/O port also functioning as A/D converter inputs	7	—	P47/AN11	—	—	—	—
		6	—	P46/AN10	—			
		5	—	P45/AN9	—			
		4	—	P44/AN8	—			
		3	—	P43/AN15	—			
		2	—	P42/AN14	—			
		1	—	P41/AN13	—			
		0	—	P40/AN12	—			
Port 5	General I/O port also functioning as A/D converter inputs	7	—	P57/AN7	—	—	—	—
		6	—	P56/AN6	—			
		5	—	P55/AN5	—			
		4	—	P54/AN4	—			
		3	—	P53/AN3	—			
		2	—	P52/AN2	—			
		1	—	P51/AN1	—			
		0	—	P50/AN0	—			
Port 6	General I/O port also functioning as SCI I/Os, interrupt inputs, and RCAN-ET I/Os	6	P66	$\overline{\text{IRQ14}}$	—	$\overline{\text{IRQ14}}$		
		5	P65	$\overline{\text{IRQ13}}$/ CRx	—	$\overline{\text{IRQ13}}$		
		4	P64	$\overline{\text{IRQ12}}$	CTx	$\overline{\text{IRQ12}}$		
		3	P63	$\overline{\text{IRQ11}}$-B	—	$\overline{\text{IRQ11}}$-B		
		2	P62/SCK4	$\overline{\text{IRQ10}}$-B	—	$\overline{\text{IRQ10}}$-B		
		1	P61	RxD4/ $\overline{\text{IRQ9}}$-B	—	$\overline{\text{IRQ9}}$-B		
		0	P60	$\overline{\text{IRQ8}}$-B	TxD4	$\overline{\text{IRQ8}}$-B		
Port A	General I/O port also functioning as SSU I/Os and Bϕ output	7	—	PA7	Bϕ	—	—	O
		6	PA6	—	—			Only for SSU
		5	PA5	—	—			
		4	PA4	—	—			
		3	PA3/SSO2	—	—			
		2	PA2/SSI2	—	—			
		1	PA1/SSCK2	—	—			

Port	Description	Bit	Function: I/O	Function: Input	Function: Output	Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port D	General I/O port also functioning as SSU I/Os	7	PD7/$\overline{\text{SCS1}}$	—	—	—	O	O Only for SSU
		6	PD6/SSCK1	—	—			
		5	PD5/SSI1	—	—			
		4	PD4/SSO1	—	—			
		3	PD3/$\overline{\text{SCS0}}$	—	—			
		2	PD2/SSCK0	—	—			
		1	PD1/SSI0	—	—			
		0	PD0/SSO0	—	—			
Port H	General I/O port	7	PH7	—	—	—	O	—
		6	PH6	—	—			
		5	PH5	—	—			
		4	PH4	—	—			
		3	PH3	—	—			
		2	PH2	—	—			
		1	PH1	—	—			
		0	PH0	—	—			
Port J	General I/O port also functioning as TPU I/Os	7	PJ7/ TIOCB8	TIOCA8/ TCLKH	—	PJ7, TIOCA8, TIOCB8, TCLKH	O	—
		6	PJ6/ TIOCA8	—	—	PJ6, TIOCA8		
		5	PJ5/ TIOCB7	TIOCA7/ TCLKG	—	PJ5, TIOCA7, TIOCB7, TCLKG		
		4	PJ4/ TIOCA7	—	—	PJ4, TIOCA7		
		3	PJ3/ TIOCD6	TIOCC6/ TCLKF	—	PJ3, TIOCC6, TIOCD6, TCLKF		
		2	PJ2/ TIOCC6	TCLKE	—	PJ2, TIOCC6, TCLKE		

Port	Description	Bit	Function: I/O	Function: Input	Function: Output	Schmitt-Trigger Input *[1]	Input Pull-up MOS Function	Open-Drain Output Function
Port J	General I/O port also functioning as TPU I/Os	1	PJ1/ TIOCB6	TIOCA6	—	PJ1, TIOCA6, TIOCB6	O	—
		0	PJ0/ TIOCA6	—	—	PJ0, TIOCA6		
Port K	General I/O port also functioning as TPU I/Os	7	PK7/ TIOCB11	TIOCA11	—	PK7, TIOCA11, TIOCB11	O	—
		6	PK6/ TIOCA11	—	—	PK6, TIOCA11		
		5	PK5/ TIOCB10	TIOCA10	—	PK5, TIOCA10, TIOCB10		
		4	PK4/ TIOCA10	—	—	PK4, TIOCA10		
		3	PK3/ TIOCD9	TIOCC9	—	PK3, TIOCC9, TIOCD9		
		2	PK2/ TIOCC9	—	—	PK2, TIOCC9		
		1	PK1/ TIOCB9	TIOCA9	—	PK1, TIOCA9, TIOCB9		
		0	PK0/ TIOCA9	—	—	PK0, TIOCA9		

Notes: 1. Pins without Schmitt-trigger input buffer have CMOS input buffer.
2. Supported only by the H8SX/1527R.

8.1 Register Descriptions

Table 8.2 lists each port registers.

Table 8.2 Register Configuration in Each Port

Port	Number of Pins	Registers						
		DDR	DR	PORT	ICR	PCR	ODR	PHRTIDR
Port 1	8	O	O	O	O	—	—	—
Port 2*[1]	4	O	O	O	O	—	O	—
Port 3	8	O	O	O	O	—	—	—
Port 4	8	—	—	O	O	—	—	—
Port 5	8	—	—	O	O	—	—	—
Port 6*[2]	7	O	O	O	O	—	—	—
Port A*[3]	7	O	O	O	O	—	—	—
Port D	8	O	O	O	O	O	—	—
Port H	8	O	O	O	O	O	—	O
Port J	8	O	O	O	O	O	—	—
Port K	8	O	O	O	O	O	—	—

[Legend]

O: Register exists

—: No register exists

Notes: 1. The lower four bits are valid and the upper four bits are reserved. The write value should always be the initial value.

2. The lower seven bits are valid and the upper one bit is reserved. The write value should always be the initial value.

3. The upper seven bits are valid and the lower one bit is reserved. The write value should always be the initial value.

Figure 8.1 is a port block diagram.

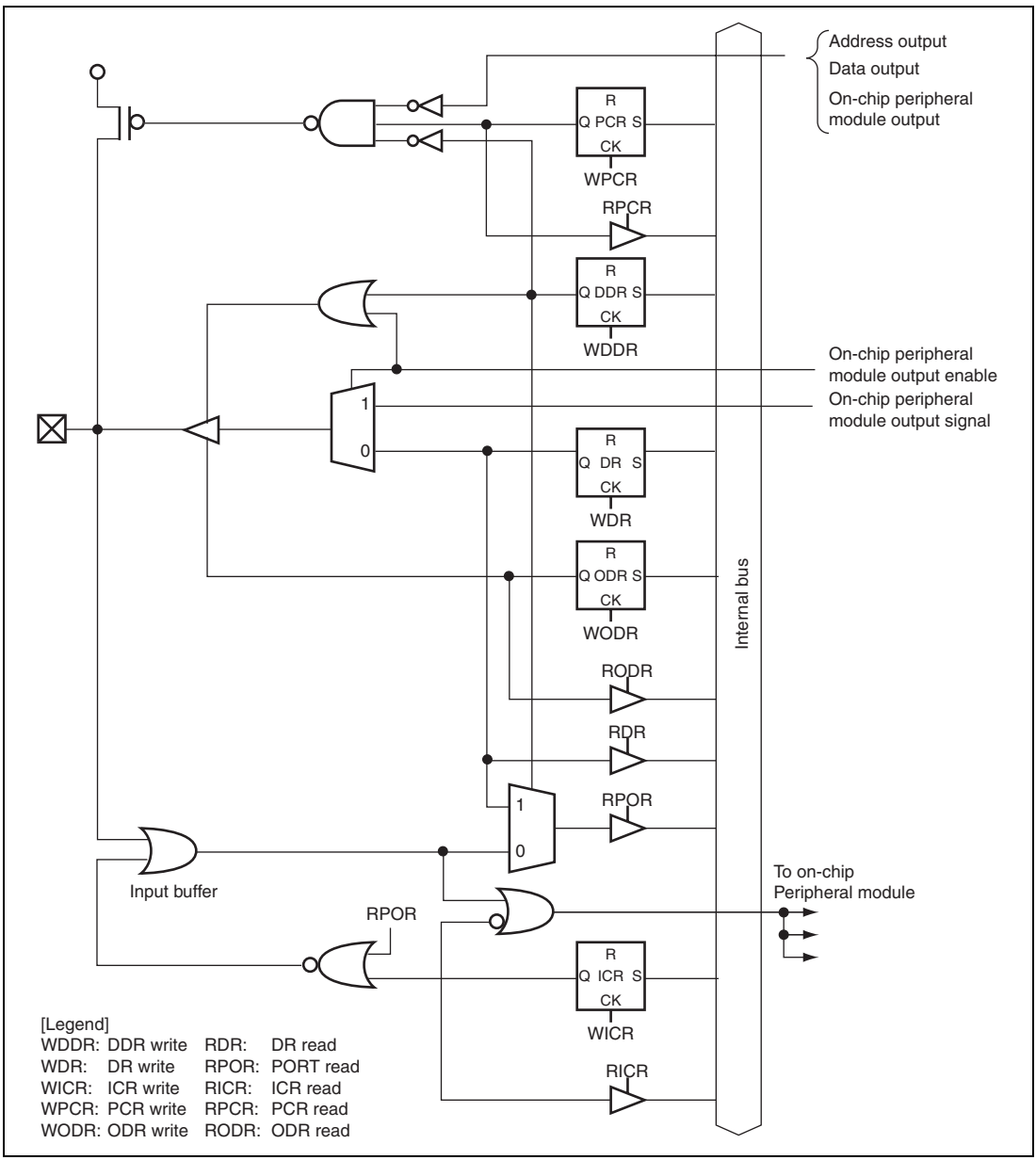


Figure 8.1 Port Block Diagram

8.1.1 Data Direction Register (PnDDR) (n = 1 to 3, 6, A, D, H, J, and K)

DDR is an 8-bit write-only register that specifies the port input or output for each bit. A read from the DDR is invalid and DDR is always read as an undefined value.

When the general I/O port function is selected, the corresponding pin functions as an output port by setting the corresponding DDR bit to 1; the corresponding pin functions as an input port by clearing the corresponding DDR bit to 0.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DDR	Pn6DDR	Pn5DDR	Pn4DDR	Pn3DDR	Pn2DDR	Pn1DDR	Pn0DDR
Initial Value	0	0	0	0	0	0	0	0
R/W	W	W	W	W	W	W	W	W

Note: The lower four bits are valid and the upper four bits are reserved for port 2 data direction register (P2DDR).
The lower seven bits are valid and the upper one bit is reserved for port 6 data direction register (P6DDR).
The upper seven bits are valid and the lower one bit is reserved for port A data direction register (PADDR).

8.1.2 Data Register (PnDR) (n = 1 to 3, 6, A, D, H, J, and K)

DR is an 8-bit readable/writable register that stores the output data of the pins to be used as the general output port.

The initial value of DR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7DR	Pn6DR	Pn5DR	Pn4DR	Pn3DR	Pn2DR	Pn1DR	Pn0DR
Initial Value	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

Note: The lower four bits are valid and the upper four bits are reserved for port 2 data register (P2DR).
The lower seven bits are valid and the upper one bit is reserved for port 6 data register (P6DR).
The upper seven bits are valid and the lower one bit is reserved for port A data register (PADR).

8.1.3 Port Register (PORTn) (n = 1 to 6, A, D, H, J, and K)

PORT is an 8-bit read-only register that reflects the port pin status. A write to PORT is invalid. When PORT is read, the DR bits that correspond to the respective DDR bits set to 1 are read and the status of each pin whose corresponding DDR bit is cleared to 0 is also read regardless of the ICR value.

The initial value of PORT is undefined and is determined based on the port pin status.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7	Pn6	Pn5	Pn4	Pn3	Pn2	Pn1	Pn0
Initial Value	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	R	R	R	R	R	R	R	R

Note: The lower four bits are valid and the upper four bits are reserved for port 2 register (PORT2).
The lower seven bits are valid and the upper one bit is reserved for port 6 register (PORT6).
The upper seven bits are valid and the lower one bit is reserved for port A register (PORTA).

8.1.4 Input Buffer Control Register (PnICR) (n = 1 to 6, A, D, H, J, and K)

ICR is an 8-bit readable/writable register that controls the port input buffers.

For bits in ICR set to 1, the input buffers of the corresponding pins are valid. For bits in ICR cleared to 0, the input buffers of the corresponding pins are invalid and the input signals are fixed high.

When the pin functions as an input for the peripheral modules, the corresponding bits should be set to 1. The initial value should be written to a bit whose corresponding pin is not used as an input or is used as an analog input/output pin.

When PORT is read, the pin status is always read regardless of the ICR value. On-chip modules are not affected by the pin status when the ICR value is cleared to 0.

If ICR is modified, an internal edge may occur depending on the pin status. Accordingly, ICR should be modified when the corresponding input pins are not used. For example, in $\overline{\text{IRQ}}$ input, modify ICR while the corresponding interrupt is disabled, clear the IRQF flag in ISR of the interrupt controller to 0, and then enable the corresponding interrupt. If an edge occurs after the ICR setting, the edge should be cancelled.

The initial value of ICR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ICR	Pn6ICR	Pn5ICR	Pn4ICR	Pn3ICR	Pn2ICR	Pn1ICR	Pn0ICR
Initial Value	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

Note: The lower four bits are valid and the upper four bits are reserved for port 2 input buffer control register (P2ICR).
The lower seven bits are valid and the upper one bit is reserved for port 6 input buffer control register (P6ICR).
The upper seven bits are valid and the lower one bit is reserved for port A input buffer control register (PAICR).

8.1.5 Pull-Up MOS Control Register (PnPCR) (n = D, H, J, and K)

PCR is an 8-bit readable/writable register that controls on/off of the port input pull-up MOS.

If a bit in PCR is set to 1 while the pin is in input state, the input pull-up MOS corresponding to the bit in PCR is turned on. Table 8.3 shows the input pull-up MOS status.

The initial value of PCR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7PCR	Pn6PCR	Pn5PCR	Pn4PCR	Pn3PCR	Pn2PCR	Pn1PCR	Pn0PCR
Initial Value	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

Table 8.3 Input Pull-Up MOS State

Port	Pin State	Reset	Software Standby Mode	Other Operation
Port D	On-chip peripheral module output	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF
Port H	Port output	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF
Port J	On-chip peripheral module output	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF
Port K	On-chip peripheral module output	OFF	OFF	OFF
	Port input	OFF	OFF	ON/OFF

[Legend]

OFF: The input pull-up MOS is always off.

ON/OFF: If PCR is set to 1, the input pull-up MOS is on; if PCR is cleared to 0, the input pull-up MOS is off.

8.1.6 Open-Drain Control Register (PnODR) (n = 2)

ODR is an 8-bit readable/writable register that selects the open-drain output function.

If a bit in ODR is set to 1, the pin corresponding to that bit in ODR functions as an NMOS open-drain output. If a bit in ODR is cleared to 0, the pin corresponding to that bit in ODR functions as a CMOS output.

The initial value of ODR is H'00.

Bit	7	6	5	4	3	2	1	0
Bit Name	Pn7ODR	Pn6ODR	Pn5ODR	Pn4ODR	Pn3ODR	Pn2ODR	Pn1ODR	Pn0ODR
Initial Value	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

Note: The lower four bits are valid and the upper four bits are reserved for port 2 open drain control register (P2ODR).

8.1.7 Port H Realtime Input Data Register (PHRTIDR)

PHRTIDR stores the status of port H using pin $\overline{\text{IRQ14}}$ as a trigger. The detection method is specified by the IRQ14SR and IRQ14SF bits in the IRQ sense control register H (ISCRH) and is selected from a low level, a falling edge, a rising edge of pin, and both edges of pin $\overline{\text{IRQ14}}$. For details, see section 5.3.5, IRQ Sense Control Registers H and L (ISCRH and ISCRL).

Bit	7	6	5	4	3	2	1	0
Bit Name	PHRTIDR7	PHRTIDR6	PHRTIDR5	PHRTIDR4	PHRTIDR3	PHRTIDR2	PHRTIDR1	PHRTIDR0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

8.2 Output Buffer Control

This section describes the output priority of each pin.

The name of each peripheral module pin is followed by "_OE". This (for example: MIOCA4_OE) indicates whether the output of the corresponding function is valid (1) or if another setting is specified (0). Table 8.4 lists each port output signal's valid setting. For details on the corresponding output signals, see the register description of each peripheral module. If the name of each peripheral module pin is followed by A or B, the pin function can be modified by the port function control register (PFCR). For details, see section 8.3.3, Port Function Control Register B (PFCRB).

8.2.1 Port 1

(1) P17/ADTRG1/IRQ7

The pin function is switched as shown below according to the P17DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P17DDR
I/O port	P17 output	1
	P17 input (initial setting)	0

(2) P16/SCK3/$\overline{\text{IRQ6}}$

The pin function is switched as shown below according to the combination of the SCI_3 and P16DDR bit settings.

		Setting	
		SCI_3	I/O Port
Module Name	Pin Function	SCK3_OE	P16DDR
SCI_3	SCK3 output	1	—
I/O port	P16 output	0	1
	P16 input (initial setting)	0	0

(3) P15/RxD3/$\overline{\text{IRQ5}}$

The pin function is switched as shown below according to the P15DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P15DDR
I/O port	P15 output	1
	P15 input (initial setting)	0

(4) P14/TxD3/$\overline{\text{IRQ4}}$

The pin function is switched as shown below according to the combination of the SCI_3 and P14DDR bit settings.

		Setting	
		SCI_4	I/O Port
Module Name	Pin Function	TxD3_OE	P14DDR
SCI_3	TxD3 output	1	—
I/O port	P14 output	0	1
	P14 input (initial setting)	0	0

(5) P13/ADTRG0/IRQ3

The pin function is switched as shown below according to the P13DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P13DDR
I/O port	P13 output	1
	P13 input (initial setting)	0

(6) P12/$\overline{\text{IRQ2}}$

The pin function is switched as shown below according to the P12DDR bit setting.

		Setting
		I/O Port
Module Name	**Pin Function**	**P12DDR**
I/O port	P12 output	1
	P12 input (initial setting)	0

(7) P11/$\overline{\text{IRQ1}}$

The pin function is switched as shown below according to the P11DDR bit setting.

		Setting
		I/O Port
Module Name	**Pin Function**	**P11DDR**
I/O port	P11 output	1
	P11 input (initial setting)	0

(8) P10/$\overline{\text{IRQ0}}$

The pin function is switched as shown below according to the P10DDR bit setting.

		Setting
		I/O Port
Module Name	**Pin Function**	**P10DDR**
I/O port	P10 output	1
	P10 input (initial setting)	0

8.2.2 Port 2

(1) P23/TIOCC3/TIOCD3/$\overline{\text{IRQ11}}$-A

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_3, and P23DDR bit settings.

		Setting	
		TPU_3*	**I/O Port**
Module Name	**Pin Function**	**TIOCD3_OE**	**P23DDR**
TPU_3*	TIOCD3 output	1	—
I/O port	P23 output	0	1
	P23 input (initial setting)	0	0

Note: * Supported only by the H8SX/1527R.

(2) P22/TIOCC3/$\overline{\text{IRQ10}}$-A

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_3, and P22DDR bit settings.

		Setting	
		TPU_3*	**I/O Port**
Module Name	**Pin Function**	**TIOCD3_OE**	**P22DDR**
TPU_3*	TIOCC3 output	1	—
I/O port	P22 output	0	1
	P22 input (initial setting)	0	0

Note: * Supported only by the H8SX/1527R.

(3) P21/TIOCA3/$\overline{\text{IRQ9}}$-A/$\overline{\text{SCS2}}$

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), SSU_2, TPU_3, and P21DDR bit settings.

		Setting		
		SSU_2	**TPU_3***	**I/O Port**
Module Name	**Pin Function**	**$\overline{\text{SCS2}}$_OE**	**TIOCA3_OE**	**P21DDR**
SSU_2	$\overline{\text{SCS2}}$ output	1	—	—
TPU_3*	TIOCA3 output	0	1	—
I/O port	P21 output	0	0	1
	P21 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(4) P20/TIOCA3/TIOCB3/$\overline{\text{IRQ8}}$-A

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_3, and P20DDR bit settings.

		Setting	
		TPU_3*	**I/O Port**
Module Name	**Pin Function**	**TIOCB3_OE**	**P20DDR**
TPU_3*	TIOCB3 output	1	—
I/O port	P20 output	0	1
	P20 input (initial setting)	0	0

Note: * Supported only by the H8SX/1527R.

8.2.3 Port 3

(1) P37/PO15/TIOCA2/TIOCB2/TCLKD

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_2, PPG, and P37DDR bit settings.

		Setting		
		TPU_2*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCB2_OE**	**PO15_OE**	**P37DDR**
TPU_2*	TIOCB2 output	1	—	—
PPG*	PO15 output	0	1	—
I/O port	P37 output	0	0	1
	P37 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(2) P36/PO14/TIOCA2

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_2, PPG, and P36DDR bit settings.

		Setting		
		TPU_2*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCA2_OE**	**PO14_OE**	**P36DDR**
TPU_2*	TIOCA2 output	1	—	—
PPG*	PO14 output	0	1	—
I/O port	P36 output	0	0	1
	P36 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(3) P35/PO13/TIOCA1/TIOCB1/TCLKC

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_1, PPG, and P35DDR bit settings.

		Setting		
		TPU_1*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCB1_OE**	**PO13_OE**	**P35DDR**
TPU_1*	TIOCB1 output	1	—	—
PPG*	PO13 output	0	1	—
I/O port	P35 output	0	0	1
	P35 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(4) P34/PO12/TIOCA1

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_1, PPG, and P34DDR bit settings.

		Setting		
		TPU_1*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCA1_OE**	**PO12_OE**	**P34DDR**
TPU_1*	TIOCA1 output	1	—	—
PPG*	PO12 output	0	1	—
I/O port	P34 output	0	0	1
	P34 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(5) P33/PO11/TIOCC0/TIOCD0/TCLKB

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_0, PPG, and P33DDR bit settings.

		Setting		
		TPU_0*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCD0_OE**	**PO11_OE**	**P33DDR**
TPU_0*	TIOCD0 output	1	—	—
PPG*	PO11 output	0	1	—
I/O port	P33 output	0	0	1
	P33 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(6) P32/PO10/TIOCC0/TCLKA:

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_0, PPG, and P32DDR bit settings.

		Setting		
		TPU_0*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCC0_OE**	**PO10_OE**	**P32DDR**
TPU_0*	TIOCC0 output	1	—	—
PPG*	PO10 output	0	1	—
I/O port	P32 output	0	0	1
	P32 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(7) P31/PO9/TIOCA0/TIOCB0

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_0, PPG, and P31DDR bit settings.

		Setting		
		TPU_0*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCB0_OE**	**PO9_OE**	**P31DDR**
TPU_0*	TIOCB0 output	1	—	—
PPG*	PO9 output	0	1	—
I/O port	P31 output	0	0	1
	P31 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

(8) P30/PO8/TIOCA0

The pin function is switched as shown below according to the combination of the port function control register 9 (PFCR9), TPU_0, PPG, and P30DDR bit settings.

		Setting		
		TPU_0*	**PPG***	**I/O Port**
Module Name	**Pin Function**	**TIOCA0_OE**	**PO8_OE**	**P30DDR**
TPU_0*	TIOCA0 output	1	—	—
PPG*	PO8 output	0	1	—
I/O port	P30 output	0	0	1
	P30 input (initial setting)	0	0	0

Note: * Supported only by the H8SX/1527R.

8.2.4 Port 6

(1) P66/$\overline{\text{IRQ14}}$

The pin function is switched as shown below according to the P66DDR bit setting.

		Setting
		I/O Port
Module Name	**Pin Function**	**P66DDR**
I/O port	P66 output	1
	P66 input (initial setting)	0

(2) P65/$\overline{\text{IRQ13}}$/CRx

The pin function is switched as shown below according to the P65DDR bit setting.

		Setting
		I/O Port
Module Name	**Pin Function**	**P65DDR**
I/O port	P65 output	1
	P65 input (initial setting)	0

(3) P64/$\overline{\text{IRQ12}}$/CTx

The pin function is switched as shown below according to the combination of the RCAN-ET and P64DDR bit settings.

		Setting	
		RCAN-ET	**I/O Port**
Module Name	**Pin Function**	**CTx_OE**	**P64DDR**
RCAN-ET	CTx output	1	—
I/O port	P64 output	0	1
	P64 input (initial setting)	0	0

(4) P63/$\overline{\text{IRQ11}}$-B

The pin function is switched as shown below according to the P63DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P63DDR
I/O port	P63 output	1
	P63 input (initial setting)	0

(5) P62/SCK4/$\overline{\text{IRQ10}}$-B

The pin function is switched as shown below according to the combination of the SCI_4 and P62DDR bit settings.

		Setting	
		SCI_4	I/O Port
Module Name	Pin Function	SCK4_OE	P62DDR
SCI_4	SCK4 output	1	—
I/O port	P62 output	0	1
	P62 input (initial setting)	0	0

(6) P61/RxD4/$\overline{\text{IRQ9}}$-B

The pin function is switched as shown below according to the P61DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	P61DDR
I/O port	P61 output	1
	P61 input (initial setting)	0

(7) P60/TxD4/$\overline{\text{IRQ8}}$-B

The pin function is switched as shown below according to the combination of the SCI_4 and P60DDR bit settings.

		Setting	
		SCI_4	I/O Port
Module Name	Pin Function	TxD4_OE	P60DDR
SCI_4	TxD4 output	1	—
I/O port	P60 output	0	1
	P60 input (initial setting)	0	0

8.2.5 Port A

(1) PA7

The pin function is switched as shown below according to the PA7DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PA7DDR
I/O port	Bϕ output	1
	PA7 input (initial setting)	0

(2) PA6

The pin function is switched as shown below according to the PA6DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PA6DDR
I/O port	PA6 output	1
	PA6 input (initial setting)	0

(3) PA5

The pin function is switched as shown below according to the PA5DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PA5DDR
I/O port	PA5 output	1
	PA5 input (initial setting)	0

(4) PA4

The pin function is switched as shown below according to the PA4DDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PA4DDR
I/O port	PA4 output	1
	PA4 input (initial setting)	0

(5) PA3/SSO2

The pin function is switched as shown below according to the combination of the SSU_2 and the PA3DDR bit settings.

		Setting	
		SSU_2	
Module Name	Pin Function	SSO2_OE	PA3DDR
SSU_2	SSO2 output	1	—
I/O port	PA3 output	0	1
	PA3 input (initial setting)	0	0

(6) PA2/SSI2

The pin function is switched as shown below according to the combination of the SSU_2 and the PA2DDR bit settings.

		Setting	
		SSU_2	**I/O Port**
Module Name	**Pin Function**	**SSI2_OE**	**PA2DDR**
SSU_2	SSI2 output	1	—
I/O port	PA2 output	0	1
	PA2 input (initial setting)	0	0

(7) PA1/SSCK2

The pin function is switched as shown below according to the combination of the SSU_2 and the PA1DDR bit settings.

		Setting	
		SSU_2	**I/O Port**
Module Name	**Pin Function**	**SSCK2_OE**	**PA1DDR**
SSU_2	SSCK2 output	1	—
I/O port	PA1 output	0	1
	PA1 input (initial setting)	0	0

8.2.6 Port D

(1) PD7/$\overline{\text{SCS1}}$

The pin function is switched as shown below according to the combination of the SSU_1 and the PD7DDR bit settings.

		Setting	
		SSU_2	I/O Port
Module Name	Pin Function	$\overline{\text{SCS1}}$_OE	PD7DDR
SSU_1	$\overline{\text{SCS1}}$ output	1	—
I/O port	PD7 output	0	1
	PD7 input (initial setting)	0	0

(2) PD6/SSCK1

The pin function is switched as shown below according to the combination of the SSU_1 and the PD6DDR bit settings.

		Setting	
		SSU_1	I/O Port
Module Name	Pin Function	SSCK1_OE	PD6DDR
SSU_1	SSCK1 output	1	—
I/O port	PD6 output	0	1
	PD6 input (initial setting)	0	0

(3) PD5/SSI1

The pin function is switched as shown below according to the combination of the SSU_1 and the PD5DDR bit settings.

		Setting	
		SSU_1	I/O Port
Module Name	Pin Function	SSI1_OE	PD5DDR
SSU_1	SSI1 output	1	—
I/O port	PD5 output	0	1
	PD5 input (initial setting)	0	0

(4) PD4/SSO1

The pin function is switched as shown below according to the combination of the SSU_1 and the PD4DDR bit settings.

		Setting	
		SSU_1	**I/O Port**
Module Name	**Pin Function**	**SSO1_OE**	**PD4DDR**
SSU_1	SSO1 output	1	—
I/O port	PD4 output	0	1
	PD4 input (initial setting)	0	0

(5) PD3/$\overline{\text{SCS0}}$

The pin function is switched as shown below according to the combination of the SSU_0 and the PD3DDR bit settings.

		Setting	
		SSU_0	**I/O Port**
Module Name	**Pin Function**	**$\overline{\text{SCS0}}$_OE**	**PD3DDR**
SSU_0	$\overline{\text{SCS0}}$ output	1	—
I/O port	PD3 output	0	1
	PD3 input (initial setting)	0	0

(6) PD2/SSCK0

The pin function is switched as shown below according to the combination of the SSU_0 and the PD2DDR bit settings.

		Setting	
		SSU_0	**I/O Port**
Module Name	**Pin Function**	**SSCK0_OE**	**PD2DDR**
SSU_0	SSCK0 output	1	—
I/O port	PD2 output	0	1
	PD2 input (initial setting)	0	0

(7) PD1/SSI0

The pin function is switched as shown below according to the combination of the SSU_0 and the PD1DDR bit settings.

		Setting	
		SSU_0	**I/O Port**
Module Name	**Pin Function**	**SSI0_OE**	**PD1DDR**
SSU_0	SSI0 output	1	—
I/O port	PD1 output	0	1
	PD1 input (initial setting)	0	0

(8) PD0/SSO0

The pin function is switched as shown below according to the combination of the SSU_0 and the PD0DDR bit settings.

		Setting	
		SSU_0	**I/O Port**
Module Name	**Pin Function**	**SSO0_OE**	**PD0DDR**
SSU_0	SSO0 output	1	—
I/O port	PD0 output	0	1
	PD0 input (initial setting)	0	0

8.2.7 Port H

(1) PH7, PH6, PH5, PH4, PH3, PH2, PH1, and PH0

Port H functions as an 8-bit I/O port and also functions as a realtime input port. Using port H as the realtime input port, the pin status of port H is stored in PHRTIDR by the following triggers

a low level, a falling edge, a rising edge, or both edges of pin $\overline{\text{IRQ14}}$.

The pin function is switched as shown below according to the PHnDDR bit setting.

		Setting
		I/O Port
Module Name	Pin Function	PHnDDR
I/O port	PHn output	1
	PHn input (initial setting)	0

[Legend]
n = 7 to 0

8.2.8 Port J

(1) PJ7/TIOCA8/TIOCB8/TCLKH

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_8, and PJ7DDR bit settings.

		Setting	
		TPU_8	I/O Port
Module Name	Pin Function	TIOCB8_OE	PJ7DDR
TPU_8	TIOCB8 output	1	—
I/O port	PJ7 output	0	1
	PJ7 input (initial setting)	0	0

(2) PJ6/TIOCA8

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_8, and PJ6DDR bit settings.

		Setting	
		TPU_8	**I/O Port**
Module Name	**Pin Function**	**TIOCA8_OE**	**PJ6DDR**
TPU_8	TIOCA8 output	1	—
I/O port	PJ6 output	0	1
	PJ6 input (initial setting)	0	0

(3) PJ5/TIOCA7/TIOCB7/TCLKG

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_7, and PJ5DDR bit settings.

		Setting	
		TPU_7	**I/O Port**
Module Name	**Pin Function**	**TIOCB7_OE**	**PJ5DDR**
TPU_7	TIOCB7 output	1	—
I/O port	PJ5 output	0	1
	PJ5 input (initial setting)	0	0

(4) PJ4/TIOCA7

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_7, and PJ4DDR bit settings.

		Setting	
		TPU_7	**I/O Port**
Module Name	**Pin Function**	**TIOCA7_OE**	**PJ4DDR**
TPU_7	TIOCA7 output	1	—
I/O port	PJ4 output	0	1
	PJ4 input (initial setting)	0	0

(5) PJ3/TIOCC6/TIOCD6/TCLKF

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_6, and PJ3DDR bit settings.

		Setting	
		TPU_6	I/O Port
Module Name	Pin Function	TIOCD6_OE	PJ3DDR
TPU_6	TIOCD6 output	1	—
I/O port	PJ3 output	0	1
	PJ3 input (initial setting)	0	0

(6) PJ2/TIOCC6/TCLKE

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_6, and PJ2DDR bit settings.

		Setting	
		TPU_6	I/O Port
Module Name	Pin Function	TIOCC6_OE	PJ2DDR
TPU_6	TIOCC6 output	1	—
I/O port	PJ2 output	0	1
	PJ2 input (initial setting)	0	0

(7) PJ1/TIOCA6/TIOCB6

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_6, and PJ1DDR bit settings.

		Setting	
		TPU_6	I/O Port
Module Name	Pin Function	TIOCB6_OE	PJ1DDR
TPU_6	TIOCB6 output	1	—
I/O port	PJ1 output	0	1
	PJ1 input (initial setting)	0	0

(8) PJ0/TIOCA6

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_6, and PJ0DDR bit settings.

		Setting	
		TPU_6	**I/O Port**
Module Name	**Pin Function**	**TIOCA6_OE**	**PJ0DDR**
TPU_6	TIOCA6 output	1	—
I/O port	PJ0 output	0	1
	PJ0 input (initial setting)	0	0

8.2.9 Port K

(1) PK7/TIOCA11/TIOCB11

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_11, and PK7DDR bit settings.

		Setting	
		TPU_11	**I/O Port**
Module Name	**Pin Function**	**TIOCB11_OE**	**PK7DDR**
TPU_11	TIOCB11 output	1	—
I/O port	PK7 output	0	1
	PK7 input (initial setting)	0	0

(2) PK6/TIOCA11

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_11, and PK6DDR bit settings.

		Setting	
		TPU_11	**I/O Port**
Module Name	**Pin Function**	**TIOCA11_OE**	**PK6DDR**
TPU_11	TIOCA11 output	1	—
I/O port	PK6 output	0	1
	PK6 input (initial setting)	0	0

(3) PK5/TIOCA10/TIOCB10

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_10, and PK5DDR bit settings.

		Setting	
		TPU_10	I/O Port
Module Name	Pin Function	TIOCB10_OE	PK5DDR
TPU_10	TIOCB10 output	1	—
I/O port	PK5 output	0	1
	PK5 input (initial setting)	0	0

(4) PK4/TIOCA10

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_10, and PK4DDR bit settings.

		Setting	
		TPU_10	I/O Port
Module Name	Pin Function	TIOCA10_OE	PK4DDR
TPU_10	TIOCA10 output	1	—
I/O port	PK4 output	0	1
	PK4 input (initial setting)	0	0

(5) PK3/TIOCC9/TIOCD9

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_9, and PK3DDR bit settings.

		Setting	
		TPU_9	I/O Port
Module Name	Pin Function	TIOCD9_OE	PK3DDR
TPU_9	TIOCD9 output	1	—
I/O port	PK3 output	0	1
	PK3 input (initial setting)	0	0

(6) PK2/TIOCC9

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_9, and PK2DDR bit settings.

		Setting	
		TPU_9	**I/O Port**
Module Name	**Pin Function**	**TIOCC9_OE**	**PK2DDR**
TPU_6	TIOCC9 output	1	—
I/O port	PK2 output	0	1
	PK2 input (initial setting)	0	0

(7) PK1/TIOCA6/TIOCB6

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_9, and PK1DDR bit settings.

		Setting	
		TPU_9	**I/O Port**
Module Name	**Pin Function**	**TIOCB9_OE**	**PK1DDR**
TPU_9	TIOCB9 output	1	—
I/O port	PK1 output	0	1
	PK1 input (initial setting)	0	0

(8) PK0/TIOCA9

The pin function is switched as shown below according to the combination of the port function control register A (PFCRA), TPU_9, and PK0DDR bit settings.

		Setting	
		TPU_9	**I/O Port**
Module Name	**Pin Function**	**TIOCA9_OE**	**PK0DDR**
TPU_9	TIOCA9 output	1	—
I/O port	PK0 output	0	1
	PK0 input (initial setting)	0	0

Table 8.4 Available Output Signals and Settings in Each Port

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P1	6	SCK3_OE	SCK3		When SCMR_3.SMIF = 1: SCR_3.TE = 1 or SCR_3.RE = 1 while SMR_3.GM = 0, SCR.CKE [1, 0] = 01 or while SMR.GM = 1 When SCMR_3.SMIF = 0: SCR_3.TE = 1 or SCR_3.RE = 1 while SMR_3.C/$\overline{A}$ = 0, SCR_3.CKE [1, 0] = 01 or while SMR_3.C/$\overline{A}$ = 1, SCR_3.CKE 1 = 0
	0	TxD3_OE	TxD3		SCR.TE = 1
P2	3	TIOCD3_OE*	TIOCD3		TPU.TMDR.BFB = 0, TPU.TIORL_3.IOD3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
	2	TIOCC3_OE*	TIOCC3		TPU.TMDR.BFA = 0, TPU.TIORL_3.IOC3 = 0, TPU.TIORL_3.IOD[1,0] = 01/10/11
	1	SCS2_OE	SCS2		SSU.SSCRH_2.CSS1 = 1, SSU.SSCRH_2.CSS0 = 0, or SSU.SSCRH_2.CSS1 = 1, SSU.SSCRH_2.CSS0 = 1 while SSU.SSCRL_2.SSUMS = 0, SSU.SSCRH_2.MSS = 1
		TIOCA3_OE*	TIOCA3		TPU.TIORH_3.IOA3 = 0, TPU.TIORH_3.IOA[1,0] = 01/10/11
	0	TIOCB3_OE*	TIOCB3		TPU.TIORH_3.IOB3 = 0, TPU.TIORH_3.IOB[1,0] = 01/10/11
P3	7	TIOCB2_OE*	TIOCB2		TPU.TIOR_2.IOB3 = 0, TPU.TIOR_2.IOB[1,0] = 01/10/11
		PO15_OE*	PO15		NDERH.NDER15 = 1
	6	TIOCA2_OE*	TIOCA2		TPU.TIOR_2.IOA3 = 0, TPU.TIOR_2.IOA[1,0] = 01/10/11
		PO14_OE*	PO14		NDERH.NDER14 = 1
	5	TIOCB1_OE*	TIOCB1		TPU.TIOR_1.IOB3 = 0, TPU.TIOR_1.IOB[1,0] = 01/10/11
		PO13_OE*	PO13		NDERH.NDER13 = 1
	4	TIOCA1_OE*	TIOCA1		TPU.TIOR_1.IOA3 = 0, TPU.TIOR_1.IOA[1,0] = 01/10/11
		PO12_OE*	PO12		NDERH.NDER12 = 1
	3	TIOCD0_OE*	TIOCD0		TPU.TMDR_0.BFB = 0, TPU.TIORL_0.IOD3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
		PO11_OE*	PO11		NDERH.NDER11 = 1
	2	TIOCC0_OE*	TIOCC0		TPU.TMDR_0.BFA = 0, TPU.TIORL_0.IOC3 = 0, TPU.TIORL_0.IOD[1,0] = 01/10/11
		PO10_OE*	PO10		NDERH.NDER10 = 1
	1	TIOCB0_OE*	TIOCB0		TPU.TIORH_0.IOB3 = 0, TPU.TIORH_0.IOB[1,0] = 01/10/11
		PO9_OE*	PO9		NDERH.NDER9 = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
P3	0	TIOCA0_OE*	TIOCA0		TPU.TIORH_0.IOA3 = 0, TPU.TIORH_0.IOA[1,0] = 01/10/11
		PO8_OE*	PO8		NDERH.NDER8 = 1
P6	4	CTx_OE	CTx		RCAN-ET MBCR.MBCRn = 0, RCAN-ET.TXRP.TXRn = 1 while RCAN-ET.RCANMON.RCANE = 1, RCAN-ET.RCANMONC.TxSTP = 0 (n = 1 to 15)
	2	SCK4_OE	SCK4		When SCMR_4.SMIF = 1: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.GM = 0, SCR_4.CKE [1, 0] = 01 or while SMR_4.GM = 1 When SCMR_4.SMIF = 0: SCR_4.TE = 1 or SCR_4.RE = 1 while SMR_4.C/$\overline{A}$ = 0, SCR_4.CKE [1, 0] = 01 or while SMR_4.C/$\overline{A}$ = 1, SCR_4.CKE 1 = 0
	0	TxD4_OE	TxD4		SCR.TE = 1
PA	7	Bϕ_OE	Bϕ		PADDR.PA7DDR = 1, SCKCR.PSTOP1 = 0, SCKCR.POSEL1 = 0
	3	SSO2_OE	SSI02		When SSU.SSCRL_2.SSUMS = 0, SSU.SSCRH_2.MSS = 1: SSU.SSCRH_2.BIDE = 0, SSU.SSER_2.TE = 1 or SSU.SSCRH_2.BIDE = 1, SSU.SSER_2.RE = 0, SSU.SSER_2.TE = 1 When SSU.SSCRL_2.SSUMS = 0, SSU.SSCRH_2.MSS = 0: SSU.SSCRH_2.BIDE = 1, SSU.SSER_2.RE = 0, SSU.SSER_1.TE = 1 When SSU.SSCRL_2.SSUMS = 1: SSU.SSER_2.TE = 1
	2	SSI2_OE	SSI2		SSU.SSCRL_2SSUMS = 0, SSU.SSCRH_2.MSS = 0 SSU.SSCRH_2.BIDE = 0, SSU.SSER_2.TE = 1
	1	SSCK2_OE	SSCK2		SSU.SSCRH_2.MSS = 1, SSU.SSCRH_2.SCKS = 1
PD	7	$\overline{\text{SCS1}}$_OE	$\overline{\text{SCS1}}$		SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 0 or SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 1 while SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 1
	6	SSCK1_OE	SSCK1		SSU.SSCRH_1.MSS = 1, SSU.SSCRH_1.SCKS = 1
	5	SSI1_OE	SSI1		SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 0 SSU.SSCRH_1.BIDE = 0, SSU.SSER_1.TE = 1

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PD	4	SSO1_OE	SSO1		When SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 1: SSU.SSCRH_1.BIDE = 0, SSU.SSER_1.TE = 1 or SSU.SSCRH_1.BIDE = 1, SSU.SSER_1.RE = 0, SSU.SSER_1.TE = 1 When SSU.SSCRL_1.SSUMS = 0, SSU.SSCRH_1.MSS = 0: SSU.SSCRH_1.BIDE = 1, SSU.SSER_1.RE = 0, SSU.SSER_1.TE = 1 When SSU.SSCRL_1.SSUMS = 1: SSU.SSER_1.TE = 1
	3	$\overline{\text{SCS0}}$_OE	$\overline{\text{SCS0}}$		SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 0 or SSU.SSCRH_0.CSS1 = 1, SSU.SSCRH_0.CSS0 = 1 while SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 1
PD	2	SSCK0_OE	SSCK0		SSU.SSCRH_0.MSS = 1, SSU.SSCRH_0.SCKS = 1
	1	SSI0_OE	SSI0		SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 0 SSU.SSCRH_0.BIDE = 0, SSU.SSER_0.TE = 1
	0	SSO0_OE	SSO0		When SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 1: SSU.SSCRH_0.BIDE = 0, SSU.SSER_0.TE = 1 or SSU.SSCRH_0.BIDE = 1, SSU.SSER_0.RE = 0, SSU.SSER_0.TE = 1 When SSU.SSCRL_0.SSUMS = 0, SSU.SSCRH_0.MSS = 0: SSU.SSCRH_0.BIDE = 1, SSU.SSER_0.RE = 0, SSU.SSER_0.TE = 1 When SSU.SSCRL_0.SSUMS = 1: SSU.SSER_0.TE = 1
PJ	7	TIOCB8_OE	TIOCB8		TPU.TIOR_8.IOB3 = 0, TPU.TIOR_8.IOB[1, 0] = 01/10/11
	6	TIOCA8_OE	TIOCA8		TPU.TIOR_8.IOA3 = 0, TPU.TIOR_8.IOA[1, 0] = 01/10/11
	5	TIOCB7_OE	TIOCB7		TPU.TIOR_7.IOB3 = 0, TPU.TIOR_7.IOB[1, 0] = 01/10/11
	4	TIOCA7_OE	TIOCA7		TPU.TIOR_7.IOA3 = 0, TPU.TIOR_7.IOA[1, 0] = 01/10/11
	3	TIOCD6_OE	TIOCD6		TPU.TMDR_6.BFB = 0, TPU.TIORL_6.IOD3 = 0 TPU.TIORL_6.IOD[1, 0] = 01/10/11
	2	TIOCC6_OE	TIOCC6		TPU.TMDR_6.BFA = 0, TPU.TIORL_6.IOC3 = 0 TPU.TIORL_6.IOC[1, 0] = 01/10/11
	1	TIOCB6_OE	TIOCB6		TPU.TIORH_6.IOB3 = 0, TPU.TIORH_6.IOB[1, 0] = 01/10/11
	0	TIOCA6_OE	TIOCA6		TPU.TIORH_6.IOA3 = 0, TPU.TIORH_6.IOA[1, 0] = 01/10/11

Port		Output Specification Signal Name	Output Signal Name	Signal Selection Register Settings	Peripheral Module Settings
PK	7	TIOCB11_OE	TIOCB11		TPU.TIOR_11.IOB3 = 0, TPU.TIOR_11.IOB[1, 0] = 01/10/11
	6	TIOCA11_OE	TIOCA11		TPU.TIOR_11.IOA3 = 0, TPU.TIOR_11.IOA[1, 0] = 01/10/11
	5	TIOCB10_OE	TIOCB10		TPU.TIOR_10.IOB3 = 0, TPU.TIOR_10.IOB[1, 0] = 01/10/11
	4	TIOCA10_OE	TIOCA10		TPU.TIOR_10.IOA3 = 0, TPU.TIOR_10.IOA[1, 0] = 01/10/11
	3	TIOCD9_OE	TIOCD9		TPU.TMDR_9.BFB = 0, TPU.TIORL_9.IOD3 = 0 TPU.TIORL_9.IOD[1, 0] = 01/10/11
	2	TIOCC9_OE	TIOCC9		TPU.TMDR_9.BFA = 0, TPU.TIORL_9.IOC3 = 0 TPU.TIORL_9.IOC[1, 0] = 01/10/11
	1	TIOCB9_OE	TIOCB9		TPU.TIOR_9.IOB3 = 0, TPU.TIOR_9.IOB[1, 0] = 01/10/11
	0	TIOCA9_OE	TIOCA9		TPU.TIOR_9.IOA3 = 0, TPU.TIOR_9.IOA[1, 0] = 01/10/11

Note: * Supported only by the H8SX/1527R.

8.3 Port Function Controller

The port function controller controls the I/O ports.

The port function controller incorporates the following registers.

- Port function control register 9 (PFCR9)*
- Port function control register A (PFCRA)
- Port function control register B (PFCRB)

Note: * PFCR9 is supported only by the H8SX/1527R.

8.3.1 Port Function Control Register 9 (PFCR9)

PFCR9 selects the multiple functions for the TPU (unit 0) I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	TPUMS3A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA3 function 0: Specifies P21 as output compare output and input capture 1: Specifies P20 as input capture input and P21 as output compare
4	TPUMS3B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC3 function 0: Specifies P22 as output compare output and input capture 1: Specifies P23 as input capture input and P22 as output compare

Bit	Bit Name	Initial Value	R/W	Description
3	TPUMS2	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA2 function 0: Specifies P36 as output compare output and input capture 1: Specifies P37 as input capture input and P36 as output compare
2	TPUMS1	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA1 function 0: Specifies P34 as output compare output and input capture 1: Specifies P35 as input capture input and P34 as output compare
1	TPUMS0A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA0 function 0: Specifies P30 as output compare output and input capture 1: Specifies P31 as input capture input and P30 as output compare
0	TPUMS0B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC0 function 0: Specifies P32 as output compare output and input capture 1: Specifies P33 as input capture input and P32 as output compare

8.3.2 Port Function Control Register A (PFCRA)

PFCRA selects the multiple functions for the TPU (unit 1) I/O pins.

Bit	7	6	5	4	3	2	1	0
Bit Name	TPUMS11	TPUMS10	TPUMS9A	TPUMS9B	TPUMS8	TPUMS7	TPUMS6A	TPUMS6B
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	TPUMS11	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA11 function 0: Specifies PK6 as output compare output and input capture 1: Specifies PK7 as input capture input and PK6 as output compare
6	TPUMS10	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA10 function 0: Specifies PK4 as output compare output and input capture 1: Specifies PK5 as input capture input and PK4 as output compare
5	TPUMS9A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA9 function 0: Specifies PK0 as output compare output and input capture 1: Specifies PK1 as input capture input and PK0 as output compare
4	TPUMS9B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC9 function 0: Specifies PK2 as output compare output and input capture 1: Specifies PK3 as input capture input and PK2 as output compare

Bit	Bit Name	Initial Value	R/W	Description
3	TPUMS8	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA8 function 0: Specifies PJ6 as output compare output and input capture 1: Specifies PJ7 as input capture input and PJ6 as output compare
2	TPUMS7	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA7 function 0: Specifies PJ4 as output compare output and input capture 1: Specifies PJ5 as input capture input and PJ4 as output compare
1	TPUMS6A	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCA6 function 0: Specifies PJ0 as output compare output and input capture 1: Specifies PJ1 as input capture input and PJ0 as output compare
0	TPUMS6B	0	R/W	TPU I/O Pin Multiplex Function Select Selects TIOCC6 function 0: Specifies PJ2 as output compare output and input capture 1: Specifies PJ3 as input capture input and PJ2 as output compare

8.3.3 Port Function Control Register B (PFCRB)

PFCRB selects the input pins for $\overline{\text{IRQ14}}$ to $\overline{\text{IRQ8}}$.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	ITS14	ITS13	ITS12	ITS11	ITS10	ITS9	ITS8
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6	ITS14	0	R/W	$\overline{\text{IRQ14}}$ Pin Select Selects an input pin for $\overline{\text{IRQ14}}$. 0: Pin P66 must not be used as $\overline{\text{IRQ14}}$-A input 1: Pin P66 is used as $\overline{\text{IRQ14}}$-B input
5	ITS13	0	R/W	$\overline{\text{IRQ13}}$ Pin Select Selects an input pin for $\overline{\text{IRQ13}}$. 0: Pin P65 must not be used as $\overline{\text{IRQ13}}$-A input 1: Pin P65 is used as $\overline{\text{IRQ13}}$-B input
4	ITS12	0	R/W	$\overline{\text{IRQ12}}$ Pin Select Selects an input pin for $\overline{\text{IRQ12}}$. 0: Pin P64 must not be used as $\overline{\text{IRQ12}}$-A input 1: Pin P64 is used as $\overline{\text{IRQ12}}$-B input
3	ITS11	0	R/W	$\overline{\text{IRQ11}}$ Pin Select Selects an input pin for $\overline{\text{IRQ11}}$. 0: Pin P23 is used as $\overline{\text{IRQ11}}$-A input 1: Pin P63 is used as $\overline{\text{IRQ11}}$-B input

Bit	Bit Name	Initial Value	R/W	Description
2	ITS10	0	R/W	$\overline{\text{IRQ10}}$ Pin Select Selects an input pin for $\overline{\text{IRQ10}}$. 0: Pin P22 is used as $\overline{\text{IRQ10}}$-A input 1: Pin P62 is used as $\overline{\text{IRQ10}}$-B input
1	ITS9	0	R/W	$\overline{\text{IRQ9}}$ Pin Select Selects an input pin for $\overline{\text{IRQ9}}$. 0: Pin P21 is used as $\overline{\text{IRQ9}}$-A input 1: Pin P61 is used as $\overline{\text{IRQ9}}$-B input
0	ITS8	0	R/W	$\overline{\text{IRQ8}}$ Pin Select Selects an input pin for $\overline{\text{IRQ8}}$. 0: Pin P20 is used as $\overline{\text{IRQ8}}$-A input 1: Pin P60 is used as $\overline{\text{IRQ8}}$-B input

8.4 Usage Notes

8.4.1 Notes on Input Buffer Control Register (ICR) Setting

- When the ICR setting is changed, the LSI may malfunction due to an edge occurred internally according to the pin states. To change the ICR setting, fix the pin high or disable the input function corresponding to the pin by setting the on-chip module registers.
- If an input is enabled by setting ICR while multiple input functions are assigned to the pin, the pin state is reflected in all the inputs of individual modules. Care must be taken for the settings of unused input function on each module side.
- When a pin is used as an output, data to be output from the pin will be latched as the pin state if the input function corresponding to the pin is enabled. To use the pin as an output, disable the input function for the pin by setting ICR.

8.4.2 Notes on Port Function Control Register (PFCR) Settings

- The PFC controls I/O ports. To specify the function of each pin, specify the input/output destination before enabling the input/output function.
- When the input/output destination is changed by the corresponding selection bit, an edge may occur if the previous pin level differs from the pin level after the change. To change the pin direction correctly, follow the procedure shown below.
  1. Disable the input function corresponding to the pin by the on-chip module registers.
  2. Select the input function by setting PFCR.
  3. Enable the input function.
- If a pin function has both a selection bit that modifies the input/output destination and an enable bit that enables the pin function, first specify the input/output destination by the selection bit and then enable the pin function by the enable bit.

Section 9 16-Bit Timer Pulse Unit (TPU)

This LSI has two on-chip 16-bit timer pulse units (TPU): unit 0 and unit1. Each unit comprises six 16-bit timer channels, that is, there are 12 timer channels in total. However, the H8SX/1525R does not include unit 0. Table 9.1 shows the unit configuration for each product.

Table 9.2 is a list of the functions and figure 9.1 is a block diagram for unit 0. Table 9.3 and figure 9.2 are for unit 1.

This section describes unit 0, which has the same functions as the other unit.

9.1 Features

Maximum 16-pulse input/output

Selection of eight counter input clocks for each channel

- The following operations can be set for each channel:
  - — Waveform output at compare match*
  - — Input capture function*
  - — Counter clear operation
  - — Synchronous operations:
    - Multiple timer counters (TCNT) can be written to simultaneously
    - Simultaneous clearing by compare match and input capture possible
    - Simultaneous input/output for registers possible by counter synchronous operation
    - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
- Cascaded operation
- Fast access via internal 16-bit bus
- 26 interrupt sources
- Automatic transfer of register data
- Programmable pulse generator (PPG) output trigger can be generated (supported only by unit 0)
- Conversion start trigger for the A/D converter can be generated (supported only by unit 0)
- Module stop mode can be set

Note: * The H8SX/1527R does not have pins TIOCA4, TIOCB4, TIOCA5, and TIOCB5 for channels 4 and 5. Therefore, 0-, 1-, or toggle-output waveform and PWM waveform at an input capture input and a compare match cannot be output.

Table 9.1 Unit Configuration for Each Product

Product	Unit Configuration	Channel Configuration
H8SX/1527R	Unit 0	Channels 0 to 5
	Unit 1	Channels 6 to 11
H8SX/1525R	Unit 1	Channels 6 to 11

Note: * The H8SX/1525R does not include unit 0.

Table 9.2 TPU Functions (Unit 0)

Item		Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
Count clock		Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 TCLKA TCLKB TCLKC TCLKD	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKB	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKB TCLKC	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 Pϕ/1024 Pϕ/4096 TCLKA	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKA TCLKC	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKA TCLKC TCLKD
General registers (TGR)		TGRA_0 TGRB_0	TGRA_1 TGRB_1	TGRA_2 TGRB_2	TGRA_3 TGRB_3	TGRA_4 TGRB_4	TGRA_5 TGRB_5
General registers/ buffer registers		TGRC_0 TGRD_0	—	—	TGRC_3 TGRD_3	—	—
I/O pins		TIOCA0 TIOCB0 TIOCC0 TIOCD0	TIOCA1 TIOCB1	TIOCA2 TIOCB2	TIOCA3 TIOCB3 TIOCC3 TIOCD3	TIOCA4*[1] TIOCB4*[1]	TIOCA5*[1] TIOCB5*[1]
Counter clear function		TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture*[2]	TGR compare match or input capture*[2]
Compare match output	0 output	O	O	O	O	O*[1]	O*[1]
	1 output	O	O	O	O	O*[1]	O*[1]
	Toggle output	O	O	O	O	O*[1]	O*[1]
Input capture function		O	O	O	O	O*[2]	O*[2]
Synchronous operation		O	O	O	O	O	O

Item	Channel 0	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
PWM mode	O	O	O	O	O*[1]	O*[1]
Phase counting mode	—	O	O	—	O	O
Buffer operation	O	—	—	O	—	—
DMAC activation	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture*[2]	TGR compare match or input capture*[2]
A/D converter trigger	TGRA_0 compare match or input capture	TGRA_1 compare match or input capture	TGRA_2 compare match or input capture	TGRA_3 compare match or input capture	TGRA_4 compare match or input capture*[2]	TGRA_5 compare match or input capture*[2]
PPG trigger	TGRA_0/ TGRB_0 compare match or input capture	TGRA_1/ TGRB_1 compare match or input capture	TGRA_2/ TGRB_2 compare match or input capture	TGRA_3/ TGRB_3 compare match or input capture	—	—
Interrupt sources	5 sources Compare match or input capture 0A Compare match or input capture 0B Compare match or input capture 0C Compare match or input capture 0D Overflow	4 sources Compare match or input capture 1A Compare match or input capture 1B Overflow Underflow	4 sources Compare match or input capture 2A Compare match or input capture 2B Overflow Underflow	5 sources Compare match or input capture 3A Compare match or input capture 3B Compare match or input capture 3C Compare match or input capture 3D Overflow	4 sources Compare match or input capture 4A Compare match or input capture 4B Overflow Underflow	4 sources Compare match or input capture 5A Compare match or input capture 5B Overflow Underflow

[Legend]

O : Possible

— : Not possible

Notes: 1. The H8SX/1527R does not have pins TIOCA4, TIOCB4, TIOCA5, and TIOCB5 for channels 4 and 5. Therefore, 0-, 1-, or toggle-output waveform and PWM waveform at an input capture input and a compare match cannot be output.

2. The H8SX/1527R does not have the input capture function for channels 4 and 5.

Table 9.3 TPU Functions (Unit 1)

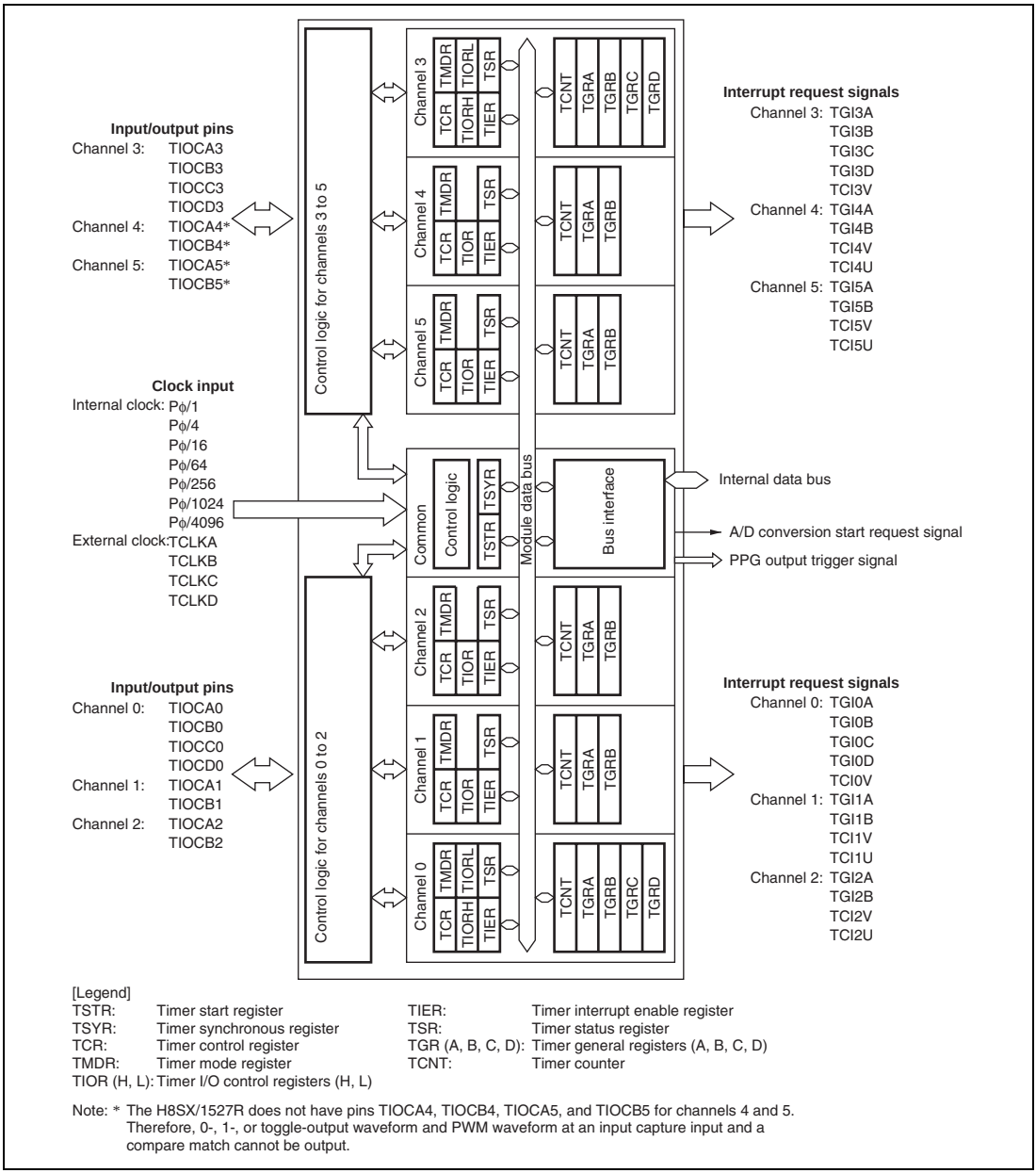
Item		Channel 6	Channel 7	Channel 8	Channel 9	Channel 10	Channel 11
Count clock		Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 TCLKE TCLKF TCLKG TCLKH	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKE TCLKF	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKE TCLKF TCLKG	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 Pϕ/1024 Pϕ/4096 TCLKE	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/1024 TCLKE TCLKG	Pϕ/1 Pϕ/4 Pϕ/16 Pϕ/64 Pϕ/256 TCLKE TCLKG TCLKH
General registers (TGR)		TGRA_6 TGRB_6	TGRA_7 TGRB_7	TGRA_8 TGRB_8	TGRA_9 TGRB_9	TGRA_10 TGRB_10	TGRA_11 TGRB_11
General registers/ buffer registers		TGRC_6 TGRD_6	—	—	TGRC_9 TGRD_9	—	—
I/O pins		TIOCA6 TIOCB6 TIOCC6 TIOCD6	TIOCA7 TIOCB7	TIOCA8 TIOCB8	TIOCA9 TIOCB9 TIOCC9 TIOCD9	TIOCA10 TIOCB10	TIOCA11 TIOCB11
Counter clear function		TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture	TGR compare match or input capture
Compare match output	0 output	O	O	O	O	O	O
	1 output	O	O	O	O	O	O
	Toggle output	O	O	O	O	O	O
Input capture function		O	O	O	O	O	O
Synchronous operation		O	O	O	O	O	O
PWM mode		O	O	O	O	O	O
Phase counting mode		—	O	O	—	O	O
Buffer operation		O	—	—	O	—	—
DMAC activation		TGRA_6 compare match or input capture	TGR_7 compare match or input capture	TGR_8 compare match or input capture	TGR_9 compare match or input capture	TGR_10 compare match or input capture	TGR_11 compare match or input capture

Item	Channel 6	Channel 7	Channel 8	Channel 9	Channel 10	Channel 11
Interrupt sources	5 sources	4 sources	4 sources	5 sources	4 sources	4 sources
	Compare match or input capture 6A	Compare match or input capture 7A	Compare match or input capture 8A	Compare match or input capture 9A	Compare match or input capture 10A	Compare match or input capture 11A
	Compare match or input capture 6B	Compare match or input capture 7B	Compare match or input capture 8B	Compare match or input capture 9B	Compare match or input capture 10B	Compare match or input capture 11B
	Compare match or input capture 6C	Overflow	Overflow	Compare match or input capture 9C	Overflow	Overflow
		Underflow	Underflow		Underflow	Underflow
	Compare match or input capture 6D			Compare match or input capture 9D		
	Overflow			Overflow		

[Legend]

○ : Possible

— : Not possible

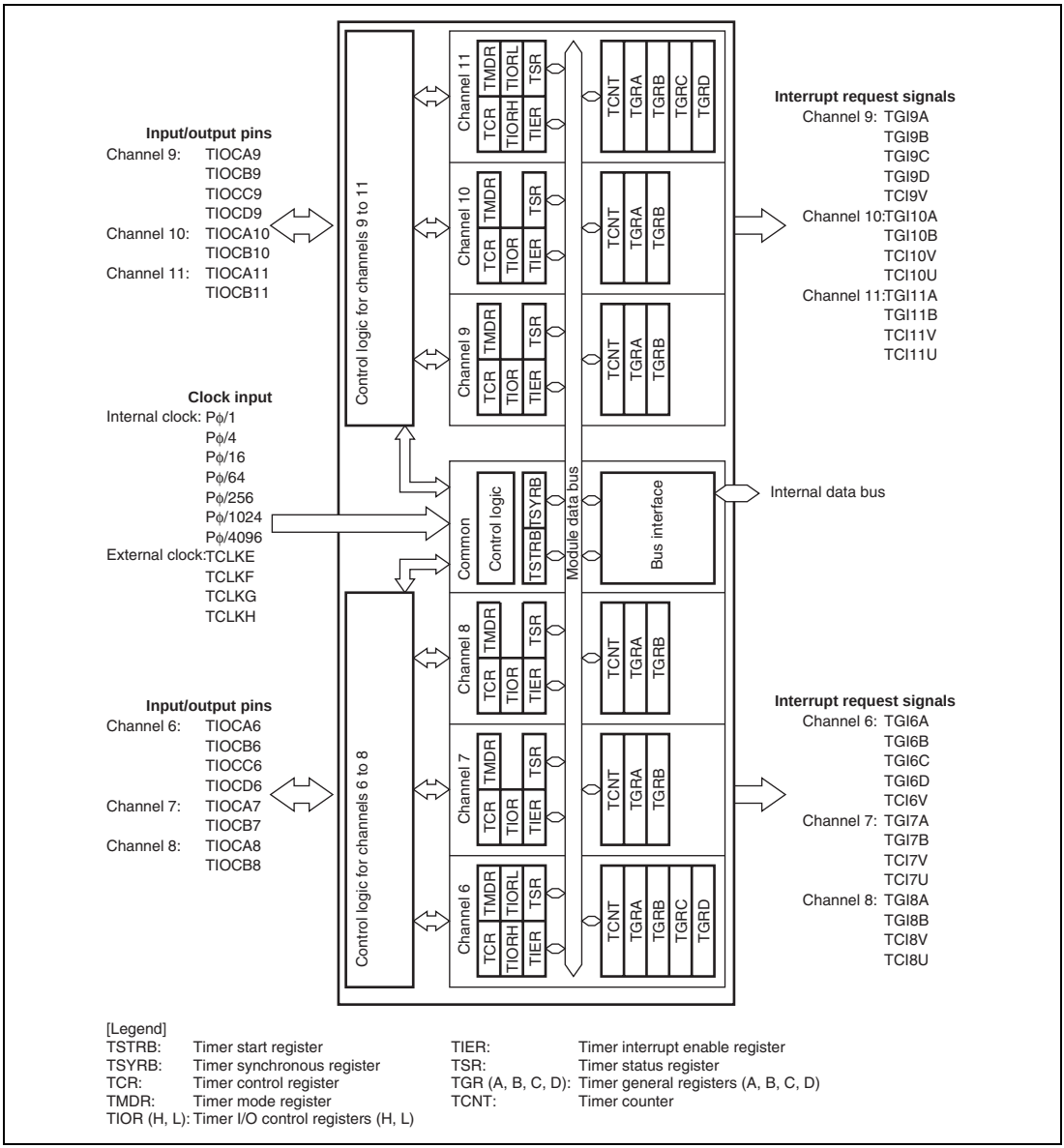


[Legend]

TSTR:	Timer start register	TIER:	Timer interrupt enable register
TSYR:	Timer synchronous register	TSR:	Timer status register
TCR:	Timer control register	TGR (A, B, C, D):	Timer general registers (A, B, C, D)
TMDR:	Timer mode register	TCNT:	Timer counter
TIOR (H, L):	Timer I/O control registers (H, L)		

Note: * The H8SX/1527R does not have pins TIOCA4, TIOCB4, TIOCA5, and TIOCB5 for channels 4 and 5. Therefore, 0-, 1-, or toggle-output waveform and PWM waveform at an input capture input and a compare match cannot be output.

Figure 9.1 Block Diagram of TPU (Unit 0)



[Legend]

TSTRB:	Timer start register	TIER:	Timer interrupt enable register
TSYRB:	Timer synchronous register	TSR:	Timer status register
TCR:	Timer control register	TGR (A, B, C, D):	Timer general registers (A, B, C, D)
TMDR:	Timer mode register	TCNT:	Timer counter
TIOR (H, L):	Timer I/O control registers (H, L)		

Figure 9.2 Block Diagram of TPU (Unit 1)

9.2 Input/Output Pins

Table 9.4 shows TPU pin configurations.

Table 9.4 Pin Configuration

Unit	Channel	Symbol	I/O	Function
0	All	TCLKA	Input	External clock A input pin (Channel 1 and 5 phase counting mode A phase input)
		TCLKB	Input	External clock B input pin (Channel 1 and 5 phase counting mode B phase input)
		TCLKC	Input	External clock C input pin (Channel 2 and 4 phase counting mode A phase input)
		TCLKD	Input	External clock D input pin (Channel 2 and 4 phase counting mode B phase input)
	0	TIOCA0	I/O	TGRA_0 input capture input/output compare output/PWM output pin
		TIOCB0	I/O	TGRB_0 input capture input/output compare output/PWM output pin
		TIOCC0	I/O	TGRC_0 input capture input/output compare output/PWM output pin
		TIOCD0	I/O	TGRD_0 input capture input/output compare output/PWM output pin
	1	TIOCA1	I/O	TGRA_1 input capture input/output compare output/PWM output pin
		TIOCB1	I/O	TGRB_1 input capture input/output compare output/PWM output pin
	2	TIOCA2	I/O	TGRA_2 input capture input/output compare output/PWM output pin
		TIOCB2	I/O	TGRB_2 input capture input/output compare output/PWM output pin
	3	TIOCA3	I/O	TGRA_3 input capture input/output compare output/PWM output pin
		TIOCB3	I/O	TGRB_3 input capture input/output compare output/PWM output pin
		TIOCC3	I/O	TGRC_3 input capture input/output compare output/PWM output pin
		TIOCD3	I/O	TGRD_3 input capture input/output compare output/PWM output pin
	4	TIOCA4*	I/O	TGRA_4 input capture input/output compare output/PWM output pin
		TIOCB4*	I/O	TGRB_4 input capture input/output compare output/PWM output pin
	5	TIOCA5*	I/O	TGRA_5 input capture input/output compare output/PWM output pin
		TIOCB5*	I/O	TGRB_5 input capture input/output compare output/PWM output pin

Unit	Channel	Symbol	I/O	Function
1	All	TCLKE	Input	External clock A input pin (Channel 7 and 11 phase counting mode A phase input)
		TCLKF	Input	External clock B input pin (Channel 7 and 11 phase counting mode B phase input)
		TCLKG	Input	External clock C input pin (Channel 8 and 10 phase counting mode A phase input)
		TCLKH	Input	External clock D input pin (Channel 8 and 10 phase counting mode B phase input)
	6	TIOCA6	I/O	TGRA_6 input capture input/output compare output/PWM output pin
		TIOCB6	I/O	TGRB_6 input capture input/output compare output/PWM output pin
		TIOCC6	I/O	TGRC_6 input capture input/output compare output/PWM output pin
		TIOCD6	I/O	TGRD_6 input capture input/output compare output/PWM output pin
	7	TIOCA7	I/O	TGRA_7 input capture input/output compare output/PWM output pin
		TIOCB7	I/O	TGRB_7 input capture input/output compare output/PWM output pin
	8	TIOCA8	I/O	TGRA_8 input capture input/output compare output/PWM output pin
		TIOCB8	I/O	TGRB_8 input capture input/output compare output/PWM output pin
	9	TIOCA9	I/O	TGRA_9 input capture input/output compare output/PWM output pin
		TIOCB9	I/O	TGRB_9 input capture input/output compare output/PWM output pin
		TIOCC9	I/O	TGRC_9 input capture input/output compare output/PWM output pin
		TIOCD9	I/O	TGRD_9 input capture input/output compare output/PWM output pin
	10	TIOCA10	I/O	TGRA_10 input capture input/output compare output/PWM output pin
		TIOCB10	I/O	TGRB_10 input capture input/output compare output/PWM output pin
	11	TIOCA11	I/O	TGRA_11 input capture input/output compare output/PWM output pin
		TIOCB11	I/O	TGRB_11 input capture input/output compare output/PWM output pin

Note: * The H8SX/1527R does not have pins TIOCA4, TIOCB4, TIOCA5, and TIOCB5 for channels 4 and 5.

9.3 Register Descriptions

The TPU has the following registers in each channel.

The registers for unit 0 and unit 1 have the same functions except bit 7 (TTGE bit for unit 0 and reserved bit for unit 1) in TIER. This section describes unit 0 registers.

Unit 0

- Channel 0:
  Timer control register_0 (TCR_0)
  Timer mode register_0 (TMDR_0)
  Timer I/O control register H_0 (TIORH_0)
  Timer I/O control register L_0 (TIORL_0)
  Timer interrupt enable register_0 (TIER_0)
  Timer status register_0 (TSR_0)
  Timer counter_0 (TCNT_0)
  Timer general register A_0 (TGRA_0)
  Timer general register B_0 (TGRB_0)
  Timer general register C_0 (TGRC_0)
  Timer general register D_0 (TGRD_0)

- Channel 1:
  Timer control register_1 (TCR_1)
  Timer mode register_1 (TMDR_1)
  Timer I/O control register _1 (TIOR_1)
  Timer interrupt enable register_1 (TIER_1)
  Timer status register_1 (TSR_1)
  Timer counter_1 (TCNT_1)
  Timer general register A_1 (TGRA_1)
  Timer general register B_1 (TGRB_1)

- Channel 2:
  Timer control register_2 (TCR_2)
  Timer mode register_2 (TMDR_2)
  Timer I/O control register_2 (TIOR_2)
  Timer interrupt enable register_2 (TIER_2)
  Timer status register_2 (TSR_2)
  Timer counter_2 (TCNT_2)
  Timer general register A_2 (TGRA_2)
  Timer general register B_2 (TGRB_2)

- Channel 3:
  Timer control register_3 (TCR_3)
  Timer mode register_3 (TMDR_3)
  Timer I/O control register H_3 (TIORH_3)
  Timer I/O control register L_3 (TIORL_3)
  Timer interrupt enable register_3 (TIER_3)
  Timer status register_3 (TSR_3)
  Timer counter_3 (TCNT_3)
  Timer general register A_3 (TGRA_3)
  Timer general register B_3 (TGRB_3)
  Timer general register C_3 (TGRC_3)
  Timer general register D_3 (TGRD_3)

- Channel 4:
  Timer control register_4 (TCR_4)
  Timer mode register_4 (TMDR_4)
  Timer I/O control register _4 (TIOR_4)
  Timer interrupt enable register_4 (TIER_4)
  Timer status register_4 (TSR_4)
  Timer counter_4 (TCNT_4)
  Timer general register A_4 (TGRA_4)
  Timer general register B_4 (TGRB_4)

- Channel 5:
  Timer control register_5 (TCR_5)
  Timer mode register_5 (TMDR_5)
  Timer I/O control register_5 (TIOR_5)
  Timer interrupt enable register_5 (TIER_5)
  Timer status register_5 (TSR_5)
  Timer counter_5 (TCNT_5)
  Timer general register A_5 (TGRA_5)
  Timer general register B_5 (TGRB_5)
- Common Registers
  Timer start register (TSTR)
  Timer synchronous register (TSYR)

Unit 1

- Channel 6:
  Timer control register_6 (TCR_6)
  Timer mode register_6 (TMDR_6)
  Timer I/O control register H_6 (TIORH_6)
  Timer I/O control register L_6 (TIORL_6)
  Timer interrupt enable register_6 (TIER_6)
  Timer status register_6 (TSR_6)
  Timer counter_6 (TCNT_6)
  Timer general register A_6 (TGRA_6)
  Timer general register B_6 (TGRB_6)
  Timer general register C_6 (TGRC_6)
  Timer general register D_6 (TGRD_6)

- Channel 7:
  Timer control register_7 (TCR_7)
  Timer mode register_7 (TMDR_7)
  Timer I/O control register _7 (TIOR_7)
  Timer interrupt enable register_7 (TIER_7)
  Timer status register_7 (TSR_7)
  Timer counter_7 (TCNT_7)
  Timer general register A_7 (TGRA_7)
  Timer general register B_7 (TGRB_7)

- Channel 8:
  Timer control register_8 (TCR_8)
  Timer mode register_8 (TMDR_8)
  Timer I/O control register_8 (TIOR_8)
  Timer interrupt enable register_8 (TIER_8)
  Timer status register_8 (TSR_8)
  Timer counter_8 (TCNT_8)
  Timer general register A_8 (TGRA_8)
  Timer general register B_8 (TGRB_8)

- Channel 9:
  Timer control register_9 (TCR_9)
  Timer mode register_9 (TMDR_9)
  Timer I/O control register H_9 (TIORH_9)
  Timer I/O control register L_9 (TIORL_9)
  Timer interrupt enable register_9 (TIER_9)
  Timer status register_9 (TSR_9)
  Timer counter_9 (TCNT_9)
  Timer general register A_9 (TGRA_9)
  Timer general register B_9 (TGRB_9)
  Timer general register C_9 (TGRC_9)
  Timer general register D_9 (TGRD_9)

- Channel 10:
  Timer control register_10 (TCR_10)
  Timer mode register_10 (TMDR_10)
  Timer I/O control register _10 (TIOR_10)
  Timer interrupt enable register_10 (TIER_10)
  Timer status register_10 (TSR_10)
  Timer counter_10 (TCNT_10)
  Timer general register A_10 (TGRA_10)
  Timer general register B_10 (TGRB_10)

- Channel 11:
  Timer control register_11 (TCR_11)
  Timer mode register_11 (TMDR_11)
  Timer I/O control register_11 (TIOR_11)
  Timer interrupt enable register_11 (TIER_11)
  Timer status register_11 (TSR_11)
  Timer counter_11 (TCNT_11)
  Timer general register A_11 (TGRA_11)
  Timer general register B_11 (TGRB_11)

- Common Registers
  Timer start register (TSTRB)
  Timer synchronous register (TSYRB)

9.3.1 Timer Control Register (TCR)

TCR controls the TCNT operation for each channel. The TPU has a total of six TCR registers, one for each channel. TCR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	CCLR2	0	R/W	Counter Clear 2 to 0
6	CCLR1	0	R/W	These bits select the TCNT counter clearing source. See tables 9.5 and 9.6 for details.
5	CCLR0	0	R/W	
4	CKEG1	0	R/W	Clock Edge 1 and 0
3	CKEG0	0	R/W	These bits select the input clock edge. For details, see table 9.7. When the input clock is counted using both edges, the input clock period is halved (e.g. Pϕ/4 both edges = Pϕ/2 rising edge). If phase counting mode is used on channels 1, 2, 4, and 5, this setting is ignored and the phase counting mode setting has priority. Internal clock edge selection is valid when the input clock is Pϕ/4 or slower. This setting is ignored if the input clock is Pϕ/1, or when overflow/underflow of another channel is selected.
2	TPSC2	0	R/W	Timer Prescaler 2 to 0
1	TPSC1	0	R/W	These bits select the TCNT counter clock. The clock source can be selected independently for each channel. See tables 9.8 to 9.13 for details. To select the external clock as the clock source, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 8, I/O Ports.
0	TPSC0	0	R/W	

Table 9.5 CCLR2 to CCLR0 (Channels 0 and 3)

Channel	Bit 7 CCLR2	Bit 6 CCLR1	Bit 5 CCLR0	Description
0, 3	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]
	1	0	0	TCNT clearing disabled
	1	0	1	TCNT cleared by TGRC compare match/input capture*[2]
	1	1	0	TCNT cleared by TGRD compare match/input capture*[2]
	1	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]

Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.
2. When TGRC or TGRD is used as a buffer register, TCNT is not cleared because the buffer register setting has priority, and compare match/input capture does not occur.

Table 9.6 CCLR2 to CCLR0 (Channels 1, 2, 4, and 5)

Channel	Bit 7*[2] Reserved	Bit 6 CCLR1	Bit 5 CCLR0	Description
1, 2, 4, 5	0	0	0	TCNT clearing disabled
	0	0	1	TCNT cleared by TGRA compare match/input capture
	0	1	0	TCNT cleared by TGRB compare match/input capture
	0	1	1	TCNT cleared by counter clearing for another channel performing synchronous clearing/ synchronous operation*[1]

Notes: 1. Synchronous operation is selected by setting the SYNC bit in TSYR to 1.
2. Bit 7 is reserved in channels 1, 2, 4, and 5. It is a read-only bit and cannot be modified.

Table 9.7 Input Clock Edge Selection

Clock Edge Selection		Input Clock	
CKEG1	**CKEG0**	**Internal Clock**	**External Clock**
0	0	Counted at falling edge	Counted at rising edge
0	1	Counted at rising edge	Counted at falling edge
1	X	Counted at both edges	Counted at both edges

[Legend]
X: Don't care

Table 9.8 TPSC2 to TPSC0 (Channel 0)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
0	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	External clock: counts on TCLKD pin input

Table 9.9 TPSC2 to TPSC0 (Channel 1)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
1	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	Counts on TCNT2 overflow/underflow

Note: This setting is ignored when channel 1 is in phase counting mode.

Table 9.10 TPSC2 to TPSC0 (Channel 2)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
2	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKB pin input
	1	1	0	External clock: counts on TCLKC pin input
	1	1	1	Internal clock: counts on Pϕ/1024

Note: This setting is ignored when channel 2 is in phase counting mode.

Table 9.11 TPSC2 to TPSC0 (Channel 3)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
3	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	Internal clock: counts on Pϕ/1024
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	Internal clock: counts on Pϕ/4096

Table 9.12 TPSC2 to TPSC0 (Channel 4)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
4	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pϕ/1024
	1	1	1	Counts on TCNT5 overflow/underflow

Note: This setting is ignored when channel 4 is in phase counting mode.

Table 9.13 TPSC2 to TPSC0 (Channel 5)

Channel	Bit 2 TPSC2	Bit 1 TPSC1	Bit 0 TPSC0	Description
5	0	0	0	Internal clock: counts on Pϕ/1
	0	0	1	Internal clock: counts on Pϕ/4
	0	1	0	Internal clock: counts on Pϕ/16
	0	1	1	Internal clock: counts on Pϕ/64
	1	0	0	External clock: counts on TCLKA pin input
	1	0	1	External clock: counts on TCLKC pin input
	1	1	0	Internal clock: counts on Pϕ/256
	1	1	1	External clock: counts on TCLKD pin input

Note: This setting is ignored when channel 5 is in phase counting mode.

9.3.2 Timer Mode Register (TMDR)

TMDR sets the operating mode for each channel. The TPU has six TMDR registers, one for each channel. TMDR register settings should be made only while TCNT operation is stopped.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	BFB	BFA	MD3	MD2	MD1	MD0
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7, 6	—	All 1	R	Reserved These are read-only bits and cannot be modified.
5	BFB	0	R/W	Buffer Operation B Specifies whether TGRB is to normally operate, or TGRB and TGRD are to be used together for buffer operation. When TGRD is used as a buffer register, TGRD input capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRD, bit 5 is reserved. It is a read-only bit and cannot be modified. 0: TGRB operates normally 1: TGRB and TGRD used together for buffer operation
4	BFA	0	R/W	Buffer Operation A Specifies whether TGRA is to normally operate, or TGRA and TGRC are to be used together for buffer operation. When TGRC is used as a buffer register, TGRC input capture/output compare is not generated. In channels 1, 2, 4, and 5, which have no TGRC, bit 4 is reserved. It is a read-only bit and cannot be modified. 0: TGRA operates normally 1: TGRA and TGRC used together for buffer operation
3	MD3	0	R/W	Modes 3 to 0 Set the timer operating mode. MD3 is a reserved bit. The write value should always be 0. See table 9.14 for details.
2	MD2	0	R/W	
1	MD1	0	R/W	
0	MD0	0	R/W	

Table 9.14 MD3 to MD0

Bit 3 MD3*[1]	Bit 2 MD2*[2]	Bit 1 MD1	Bit 0 MD0	Description
0	0	0	0	Normal operation
0	0	0	1	Reserved
0	0	1	0	PWM mode 1
0	0	1	1	PWM mode 2
0	1	0	0	Phase counting mode 1
0	1	0	1	Phase counting mode 2
0	1	1	0	Phase counting mode 3
0	1	1	1	Phase counting mode 4
1	X	X	X	—

[Legend]

X: Don't care

Notes: 1. MD3 is a reserved bit. The write value should always be 0.

2. Phase counting mode cannot be set for channels 0 and 3. In this case, the write value should always be 0.

9.3.3 Timer I/O Control Register (TIOR)

TIOR controls TGR. The TPU has eight TIOR registers, two each for channels 0 and 3, and one each for channels 1, 2, 4, and 5. Care is required since TIOR is affected by the TMDR setting.

The initial output specified by TIOR is valid when the counter is stopped (the CST bit in TSTR is cleared to 0). Note also that, in PWM mode 2, the output at the point at which the counter is cleared to 0 is specified.

When TGRC or TGRD is designated for buffer operation, this setting is invalid and the register operates as a buffer register.

To designate the input capture pin in TIOR, the DDR bit and ICR bit for the corresponding pin should be set to 0 and 1, respectively. For details, see section 8, I/O Ports.

Note: The H8SX/1527R does not include TIOR_4 and TIOR_5.

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	7	6	5	4	3	2	1	0
Bit Name	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
Initial Value	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

- TIORL_0, TORL_3

Bit	7	6	5	4	3	2	1	0
Bit Name	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
Initial Value	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

- TIORH_0, TIOR_1, TIOR_2, TIORH_3, TIOR_4, TIOR_5

Bit	Bit Name	Initial Value	R/W	Description
7	IOB3	0	R/W	I/O Control B3 to B0
6	IOB2	0	R/W	Specify the function of TGRB.
5	IOB1	0	R/W	For details, see tables 9.15, 9.17, 9.18, 9.19, 9.21, and 9.22.
4	IOB0	0	R/W	
3	IOA3	0	R/W	I/O Control A3 to A0
2	IOA2	0	R/W	Specify the function of TGRA.
1	IOA1	0	R/W	For details, see tables 9.23, 9.25, 9.26, 9.27, 9.29, and 9.30.
0	IOA0	0	R/W	

- TIORL_0, TIORL_3:

Bit	Bit Name	Initial Value	R/W	Description
7	IOD3	0	R/W	I/O Control D3 to D0
6	IOD2	0	R/W	Specify the function of TGRD.
5	IOD1	0	R/W	For details, see tables 9.16, and 9.20.
4	IOD0	0	R/W	
3	IOC3	0	R/W	I/O Control C3 to C0
2	IOC2	0	R/W	Specify the function of TGRC.
1	IOC1	0	R/W	For details, see tables 9.24, and 9.28.
0	IOC0	0	R/W	

Table 9.15 TIORH_0

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_0 Function**	**TIOCB0 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB0 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB0 pin Input capture at both edges
1	1	x	x		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*

[Legend]

X: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

Table 9.16 TIORL_0

				Description	
Bit 7 IOD3	**Bit 6 IOD2**	**Bit 5 IOD1**	**Bit 4 IOD0**	**TGRD_0 Function**	**TIOCD0 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCD0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCD0 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCD0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture at TCNT_1 count-up/count-down*1

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_0 is set to 1 and TGRD_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.17 TIOR_1

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_1 Function**	**TIOCB1 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB1 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB1 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCB1 pin Input capture at both edges
1	1	X	X		TGRC_0 compare match/input capture Input capture at generation of TGRC_0 compare match/input capture

[Legend]
X: Don't care

Table 9.18 TIOR_2

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_2 Function**	**TIOCB2 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Capture input source is TIOCB2 pin Input capture at rising edge
1	X	0	1		Capture input source is TIOCB2 pin Input capture at falling edge
1	X	1	X		Capture input source is TIOCB2 pin Input capture at both edges

[Legend]
X: Don't care

Table 9.19 TIORH_3

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_3 Function**	**TIOCB3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB3 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB3 pin Input capture at both edges
1	1	x	x		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*

[Legend]

X: Don't care

Note: When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

Table 9.20 TIORL_3

				Description	
Bit 7 IOD3	**Bit 6 IOD2**	**Bit 5 IOD1**	**Bit 4 IOD0**	**TGRD_3 Function**	**TIOCD3 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCD3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCD3 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCD3 pin Input capture at both edges
1	1	x	x		Capture input source is channel 4/count clock Input capture at TCNT_4 count-up/count-down*1

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 count clock, this setting is invalid and input capture is not generated.

2. When the BFB bit in TMDR_3 is set to 1 and TGRD_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.21 TIOR_4

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_4 Function**	**TIOCB4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCB4 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCB4 pin Input capture at falling edge
1	0	1	x		Capture input source is TIOCB4 pin Input capture at both edges
1	1	x	x		Capture input source is TGRC_3 compare match/input capture Input capture at generation of TGRC_3 compare match/input capture

[Legend]
X: Don't care

Table 9.22 TIOR_5

				Description	
Bit 7 IOB3	**Bit 6 IOB2**	**Bit 5 IOB1**	**Bit 4 IOB0**	**TGRB_5 Function**	**TIOCB5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	x	0	0	Input capture register	Capture input source is TIOCB5 pin Input capture at rising edge
1	x	0	1		Capture input source is TIOCB5 pin Input capture at falling edge
1	x	1	x		Capture input source is TIOCB5 pin Input capture at both edges

[Legend]
X: Don't care

Table 9.23 TIORH_0

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_0 Function**	**TIOCA0 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	1	Input capture register	Capture input source is TIOCA0 pin Input capture at rising edge
1	0	0	0		Capture input source is TIOCA0 pin Input capture at rising edge
1	0	1	X		Capture input source is TIOCA0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture* at TCNT_1 count-up/count-down

[Legend]

X: Don't care

Note: * When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 counter clock, this setting is ignored and an input capture interrupt is not generated.

Table 9.24 TIORL_0

				Description	
Bit 3 IOC3	**Bit 2 IOC2**	**Bit 1 IOC1**	**Bit 0 IOC0**	**TGRC_0 Function**	**TIOCC0 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCC0 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCC0 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCC0 pin Input capture at both edges
1	1	X	X		Capture input source is channel 1/count clock Input capture*1 at TCNT_1 count-up/count-down

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_1 are set to B'000 and Pϕ/1 is used as the TCNT_1 counter clock, this setting is ignored and an input capture interrupt is not generated.

2. When the BFA bit in TMDR_0 is set to 1 and TGRC_0 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.25 TIOR_1

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_1 Function**	**TIOCA1 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA1 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA1 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA1 pin Input capture at both edges
1	1	X	X		Capture input source is TGRA_0 compare match/input capture Input capture at generation of channel 0/TGRA_0 compare match/input capture

[Legend]

X: Don't care

Table 9.26 TIOR_2

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_2 Function**	**TIOCA2 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Capture input source is TIOCA2 pin Input capture at rising edge
1	X	0	1		Capture input source is TIOCA2 pin Input capture at falling edge
1	X	1	X		Capture input source is TIOCA2 pin Input capture at both edges

[Legend]
X: Don't care

Table 9.27 TIORH_3

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_3 Function**	**TIOCA3 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA3 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA3 pin Input capture at both edges
1	1	X	X		Capture input source is channel 4/count clock Input capture* at TCNT_4 count-up/count-down

[Legend]

X: Don't care

Note: * When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 counter clock, this setting is ignored and an input capture interrupt is not generated.

Table 9.28 TIORL_3

				Description	
Bit 3 IOC3	**Bit 2 IOC2**	**Bit 1 IOC1**	**Bit 0 IOC0**	**TGRC_3 Function**	**TIOCC3 Pin Function**
0	0	0	0	Output compare register*2	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register*2	Capture input source is TIOCC3 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCC3 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCC3 pin Input capture at both edges
1	1	X	X		Capture input source is channel 4/count clock Input capture*1 at TCNT_4 count-up/count-down

[Legend]

X: Don't care

Notes: 1. When bits TPSC2 to TPSC0 in TCR_4 are set to B'000 and Pϕ/1 is used as the TCNT_4 counter clock, this setting is ignored and an input capture interrupt is not generated.

2. When the BFA bit in TMDR_3 is set to 1 and TGRC_3 is used as a buffer register, this setting is invalid and input capture/output compare is not generated.

Table 9.29 TIOR_4

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_4 Function**	**TIOCA4 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	0	0	0	Input capture register	Capture input source is TIOCA4 pin Input capture at rising edge
1	0	0	1		Capture input source is TIOCA4 pin Input capture at falling edge
1	0	1	X		Capture input source is TIOCA4 pin Input capture at both edges
1	1	X	X		Capture input source is TGRA_3 compare match/input capture Input capture at generation of TGRA_3 compare match/input capture

[Legend]
X: Don't care

Table 9.30 TIOR_5

				Description	
Bit 3 IOA3	**Bit 2 IOA2**	**Bit 1 IOA1**	**Bit 0 IOA0**	**TGRA_5 Function**	**TIOCA5 Pin Function**
0	0	0	0	Output compare register	Output disabled
0	0	0	1		Initial output is 0 output 0 output at compare match
0	0	1	0		Initial output is 0 output 1 output at compare match
0	0	1	1		Initial output is 0 output Toggle output at compare match
0	1	0	0		Output disabled
0	1	0	1		Initial output is 1 output 0 output at compare match
0	1	1	0		Initial output is 1 output 1 output at compare match
0	1	1	1		Initial output is 1 output Toggle output at compare match
1	X	0	0	Input capture register	Input capture source is TIOCA5 pin Input capture at rising edge
1	X	0	1		Input capture source is TIOCA5 pin Input capture at falling edge
1	X	1	X		Input capture source is TIOCA5 pin Input capture at both edges

[Legend]
X: Don't care

9.3.4 Timer Interrupt Enable Register (TIER)

TIER controls enabling or disabling of interrupt requests for each channel. The TPU has six TIER registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TTGE*	—	TCIEU	TCIEV	TGIED	TGIEC	TGIEB	TGIEA
Initial Value	0	1	0	0	0	0	0	0
R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W

Note: * Bit 7 in TIER for unit 1 is a reserved bit and is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial value	R/W	Description
7	TTGE*	0	R/W	A/D Conversion Start Request Enable Enables/disables generation of A/D conversion start requests by TGRA input capture/compare match. 0: A/D conversion start request generation disabled 1: A/D conversion start request generation enabled
6	—	1	R	Reserved This is a read-only bit and cannot be modified.
5	TCIEU	0	R/W	Underflow Interrupt Enable Enables/disables interrupt requests (TCIU) by the TCFU flag when the TCFU flag in TSR is set to 1 in channels 1, 2, 4, and 5. In channels 0 and 3, bit 5 is reserved. It is a read-only bit and cannot be modified. 0: Interrupt requests (TCIU) by TCFU disabled 1: Interrupt requests (TCIU) by TCFU enabled
4	TCIEV	0	R/W	Overflow Interrupt Enable Enables/disables interrupt requests (TCIV) by the TCFV flag when the TCFV flag in TSR is set to 1. 0: Interrupt requests (TCIV) by TCFV disabled 1: Interrupt requests (TCIV) by TCFV enabled

Bit	Bit Name	Initial value	R/W	Description
3	TGIED	0	R/W	TGR Interrupt Enable D Enables/disables interrupt requests (TGID) by the TGFD bit when the TGFD bit in TSR is set to 1 in channels 0 and 3. In channels 1, 2, 4, and 5, bit 3 is reserved. It is a read-only bit and cannot be modified. 0: Interrupt requests (TGID) by TGFD bit disabled 1: Interrupt requests (TGID) by TGFD bit enabled
2	TGIEC	0	R/W	TGR Interrupt Enable C Enables/disables interrupt requests (TGIC) by the TGFC bit when the TGFC bit in TSR is set to 1 in channels 0 and 3. In channels 1, 2, 4, and 5, bit 2 is reserved. It is a read-only bit and cannot be modified. 0: Interrupt requests (TGIC) by TGFC bit disabled 1: Interrupt requests (TGIC) by TGFC bit enabled
1	TGIEB	0	R/W	TGR Interrupt Enable B Enables/disables interrupt requests (TGIB) by the TGFB bit when the TGFB bit in TSR is set to 1. 0: Interrupt requests (TGIB) by TGFB bit disabled 1: Interrupt requests (TGIB) by TGFB bit enabled
0	TGIEA	0	R/W	TGR Interrupt Enable A Enables/disables interrupt requests (TGIA) by the TGFA bit when the TGFA bit in TSR is set to 1. 0: Interrupt requests (TGIA) by TGFA bit disabled 1: Interrupt requests (TGIA) by TGFA bit enabled

Note: * Bit 7 in TIER for unit 1 is a reserved bit and is always read as 0. The write value should always be 0.

9.3.5 Timer Status Register (TSR)

TSR indicates the status of each channel. The TPU has six TSR registers, one for each channel.

Bit	7	6	5	4	3	2	1	0
Bit Name	TCFD	—	TCFU	TCFV	TGFD	TGFC	TGFB	TGFA
Initial Value	1	1	0	0	0	0	0	0
R/W	R	R	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*

Note: * Only 0 can be written to bits 5 to 0, to clear flags.

Bit	Bit Name	Initial value	R/W	Description
7	TCFD	1	R	Count Direction Flag Status flag that shows the direction in which TCNT counts in channels 1, 2, 4, and 5. In channels 0 and 3, bit 7 is reserved. It is a read-only bit and cannot be modified. 0: TCNT counts down 1: TCNT counts up
6	—	1	R	Reserved This is a read-only bit and cannot be modified.
5	TCFU	0	R/(W)*	Underflow Flag Status flag that indicates that a TCNT underflow has occurred when channels 1, 2, 4, and 5 are set to phase counting mode. In channels 0 and 3, bit 5 is reserved. It is a read-only bit and cannot be modified. [Setting condition] • When the TCNT value underflows (changes from H'0000 to H'FFFF) [Clearing condition] • When a 0 is written to TCFU after reading TCFU = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
4	TCFV	0	R/(W)*	Overflow Flag Status flag that indicates that a TCNT overflow has occurred. [Setting condition] • When the TCNT value overflows (changes from H'FFFF to H'0000) [Clearing condition] • When a 0 is written to TCFV after reading TCFV = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
3	TGFD	0	R/(W)*	Input Capture/Output Compare Flag D Status flag that indicates the occurrence of TGRD input capture or compare match in channels 0 and 3. In channels 1, 2, 4, and 5, bit 3 is reserved. It is a read-only bit and cannot be modified. [Setting conditions] • When TCNT = TGRD while TGRD is functioning as output compare register • When TCNT value is transferred to TGRD by input capture signal while TGRD is functioning as input capture register [Clearing condition] • When 0 is written to TGFD after reading TGFD = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
2	TGFC	0	R/(W)*	Input Capture/Output Compare Flag C Status flag that indicates the occurrence of TGRC input capture or compare match in channels 0 and 3. In channels 1, 2, 4, and 5, bit 2 is reserved. It is a read-only bit and cannot be modified. [Setting conditions] • When TCNT = TGRC while TGRC is functioning as output compare register • When TCNT value is transferred to TGRC by input capture signal while TGRC is functioning as input capture register [Clearing condition] • When 0 is written to TGFC after reading TGFC = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
1	TGFB	0	R/(W)*	Input Capture/Output Compare Flag B Status flag that indicates the occurrence of TGRB input capture or compare match. [Setting conditions] • When TCNT = TGRB while TGRB is functioning as output compare register • When TCNT value is transferred to TGRB by input capture signal while TGRB is functioning as input capture register [Clearing condition] • When 0 is written to TGFB after reading TGFB = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial value	R/W	Description
0	TGFA	0	R/(W)*	Input Capture/Output Compare Flag A Status flag that indicates the occurrence of TGRA input capture or compare match. [Setting conditions] • When TCNT = TGRA while TGRA is functioning as output compare register • When TCNT value is transferred to TGRA by input capture signal while TGRA is functioning as input capture register [Clearing conditions] • When DMAC is activated by a TGIA interrupt while the DTA bit in DMDR of DMAC is 1 • When 0 is written to TGFA after reading TGFA = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Note: * Only 0 can be written to clear the flag.

9.3.6 Timer Counter (TCNT)

TCNT is a 16-bit readable/writable counter. The TPU has six TCNT counters, one for each channel.

TCNT is initialized to H'0000 by a reset or in hardware standby mode.

TCNT cannot be accessed in 8-bit units. TCNT must always be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

9.3.7 Timer General Register (TGR)

TGR is a 16-bit readable/writable register with a dual function as output compare and input capture registers. The TPU has 16 TGR registers, four each for channels 0 and 3 and two each for channels 1, 2, 4, and 5. TGRC and TGRD for channels 0 and 3 can also be designated for operation as buffer registers. The TGR registers cannot be accessed in 8-bit units; they must always be accessed in 16-bit units. TGR and buffer register combinations during buffer operations are TGRA–TGRC and TGRB–TGRD.

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

9.3.8 Timer Start Register (TSTR)

TSTR starts or stops operation for channels 0 to 5. When setting the operating mode in TMDR or setting the count clock in TCR, first stop the TCNT counter.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	CST5	CST4	CST3	CST2	CST1	CST0
Initial Value	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

Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	CST5	0	R/W	Counter Start 5 to 0 These bits select operation or stoppage for TCNT. If 0 is written to the CST bit during operation with the TIOC pin designated for output, the counter stops but the TIOC pin output compare output level is retained. If TIOR is written to when the CST bit is cleared to 0, the pin output level will be changed to the set initial output value. 0: TCNT_5 to TCNT_0 count operation is stopped 1: TCNT_5 to TCNT_0 performs count operation
4	CST4	0	R/W	
3	CST3	0	R/W	
2	CST2	0	R/W	
1	CST1	0	R/W	
0	CST0	0	R/W	

9.3.9 Timer Synchronous Register (TSYR)

TSYR selects independent operation or synchronous operation for the TCNT counters of channels 0 to 5. A channel performs synchronous operation when the corresponding bit in TSYR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
Initial Value	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

Bit	Bit Name	Initial value	R/W	Description
7, 6	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
5	SYNC5	0	R/W	Timer Synchronization 5 to 0 These bits select whether operation is independent of or synchronized with other channels. When synchronous operation is selected, synchronous presetting of multiple channels, and synchronous clearing through counter clearing on another channel are possible. To set synchronous operation, the SYNC bits for at least two channels must be set to 1. To set synchronous clearing, in addition to the SYNC bit, the TCNT clearing source must also be set by means of bits CCLR2 to CCLR0 in TCR. 0: TCNT_5 to TCNT_0 operate independently (TCNT presetting/clearing is unrelated to other channels) 1: TCNT_5 to TCNT_0 perform synchronous operation (TCNT synchronous presetting/synchronous clearing is possible)
4	SYNC4	0	R/W	
3	SYNC3	0	R/W	
2	SYNC2	0	R/W	
1	SYNC1	0	R/W	
0	SYNC0	0	R/W	

9.4 Operation

9.4.1 Basic Functions

Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, periodic counting, and external event counting.

Each TGR can be used as an input capture register or output compare register.

(1) Counter Operation

When one of bits CST0 to CST5 is set to 1 in TSTR, the TCNT counter for the corresponding channel starts counting. TCNT can operate as a free-running counter, periodic counter, and so on.

(a) Example of count operation setting procedure

Figure 9.3 shows an example of the count operation setting procedure.

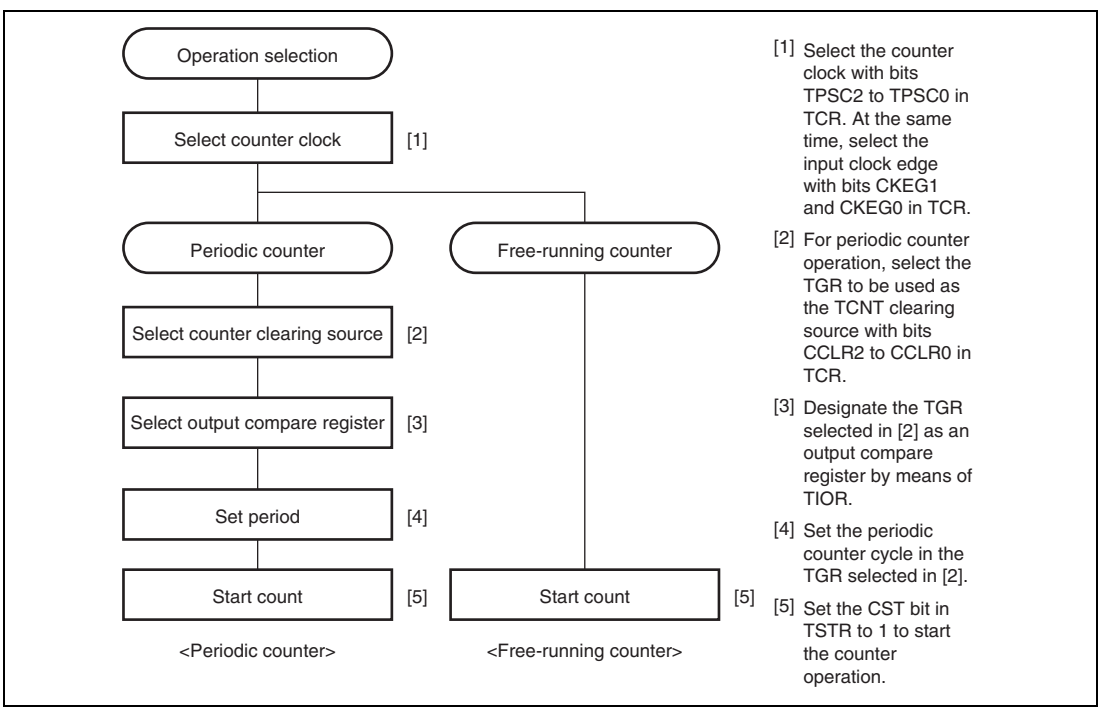


Figure 9.3 Example of Counter Operation Setting Procedure

(b) Free-running count operation and periodic count operation

Immediately after a reset, the TPU's TCNT counters are all designated as free-running counters. When the relevant bit in TSTR is set to 1 the corresponding TCNT counter starts up-count operation as a free-running counter. When TCNT overflows (changes from H'FFFF to H'0000), the TCFV bit in TSR is set to 1. If the value of the corresponding TCIEV bit in TIER is 1 at this point, the TPU requests an interrupt. After overflow, TCNT starts counting up again from H'0000.

Figure 9.4 illustrates free-running counter operation.

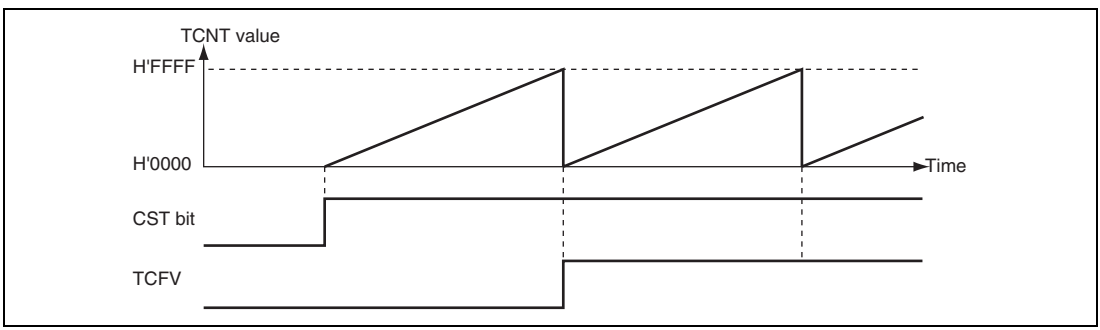


Figure 9.4 Free-Running Counter Operation

When compare match is selected as the TCNT clearing source, the TCNT counter for the relevant channel performs periodic count operation. The TGR register for setting the period is designated as an output compare register, and counter clearing by compare match is selected by means of bits CCLR2 to CCLR0 in TCR. After the settings have been made, TCNT starts count-up operation as a periodic counter when the corresponding bit in TSTR is set to 1. When the count value matches the value in TGR, the TGF bit in TSR is set to 1 and TCNT is cleared to H'0000.

If the value of the corresponding TGIE bit in TIER is 1 at this point, the TPU requests an interrupt. After a compare match, TCNT starts counting up again from H'0000.

Figure 9.5 illustrates periodic counter operation.

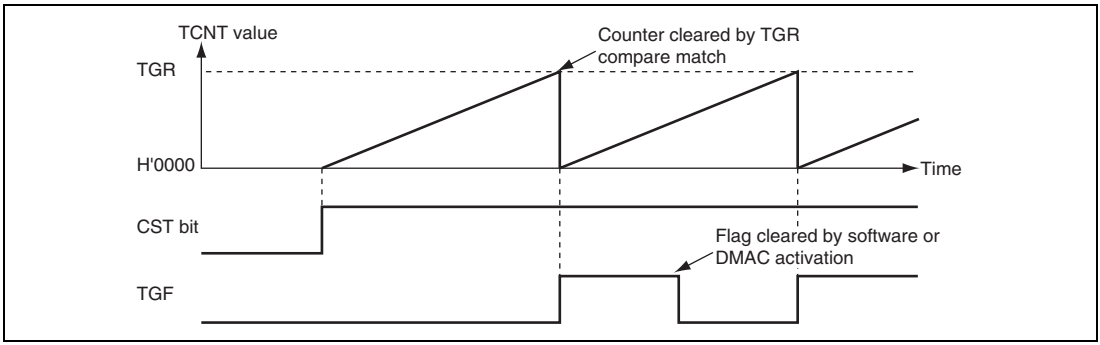


Figure 9.5 Periodic Counter Operation

(2) Waveform Output by Compare Match

The TPU can perform 0, 1, or toggle output from the corresponding output pin using a compare match.

Note: * The H8SX/1527R does not have pins TIOCA4, TIOCB4, TIOCA5, and TIOCB5 for channels 4 and 5. Therefore, 0-, 1-, or toggle-output waveform and PWM waveform at an input capture input and a compare match cannot be output.

(a) Example of setting procedure for waveform output by compare match

Figure 9.6 shows an example of the setting procedure for waveform output by a compare match.

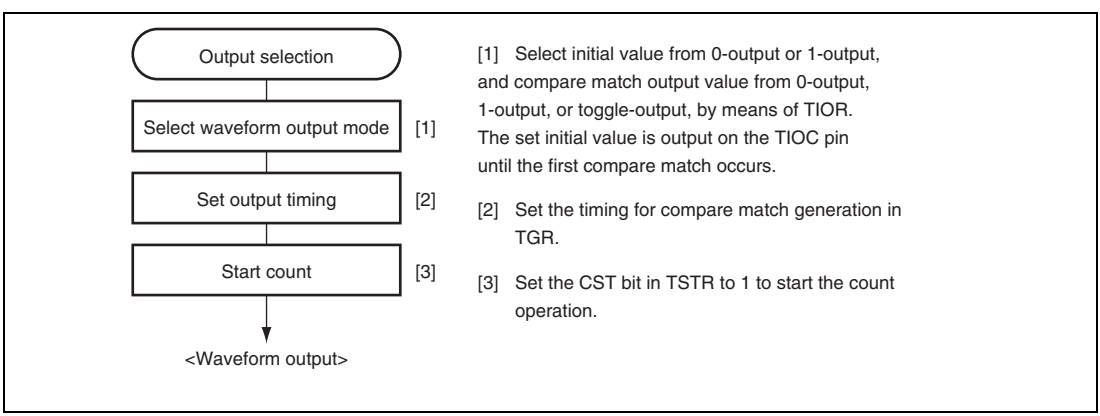


Figure 9.6 Example of Setting Procedure for Waveform Output by Compare Match

(b) Examples of waveform output operation

Figure 9.7 shows an example of 0-output and 1-output.

In this example, TCNT has been designated as a free-running counter, and settings have been made so that 1 is output by compare match A, and 0 is output by compare match B. When the set level and the pin level match, the pin level does not change.

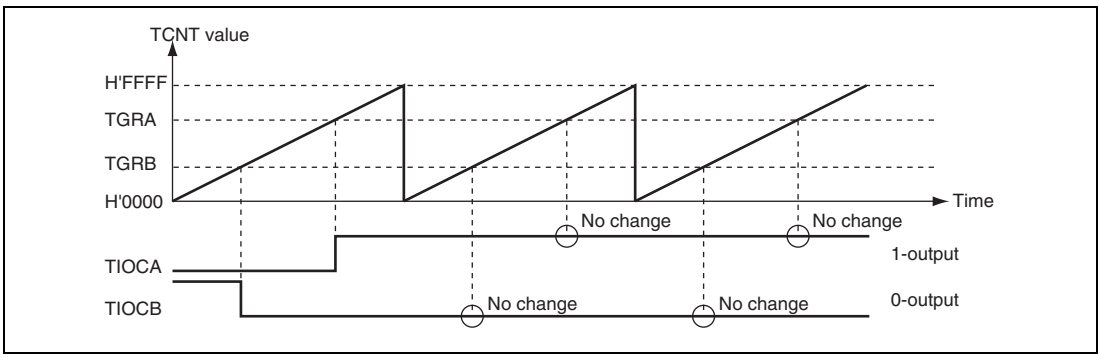


Figure 9.7 Example of 0-Output/1-Output Operation

Figure 9.8 shows an example of toggle output.

In this example, TCNT has been designated as a periodic counter (with counter clearing performed by compare match B), and settings have been made so that output is toggled by both compare match A and compare match B.

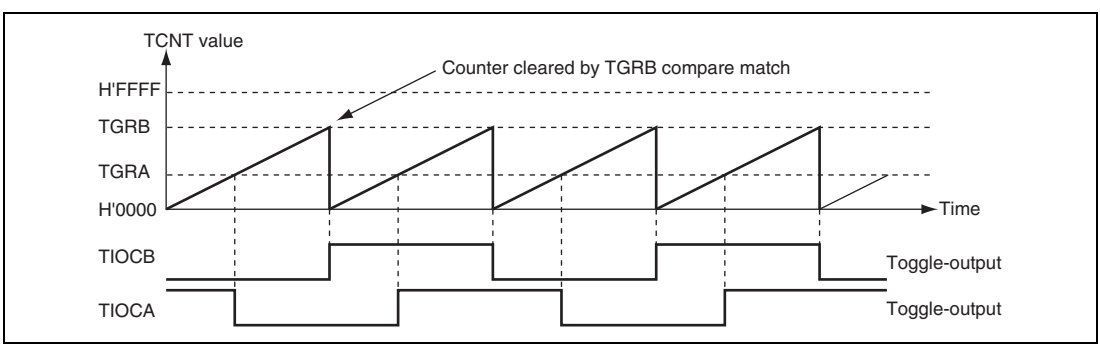


Figure 9.8 Example of Toggle Output Operation

(3) Input Capture Function

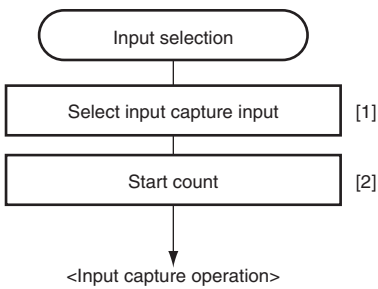
The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the detection edge. For channels 0, 1, 3, and 4, it is also possible to specify another channel's counter input clock or compare match signal as the input capture source.

Note: When another channel's counter input clock is used as the input capture input for channels 0 and 3, Pϕ/1 should not be selected as the counter input clock used for input capture input. Input capture will not be generated if Pϕ/1 is selected.

(a) Example of setting procedure for input capture operation

Figure 9.9 shows an example of the setting procedure for input capture operation.



[1] Designate TGR as an input capture register by means of TIOR, and select the input capture source and input signal edge (rising edge, falling edge, or both edges).

[2] Set the CST bit in TSTR to 1 to start the count operation.

Figure 9.9 Example of Setting Procedure for Input Capture Operation

(b) Example of input capture operation

Figure 9.10 shows an example of input capture operation.

In this example, both rising and falling edges have been selected as the TIOCA pin input capture input edge, falling edge has been selected as the TIOCB pin input capture input edge, and counter clearing by TGRB input capture has been designated for TCNT.

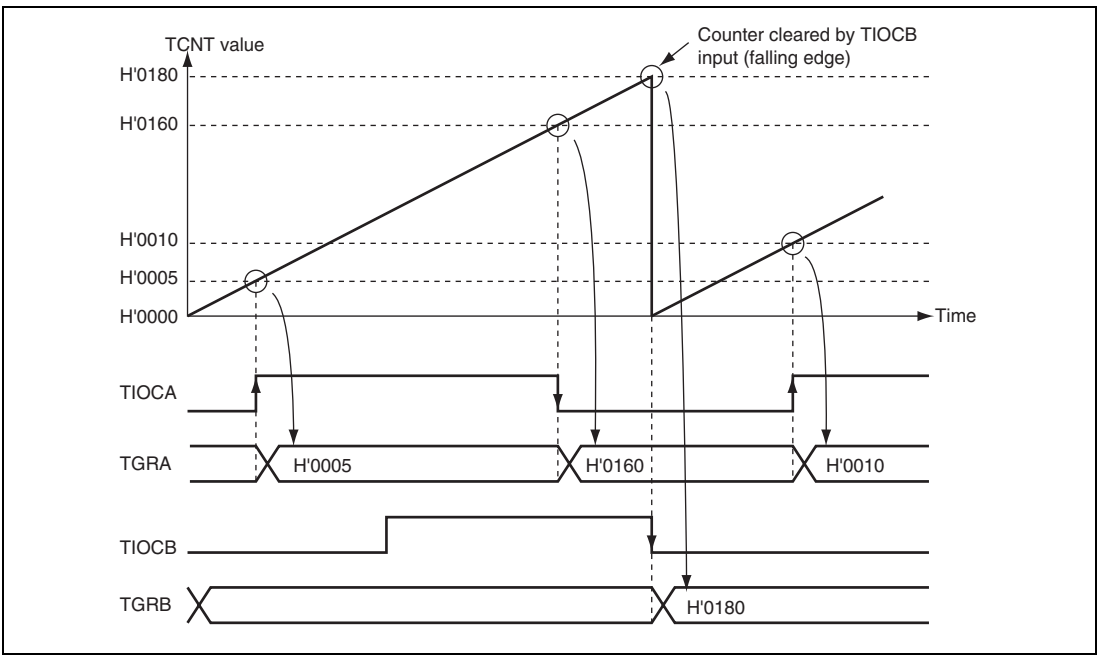


Figure 9.10 Example of Input Capture Operation

9.4.2 Synchronous Operation

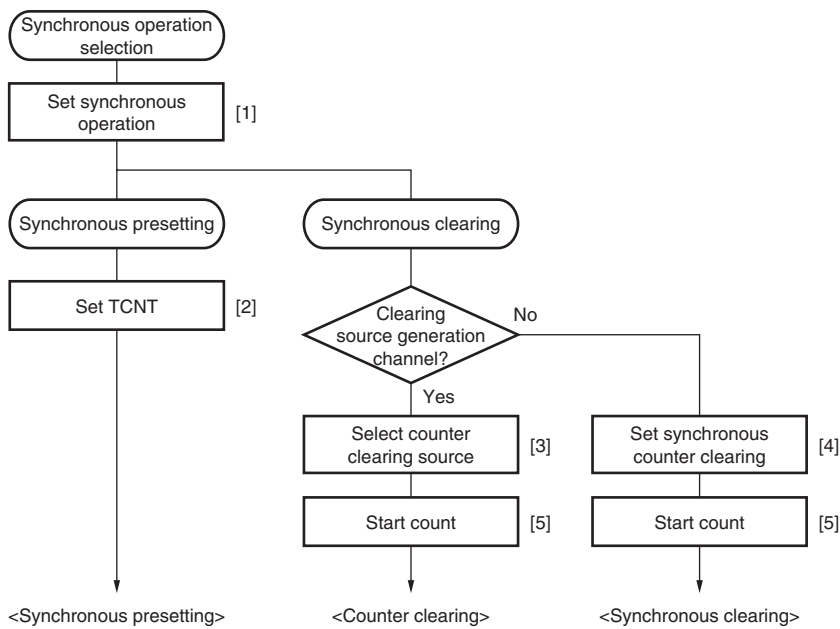
In synchronous operation, the values in multiple TCNT counters can be rewritten simultaneously (synchronous presetting). Also, multiple TCNT counters can be cleared simultaneously (synchronous clearing) by making the appropriate setting in TCR.

Synchronous operation enables TGR to be incremented with respect to a single time base.

Channels 0 to 5 can all be designated for synchronous operation.

(1) Example of Synchronous Operation Setting Procedure

Figure 9.11 shows an example of the synchronous operation setting procedure.



[1] Set the SYNC bits in TSYR corresponding to the channels to be designated for synchronous operation to 1.

[2] When the TCNT counter of any of the channels designated for synchronous operation is written to, the same value is simultaneously written to the other TCNT counters.

[3] Use bits CCLR2 to CCLR0 in TCR to specify TCNT clearing by input capture/output compare, etc.

[4] Use bits CCLR2 to CCLR0 in TCR to designate synchronous clearing for the counter clearing source.

[5] Set the CST bits in TSTR for the relevant channels to 1, to start the count operation.

Figure 9.11 Example of Synchronous Operation Setting Procedure

(2) Example of Synchronous Operation

Figure 9.12 shows an example of synchronous operation.

In this example, synchronous operation and PWM mode 1 have been designated for channels 0 to 2, TGRB_0 compare match has been set as the channel 0 counter clearing source, and synchronous clearing has been set for the channel 1 and 2 counter clearing source.

Three-phase PWM waveforms are output from pins TIOCA0, TIOCA1, and TIOCA2. At this time, synchronous presetting and synchronous clearing by TGRB_0 compare match are performed for channel 0 to 2 TCNT counters, and the data set in TGRB_0 is used as the PWM cycle.

For details on PWM modes, see section 9.4.5, PWM Modes.

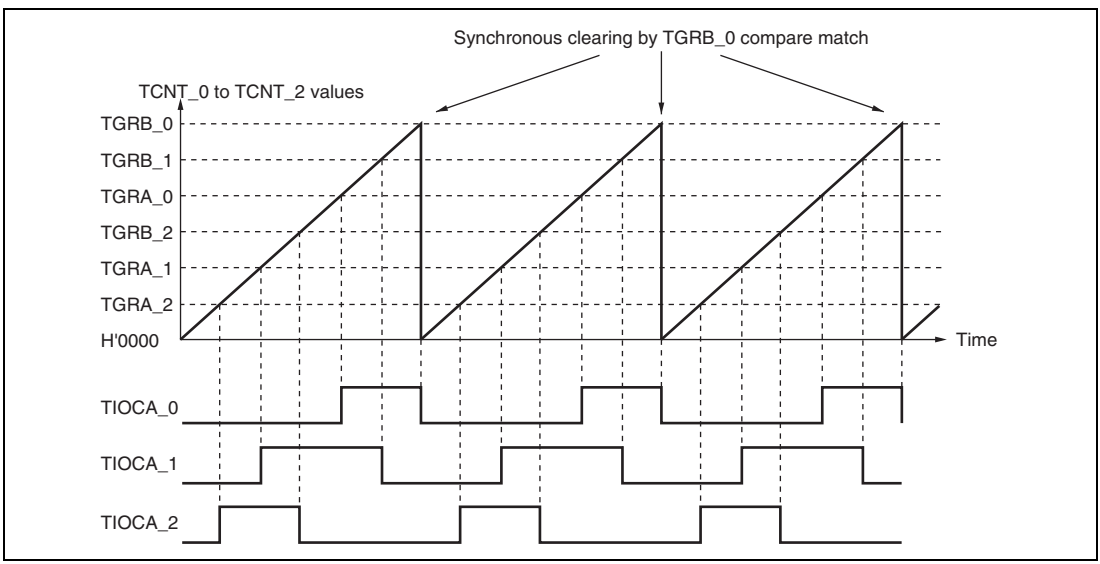


Figure 9.12 Example of Synchronous Operation

9.4.3 Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

Buffer operation differs depending on whether TGR has been designated as an input capture register or a compare match register.

Table 9.31 shows the register combinations used in buffer operation.

Table 9.31 Register Combinations in Buffer Operation

Channel	Timer General Register	Buffer Register
0	TGRA_0	TGRC_0
	TGRB_0	TGRD_0
3	TGRA_3	TGRC_3
	TGRB_3	TGRD_3

- When TGR is an output compare register

  When a compare match occurs, the value in the buffer register for the corresponding channel is transferred to the timer general register.

  This operation is illustrated in figure 9.13.

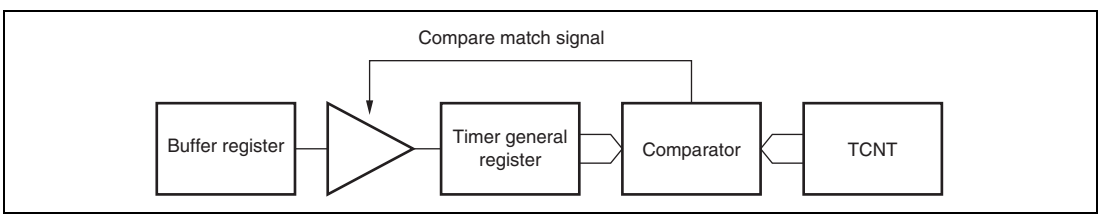


Figure 9.13 Compare Match Buffer Operation

- When TGR is an input capture register

  When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in TGR is transferred to the buffer register.

  This operation is illustrated in figure 9.14.

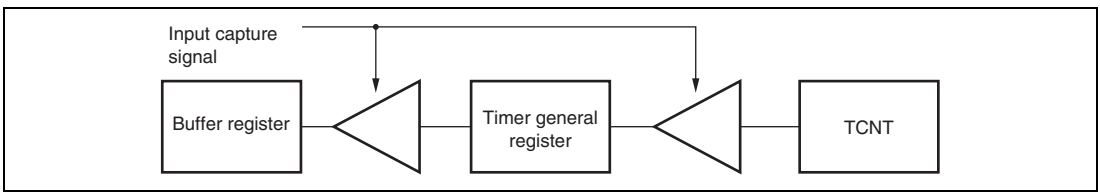


Figure 9.14 Input Capture Buffer Operation

(1) Example of Buffer Operation Setting Procedure

Figure 9.15 shows an example of the buffer operation setting procedure.

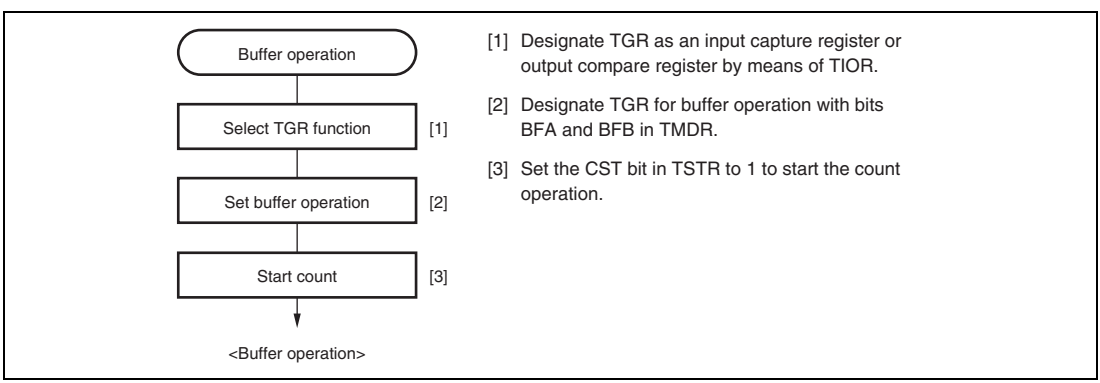


Figure 9.15 Example of Buffer Operation Setting Procedure

(2) Examples of Buffer Operation

(a) When TGR is an output compare register

Figure 9.16 shows an operation example in which PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used in this example are TCNT clearing by compare match B, 1 output at compare match A, and 0 output at compare match B.

As buffer operation has been set, when compare match A occurs, the output changes and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA. This operation is repeated each time compare match A occurs.

For details on PWM modes, see section 9.4.5, PWM Modes.

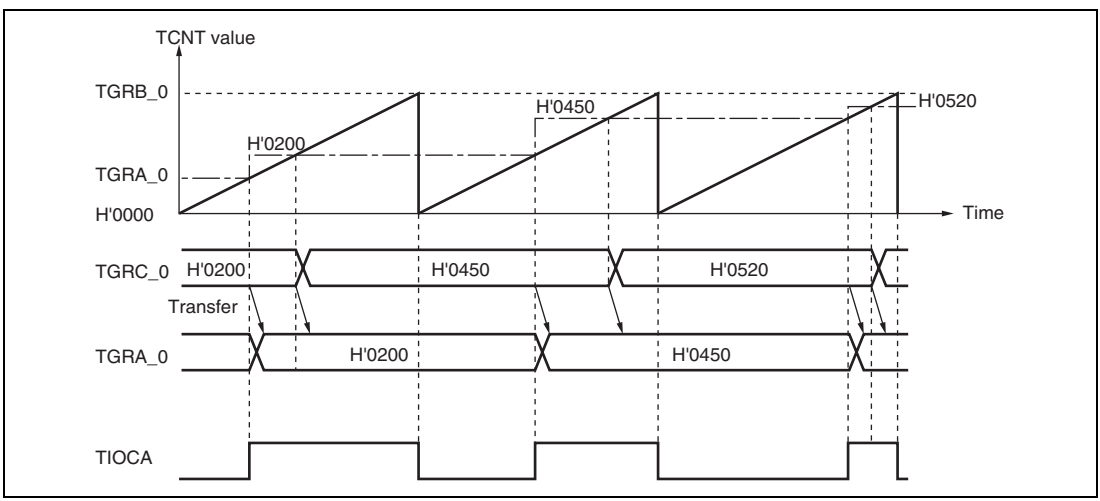


Figure 9.16 Example of Buffer Operation (1)

(b) When TGR is an input capture register

Figure 9.17 shows an operation example in which TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC.

Counter clearing by TGRA input capture has been set for TCNT, and both rising and falling edges have been selected as the TIOCA pin input capture input edge.

As buffer operation has been set, when the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

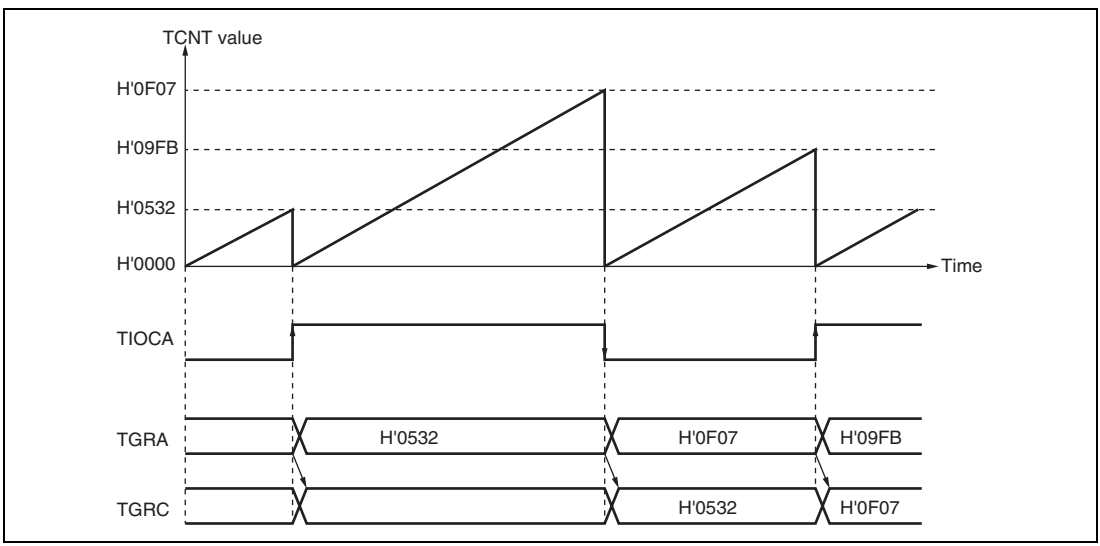


Figure 9.17 Example of Buffer Operation (2)

9.4.4 Cascaded Operation

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter.

This function works by counting the channel 1 (channel 4) counter clock at overflow/underflow of TCNT_2 (TCNT_5) as set in bits TPSC2 to TPSC0 in TCR.

Underflow occurs only when the lower 16-bit TCNT is in phase-counting mode.

Table 9.32 shows the register combinations used in cascaded operation.

Note: When phase counting mode is set for channel 1 or 4, the counter clock setting is invalid and the counter operates independently in phase counting mode.

Table 9.32 Cascaded Combinations

Combination	Upper 16 Bits	Lower 16 Bits
Channels 1 and 2	TCNT_1	TCNT_2
Channels 4 and 5	TCNT_4	TCNT_5

(1) Example of Cascaded Operation Setting Procedure

Figure 9.18 shows an example of the setting procedure for cascaded operation.

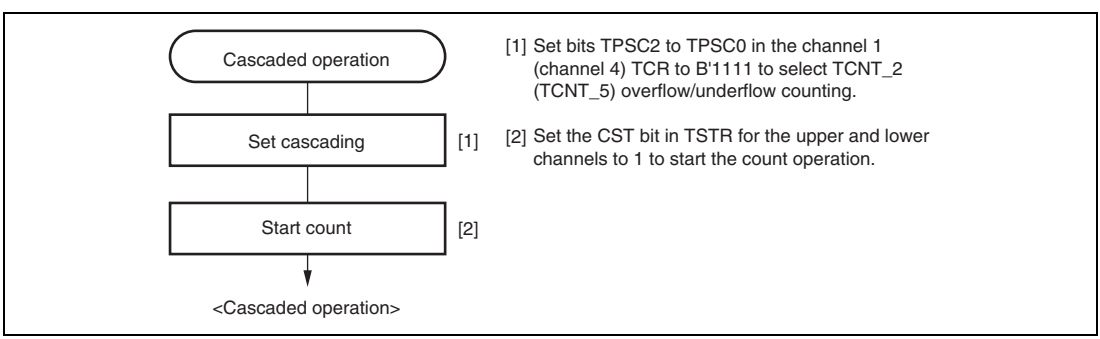


Figure 9.18 Example of Cascaded Operation Setting Procedure

(2) Examples of Cascaded Operation

Figure 9.19 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, TGRA_1 and TGRA_2 have been designated as input capture registers, and the TIOC pin rising edge has been selected.

When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGRA_1, and the lower 16 bits to TGRA_2.

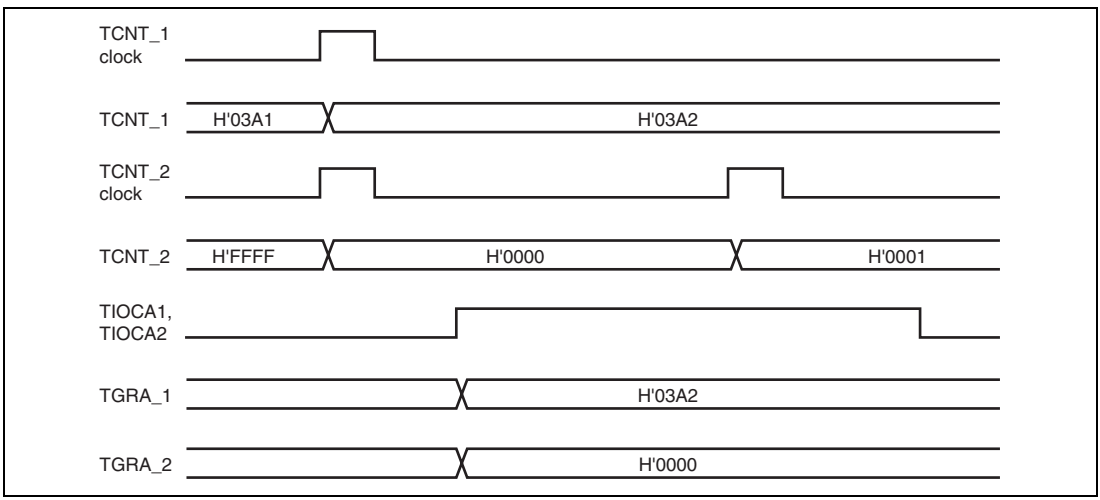


Figure 9.19 Example of Cascaded Operation (1)

Figure 9.20 illustrates the operation when counting upon TCNT_2 overflow/underflow has been set for TCNT_1, and phase counting mode has been designated for channel 2.

TCNT_1 is incremented by TCNT_2 overflow and decremented by TCNT_2 underflow.

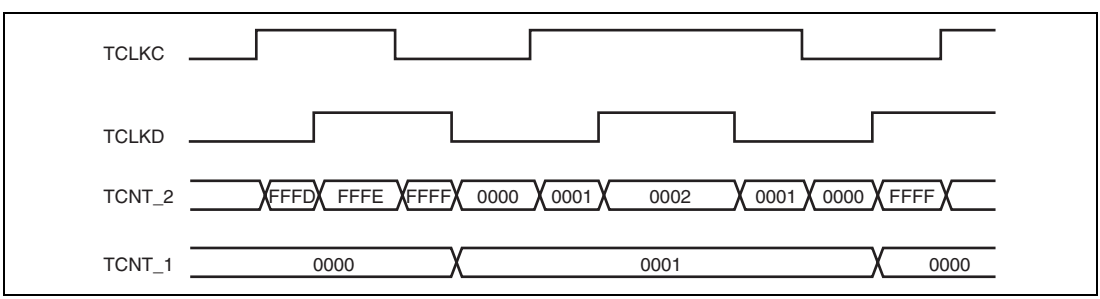


Figure 9.20 Example of Cascaded Operation (2)

9.4.5 PWM Modes

In PWM mode, PWM waveforms are output from the output pins. 0-, 1-, or toggle-output can be selected as the output level in response to compare match of each TGR.

Settings of TGR registers can output a PWM waveform in the range of 0% to 100% duty cycle.

Designating TGR compare match as the counter clearing source enables the cycle to be set in that register. All channels can be designated for PWM mode independently. Synchronous operation is also possible.

There are two PWM modes, as described below.

(1) PWM mode 1

PWM output is generated from the TIOCA and TIOCC pins by pairing TGRA with TGRB and TGRC with TGRD. The outputs specified by bits IOA3 to IOA0 and IOC3 to IOC0 in TIOR are output from the TIOCA and TIOCC pins at compare matches A and C, respectively. The outputs specified by bits IOB3 to IOB0 and IOD3 to IOD0 in TIOR are output at compare matches B and D, respectively. The initial output value is the value set in TGRA or TGRC. If the set values of paired TGRs are identical, the output value does not change when a compare match occurs.

In PWM mode 1, a maximum 8-phase PWM output is possible.

(2) PWM mode 2

PWM output is generated using one TGR as the cycle register and the others as duty cycle registers. The output specified in TIOR is performed by means of compare matches. Upon counter clearing by a cycle register compare match, the output value of each pin is the initial value set in TIOR. If the set values of the cycle and duty cycle registers are identical, the output value does not change when a compare match occurs.

In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

The correspondence between PWM output pins and registers is shown in table 9.33.

Table 9.33 PWM Output Registers and Output Pins

		Output Pins	
Channel	**Registers**	**PWM Mode 1**	**PWM Mode 2**
0	TGRA_0	TIOCA0	TIOCA0
	TGRB_0		TIOCB0
	TGRC_0	TIOCC0	TIOCC0
	TGRD_0		TIOCD0
1	TGRA_1	TIOCA1	TIOCA1
	TGRB_1		TIOCB1
2	TGRA_2	TIOCA2	TIOCA2
	TGRB_2		TIOCB2
3	TGRA_3	TIOCA3	TIOCA3
	TGRB_3		TIOCB3
	TGRC_3	TIOCC3	TIOCC3
	TGRD_3		TIOCD3
4	TGRA_4	TIOCA4	TIOCA4
	TGRB_4		TIOCB4
5	TGRA_5	TIOCA5	TIOCA5
	TGRB_5		TIOCB5

Note: In PWM mode 2, PWM output is not possible for the TGR register in which the cycle is set.

(3) Example of PWM Mode Setting Procedure

Figure 9.21 shows an example of the PWM mode setting procedure.

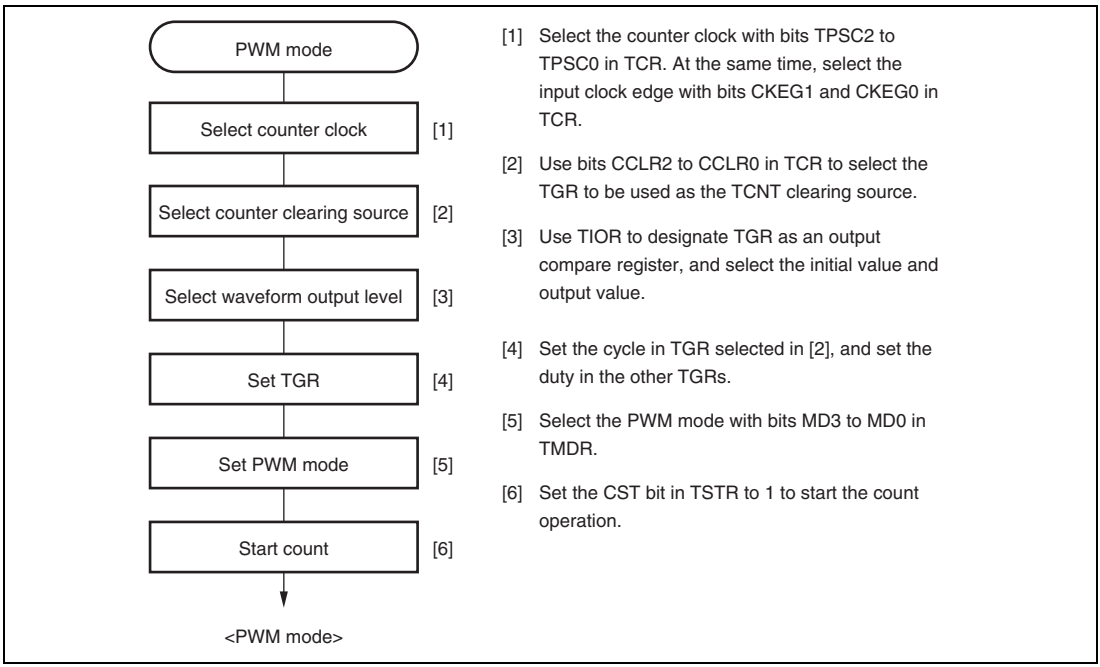


Figure 9.21 Example of PWM Mode Setting Procedure

(4) Examples of PWM Mode Operation

Figure 9.22 shows an example of PWM mode 1 operation.

In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value.

In this case, the value set in TGRA is used as the cycle, and the value set in TGRB register as the duty cycle.

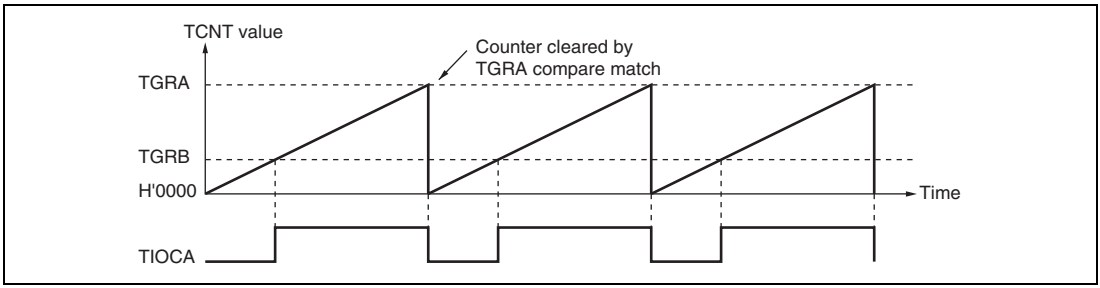


Figure 9.22 Example of PWM Mode Operation (1)

Figure 9.23 shows an example of PWM mode 2 operation.

In this example, synchronous operation is designated for channels 0 and 1, TGRB_1 compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers (TGRA_0 to TGRD_0, TGRA_1), to output a 5-phase PWM waveform.

In this case, the value set in TGRB_1 is used as the cycle, and the values set in the other TGRs as the duty cycle.

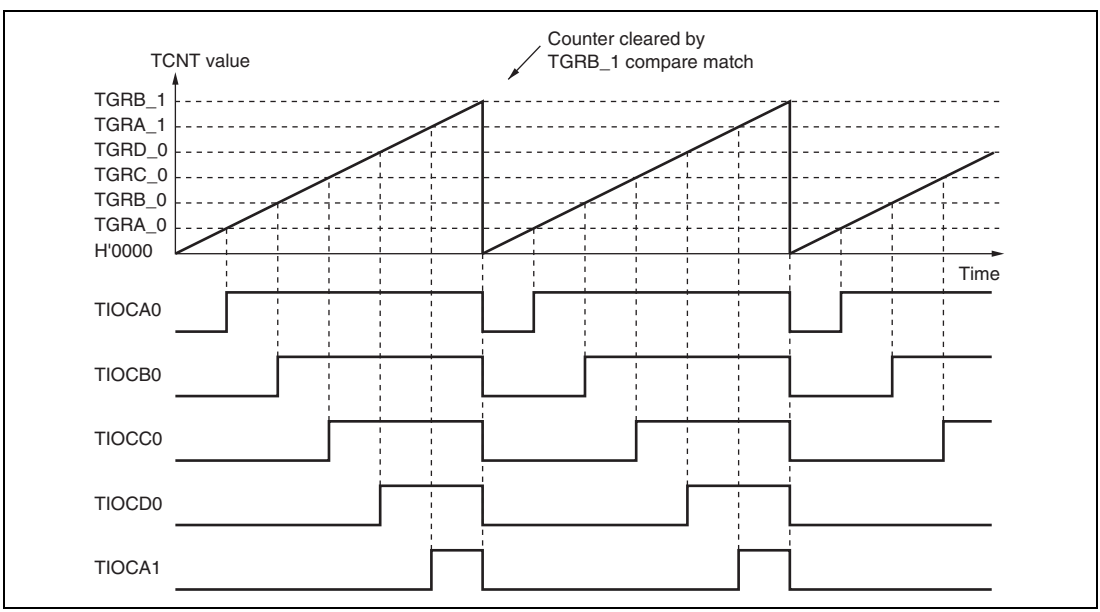


Figure 9.23 Example of PWM Mode Operation (2)

Figure 9.24 shows examples of PWM waveform output with 0% duty cycle and 100% duty cycle in PWM mode.

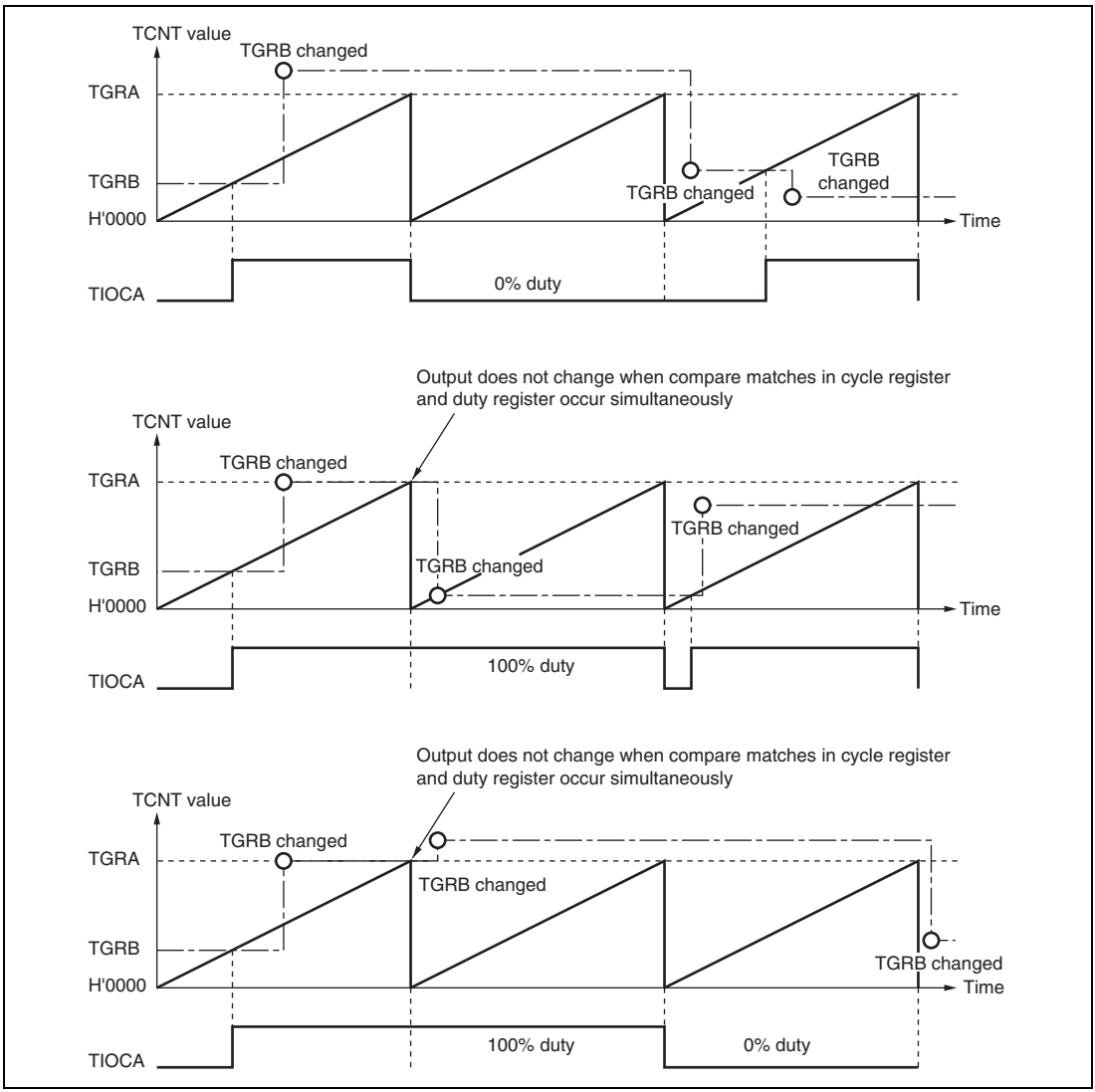


Figure 9.24 Example of PWM Mode Operation (3)

9.4.6 Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT is incremented/decremented accordingly. This mode can be set for channels 1, 2, 4, and 5.

When phase counting mode is set, an external clock is selected as the counter input clock and TCNT operates as an up/down-counter regardless of the setting of bits TPSC2 to TPSC0 and bits CKEG1 and CKEG0 in TCR. However, the functions of bits CCLR1 and CCLR0 in TCR, and of TIOR, TIER, and TGR are valid, and input capture/compare match and interrupt functions can be used.

This can be used for two-phase encoder pulse input.

When overflow occurs while TCNT is counting up, the TCFV flag in TSR is set; when underflow occurs while TCNT is counting down, the TCFU flag is set.

The TCFD bit in TSR is the count direction flag. Reading the TCFD flag provides an indication of whether TCNT is counting up or down.

Table 9.34 shows the correspondence between external clock pins and channels.

Table 9.34 Clock Input Pins in Phase Counting Mode

	External Clock Pins	
Channels	**A-Phase**	**B-Phase**
When channel 1 or 5 is set to phase counting mode	TCLKA	TCLKB
When channel 2 or 4 is set to phase counting mode	TCLKC	TCLKD

(1) Example of Phase Counting Mode Setting Procedure

Figure 9.25 shows an example of the phase counting mode setting procedure.

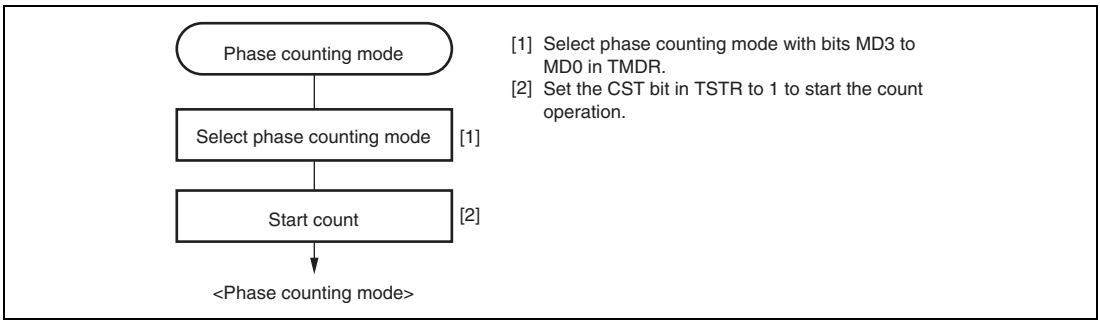


Figure 9.25 Example of Phase Counting Mode Setting Procedure

(2) Examples of Phase Counting Mode Operation

In phase counting mode, TCNT counts up or down according to the phase difference between two external clocks. There are four modes, according to the count conditions.

(a) Phase counting mode 1

Figure 9.26 shows an example of phase counting mode 1 operation, and table 9.35 summarizes the TCNT up/down-count conditions.

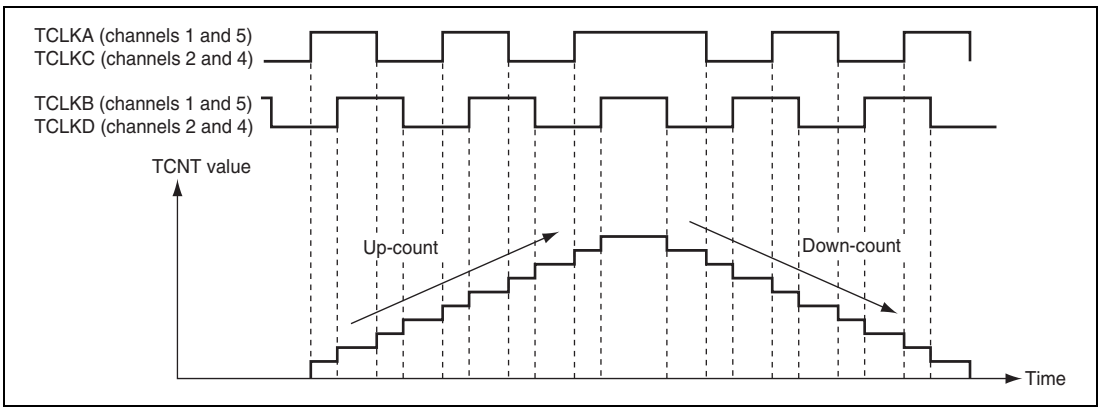


Figure 9.26 Example of Phase Counting Mode 1 Operation

Table 9.35 Up/Down-Count Conditions in Phase Counting Mode 1

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Up-count
Low level	↓	
↑	Low level	
↓	High level	
High level	↓	Down-count
Low level	↑	
↑	High level	
↓	Low level	

[Legend]
↑ : Rising edge
↓ : Falling edge

(b) Phase counting mode 2

Figure 9.27 shows an example of phase counting mode 2 operation, and table 9.36 summarizes the TCNT up/down-count conditions.

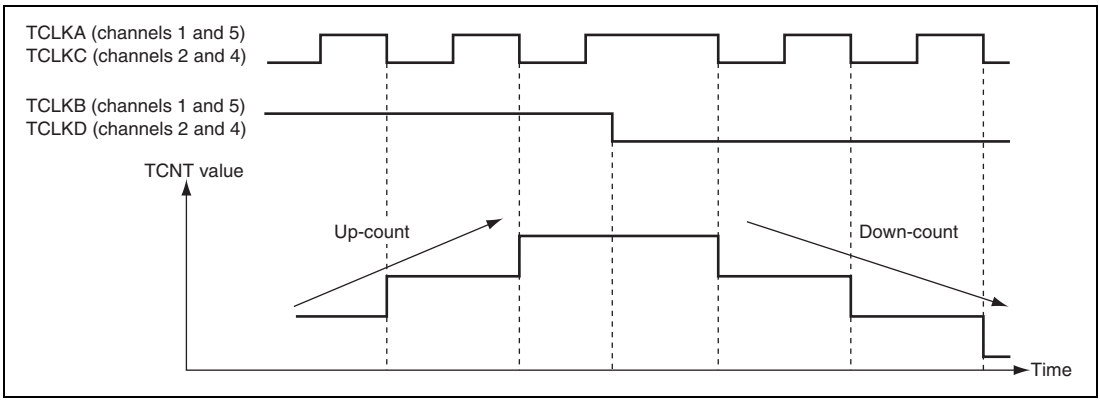


Figure 9.27 Example of Phase Counting Mode 2 Operation

Table 9.36 Up/Down-Count Conditions in Phase Counting Mode 2

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Don't care
Low level	↓	Don't care
↑	Low level	Don't care
↓	High level	Up-count
High level	↓	Don't care
Low level	↑	Don't care
↑	High level	Don't care
↓	Low level	Down-count

[Legend]
↑ : Rising edge
↓ : Falling edge

(c) Phase counting mode 3

Figure 9.28 shows an example of phase counting mode 3 operation, and table 9.37 summarizes the TCNT up/down-count conditions.

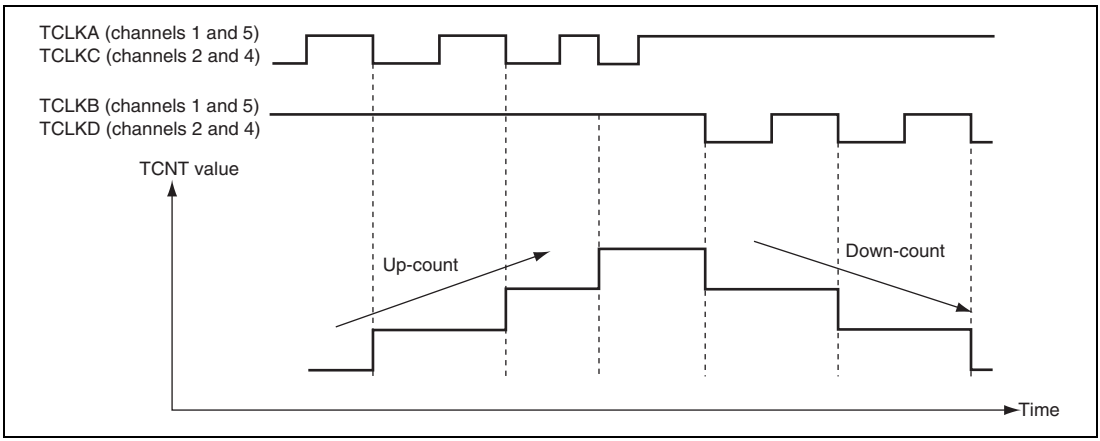


Figure 9.28 Example of Phase Counting Mode 3 Operation

Table 9.37 Up/Down-Count Conditions in Phase Counting Mode 3

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Don't care
Low level	↓	Don't care
↑	Low level	Don't care
↓	High level	Up-count
High level	↓	Down-count
Low level	↑	Don't care
↑	High level	Don't care
↓	Low level	Don't care

[Legend]
↑ : Rising edge
↓ : Falling edge

(d) Phase counting mode 4

Figure 9.29 shows an example of phase counting mode 4 operation, and table 9.38 summarizes the TCNT up/down-count conditions.

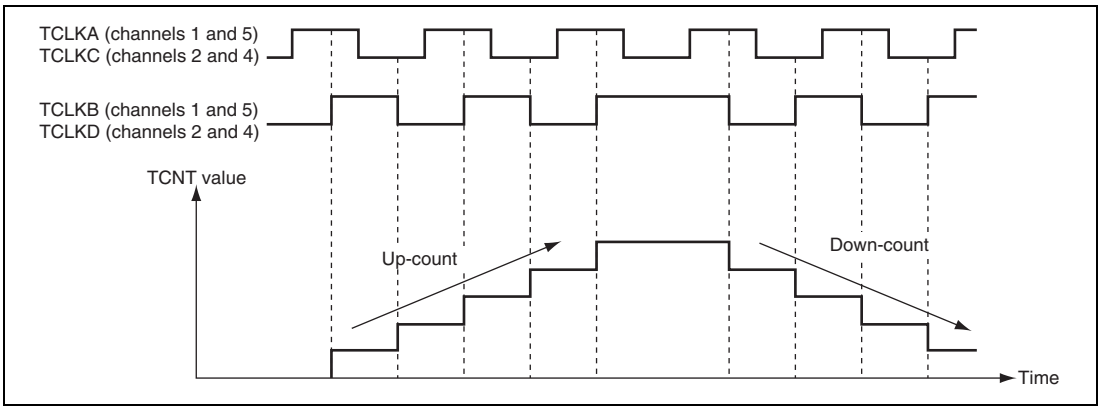


Figure 9.29 Example of Phase Counting Mode 4 Operation

Table 9.38 Up/Down-Count Conditions in Phase Counting Mode 4

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Operation
High level	↑	Up-count
Low level	↓	
↑	Low level	Don't care
↓	High level	
High level	↓	Down-count
Low level	↑	
↑	High level	Don't care
↓	Low level	

[Legend]
↑ : Rising edge
↓ : Falling edge

(3) Phase Counting Mode Application Example

Figure 9.30 shows an example in which phase counting mode is designated for channel 1, and channel 1 is coupled with channel 0 to input servo motor 2-phase encoder pulses in order to detect the position or speed.

Channel 1 is set to phase counting mode 1, and the encoder pulse A-phase and B-phase are input to TCLKA and TCLKB.

Channel 0 operates with TCNT counter clearing by TGRC_0 compare match; TGRA_0 and TGRC_0 are used for the compare match function and are set with the speed control cycle and position control cycle. TGRB_0 is used for input capture, with TGRB_0 and TGRD_0 operating in buffer mode. The channel 1 counter input clock is designated as the TGRB_0 input capture source, and the pulse width of 2-phase encoder 4-multiplication pulses is detected.

TGRA_1 and TGRB_1 for channel 1 are designated for input capture, channel 0 TGRA_0 and TGRC_0 compare matches are selected as the input capture source, and the up/down-counter values for the control cycles are stored.

This procedure enables accurate position/speed detection to be achieved.

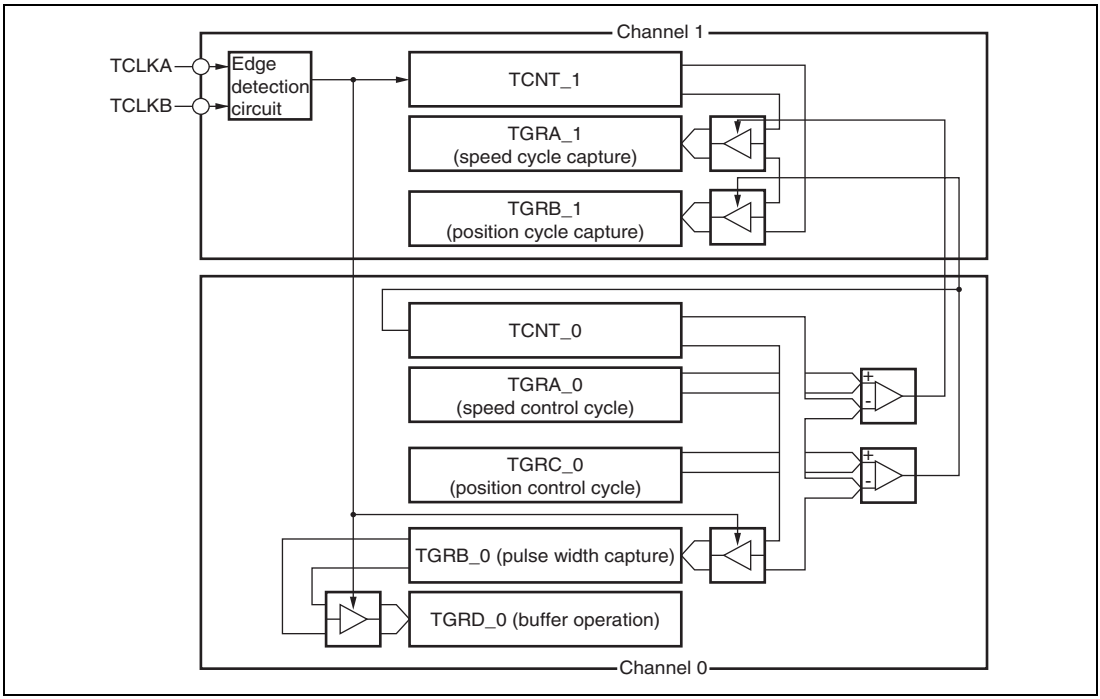


Figure 9.30 Phase Counting Mode Application Example

9.5 Interrupt Sources

There are three kinds of TPU interrupt sources: TGR input capture/compare match, TCNT overflow, and TCNT underflow. Each interrupt source has its own status flag and enable/disable bit, allowing generation of interrupt request signals to be enabled or disabled individually.

When an interrupt request is generated, the corresponding status flag in TSR is set to 1. If the corresponding enable/disable bit in TIER is set to 1 at this time, an interrupt is requested. The interrupt request is cleared by clearing the status flag to 0.

Relative channel priority levels can be changed by the interrupt controller, but the priority within a channel is fixed. For details, see section 5, Interrupt Controller.

Table 9.39 lists the TPU interrupt sources.

Table 9.39 TPU Interrupts

Channel	Name	Interrupt Source	Interrupt Flag	DMAC Activation
0	TGI0A	TGRA_0 input capture/compare match	TGFA_0	Possible
	TGI0B	TGRB_0 input capture/compare match	TGFB_0	Not possible
	TGI0C	TGRC_0 input capture/compare match	TGFC_0	Not possible
	TGI0D	TGRD_0 input capture/compare match	TGFD_0	Not possible
	TCI0V	TCNT_0 overflow	TCFV_0	Not possible
1	TGI1A	TGRA_1 input capture/compare match	TGFA_1	Possible
	TGI1B	TGRB_1 input capture/compare match	TGFB_1	Not possible
	TCI1V	TCNT_1 overflow	TCFV_1	Not possible
	TCI1U	TCNT_1 underflow	TCFU_1	Not possible
2	TGI2A	TGRA_2 input capture/compare match	TGFA_2	Possible
	TGI2B	TGRB_2 input capture/compare match	TGFB_2	Not possible
	TCI2V	TCNT_2 overflow	TCFV_2	Not possible
	TCI2U	TCNT_2 underflow	TCFU_2	Not possible

Channel	Name	Interrupt Source	Interrupt Flag	DMAC Activation
3	TGI3A	TGRA_3 input capture/compare match	TGFA_3	Possible
	TGI3B	TGRB_3 input capture/compare match	TGFB_3	Not possible
	TGI3C	TGRC_3 input capture/compare match	TGFC_3	Not possible
	TGI3D	TGRD_3 input capture/compare match	TGFD_3	Not possible
	TCI3V	TCNT_3 overflow	TCFV_3	Not possible
4	TGI4A*	TGRA_4 input capture/compare match	TGFA_4	Possible
	TGI4B*	TGRB_4 input capture/compare match	TGFB_4	Not possible
	TCI4V	TCNT_4 overflow	TCFV_4	Not possible
	TCI4U	TCNT_4 underflow	TCFU_4	Not possible
5	TGI5A*	TGRA_5 input capture/compare match	TGFA_5	Possible
	TGI5B*	TGRB_5 input capture/compare match	TGFB_5	Not possible
	TCI5V	TCNT_5 overflow	TCFV_5	Not possible
	TCI5U	TCNT_5 underflow	TCFU_5	Not possible
6	TGI6A	TGRA_0 input capture/compare match	TGFA_0	Possible
	TGI6B	TGRB_0 input capture/compare match	TGFB_0	Not possible
	TGI6C	TGRC_0 input capture/compare match	TGFC_0	Not possible
	TGI6D	TGRD_0 input capture/compare match	TGFD_0	Not possible
	TCI6V	TCNT_0 overflow	TCFV_0	Not possible
7	TGI7A	TGRA_1 input capture/compare match	TGFA_1	Possible
	TGI7B	TGRB_1 input capture/compare match	TGFB_1	Not possible
	TCI7V	TCNT_1 overflow	TCFV_1	Not possible
	TCI7U	TCNT_1 underflow	TCFU_1	Not possible
8	TGI8A	TGRA_2 input capture/compare match	TGFA_2	Possible
	TGI8B	TGRB_2 input capture/compare match	TGFB_2	Not possible
	TCI8V	TCNT_2 overflow	TCFV_2	Not possible
	TCI8U	TCNT_2 underflow	TCFU_2	Not possible
9	TGI9A	TGRA_3 input capture/compare match	TGFA_3	Possible
	TGI9B	TGRB_3 input capture/compare match	TGFB_3	Not possible
	TGI9C	TGRC_3 input capture/compare match	TGFC_3	Not possible
	TGI9D	TGRD_3 input capture/compare match	TGFD_3	Not possible
	TCI9V	TCNT_3 overflow	TCFV_3	Not possible

Channel	Name	Interrupt Source	Interrupt Flag	DMAC Activation
10	TGI10A	TGRA_4 input capture/compare match	TGFA_4	Possible
	TGI10B	TGRB_4 input capture/compare match	TGFB_4	Not possible
	TCI10V	TCNT_4 overflow	TCFV_4	Not possible
	TCI10U	TCNT_4 underflow	TCFU_4	Not possible
11	TGI11A	TGRA_5 input capture/compare match	TGFA_5	Possible
	TGI11B	TGRB_5 input capture/compare match	TGFB_5	Not possible
	TCI11V	TCNT_5 overflow	TCFV_5	Not possible
	TCI11U	TCNT_5 underflow	TCFU_5	Not possible

Note: 1. This table shows the initial state immediately after a reset. The relative channel priority levels can be changed by the interrupt controller.
2. The H8SX/1527R does not have the input capture function for channels 4 and 5.

(1) Input Capture/Compare Match Interrupt

An interrupt is requested if the TGIE bit in TIER is set to 1 when the TGF flag in TSR is set to 1 by the occurrence of a TGR input capture/compare match on a channel. The interrupt request is cleared by clearing the TGF flag to 0. The TPU has 16 input capture/compare match interrupts, four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5.

(2) Overflow Interrupt

An interrupt is requested if the TCIEV bit in TIER is set to 1 when the TCFV flag in TSR is set to 1 by the occurrence of a TCNT overflow on a channel. The interrupt request is cleared by clearing the TCFV flag to 0. The TPU has six overflow interrupts, one for each channel.

(3) Underflow Interrupt

An interrupt is requested if the TCIEU bit in TIER is set to 1 when the TCFU flag in TSR is set to 1 by the occurrence of a TCNT underflow on a channel. The interrupt request is cleared by clearing the TCFU flag to 0. The TPU has four underflow interrupts, one each for channels 1, 2, 4, and 5.

9.6 DMAC Activation

The DMAC can be activated by the TGR input capture/compare match interrupt for a channel. For details, see section 7, DMA Controller (DMAC).

A total of six TPU input capture/compare match interrupts can be used as DMAC activation sources, one for each channel.

9.7 A/D Converter Activation

The TGRA input capture/compare match for each channel of unit 0 can activate the A/D converter (this function is not available for unit 1).

If the TTGE bit in TIER is set to 1 when the TGFA flag in TSR is set to 1 by the occurrence of a TGRA input capture/compare match on a particular channel, a request to start A/D conversion is sent to the A/D converter. If the TPU conversion start trigger has been selected on the A/D converter side at this time, A/D conversion is started.

In the TPU, a total of six TGRA input capture/compare match interrupts can be used as A/D converter conversion start sources, one for each channel.

The A/D converter cannot be activated by unit 1.

9.8 Operation Timing

9.8.1 Input/Output Timing

(1) TCNT Count Timing

Figure 9.31 shows TCNT count timing in internal clock operation, and figure 9.32 shows TCNT count timing in external clock operation.

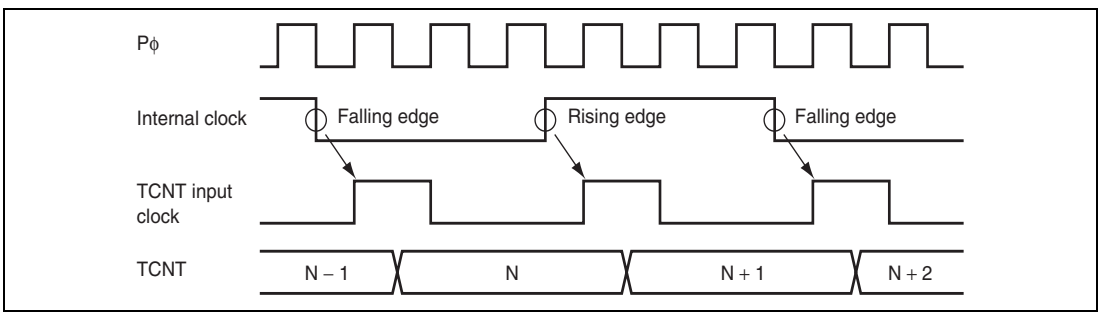


Figure 9.31 Count Timing in Internal Clock Operation

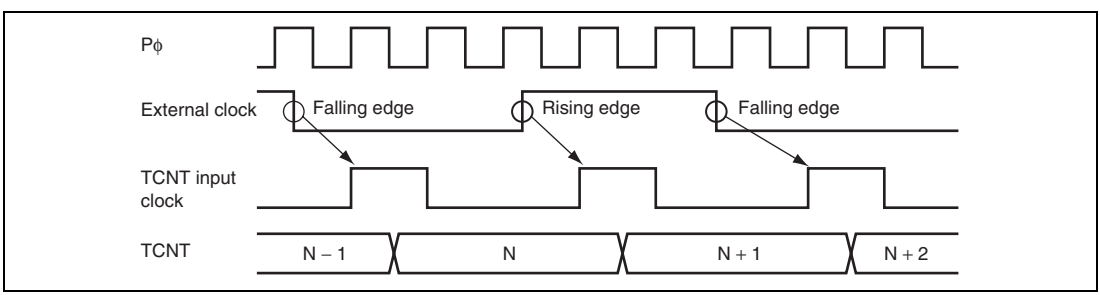


Figure 9.32 Count Timing in External Clock Operation

(2) Output Compare Output Timing

A compare match signal is generated in the final state in which TCNT and TGR match (the point at which the count value matched by TCNT is updated). When a compare match signal is generated, the output value set in TIOR is output at the output compare output pin (TIOC pin). After a match between TCNT and TGR, the compare match signal is not generated until the TCNT input clock is generated.

Figure 9.33 shows output compare output timing.

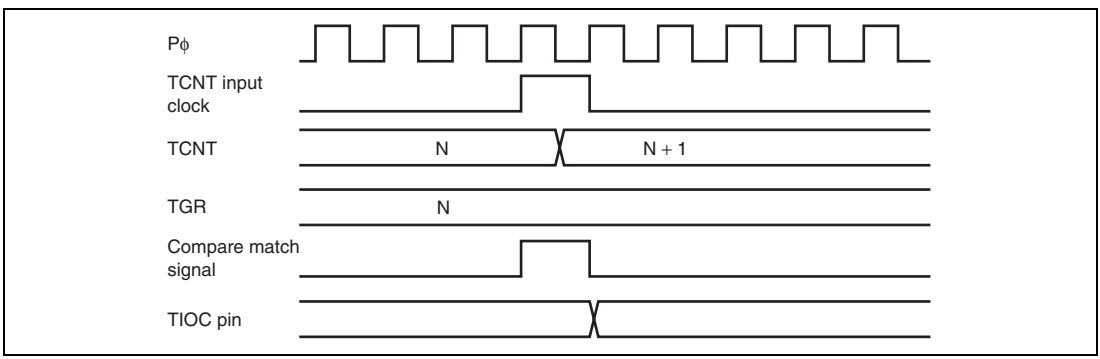


Figure 9.33 Output Compare Output Timing

(3) Input Capture Signal Timing

Figure 9.34 shows input capture signal timing.

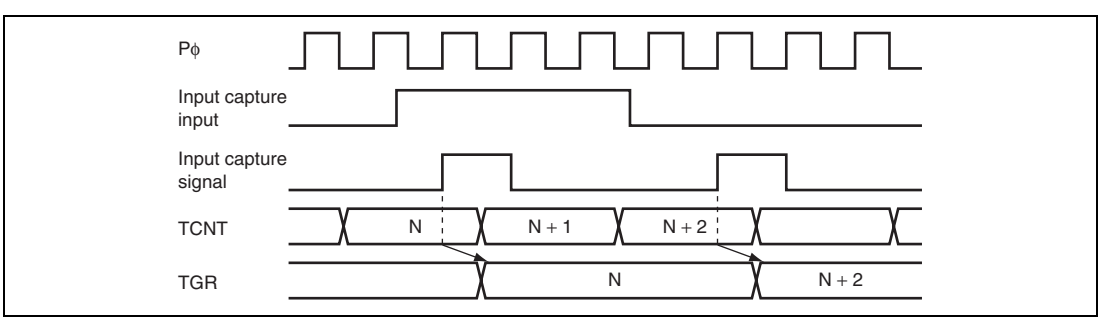


Figure 9.34 Input Capture Input Signal Timing

(4) Timing for Counter Clearing by Compare Match/Input Capture

Figure 9.35 shows the timing when counter clearing by compare match occurrence is specified, and figure 9.36 shows the timing when counter clearing by input capture occurrence is specified.

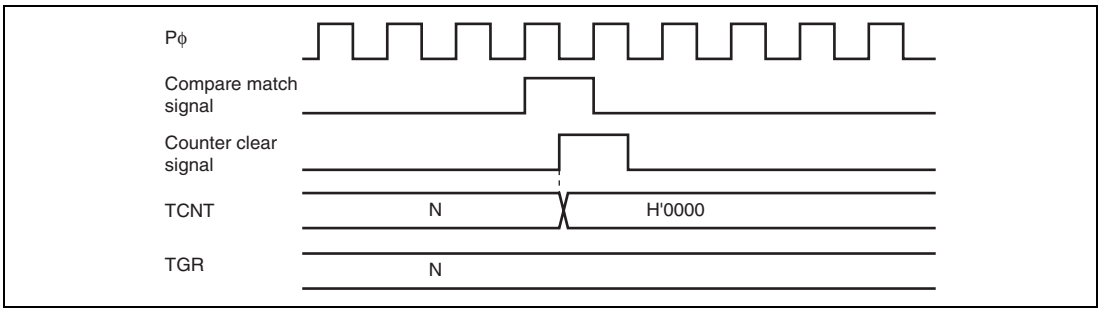


Figure 9.35 Counter Clear Timing (Compare Match)

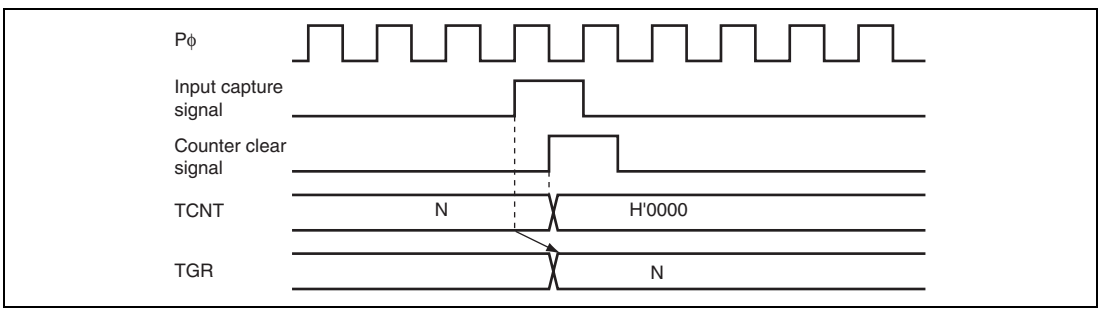


Figure 9.36 Counter Clear Timing (Input Capture)

(5) Buffer Operation Timing

Figures 9.37 and 9.38 show the timings in buffer operation.

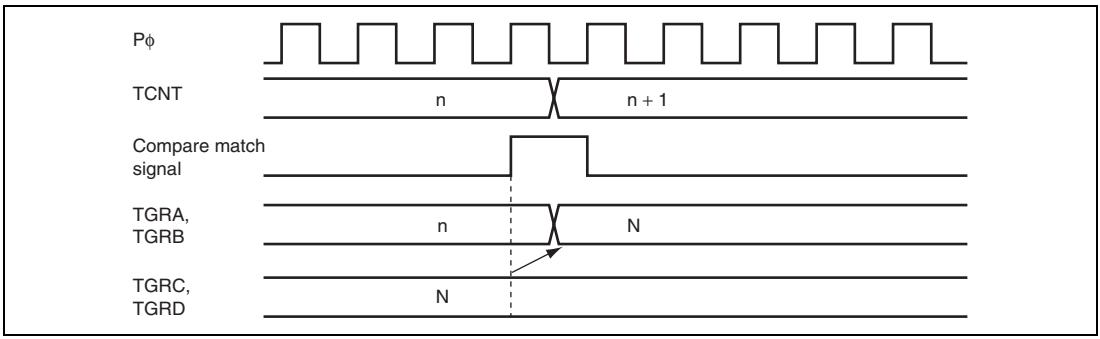


Figure 9.37 Buffer Operation Timing (Compare Match)

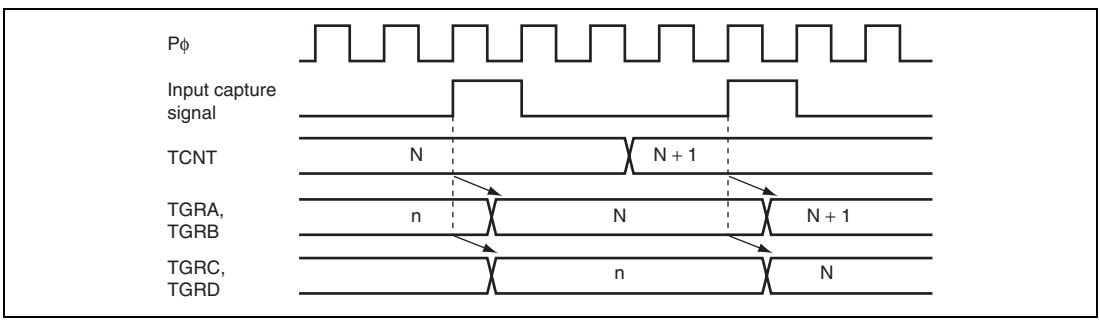


Figure 9.38 Buffer Operation Timing (Input Capture)

9.8.2 Interrupt Signal Timing

(1) TGF Flag Setting Timing in Case of Compare Match

Figure 9.39 shows the timing for setting of the TGF flag in TSR by compare match occurrence, and the TGI interrupt request signal timing.

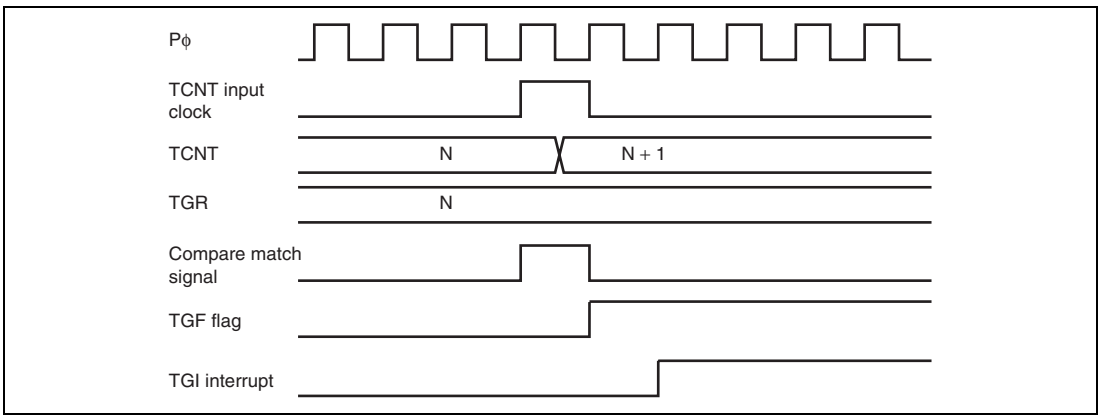


Figure 9.39 TGI Interrupt Timing (Compare Match)

(2) TGF Flag Setting Timing in Case of Input Capture

Figure 9.40 shows the timing for setting of the TGF flag in TSR by input capture occurrence, and the TGI interrupt request signal timing.

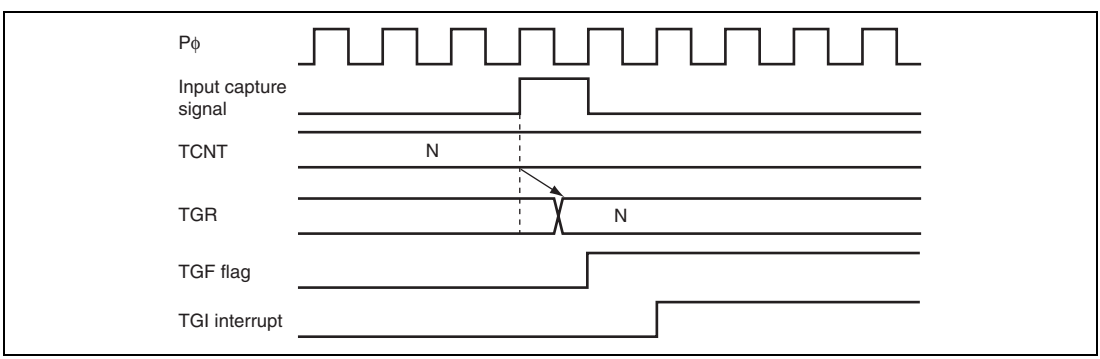


Figure 9.40 TGI Interrupt Timing (Input Capture)

(3) TCFV Flag/TCFU Flag Setting Timing

Figure 9.41 shows the timing for setting of the TCFV flag in TSR by overflow occurrence, and the TCIV interrupt request signal timing.

Figure 9.42 shows the timing for setting of the TCFU flag in TSR by underflow occurrence, and the TCIU interrupt request signal timing.

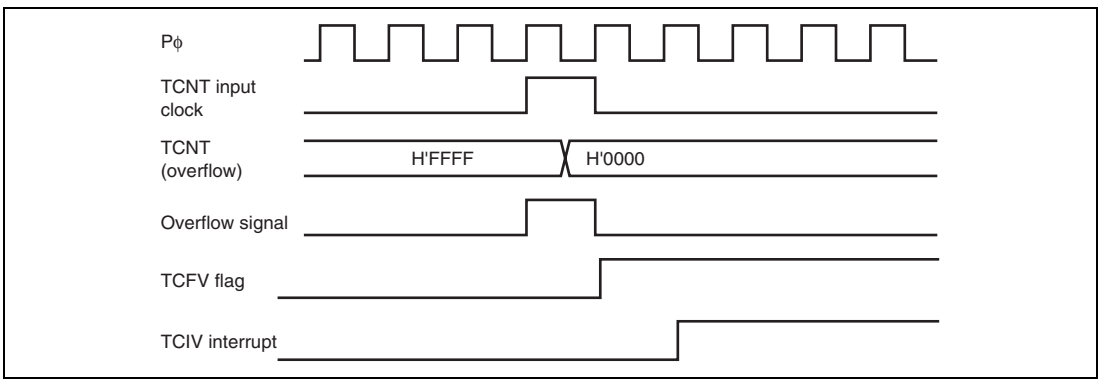


Figure 9.41 TCIV Interrupt Setting Timing

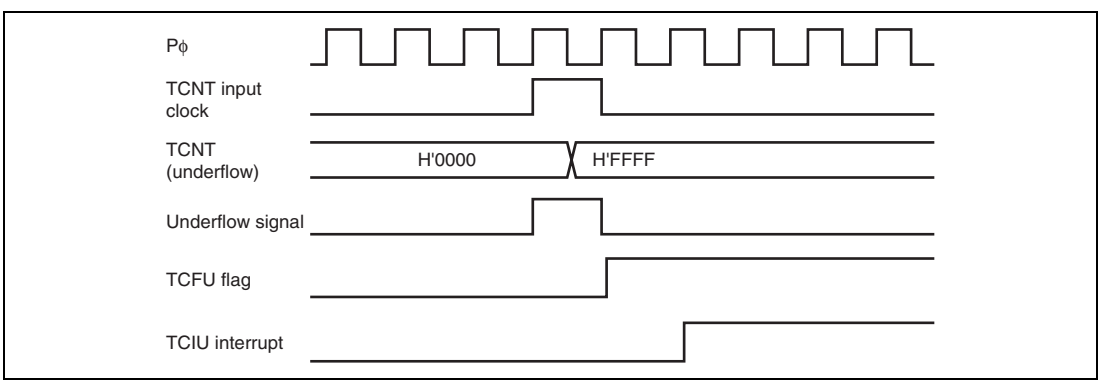


Figure 9.42 TCIU Interrupt Setting Timing

(4) Status Flag Clearing Timing

After a status flag is read as 1 by the CPU, it is cleared by writing 0 to it. When the DMAC is activated, the flag is cleared automatically. Figure 9.43 shows the timing for status flag clearing by the CPU, and figure 9.44 shows the timing for status flag clearing by the DMAC.

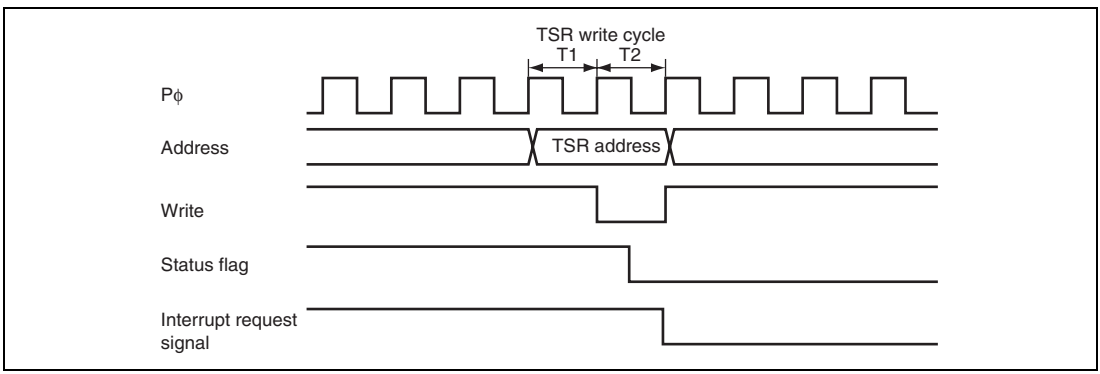


Figure 9.43 Timing for Status Flag Clearing by CPU

The status flag and interrupt request signal are cleared in synchronization with Pϕ after the DMAC transfer has started, as shown in figure 9.44. If conflict occurs for clearing the status flag and interrupt request signal due to activation of multiple DMAC transfers, it will take up to five clock cycles (Pϕ) for clearing them, as shown in figure 9.45. The next transfer request is masked for a longer period of either a period until the current transfer ends or a period for five clock cycles (Pϕ) from the beginning of the transfer.

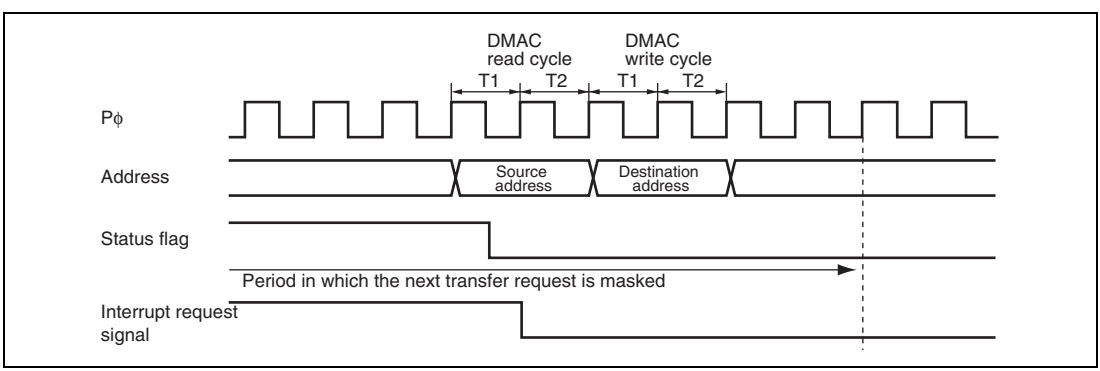


Figure 9.44 Timing for Status Flag Clearing by DMAC Activation (1)

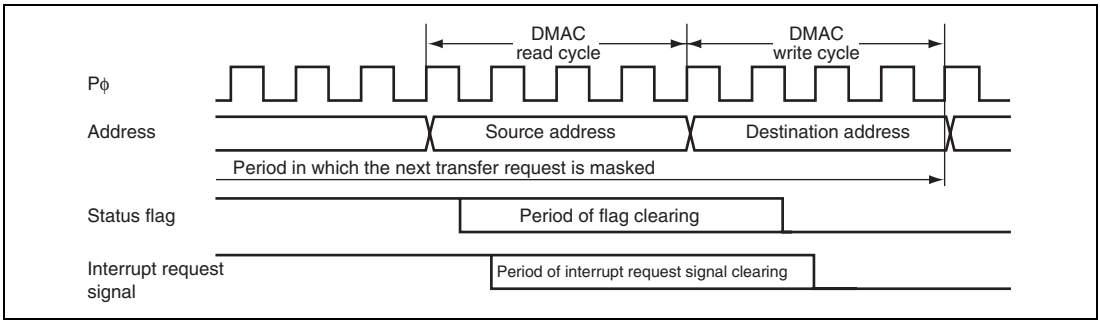


Figure 9.45 Timing for Status Flag Clearing by DMAC Activation (2)

9.9 Usage Notes

9.9.1 Module Stop Mode Setting

Operation of the TPU can be disabled or enabled using the module stop control register. The initial setting is for operation of the TPU to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 19, Power-Down Modes.

9.9.2 Input Clock Restrictions

The input clock pulse width must be at least 1.5 states in the case of single-edge detection, and at least 2.5 states in the case of both-edge detection. The TPU will not operate properly with a narrower pulse width.

In phase counting mode, the phase difference and overlap between the two input clocks must be at least 1.5 states, and the pulse width must be at least 2.5 states. Figure 9.46 shows the input clock conditions in phase counting mode.

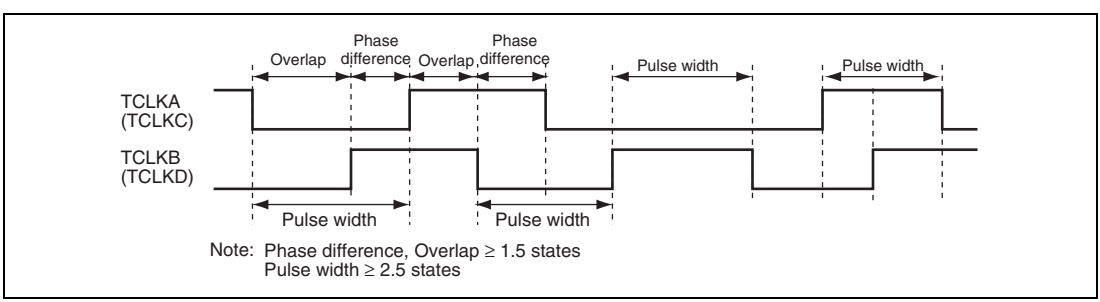


Figure 9.46 Phase Difference, Overlap, and Pulse Width in Phase Counting Mode

9.9.3 Caution on Cycle Setting

When counter clearing by compare match is set, TCNT is cleared in the final state in which it matches the TGR value (the point at which the count value matched by TCNT is updated). Consequently, the actual counter frequency is given by the following formula:

$$f = \frac{P\phi}{(N + 1)}$$

f: Counter frequency

Pϕ: Operating frequency

N: TGR set value

9.9.4 Conflict between TCNT Write and Clear Operations

If the counter clearing signal is generated in the T2 state of a TCNT write cycle, TCNT clearing takes precedence and the TCNT write is not performed. Figure 9.47 shows the timing in this case.

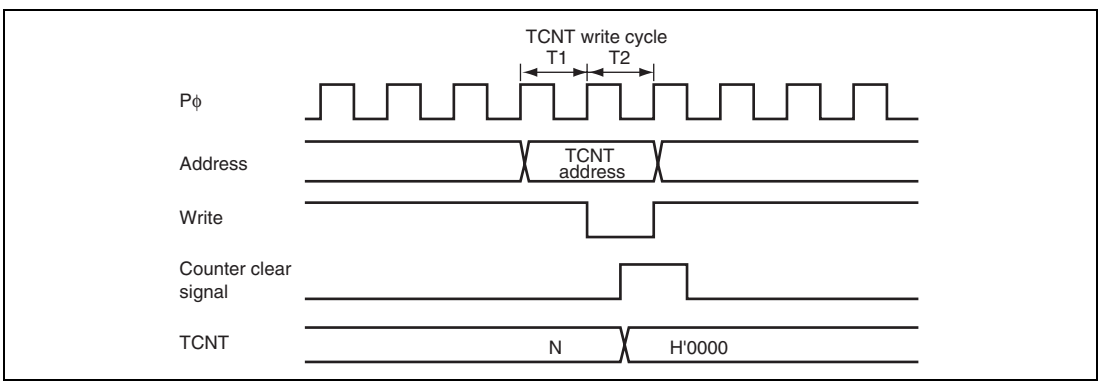


Figure 9.47 Conflict between TCNT Write and Clear Operations

9.9.5 Conflict between TCNT Write and Increment Operations

If incrementing occurs in the T2 state of a TCNT write cycle, the TCNT write takes precedence and TCNT is not incremented. Figure 9.48 shows the timing in this case.

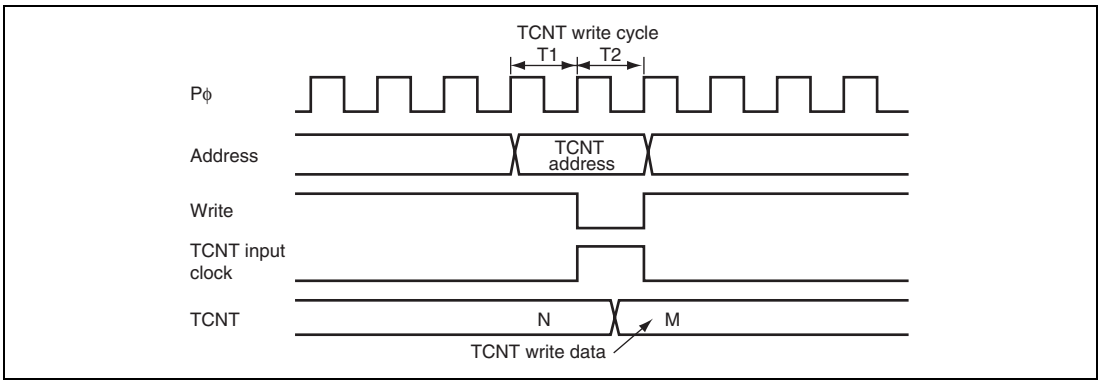


Figure 9.48 Conflict between TCNT Write and Increment Operations

9.9.6 Conflict between TGR Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the TGR write takes precedence and the compare match signal is disabled. A compare match also does not occur when the same value as before is written.

Figure 9.49shows the timing in this case.

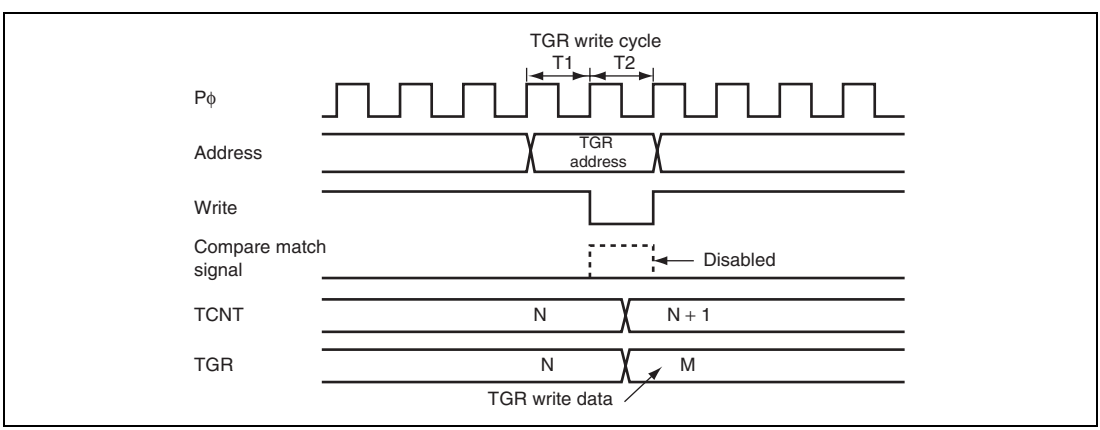


Figure 9.49 Conflict between TGR Write and Compare Match

9.9.7 Conflict between Buffer Register Write and Compare Match

If a compare match occurs in the T2 state of a TGR write cycle, the data transferred to TGR by the buffer operation will be the write data.

Figure 9.50 shows the timing in this case.

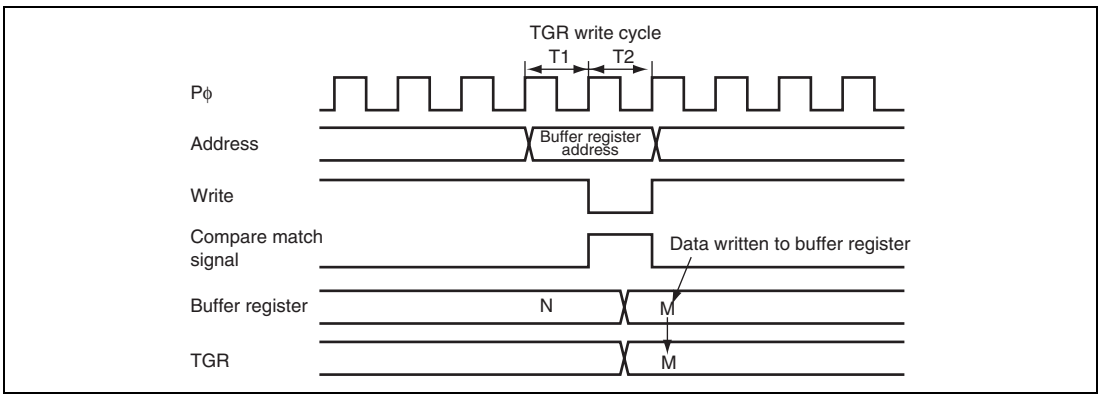


Figure 9.50 Conflict between Buffer Register Write and Compare Match

9.9.8 Conflict between TGR Read and Input Capture

If the input capture signal is generated in the T1 state of a TGR read cycle, the data that is read will be the data after input capture transfer.

Figure 9.51 shows the timing in this case.

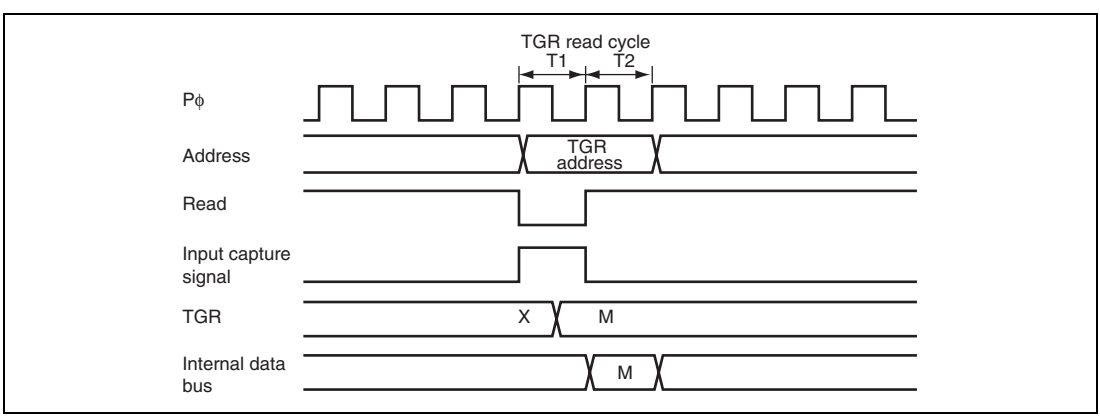


Figure 9.51 Conflict between TGR Read and Input Capture

9.9.9 Conflict between TGR Write and Input Capture

If the input capture signal is generated in the T2 state of a TGR write cycle, the input capture operation takes precedence and the write to TGR is not performed.

Figure 9.52 shows the timing in this case.

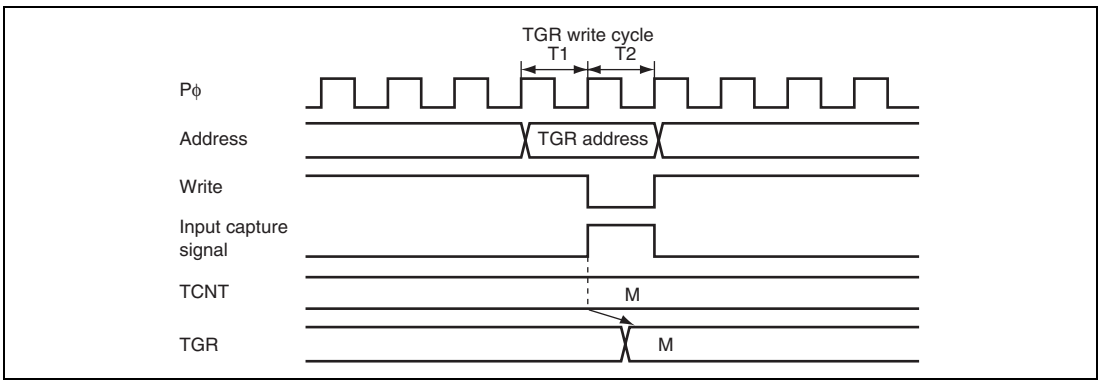


Figure 9.52 Conflict between TGR Write and Input Capture

9.9.10 Conflict between Buffer Register Write and Input Capture

If the input capture signal is generated in the T2 state of a buffer register write cycle, the buffer operation takes precedence and the write to the buffer register is not performed.

Figure 9.53 shows the timing in this case.

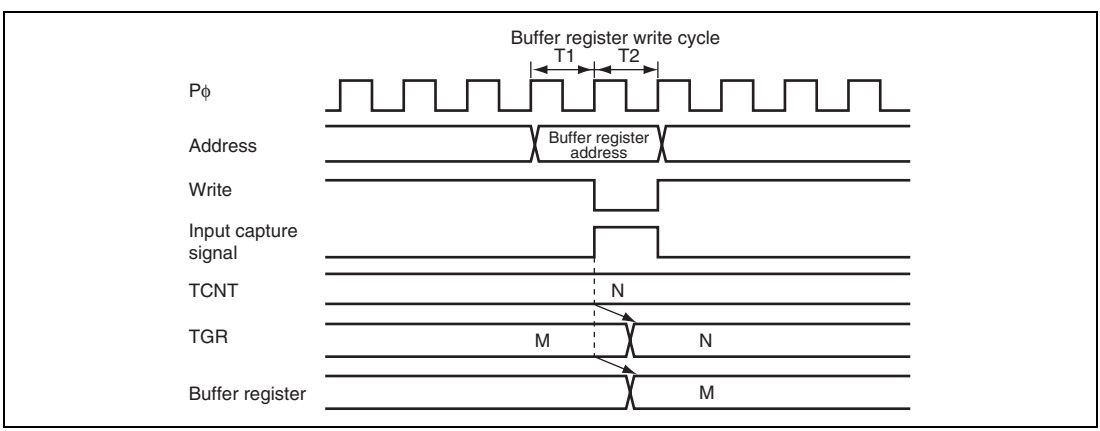


Figure 9.53 Conflict between Buffer Register Write and Input Capture

9.9.11 Conflict between Overflow/Underflow and Counter Clearing

If overflow/underflow and counter clearing occur simultaneously, the TCFV/TCFU flag in TSR is not set and TCNT clearing takes precedence.

Figure 9.54 shows the operation timing when a TGR compare match is specified as the clearing source, and H'FFFF is set in TGR.

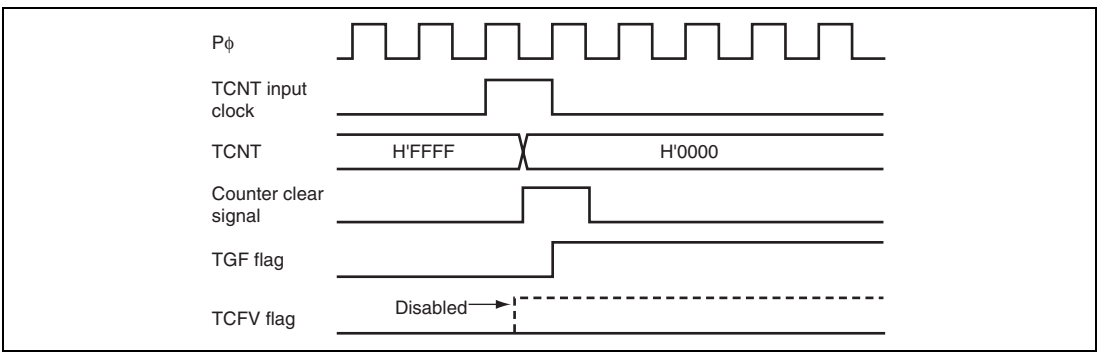


Figure 9.54 Conflict between Overflow and Counter Clearing

9.9.12 Conflict between TCNT Write and Overflow/Underflow

If an overflow/underflow occurs due to increment/decrement in the T2 state of a TCNT write cycle, the TCNT write takes precedence and the TCFV/TCFU flag in TSR is not set.

Figure 9.55 shows the operation timing when there is conflict between TCNT write and overflow.

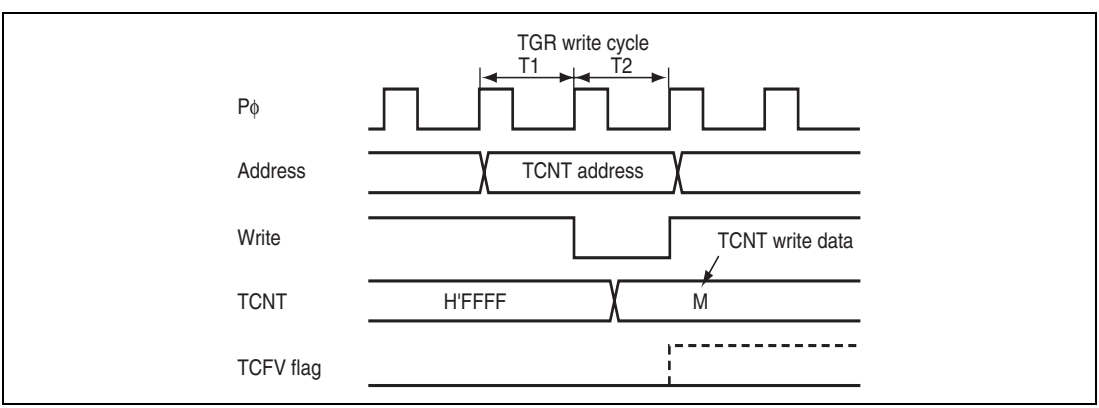


Figure 9.55 Conflict between TCNT Write and Overflow

9.9.13 Multiplexing of I/O Pins

In this LSI, the TCLKA input pin is multiplexed with the TIOCC0 I/O pin, the TCLKB input pin with the TIOCD0 I/O pin, the TCLKC input pin with the TIOCB1 I/O pin, and the TCLKD input pin with the TIOCB2 I/O pin. When an external clock is input, compare match output should not be performed from a multiplexed pin.

9.9.14 Interrupts and Module Stop Mode

If module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC activation source. Interrupts should therefore be disabled before entering module stop mode.

Section 10 Programmable Pulse Generator (PPG)

The programmable pulse generator (PPG) provides pulse outputs by using the 16-bit timer pulse unit (TPU) as a time base. The PPG pulse outputs are divided into 4-bit groups (groups 3 and 2) that can operate both simultaneously and independently. Figure 10.1 shows a block diagram of the PPG.

10.1 Features

- 8-bit output data
- Two output groups
- Selectable output trigger signals
- Non-overlapping mode
- Can operate together with the DMA controller (DMAC)
- Inverted output can be set
- Module stop mode can be set

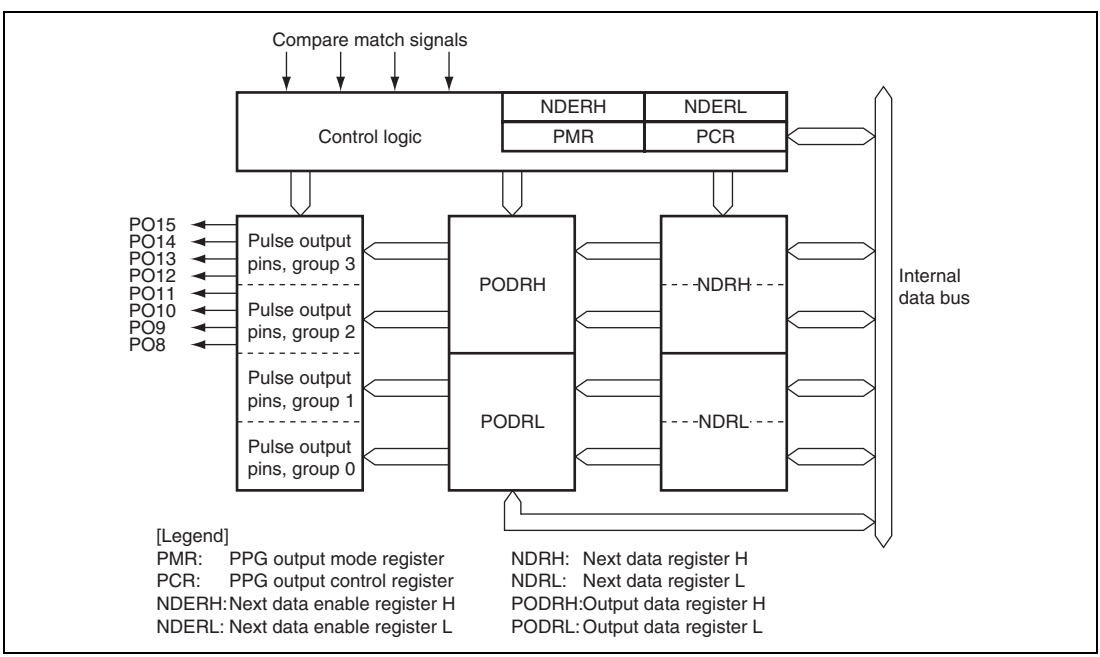


Figure 10.1 Block Diagram of PPG

10.2 Input/Output Pins

Table 10.1 shows the PPG pin configuration.

Table 10.1 Pin Configuration

Pin Name	I/O	Function
PO15	Output	Group 3 pulse output
PO14	Output	
PO13	Output	
PO12	Output	
PO11	Output	Group 2 pulse output
PO10	Output	
PO9	Output	
PO8	Output	

10.3 Register Descriptions

The PPG has the following registers.

- Next data enable register H (NDERH)
- Next data enable register L (NDERL)
- Output data register H (PODRH)
- Output data register L (PODRL)
- Next data register H (NDRH)
- Next data register L (NDRL)
- PPG output control register (PCR)
- PPG output mode register (PMR)

10.3.1 Next Data Enable Registers H, L (NDERH, NDERL)

NDERH and NDERL enable/disable pulse output on a bit-by-bit basis.

- NDERH

Bit	7	6	5	4	3	2	1	0
Bit Name	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8
Initial Value	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

- NDERL

Bit	7	6	5	4	3	2	1	0
Bit Name	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0
Initial Value	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

- NDERH

Bit	Bit Name	Initial Value	R/W	Description
7	NDER15	0	R/W	Next Data Enable 15 to 8
6	NDER14	0	R/W	When a bit is set to 1, the value in the corresponding NDRH bit is transferred to the PODRH bit by the selected output trigger. Values are not transferred from NDRH to PODRH for cleared bits.
5	NDER13	0	R/W	
4	NDER12	0	R/W	
3	NDER11	0	R/W	
2	NDER10	0	R/W	
1	NDER9	0	R/W	
0	NDER8	0	R/W	

- NDERL

Bit	Bit Name	Initial Value	R/W	Description
7	NDER7	0	R/W	Next Data Enable 7 to 0
6	NDER6	0	R/W	When a bit is set to 1, the value in the corresponding NDRL bit is transferred to the PODRL bit by the selected output trigger. Values are not transferred from NDRL to PODRL for cleared bits.
5	NDER5	0	R/W	
4	NDER4	0	R/W	
3	NDER3	0	R/W	
2	NDER2	0	R/W	
1	NDER1	0	R/W	
0	NDER0	0	R/W	

10.3.2 Output Data Registers H, L (PODRH, PODRL)

PODRH and PODRL store output data for use in pulse output. A bit that has been set for pulse output by NDER is read-only and cannot be modified.

- PODRH

Bit	7	6	5	4	3	2	1	0
Bit Name	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8
Initial Value	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

- PODRL

Bit	7	6	5	4	3	2	1	0
Bit Name	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0
Initial Value	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

- PODRH

Bit	Bit Name	Initial Value	R/W	Description
7	POD15	0	R/W	Output Data Register 15 to 8
6	POD14	0	R/W	For bits which have been set to pulse output by NDERH, the output trigger transfers NDRH values to this register during PPG operation. While NDERH is set to 1, the CPU cannot write to this register. While NDERH is cleared, the initial output value of the pulse can be set.
5	POD13	0	R/W	
4	POD12	0	R/W	
3	POD11	0	R/W	
2	POD10	0	R/W	
1	POD9	0	R/W	
0	POD8	0	R/W	

- PODRL

Bit	Bit Name	Initial Value	R/W	Description
7	POD7	0	R/W	Output Data Register 7 to 0
6	POD6	0	R/W	For bits which have been set to pulse output by NDERL, the output trigger transfers NDRL values to this register during PPG operation. While NDERL is set to 1, the CPU cannot write to this register. While NDERL is cleared, the initial output value of the pulse can be set.
5	POD5	0	R/W	
4	POD4	0	R/W	
3	POD3	0	R/W	
2	POD2	0	R/W	
1	POD1	0	R/W	
0	POD0	0	R/W	

10.3.3 Next Data Registers H, L (NDRH, NDRL)

NDRH and NDRL store the next data for pulse output. The NDR addresses differ depending on whether pulse output groups have the same output trigger or different output triggers.

- NDRH

Bit	7	6	5	4	3	2	1	0
Bit Name	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8
Initial Value	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

- NDRL

Bit	7	6	5	4	3	2	1	0
Bit Name	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0
Initial Value	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

- NDRH

  If pulse output groups 2 and 3 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 8
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3	NDR11	0	R/W	
2	NDR10	0	R/W	
1	NDR9	0	R/W	
0	NDR8	0	R/W	

If pulse output groups 2 and 3 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR15	0	R/W	Next Data Register 15 to 12
6	NDR14	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
5	NDR13	0	R/W	
4	NDR12	0	R/W	
3 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These are read-only bits and cannot be modified.
3	NDR11	0	R/W	Next Data Register 11 to 8
2	NDR10	0	R/W	The register contents are transferred to the corresponding PODRH bits by the output trigger specified with PCR.
1	NDR9	0	R/W	
0	NDR8	0	R/W	

- NDRL

  If pulse output groups 0 and 1 have the same output trigger, all eight bits are mapped to the same address and can be accessed at one time, as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 0
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3	NDR3	0	R/W	
2	NDR2	0	R/W	
1	NDR1	0	R/W	
0	NDR0	0	R/W	

If pulse output groups 0 and 1 have different output triggers, the upper four bits and lower four bits are mapped to different addresses as shown below.

Bit	Bit Name	Initial Value	R/W	Description
7	NDR7	0	R/W	Next Data Register 7 to 4
6	NDR6	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
5	NDR5	0	R/W	
4	NDR4	0	R/W	
3 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These are read-only bits and cannot be modified.
3	NDR3	0	R/W	Next Data Register 3 to 0
2	NDR2	0	R/W	The register contents are transferred to the corresponding PODRL bits by the output trigger specified with PCR.
1	NDR1	0	R/W	
0	NDR0	0	R/W	

10.3.4 PPG Output Control Register (PCR)

PCR selects output trigger signals on a group-by-group basis. For details on output trigger selection, refer to section 10.3.5, PPG Output Mode Register (PMR).

Bit	7	6	5	4	3	2	1	0
Bit Name	G3CMS1	G3CMS0	G2CMS1	G2CMS0	—	—	—	—
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	G3CMS1	1	R/W	Group 3 Compare Match Select 1 and 0
6	G3CMS0	1	R/W	These bits select output trigger of pulse output group 3. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
5	G2CMS1	1	R/W	Group 2 Compare Match Select 1 and 0
4	G2CMS0	1	R/W	These bits select output trigger of pulse output group 2. 00: Compare match in TPU channel 0 01: Compare match in TPU channel 1 10: Compare match in TPU channel 2 11: Compare match in TPU channel 3
3 to 0	—	All 1	R/W	Reserved These bits are always read as 1. The write value should always be 1.

10.3.5 PPG Output Mode Register (PMR)

PMR selects the pulse output mode of the PPG for each group. If inverted output is selected, a low-level pulse is output when PODRH is 1 and a high-level pulse is output when PODRH is 0. If non-overlapping operation is selected, PPG updates its output values at compare match A or B of the TPU that becomes the output trigger. For details, refer to section 10.4.4, Non-Overlapping Pulse Output.

Bit	7	6	5	4	3	2	1	0
Bit Name	G3INV	G2INV	—	—	G3NOV	G2NOV	—	—
Initial Value	1	1	1	1	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	G3INV	1	R/W	Group 3 Inversion Selects direct output or inverted output for pulse output group 3. 0: Inverted output 1: Direct output
6	G2INV	1	R/W	Group 2 Inversion Selects direct output or inverted output for pulse output group 2. 0: Inverted output 1: Direct output
5, 4	—	All 1	R/W	Reserved These bits are always read as 1. The write value should always be 1.

Bit	Bit Name	Initial Value	R/W	Description
3	G3NOV	0	R/W	Group 3 Non-Overlap Selects normal or non-overlapping operation for pulse output group 3. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
2	G2NOV	0	R/W	Group 2 Non-Overlap Selects normal or non-overlapping operation for pulse output group 2. 0: Normal operation (output values updated at compare match A in the selected TPU channel) 1: Non-overlapping operation (output values updated at compare match A or B in the selected TPU channel)
1, 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

10.4 Operation

Figure 10.2 shows a schematic diagram of the PPG. PPG pulse output is enabled when the corresponding bits in NDER are set to 1. An initial output value is determined by its corresponding PODR initial setting. When the compare match event specified by PCR occurs, the corresponding NDR bit contents are transferred to PODR to update the output values. Sequential output of data of up to eight bits is possible by writing new output data to NDR before the next compare match.

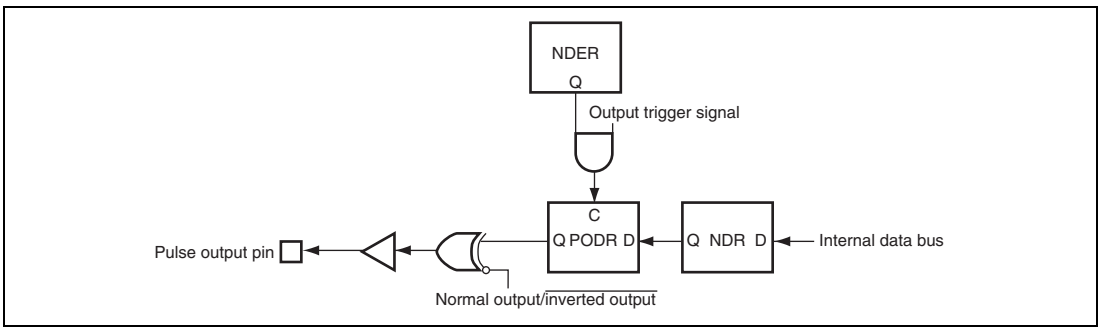


Figure 10.2 Schematic Diagram of PPG

10.4.1 Output Timing

If pulse output is enabled, the NDR contents are transferred to PODR and output when the specified compare match event occurs. Figure 10.3 shows the timing of these operations for the case of normal output in groups 2 and 3, triggered by compare match A.

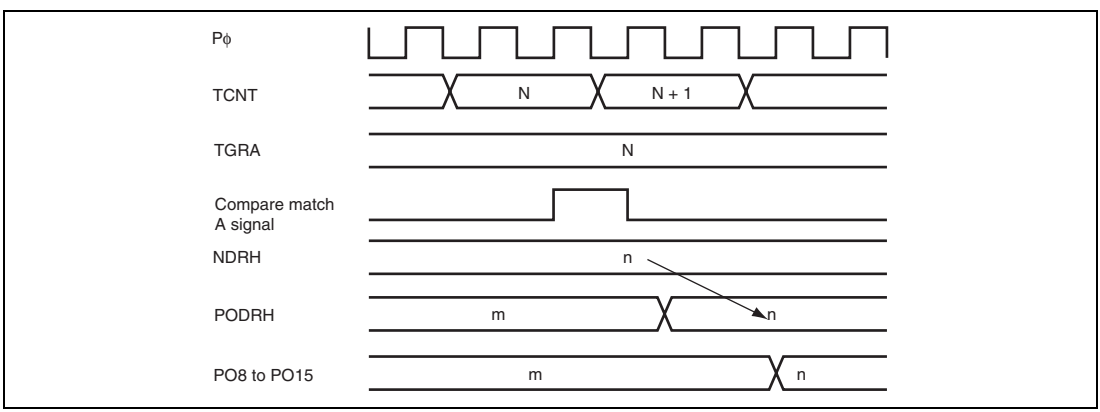


Figure 10.3 Timing of Transfer and Output of NDR Contents (Example)

10.4.2 Sample Setup Procedure for Normal Pulse Output

Figure 10.4 shows a sample procedure for setting up normal pulse output.

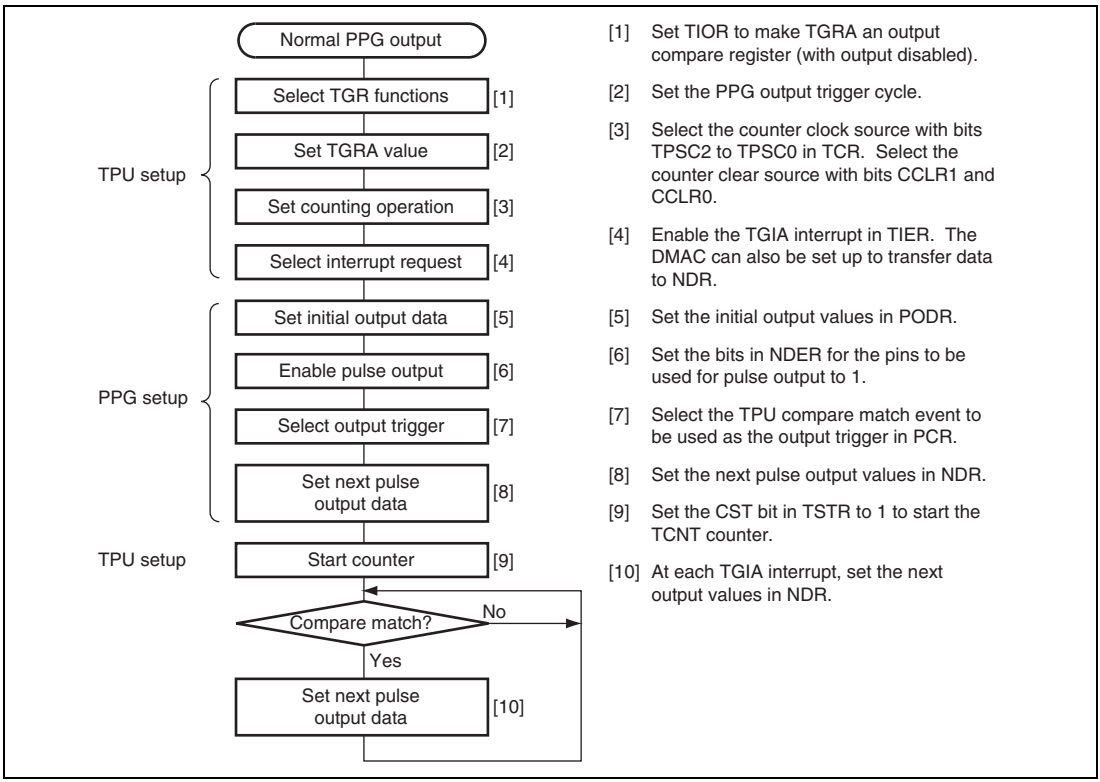


Figure 10.4 Setup Procedure for Normal Pulse Output (Example)

10.4.3 Example of Normal Pulse Output (Example of 5-Phase Pulse Output)

Figure 10.5 shows an example in which pulse output is used for cyclic 5-phase pulse output.

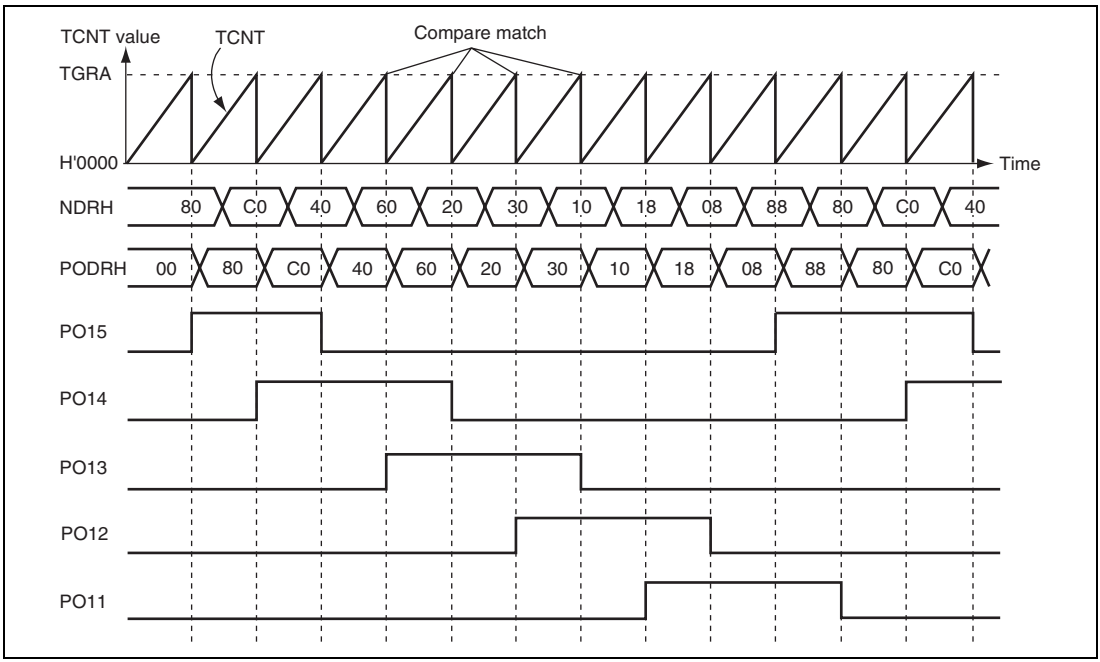


Figure 10.5 Normal Pulse Output Example (5-Phase Pulse Output)

1. Set up TGRA in TPU which is used as the output trigger to be an output compare register. Set a cycle in TGRA so the counter will be cleared by compare match A. Set the TGIEA bit in TIER to 1 to enable the compare match/input capture A (TGIA) interrupt.
2. Write H'F8 to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger. Write output data H'80 in NDRH.
3. The timer counter in the TPU channel starts. When compare match A occurs, the NDRH contents are transferred to PODRH and output. The TGIA interrupt handling routine writes the next output data (H'C0) in NDRH.
4. 5-phase pulse output (one or two phases active at a time) can be obtained subsequently by writing H'40, H'60, H'20, H'30, H'10, H'18, H'08, H'88... at successive TGIA interrupts.

   If the DMAC is set for activation by the TGIA interrupt, pulse output can be obtained without imposing a load on the CPU.

10.4.4 Non-Overlapping Pulse Output

During non-overlapping operation, transfer from NDR to PODR is performed as follows:

- At compare match A, the NDR bits are always transferred to PODR.
- At compare match B, the NDR bits are transferred only if their value is 0. The NDR bits are not transferred if their value is 1.

Figure 10.6 illustrates the non-overlapping pulse output operation.

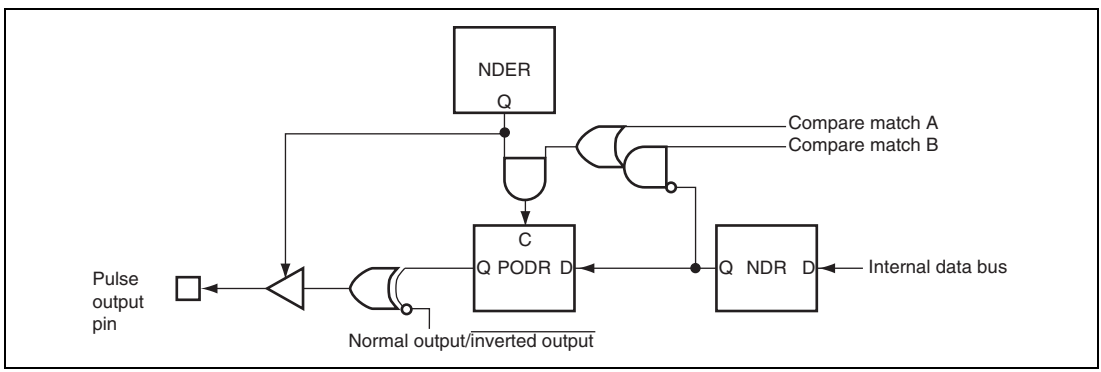


Figure 10.6 Non-Overlapping Pulse Output

Therefore, 0 data can be transferred ahead of 1 data by making compare match B occur before compare match A.

The NDR contents should not be altered during the interval from compare match B to compare match A (the non-overlapping margin).

This can be accomplished by having the TGIA interrupt handling routine write the next data in NDR, or by having the TGIA interrupt activate the DMAC. Note, however, that the next data must be written before the next compare match B occurs.

Figure 10.7 shows the timing of this operation.

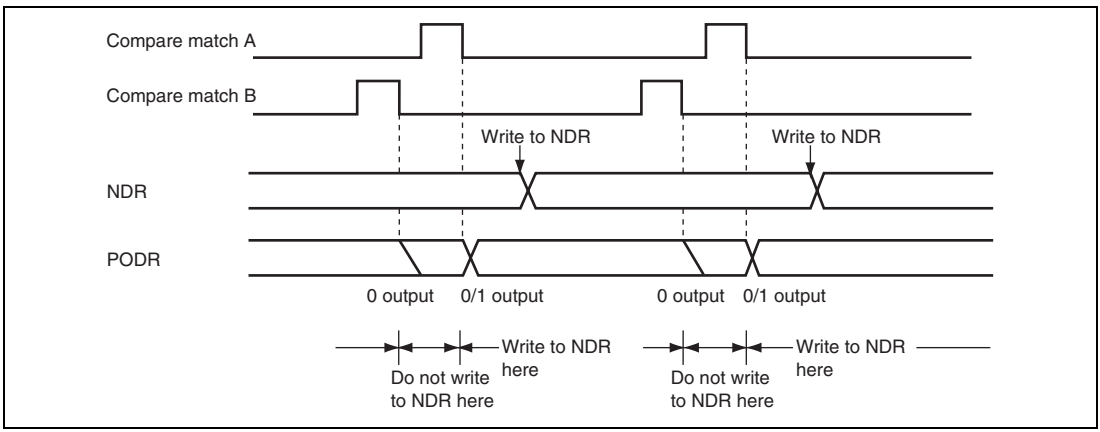
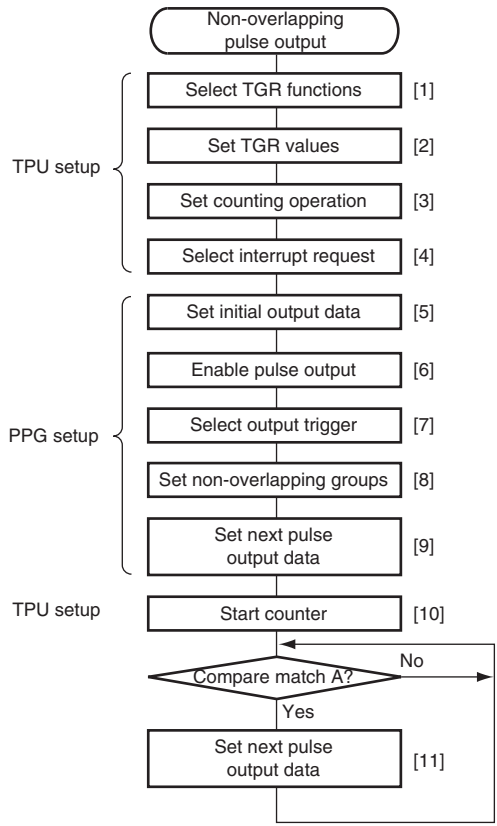


Figure 10.7 Non-Overlapping Operation and NDR Write Timing

10.4.5 Sample Setup Procedure for Non-Overlapping Pulse Output

Figure 10.8 shows a sample procedure for setting up non-overlapping pulse output.



[1] Set TIOR to make TGRA and TGRB output compare registers (with output disabled).

[2] Set the pulse output trigger cycle in TGRB and the non-overlapping margin in TGRA.

[3] Select the counter clock source with bits TPSC2 to TPSC0 in TCR. Select the counter clear source with bits CCLR1 and CCLR0.

[4] Enable the TGIA interrupt in TIER. The DMAC can also be set up to transfer data to NDR.

[5] Set the initial output values in PODR.

[6] Set the bits in NDER for the pins to be used for pulse output to 1.

[7] Select the TPU compare match event to be used as the pulse output trigger in PCR.

[8] In PMR, select the groups that will operate in non-overlapping mode.

[9] Set the next pulse output values in NDR.

[10] Set the CST bit in TSTR to 1 to start the TCNT counter.

[11] At each TGIA interrupt, set the next output values in NDR.

Figure 10.8 Setup Procedure for Non-Overlapping Pulse Output (Example)

10.4.6 Example of Non-Overlapping Pulse Output (Example of 4-Phase Complementary Non-Overlapping Pulse Output)

Figure 10.9 shows an example in which pulse output is used for 4-phase complementary non-overlapping pulse output.

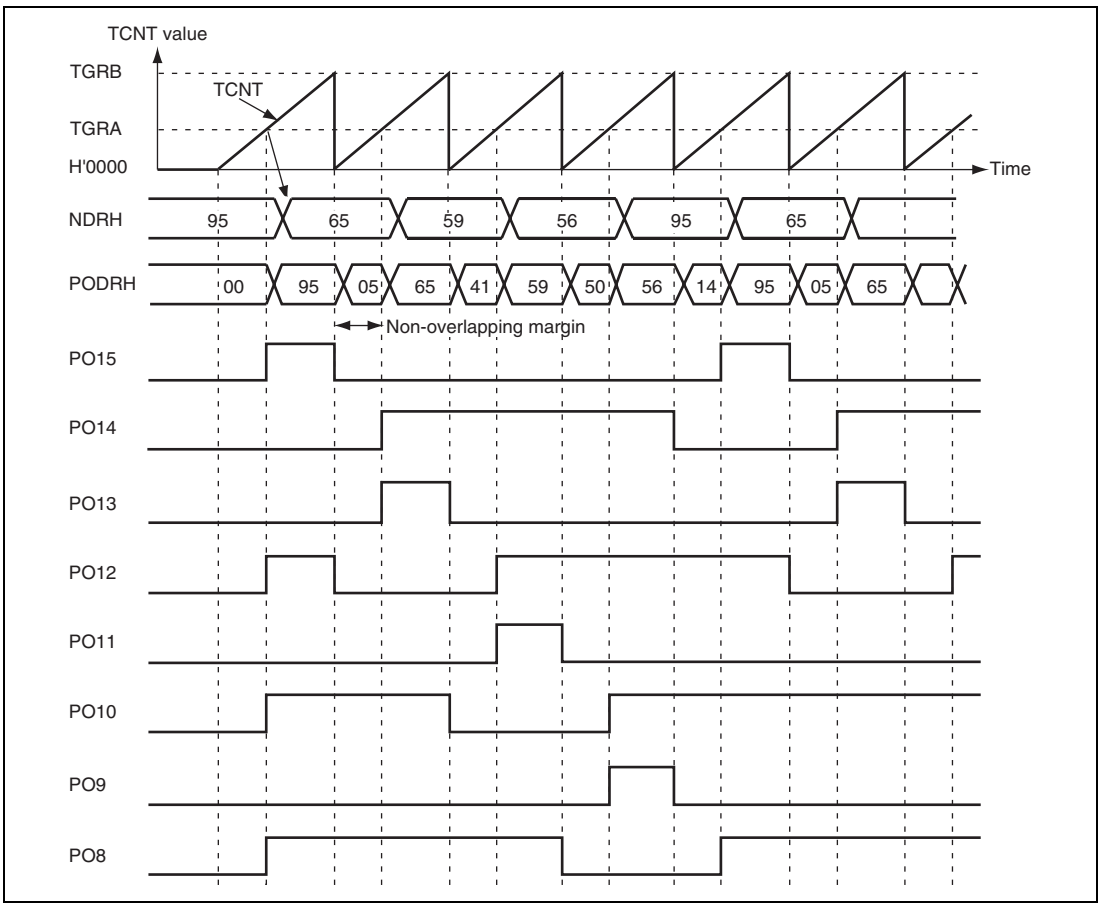


Figure 10.9 Non-Overlapping Pulse Output Example (4-Phase Complementary)

1. Set up the TPU channel to be used as the output trigger channel so that TGRA and TGRB are output compare registers. Set the cycle in TGRB and the non-overlapping margin in TGRA, and set the counter to be cleared by compare match B. Set the TGIEA bit in TIER to 1 to enable the TGIA interrupt.
2. Write H'FF to NDERH, and set bits G3CMS1, G3CMS0, G2CMS1, and G2CMS0 in PCR to select compare match in the TPU channel set up in the previous step to be the output trigger.

   Set bits G3NOV and G2NOV in PMR to 1 to select non-overlapping pulse output.

   Write output data H'95 to NDRH.
3. The timer counter in the TPU channel starts. When a compare match with TGRB occurs, outputs change from 1 to 0. When a compare match with TGRA occurs, outputs change from 0 to 1 (the change from 0 to 1 is delayed by the value set in TGRA).

   The TGIA interrupt handling routine writes the next output data (H'65) to NDRH.
4. 4-phase complementary non-overlapping pulse output can be obtained subsequently by writing H'59, H'56, H'95... at successive TGIA interrupts.

   If the DMAC is set for activation by a TGIA interrupt, pulse can be output without imposing a load on the CPU.

10.4.7 Inverted Pulse Output

If the G3INV and G2INV bits in PMR are cleared to 0, values that are the inverse of the PODR contents can be output.

Figure 10.10 shows the outputs when the G3INV and G2INV bits are cleared to 0, in addition to the settings of figure 10.9.

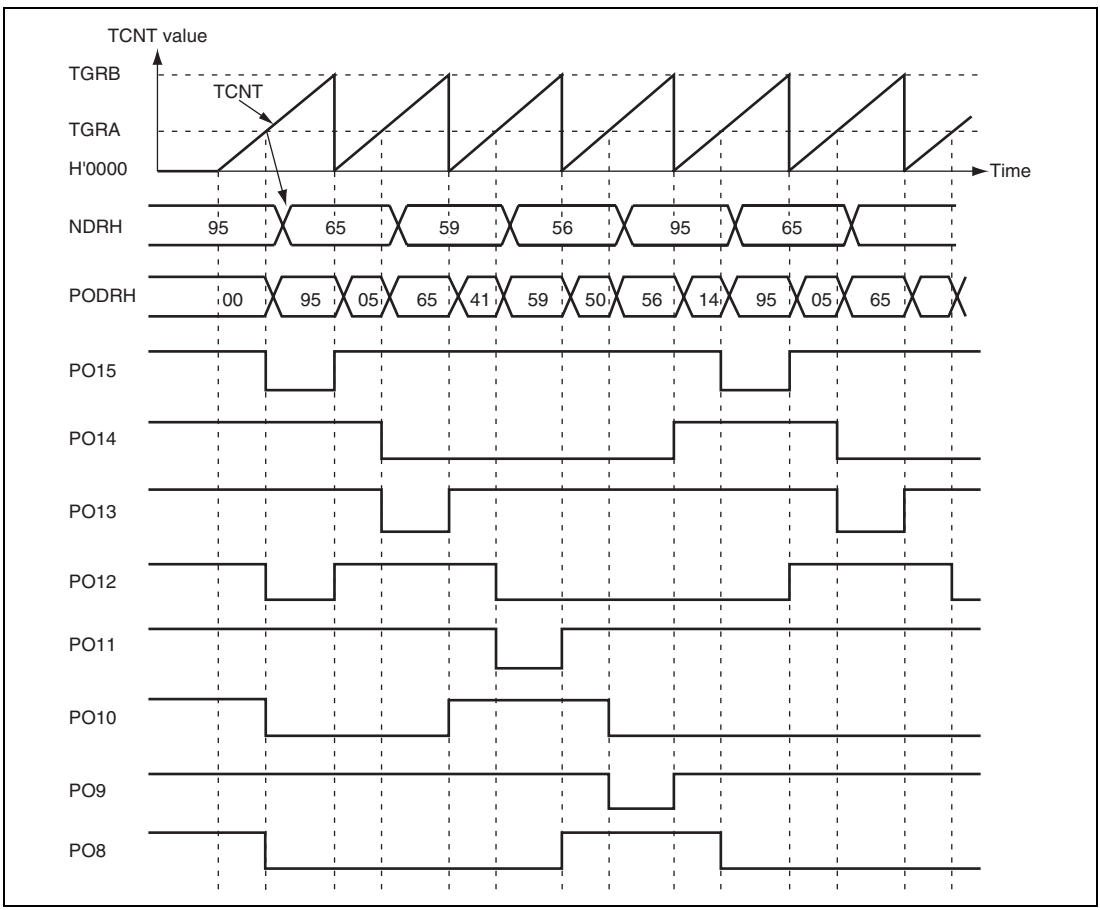


Figure 10.10 Inverted Pulse Output (Example)

10.4.8 Pulse Output Triggered by Input Capture

Pulse output can be triggered by TPU input capture as well as by compare match. If TGRA functions as an input capture register in the TPU channel selected by PCR, pulse output will be triggered by the input capture signal.

Figure 10.11 shows the timing of this output.

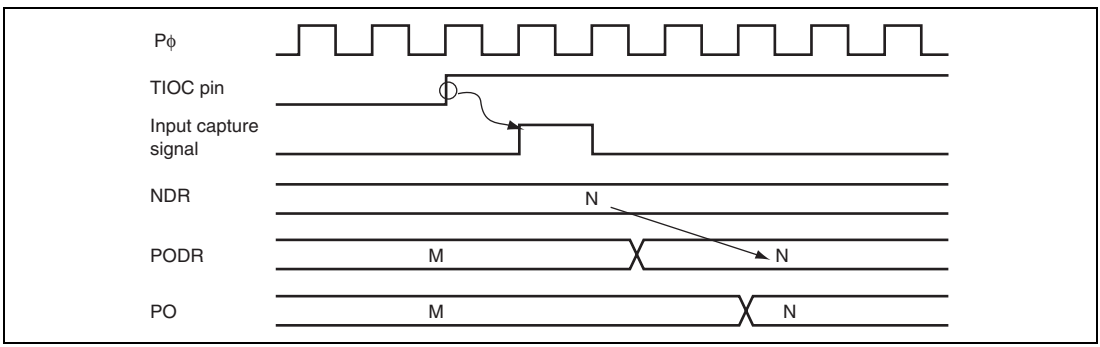


Figure 10.11 Pulse Output Triggered by Input Capture (Example)

10.5 Usage Notes

10.5.1 Module Stop Mode Setting

PPG operation can be disabled or enabled using the module stop control register. The initial value is for PPG operation to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 19, Power-Down Modes.

10.5.2 Operation of Pulse Output Pins

Pins PO0 to PO8 are also used for other peripheral functions such as the TPU. When output by another peripheral function is enabled, the corresponding pins cannot be used for pulse output. Note, however, that data transfer from NDR bits to PODR bits takes place, regardless of the usage of the pins.

Pin functions should be changed only under conditions in which the output trigger event will not occur.

Section 11 Watchdog Timer (WDT)

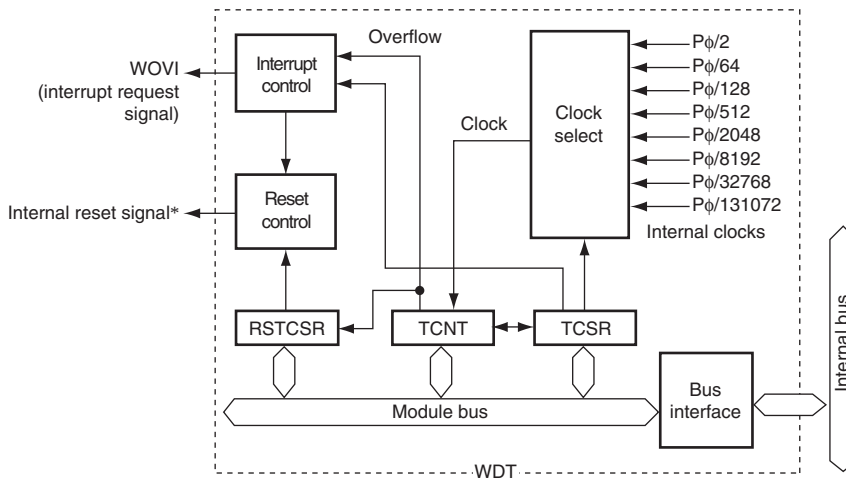
The watchdog timer (WDT) is an 8-bit timer that outputs an internal reset signal if a system crash prevents the CPU from writing to the timer counter, thus allowing it to overflow.

When this watchdog function is not needed, the WDT can be used as an interval timer. In interval timer operation, an interval timer interrupt is generated each time the counter overflows.

Figure 11.1 shows a block diagram of the WDT.

11.1 Features

- Selectable from eight counter input clocks
- Switchable between watchdog timer mode and interval timer mode
  - — In watchdog timer mode
    If the counter overflows, this LSI can be initialized internally.
  - — In interval timer mode
    If the counter overflows, the WDT generates an interval timer interrupt (WOVI).



[Legend]
TCSR: Timer control/status register
TCNT: Timer counter
RSTCSR: Reset control/status register

Note: * An internal reset signal can be generated by the RSTCSR setting.

Figure 11.1 Block Diagram of WDT

11.2 Register Descriptions

The WDT has the following three registers. To prevent accidental overwriting, TCSR, TCNT, and RSTCSR have to be written to in a method different from normal registers. For details, see section 11.5.1, Notes on Register Access.

- Timer counter (TCNT)
- Timer control/status register (TCSR)
- Reset control/status register (RSTCSR)

11.2.1 Timer Counter (TCNT)

TCNT is an 8-bit readable/writable up-counter. TCNT is initialized to H'00 when the TME bit in TCSR is cleared to 0.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

11.2.2 Timer Control/Status Register (TCSR)

TCSR selects the clock source to be input to TCNT, and the timer mode.

Bit	7	6	5	4	3	2	1	0
Bit Name	OVF	WT/$\overline{\text{IT}}$	TME	—	—	CKS2	CKS1	CKS0
Initial Value	0	0	0	1	1	0	0	0
R/W	R/(W)*	R/W	R/W	R	R	R/W	R/W	R/W

Note: * Only 0 can be written to this bit, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
7	OVF	0	R/(W)*	Overflow Flag Indicates that TCNT has overflowed in interval timer mode. Only 0 can be written to this bit, to clear the flag. [Setting condition] When TCNT overflows in interval timer mode (changes from H'FF to H'00) When internal reset request generation is selected in watchdog timer mode, OVF is cleared automatically by the internal reset. [Clearing condition] Cleared by reading TCSR when OVF = 1, then writing 0 to OVF (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
6	WT/$\overline{\text{IT}}$	0	R/W	Timer Mode Select Selects whether the WDT is used as a watchdog timer or interval timer. 0: Interval timer mode When TCNT overflows, an interval timer interrupt (WOVI) is requested. 1: Watchdog timer mode When TCNT overflows while RSTE = 1, this LSI is initialized initially.

Bit	Bit Name	Initial Value	R/W	Description
5	TME	0	R/W	Timer Enable When this bit is set to 1, TCNT starts counting. When this bit is cleared, TCNT stops counting and is initialized to H'00.
4, 3	—	All 1	R	Reserved These are read-only bits and cannot be modified.
2 1 0	CKS2 CKS1 CKS0	0 0 0	R/W R/W R/W	Clock Select 2 to 0 Select the clock source to be input to TCNT. The overflow cycle for Pϕ = 20 MHz is indicated in parentheses. 000: Clock Pϕ/2 (cycle: 25.6 μs) 001: Clock Pϕ/64 (cycle: 819.2 μs) 010: Clock Pϕ/128 (cycle: 1.6 ms) 011: Clock Pϕ/512 (cycle: 6.6 ms) 100: Clock Pϕ/2048 (cycle: 26.2 ms) 101: Clock Pϕ/8192 (cycle: 104.9 ms) 110: Clock Pϕ/32768 (cycle: 419.4 ms) 111: Clock Pϕ/131072 (cycle: 1.68 s)

Note: * Only 0 can be written to this bit, to clear the flag.

11.2.3 Reset Control/Status Register (RSTCSR)

RSTCSR controls the generation of the internal reset signal when TCNT overflows, and selects the type of internal reset signal. RSTCSR is initialized to H'1F by a reset signal from the $\overline{\text{RES}}$ pin, but not by the WDT internal reset signal caused by WDT overflows.

Bit	7	6	5	4	3	2	1	0
Bit Name	WOVF	RSTE	—	—	—	—	—	—
Initial Value	0	0	0	1	1	1	1	1
R/W	R/(W)*	R/W	R/W	R	R	R	R	R

Note: * Only 0 can be written to this bit, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
7	WOVF	0	R/(W)*	Watchdog Timer Overflow Flag This bit is set when TCNT overflows in watchdog timer mode. This bit cannot be set in interval timer mode, and only 0 can be written. [Setting condition] When TCNT overflows (changed from H'FF to H'00) in watchdog timer mode [Clearing condition] Reading RSTCSR when WOVF = 1, and then writing 0 to WOVF
6	RSTE	0	R/W	Reset Enable Specifies whether or not this LSI is internally reset if TCNT overflows during watchdog timer operation. 0: LSI is not reset even if TCNT overflows (Though this LSI is not reset, TCNT and TCSR in WDT are reset) 1: LSI is reset if TCNT overflows
5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
4 to 0	—	All 1	R	Reserved These are read-only bits and cannot be modified.

Note: * Only 0 can be written to this bit, to clear the flag.

11.3 Operation

11.3.1 Watchdog Timer Mode

To use the WDT in watchdog timer mode, set both the WT/$\overline{\text{IT}}$ and TME bits in TCSR to 1.

When TCNT overflows in watchdog timer mode, the WOVF bit in RSTCSR is set to 1.

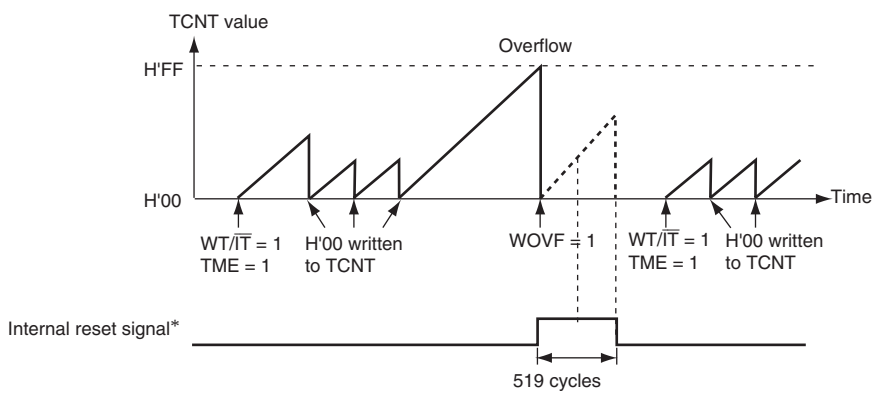
When the watchdog timer mode is selected and the RSTE bit in RSTCSR is set to 1, if TCNT overflows without being rewritten because of a system crash or other error, this LSI is initialized internally. This ensures that TCNT does not overflow while the system is operating normally. Software must prevent TCNT overflows by rewriting the TCNT value (normally H'00 is written) before overflow occurs.

If a reset caused by a signal input to the $\overline{\text{RES}}$ pin occurs at the same time as a reset caused by a WDT overflow (TCNT has overflowed), the $\overline{\text{RES}}$ pin reset has priority and the WOVF bit in RSTCSR is cleared to 0.

The internal reset signal is output for 519 cycles of Pϕ.

When RSTE = 1, a signal to initialize this LSI internally is generated. Since this signal initializes the system click control register (SCKCR), the multiplication ratio of P ϕ clock is also initialized.

When RSTE = 0, the signal is not generated, meaning that the SCKCR value and multiplication ratio of P ϕ clock remain unchanged.



Notes: * If TCNT overflows when the RSTE bit is set to 1, an internal reset signal is generated.

Figure 11.2 Operation in Watchdog Timer Mode

11.3.2 Interval Timer Mode

To use the WDT as an interval timer, set the WT/$\overline{\text{IT}}$ bit to 0 and the TME bit to 1 in TCSR.

When the WDT is used as an interval timer, an interval timer interrupt (WOVI) is generated each time the TCNT overflows. Therefore, an interrupt can be generated at intervals.

When the TCNT overflows in interval timer mode, an interval timer interrupt (WOVI) is requested at the same time the OVF bit in the TCSR is set to 1.

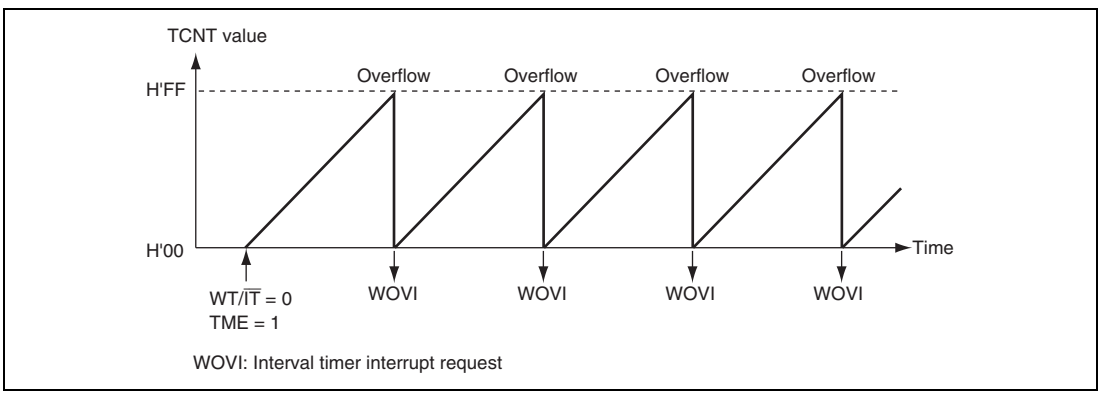


Figure 11.3 Operation in Interval Timer Mode

11.4 Interrupt Source

During interval timer mode operation, an overflow generates an interval timer interrupt (WOVI). The interval timer interrupt is requested whenever the OVF flag is set to 1 in TCSR. The OVF flag must be cleared to 0 in the interrupt handling routine.

Table 11.1 WDT Interrupt Source

Name	Interrupt Source	Interrupt Flag	DMAC Activation
WOVI	TCNT overflow	OVF	Impossible

11.5 Usage Notes

11.5.1 Notes on Register Access

The watchdog timer's TCNT, TCSR, and RSTCSR registers differ from other registers in being more difficult to write to. The procedures for writing to and reading these registers are given below.

(1) Writing to TCNT, TCSR, and RSTCSR

TCNT and TCSR must be written to by a word transfer instruction. They cannot be written to by a byte transfer instruction.

For writing, TCNT and TCSR are assigned to the same address. Accordingly, perform data transfer as shown in figure 11.4. The transfer instruction writes the lower byte data to TCNT or TCSR.

To write to RSTCSR, execute a word transfer instruction for address H'FFA6. A byte transfer instruction cannot be used to write to RSTCSR.

The method of writing 0 to the WOVF bit in RSTCSR differs from that of writing to the RSTE bit in RSTCSR. Perform data transfer as shown in figure 11.4.

At data transfer, the transfer instruction clears the WOVF bit to 0, but has no effect on the RSTE bit. To write to the RSTE bit, perform data transfer as shown in figure 11.4. In this case, the transfer instruction writes the value in bit 6 of the lower byte to the RSTE bit, but has no effect on the WOVF bit.

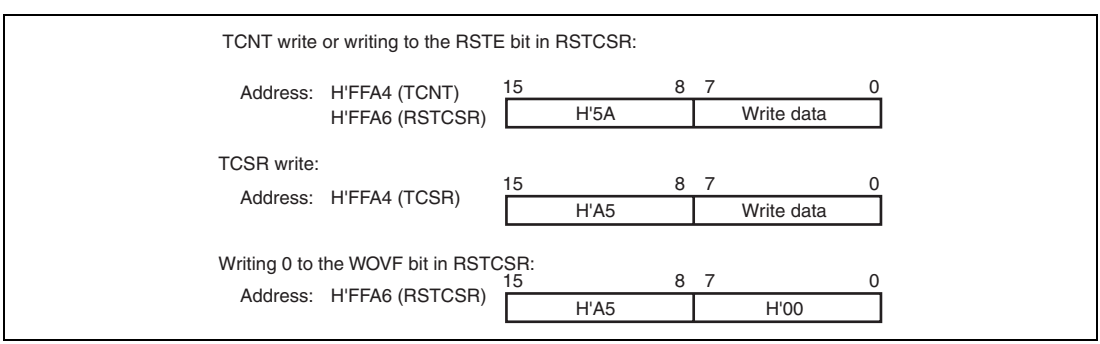


Figure 11.4 Writing to TCNT, TCSR, and RSTCSR

(2) Reading from TCNT, TCSR, and RSTCSR

These registers can be read from in the same way as other registers. For reading, TCSR is assigned to address H'FFA4, TCNT to address H'FFA5, and RSTCSR to address H'FFA7.

11.5.2 Conflict between Timer Counter (TCNT) Write and Increment

If a TCNT clock pulse is generated during the T2 state of a TCNT write cycle, the write takes priority and the timer counter is not incremented. Figure 11.5 shows this operation.

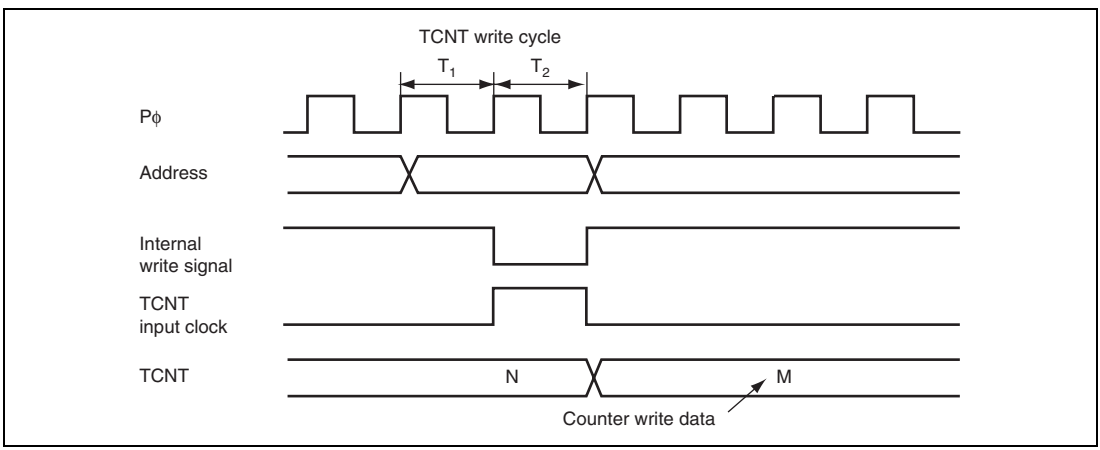


Figure 11.5 Conflict between TCNT Write and Increment

11.5.3 Changing Values of Bits CKS2 to CKS0

If bits CKS2 to CKS0 in TCSR are written to while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before the values of bits CKS2 to CKS0 are changed.

11.5.4 Switching between Watchdog Timer Mode and Interval Timer Mode

If the timer mode is switched from watchdog timer mode to interval timer mode while the WDT is operating, errors could occur in the incrementation. The watchdog timer must be stopped (by clearing the TME bit to 0) before switching the timer mode.

11.5.5 Transition to Watchdog Timer Mode or Software Standby Mode

When the WDT operates in watchdog timer mode, a transition to software standby mode is not made even when the SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1. Instead, a transition to sleep mode is made.

To transit to software standby mode, the SLEEP instruction must be executed after halting the WDT (clearing the TME bit to 0).

When the WDT operates in interval timer mode, a transition to software standby mode is made through execution of the SLEEP instruction when the SSBY bit in SBYCR is set to 1.

Section 12 Serial Communication Interface (SCI)

This LSI has two independent serial communication interface (SCI) channels. The SCI can handle both asynchronous and clocked synchronous serial communication. Asynchronous serial data communication can be carried out with standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or Asynchronous Communication Interface Adapter (ACIA). A function is also provided for serial communication between processors (multiprocessor communication function). The SCI also supports the smart card (IC card) interface conforming to ISO/IEC 7816-3 (Identification Card) as an extended asynchronous communication mode. Figure 12.1 shows a block diagram of the SCI.

12.1 Features

- Choice of asynchronous or clocked synchronous serial communication mode
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously. Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected

  The external clock can be selected as a transfer clock source (except for the smart card interface).
- Choice of LSB-first or MSB-first transfer (except in the case of asynchronous mode 7-bit data)
- Four interrupt sources

  The interrupt sources are transmit-end, transmit-data-empty, receive-data-full, and receive error. The transmit-data-empty and receive-data-full interrupt sources can activate the DMAC.
- Module stop mode can be set

Asynchronous Mode:

- Data length: 7 or 8 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RxD pin level directly in case of a framing error

Clocked Synchronous Mode:

- Data length: 8 bits
- Receive error detection: Overrun errors

Smart Card Interface:

- An error signal can be automatically transmitted on detection of a parity error during reception
- Data can be automatically re-transmitted on receiving an error signal during transmission
- Both direct convention and inverse convention are supported

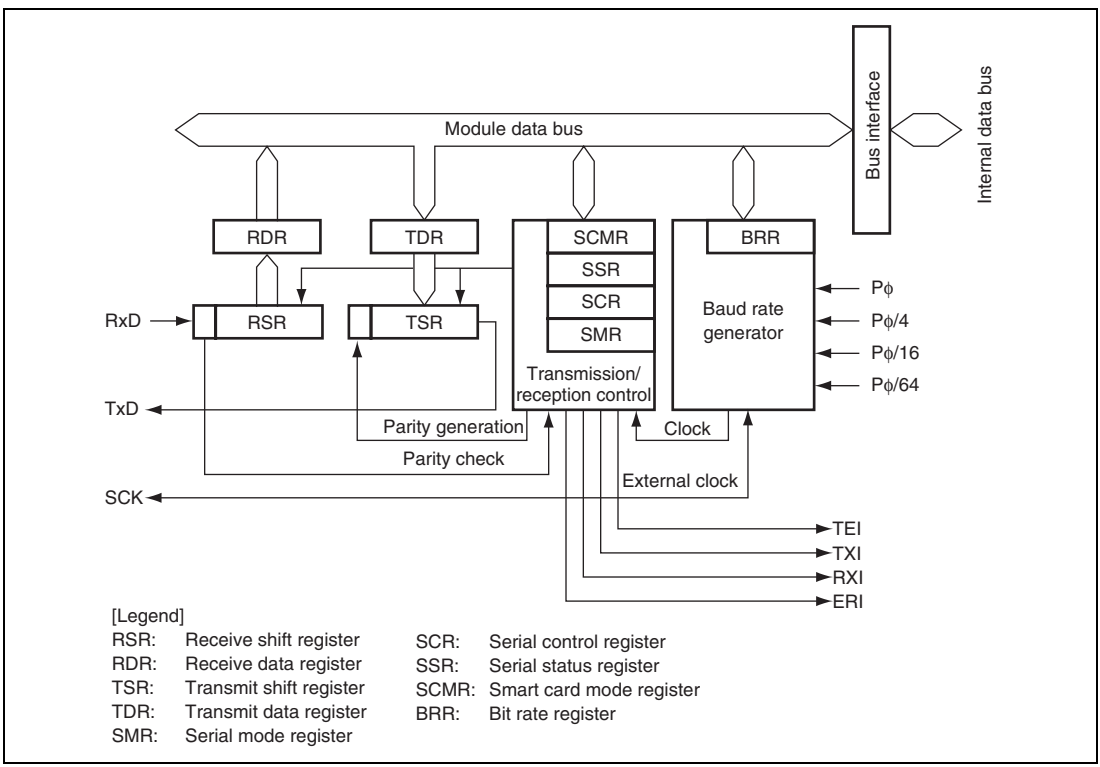


Figure 12.1 Block Diagram of SCI

12.2 Input/Output Pins

Table 12.1 lists the pin configuration of the SCI.

Table 12.1 Pin Configuration

Channel	Pin Name*	I/O	Function
3	SCK3	I/O	Channel 3 clock input/output
	RxD3	Input	Channel 3 receive data input
	TxD3	Output	Channel 3 transmit data output
4	SCK4	I/O	Channel 4 clock input/output
	RxD4	Input	Channel 4 receive data input
	TxD4	Output	Channel 4 transmit data output

Note: * Pin names SCK, RxD, and TxD are used in the text for all channels, omitting the channel designation.

12.3 Register Descriptions

The SCI has the following registers. Some bits in the serial mode register (SMR), serial status register (SSR), and serial control register (SCR) have different functions in different modes: Normal serial communication interface mode and smart card interface mode. The bits, therefore, are described separately for each mode in the corresponding register sections.

Channel 3:

Receive shift register_3 (RSR_3)
Transmit shift register_3 (TSR_3)
Receive data register_3 (RDR_3)
Transmit data register_3 (TDR_3)
Serial mode register_3 (SMR_3)
Serial control register_3 (SCR_3)
Serial status register_3 (SSR_3)
Smart card mode register_3 (SCMR_3)
Bit rate register_3 (BRR_3)

Channel 4:

Receive shift register_4 (RSR_4)
Transmit shift register_4 (TSR_4)
Receive data register_4 (RDR_4)
Transmit data register_4 (TDR_4)
Serial mode register_4 (SMR_4)
Serial control register_4 (SCR_4)
Serial status register_4 (SSR_4)
Smart card mode register_4 (SCMR_4)
Bit rate register_4 (BRR_4)

12.3.1 Receive Shift Register (RSR)

RSR is a shift register which is used to receive serial data input from the RxD pin and converts it into parallel data. When one frame of data has been received, it is transferred to RDR automatically. RSR cannot be directly accessed by the CPU.

12.3.2 Receive Data Register (RDR)

RDR is an 8-bit register that stores receive data. When the SCI has received one frame of serial data, it transfers the received serial data from RSR to RDR where it is stored. This allows RSR to receive the next data. Since RSR and RDR function as a double buffer in this way, continuous receive operations can be performed. After confirming that the RDRF bit in SSR is set to 1, read RDR only once. RDR cannot be written to by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

12.3.3 Transmit Data Register (TDR)

TDR is an 8-bit register that stores transmit data. When the SCI detects that TSR is empty, it transfers the transmit data written in TDR to TSR and starts transmission. The double-buffered structures of TDR and TSR enables continuous serial transmission. If the next transmit data has already been written to TDR when one frame of data is transmitted, the SCI transfers the written data to TSR to continue transmission. Although TDR can be read from or written to by the CPU at all times, to achieve reliable serial transmission, write transmit data to TDR for only once after confirming that the TDRE bit in SSR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

12.3.4 Transmit Shift Register (TSR)

TSR is a shift register that transmits serial data. To perform serial data transmission, the SCI first automatically transfers transmit data from TDR to TSR, then sends the data to the TxD pin. TSR cannot be directly accessed by the CPU.

12.3.5 Serial Mode Register (SMR)

SMR is used to set the SCI's serial transfer format and select the baud rate generator clock source. Some bits in SMR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	C/$\overline{A}$	CHR	PE	O/$\overline{E}$	STOP	MP	CKS1	CKS0
Initial Value	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	GM	BLK	PE	O/$\overline{E}$	BCP1	BCP0	CKS1	CKS0
Initial Value	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

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	C/$\overline{A}$	0	R/W	Communication Mode 0: Asynchronous mode 1: Clocked synchronous mode
6	CHR	0	R/W	Character Length (valid only in asynchronous mode) 0: Selects 8 bits as the data length. 1: Selects 7 bits as the data length. LSB-first is fixed and the MSB (bit 7) in TDR is not transmitted in transmission. In clocked synchronous mode, a fixed data length of 8 bits is used.

Bit	Bit Name	Initial Value	R/W	Description
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. For a multiprocessor format, parity bit addition and checking are not performed regardless of the PE bit setting.
4	O/$\overline{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity. 1: Selects odd parity.
3	STOP	0	R/W	Stop Bit Length (valid only in asynchronous mode) Selects the stop bit length in transmission. 0: 1 stop bit 1: 2 stop bits In reception, only the first stop bit is checked. If the second stop bit is 0, it is treated as the start bit of the next transmit frame.
2	MP	0	R/W	Multiprocessor Mode (valid only in asynchronous mode) When this bit is set to 1, the multiprocessor function is enabled. The PE bit and O/$\overline{E}$ bit settings are invalid in multiprocessor mode.
1	CKS1	0	R/W	Clock Select 1, 0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: Pϕ clock (n = 0) 01: Pϕ/4 clock (n = 1) 10: Pϕ/16 clock (n = 2) 11: Pϕ/64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 12.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 12.3.9, Bit Rate Register (BRR)).

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	GM	0	R/W	GSM Mode Setting this bit to 1 allows GSM mode operation. In GSM mode, the TEND set timing is put forward to 11.0 etu from the start and the clock output control function is appended. For details, see sections 12.7.6, Data Transmission (Except in Block Transfer Mode) and 12.7.8, Clock Output Control.
6	BLK	0	R/W	Setting this bit to 1 allows block transfer mode operation. For details, see section 12.7.3, Block Transfer Mode.
5	PE	0	R/W	Parity Enable (valid only in asynchronous mode) When this bit is set to 1, the parity bit is added to transmit data before transmission, and the parity bit is checked in reception. Set this bit to 1 in smart card interface mode.
4	O/$\overline{E}$	0	R/W	Parity Mode (valid only when the PE bit is 1 in asynchronous mode) 0: Selects even parity 1: Selects odd parity For details on the usage of this bit in smart card interface mode, see section 12.7.2, Data Format (Except in Block Transfer Mode).
3	BCP1	0	R/W	Basic Clock Pulse 1,0
2	BCP0	0	R/W	These bits select the number of basic clock cycles in a 1-bit data transfer time in smart card interface mode. 00: 32 clock cycles (S = 32) 01: 64 clock cycles (S = 64) 10: 372 clock cycles (S = 372) 11: 256 clock cycles (S = 256) For details, see section 12.7.4, Receive Data Sampling Timing and Reception Margin. S is described in section 12.3.9, Bit Rate Register (BRR).

Bit	Bit Name	Initial Value	R/W	Description
1	CKS1	0	R/W	Clock Select 1,0
0	CKS0	0	R/W	These bits select the clock source for the baud rate generator. 00: Pϕ clock (n = 0) 01: Pϕ/4 clock (n = 1) 10: Pϕ/16 clock (n = 2) 11: Pϕ/64 clock (n = 3) For the relation between the settings of these bits and the baud rate, see section 12.3.9, Bit Rate Register (BRR). n is the decimal display of the value of n in BRR (see section 12.3.9, Bit Rate Register (BRR)).

Note: etu (Elementary Time Unit): 1-bit transfer time

12.3.6 Serial Control Register (SCR)

SCR is a register that enables/disables the following SCI transfer operations and interrupt requests, and selects the transfer clock source. For details on interrupt requests, see section 12.8, Interrupt Sources. Some bits in SCR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial Value	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

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0
Initial Value	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

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled. A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 Sin order to designate the transmission format. If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format. Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.

Bit	Bit Name	Initial Value	R/W	Description
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode) When this bit is set to 1, receive data in which the multiprocessor bit is 0 is skipped, and setting of the RDRF, FER, and ORER status flags in SSR is disabled. On receiving data in which the multiprocessor bit is 1, this bit is automatically cleared and normal reception is resumed. For details, see section 12.5, Multiprocessor Communication Function. When receive data including MPB = 0 in SSR is being received, transfer of the received data from RSR to RDR, detection of reception errors, and the settings of RDRF, FER, and ORER flags in SSR are not performed. When receive data including MPB = 1 is received, the MPB bit in SSR is set to 1, the MPIE bit is automatically cleared to 0, and RXI and ERI interrupt requests (in the case where the TIE and RIE bits in SCR are set to 1) and setting of the FER and ORER flags are enabled.
2	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, a TEI interrupt request is enabled. A TEI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0 in order to clear the TEND flag to 0, or by clearing the TEIE bit to 0.
1	CKE1	0	R/W	Clock Enable 1, 0
0	CKE0	0	R/W	These bits select the clock source and SCK pin function. • Asynchronous mode 00: On-chip baud rate generator (SCK pin functions as I/O port.) 01: On-chip baud rate generator (Outputs a clock with the same frequency as the bit rate from the SCK pin.) 1X: External clock (Inputs a clock with a frequency 16 times the bit rate from the SCK pin.) • Clocked synchronous mode 0X: Internal clock (SCK pin functions as clock output.) 1X: External clock (SCK pin functions as clock input.)

Note: X: Don't care

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1,a TXI interrupt request is enabled. A TXI interrupt request can be cancelled by reading 1 from the TDRE flag and then clearing the flag to 0, or by clearing the TIE bit to 0.
6	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, RXI and ERI interrupt requests are enabled. RXI and ERI interrupt requests can be cancelled by reading 1 from the RDRF, FER, PER, or ORER flag and then clearing the flag to 0, or by clearing the RIE bit to 0.
5	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled. Under this condition, serial transmission is started by writing transmit data to TDR, and clearing the TDRE flag in SSR to 0. Note that SMR should be set prior to setting the TE bit to 1 in order to designate the transmission format. If transmission is halted by clearing this bit to 0, the TDRE flag in SSR is fixed 1.
4	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled. Under this condition, serial reception is started by detecting the start bit in asynchronous mode or the synchronous clock input in clocked synchronous mode. Note that SMR should be set prior to setting the RE bit to 1 in order to designate the reception format. Even if reception is halted by clearing this bit to 0, the RDRF, FER, PER, and ORER flags are not affected and the previous value is retained.

Bit	Bit Name	Initial Value	R/W	Description
3	MPIE	0	R/W	Multiprocessor Interrupt Enable (valid only when the MP bit in SMR is 1 in asynchronous mode) Write 0 to this bit in smart card interface mode.
2	TEIE	0	R/W	Transmit End Interrupt Enable Write 0 to this bit in smart card interface mode.
1	CKE1	0	R/W	Clock Enable 1, 0
0	CKE0	0	R/W	These bits control the clock output from the SCK pin. In GSM mode, clock output can be dynamically switched. For details, see section 12.7.8, Clock Output Control. • When GM in SMR = 0 00: Output disabled (SCK pin functions as I/O port.) 01: Clock output 1X: Reserved • When GM in SMR = 1 00: Output fixed low 01: Clock output 10: Output fixed high 11: Clock output

12.3.7 Serial Status Register (SSR)

SSR is a register containing status flags of the SCI and multiprocessor bits for transfer. TDRE, RDRF, ORER, PER, and FER can only be cleared. Some bits in SSR have different functions in normal mode and smart card interface mode.

- When SMIF in SCMR = 0

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	FER	PER	TEND	MPB	MPBT
Initial Value	1	0	0	0	0	1	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: * Only 0 can be written, to clear the flag.

- When SMIF in SCMR = 1

Bit	7	6	5	4	3	2	1	0
Bit Name	TDRE	RDRF	ORER	ERS	PER	TEND	MPB	MPBT
Initial Value	1	0	0	0	0	1	0	0
R/W	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R/(W)*	R	R	R/W

Note: * Only 0 can be written, to clear the flag.

Bit Functions in Normal Serial Communication Interface Mode (When SMIF in SCMR = 0):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When a TXI interrupt request is issued allowing DMAC to write data to TDR
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates whether receive data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When an RXI interrupt request is issued allowing DMAC to read data from RDR The RDRF flag is not affected and retains its previous value when the RE bit in SCR is cleared to 0. Note that when the next serial reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error Indicates that an overrun error has occurred during reception and the reception ends abnormally. [Setting condition] • When the next serial reception is completed while RDRF = 1 In RDR, receive data prior to an overrun error occurrence is retained, but data received after the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to ORER after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.
4	FER	0	R/(W)*	Framing Error Indicates that a framing error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When the stop bit is 0 In 2-stop-bit mode, only the first stop bit is checked whether it is 1 but the second stop bit is not checked. Note that receive data when the framing error occurs is transferred to RDR, however, the RDRF flag is not set. In addition, when the FER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to FER after reading FER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the FER flag is not affected and retains its previous value.

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to PER after reading PER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the PER bit is not affected and retains its previous value.
2	TEND	1	R	Transmit End [Setting conditions] • When the TE bit in SCR is 0 • When TDRE = 1 at transmission of the last bit of a transmit character [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When a TXI interrupt request is issued allowing DMAC to write data to TDR
1	MPB	0	R	Multiprocessor Bit Stores the multiprocessor bit value in the receive frame. When the RE bit in SCR is cleared to 0 its previous state is retained.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Sets the multiprocessor bit value to be added to the transmit frame.

Note: * Only 0 can be written, to clear the flag.

Bit Functions in Smart Card Interface Mode (When SMIF in SCMR = 1):

Bit	Bit Name	Initial Value	R/W	Description
7	TDRE	1	R/(W)*	Transmit Data Register Empty Indicates whether TDR contains transmit data. [Setting conditions] • When the TE bit in SCR is 0 • When data is transferred from TDR to TSR [Clearing conditions] • When 0 is written to TDRE after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When a TXI interrupt request is issued allowing DMAC to write data to TDR
6	RDRF	0	R/(W)*	Receive Data Register Full Indicates whether receive data is stored in RDR. [Setting condition] • When serial reception ends normally and receive data is transferred from RSR to RDR [Clearing conditions] • When 0 is written to RDRF after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When an RXI interrupt request is issued allowing DMAC to read data from RDR The RDRF flag is not affected and retains its previous value even when the RE bit in SCR is cleared to 0. Note that when the next reception is completed while the RDRF flag is being set to 1, an overrun error occurs and the received data is lost.

Bit	Bit Name	Initial Value	R/W	Description
5	ORER	0	R/(W)*	Overrun Error Indicates that an overrun error has occurred during reception and the reception ends abnormally. [Setting condition] • When the next serial reception is completed while RDRF = 1 In RDR, the receive data prior to an overrun error occurrence is retained, but data received following the overrun error occurrence is lost. When the ORER flag is set to 1, subsequent serial reception cannot be performed. Note that, in clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to ORER after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the ORER flag is not affected and retains its previous value.
4	ERS	0	R/(W)*	Error Signal Status [Setting condition] • When a low error signal is sampled [Clearing condition] • When 0 is written to ERS after reading ERS = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

Bit	Bit Name	Initial Value	R/W	Description
3	PER	0	R/(W)*	Parity Error Indicates that a parity error has occurred during reception in asynchronous mode and the reception ends abnormally. [Setting condition] • When a parity error is detected during reception Receive data when the parity error occurs is transferred to RDR, however, the RDRF flag is not set. Note that when the PER flag is being set to 1, the subsequent serial reception cannot be performed. In clocked synchronous mode, serial transmission also cannot continue. [Clearing condition] • When 0 is written to PER after reading PER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) Even when the RE bit in SCR is cleared, the PER flag is not affected and retains its previous value.

Bit	Bit Name	Initial Value	R/W	Description
2	TEND	1	R	Transmit End This bit is set to 1 when no error signal is sent from the receiving side and the next transmit data is ready to be transferred to TDR. [Setting conditions] • When both the TE and ERS bits in SCR are 0 • When ERS = 0 and TDRE = 1 after a specified time passed after completion of 1-byte data transfer. The set timing depends on the register setting as follows: When GM = 0 and BLK = 0, 2.5 etu after transmission start When GM = 0 and BLK = 1, 1.5 etu after transmission start When GM = 1 and BLK = 0, 1.0 etu after transmission start When GM = 1 and BLK = 1, 1.0 etu after transmission start [Clearing conditions] • When 0 is written to TEND after reading TEND = 1 • When a TXI interrupt request is issued allowing DMAC to write the next data to TDR
1	MPB	0	R	Multiprocessor Bit Not used in smart card interface mode.
0	MPBT	0	R/W	Multiprocessor Bit Transfer Write 0 to this bit in smart card interface mode.

Note: * Only 0 can be written, to clear the flag.

12.3.8 Smart Card Mode Register (SCMR)

SCMR selects smart card interface mode and its format.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	SDIR	SINV	—	SMIF
Initial Value	1	1	1	1	0	0	1	0
R/W	R	R	R	R	R/W	R/W	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	All 1	R	Reserved These are read-only bits and cannot be modified.
3	SDIR	0	R/W	Smart Card Data Transfer Direction Selects the serial/parallel conversion format. 0: Transfer with LSB-first 1: Transfer with MSB-first This bit is valid only when the 8-bit data format is used for transmission/reception; when the 7-bit data format is used, data is always transmitted/received with LSB-first.
2	SINV	0	R/W	Smart Card Data Invert Inverts the transmit/receive data logic level. This bit does not affect the logic level of the parity bit. To invert the parity bit, invert the O/$\overline{E}$ bit in SMR. 0: TDR contents are transmitted as they are. Receive data is stored as it is in RDR. 1: TDR contents are inverted before being transmitted. Receive data is stored in inverted form in RDR.
1	—	1	R	Reserved This is a read-only bit and cannot be modified.
0	SMIF	0	R/W	Smart Card Interface Mode Select When this bit is set to 1, smart card interface mode is selected. 0: Normal asynchronous or clocked synchronous mode 1: Smart card interface mode

12.3.9 Bit Rate Register (BRR)

BRR is an 8-bit register that adjusts the bit rate. As the SCI performs baud rate generator control independently for each channel, different bit rates can be set for each channel. Table 12.2 shows the relationships between the N setting in BRR and bit rate B for normal asynchronous mode and clocked synchronous mode, and smart card interface mode. The initial value of BRR is H'FF, and it can be read from or written to by the CPU at all times.

Table 12.2 Relationships between N Setting in BRR and Bit Rate B

Mode	Bit Rate	Error
Asynchronous mode	$N = \frac{P\phi \times 10^6}{64 \times 2^{2n-1} \times B} - 1$	$\text{Error (\%)} = \{\frac{P\phi \times 10^6}{B \times 64 \times 2^{2n-1} \times (N+1)} - 1\} \times 100$
Clocked synchronous mode	$N = \frac{P\phi \times 10^6}{8 \times 2^{2n-1} \times B} - 1$	
Smart card interface mode	$N = \frac{P\phi \times 10^6}{S \times 2^{2n+1} \times B} - 1$	$\text{Error (\%)} = \{\frac{P\phi \times 10^6}{B \times S \times 2^{2n+1} \times (N+1)} - 1\} \times 100$

[Legend]

B: Bit rate (bit/s)

N: BRR setting for baud rate generator ($0 \leq N \leq 255$)

Pϕ: Operating frequency (MHz)

n and S: Determined by the SMR settings shown in the following table.

SMR Setting			SMR Setting		
CKS1	**CKS0**	**n**	**BCP1**	**BCP0**	**S**
0	0	0	0	0	32
0	1	1	0	1	64
1	0	2	1	0	372
1	1	3	1	1	256

Table 12.3 shows sample N settings in BRR in normal asynchronous mode. Table 12.4 shows the maximum bit rate settable for each operating frequency. Tables 12.6 and 12.8 show sample N settings in BRR in clocked synchronous mode and smart card interface mode, respectively. In smart card interface mode, the number of basic clock cycles S in a 1-bit data transfer time can be selected. For details, see section 12.7.4, Receive Data Sampling Timing and Reception Margin. Tables 12.5 and 12.7 show the maximum bit rates with external clock input.

Table 12.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (1)

	Operating Frequency Pϕ (MHz)											
	8			9.8304			10			12		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	141	0.03	2	174	–0.26	2	177	–0.25	2	212	0.03
150	2	103	0.16	2	127	0.00	2	129	0.16	2	155	0.16
300	1	207	0.16	1	255	0.00	2	64	0.16	2	77	0.16
600	1	103	0.16	1	127	0.00	1	129	0.16	1	155	0.16
1200	0	207	0.16	0	255	0.00	1	64	0.16	1	77	0.16
2400	0	103	0.16	0	127	0.00	0	129	0.16	0	155	0.16
4800	0	51	0.16	0	63	0.00	0	64	0.16	0	77	0.16
9600	0	25	0.16	0	31	0.00	0	32	–1.36	0	38	0.16
19200	0	12	0.16	0	15	0.00	0	15	1.73	0	19	–2.34
31250	0	7	0.00	0	9	–1.70	0	9	0.00	0	11	0.00
38400	—	—	—	0	7	0.00	0	7	1.73	0	9	–2.34

	Operating Frequency Pϕ (MHz)											
	12.288			14			14.7456			16		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	2	217	0.08	2	248	–0.17	3	64	0.70	3	70	0.03
150	2	159	0.00	2	181	0.16	2	191	0.00	2	207	0.16
300	2	79	0.00	2	90	0.16	2	95	0.00	2	103	0.16
600	1	159	0.00	1	181	0.16	1	191	0.00	1	207	0.16
1200	1	79	0.00	1	90	0.16	1	95	0.00	1	103	0.16
2400	0	159	0.00	0	181	0.16	0	191	0.00	0	207	0.16
4800	0	79	0.00	0	90	0.16	0	95	0.00	0	103	0.16
9600	0	39	0.00	0	45	–0.93	0	47	0.00	0	51	0.16
19200	0	19	0.00	0	22	–0.93	0	23	0.00	0	25	0.16
31250	0	11	2.40	0	13	0.00	0	14	–1.70	0	15	0.00
38400	0	9	0.00	—	—	—	0	11	0.00	0	12	0.16

Table 12.3 Examples of BRR Settings for Various Bit Rates (Asynchronous Mode) (2)

	Operating Frequency Pϕ (MHz)											
	17.2032			18			19.6608			20		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	75	0.48	3	79	–0.12	3	86	0.31	3	88	–0.25
150	2	223	0.00	2	233	0.16	2	255	0.00	3	64	0.16
300	2	111	0.00	2	116	0.16	2	127	0.00	2	129	0.16
600	1	223	0.00	1	233	0.16	1	255	0.00	2	64	0.16
1200	1	111	0.00	1	116	0.16	1	127	0.00	1	129	0.16
2400	0	223	0.00	0	233	0.16	0	255	0.00	1	64	0.16
4800	0	111	0.00	0	116	0.16	0	127	0.00	0	129	0.16
9600	0	55	0.00	0	58	–0.69	0	63	0.00	0	64	0.16
19200	0	27	0.00	0	28	1.02	0	31	0.00	0	32	–1.36
31250	0	16	1.20	0	17	0.00	0	19	–1.70	0	19	0.00
38400	0	13	0.00	0	14	–2.34	0	15	0.00	0	15	1.73

	Operating Frequency Pϕ (MHz)											
	25			30			33			35		
Bit Rate (bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
110	3	110	–0.02	3	132	0.13	3	145	0.33	3	154	0.23
150	3	80	–0.47	3	97	–0.35	3	106	0.39	3	113	–0.06
300	2	162	0.15	2	194	0.16	2	214	–0.07	2	227	0.00
600	2	80	–0.47	2	97	–0.35	2	106	0.39	2	113	0.00
1200	1	162	0.15	1	194	0.16	1	214	–0.07	1	227	0.00
2400	1	80	–0.47	1	97	–0.35	1	106	0.39	1	113	0.00
4800	0	162	0.15	0	194	0.16	0	214	–0.07	0	227	0.00
9600	0	80	–0.47	0	97	–0.35	0	106	0.39	0	113	0.00
19200	0	40	–0.76	0	48	–0.35	0	53	–0.54	0	56	0.00
31250	0	24	0.00	0	29	0	0	32	0	0	34	0.00
38400	0	19	1.73	0	23	1.73	0	26	–0.54	0	28	–1.78

Table 12.4 Maximum Bit Rate for Each Operating Frequency (Asynchronous Mode)

Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
8	250000	0	0	17.2032	537600	0	0
9.8304	307200	0	0	18	562500	0	0
10	312500	0	0	19.6608	614400	0	0
12	375000	0	0	20	625000	0	0
12.288	384000	0	0	25	781250	0	0
14	437500	0	0	30	937500	0	0
14.7456	460800	0	0	33	1031250	0	0
16	500000	0	0	35	1093750	0	0

Table 12.5 Maximum Bit Rate with External Clock Input (Asynchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	2.0000	125000	17.2032	4.3008	268800
9.8304	2.4576	153600	18	4.5000	281250
10	2.5000	156250	19.6608	4.9152	307200
12	3.0000	187500	20	5.0000	312500
12.288	3.0720	192000	25	6.2500	390625
14	3.5000	218750	30	7.5000	468750
14.7456	3.6864	230400	33	8.2500	515625
16	4.0000	250000	35	8.7500	546875

Table 12.6 BRR Settings for Various Bit Rates (Clocked Synchronous Mode)

Bit Rate (bit/s)	Operating Frequency Pϕ (MHz)							
	8		10		16		20	
	n	N	n	N	n	N	n	N
110								
250	3	124	—	—	3	249		
500	2	249	—	—	3	124	—	—
1k	2	124	—	—	2	249	—	—
2.5k	1	199	1	249	2	99	2	124
5k	1	99	1	124	1	199	1	249
10k	0	199	0	249	1	99	1	124
25k	0	79	0	99	0	159	0	199
50k	0	39	0	49	0	79	0	99
100k	0	19	0	24	0	39	0	49
250k	0	7	0	9	0	15	0	19
500k	0	3	0	4	0	7	0	9
1M	0	1			0	3	0	4
2.5M			0	0*			0	1
5M							0	0*

Bit Rate (bit/s)	Operating Frequency Pϕ (MHz)							
	25		30		33		35	
	n	N	n	N	n	N	n	N
110								
250								
500			3	233				
1k	3	97	3	116	3	128	3	136
2.5k	2	155	2	187	2	205	2	218
5k	2	77	2	93	2	102	2	108
10k	1	155	1	187	1	205	1	218
25k	0	249	1	74	1	82	1	87
50k	0	124	0	149	0	164	0	174
100k	0	62	0	74	0	82	0	87
250k	0	24	0	29	0	32	0	34
500k	—	—	0	14	—	—	—	—
1M	—	—	—	—	—	—	—	—
2.5M	—	—	0	2	—	—	—	—
5M	—	—	—	—	—	—	—	—

[Legend]

Space : Setting prohibited.

— : Can be set, but there will be error.

* : Continuous transmission or reception is not possible.

Table 12.7 Maximum Bit Rate with External Clock Input (Clocked Synchronous Mode)

Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)	Pϕ (MHz)	External Input Clock (MHz)	Maximum Bit Rate (bit/s)
8	1.3333	1333333.3	20	3.3333	3333333.3
10	1.6667	1666666.7	25	4.1667	4166666.7
12	2.0000	2000000.0	30	5.0000	5000000.0
14	2.3333	2333333.3	33	5.5000	5500000.0
16	2.6667	2666666.7	35	5.8336	5833625.0
18	3.0000	3000000.0			

Table 12.8 BRR Settings for Various Bit Rates (Smart Card Interface Mode, n = 0, S = 372)

Bit Rate	Operating Frequency Pϕ (MHz)											
	7.1424			10.00			10.7136			13.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	0	0.00	0	1	30	0	1	25	0	1	8.99

Bit Rate	Operating Frequency Pϕ (MHz)											
	14.2848			16.00			18.00			20.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	1	0.00	0	1	12.01	0	2	15.99	0	2	6.60

Bit Rate	Operating Frequency Pϕ (MHz)											
	25.00			30.00			33.00			35.00		
(bit/s)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)	n	N	Error (%)
9600	0	3	12.49	0	3	5.01	0	4	7.59	0	4	1.99

Table 12.9 Maximum Bit Rate for Each Operating Frequency (Smart Card Interface Mode, S = 372)

Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N	Pϕ (MHz)	Maximum Bit Rate (bit/s)	n	N
7.1424	9600	0	0	18.00	24194	0	0
10.00	13441	0	0	20.00	26882	0	0
10.7136	14400	0	0	25.00	33602	0	0
13.00	17473	0	0	30.00	40323	0	0
14.2848	19200	0	0	33.00	44355	0	0
16.00	21505	0	0	35.00	47043	0	0

12.4 Operation in Asynchronous Mode

Figure 12.2 shows the general format for asynchronous serial communication. One frame consists of a start bit (low level), followed by transmit/receive data, a parity bit, and finally stop bits (high level). In asynchronous serial communication, the communication line is usually held in the mark state (high level). The SCI monitors the communication line, and when it goes to the space state (low level), recognizes a start bit and starts serial communication. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication. Both the transmitter and the receiver also have a double-buffered structure, so that data can be read or written during transmission or reception, enabling continuous data transmission and reception.

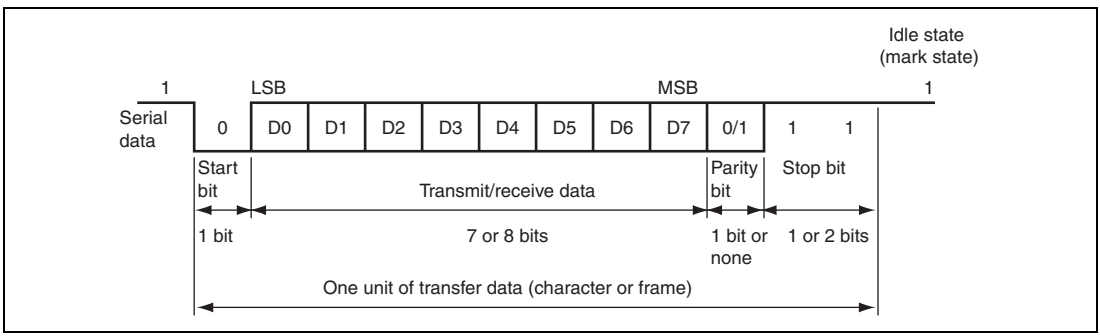


Figure 12.2 Data Format in Asynchronous Communication (Example with 8-Bit Data, Parity, Two Stop Bits)

12.4.1 Data Transfer Format

Table 12.10 shows the data transfer formats that can be used in asynchronous mode. Any of 12 transfer formats can be selected according to the SMR setting. For details on the multiprocessor bit, see section 12.5, Multiprocessor Communication Function.

Table 12.10 Serial Transfer Formats (Asynchronous Mode)

SMR Settings				Serial Transmit/Receive Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data							STOP			
1	0	0	1	S	7-bit data							STOP	STOP		
1	1	0	0	S	7-bit data							P	STOP		
1	1	0	1	S	7-bit data							P	STOP	STOP	
0	—	1	0	S	8-bit data								MPB	STOP	
0	—	1	1	S	8-bit data								MPB	STOP	STOP
1	—	1	0	S	7-bit data							MPB	STOP		
1	—	1	1	S	7-bit data							MPB	STOP	STOP	

[Legend]
S: Start bit
STOP: Stop bit
P: Parity bit
MPB: Multiprocessor bit

12.4.2 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI operates on a basic clock with a frequency of 16 times the bit rate. In reception, the SCI samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Since receive data is sampled at the rising edge of the 8th pulse of the basic clock, data is latched at the middle of each bit, as shown in figure 12.3. Thus the reception margin in asynchronous mode is determined by formula (1) below.

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100 \quad [\%] \quad \cdots \quad \text{Formula (1)}$$

M: Reception margin
N: Ratio of bit rate to clock (N = 16)
D: Duty cycle of clock (D = 0.5 to 1.0)
L: Frame length (L = 9 to 12)
F: Absolute value of clock frequency deviation

Assuming values of F = 0 and D = 0.5 in formula (1), the reception margin is determined by the formula below.

$$M = \left(0.5 - \frac{1}{2 \times 16}\right) \times 100 \quad [\%] = 46.875\%$$

However, this is only the computed value, and a margin of 20% to 30% should be allowed in system design.

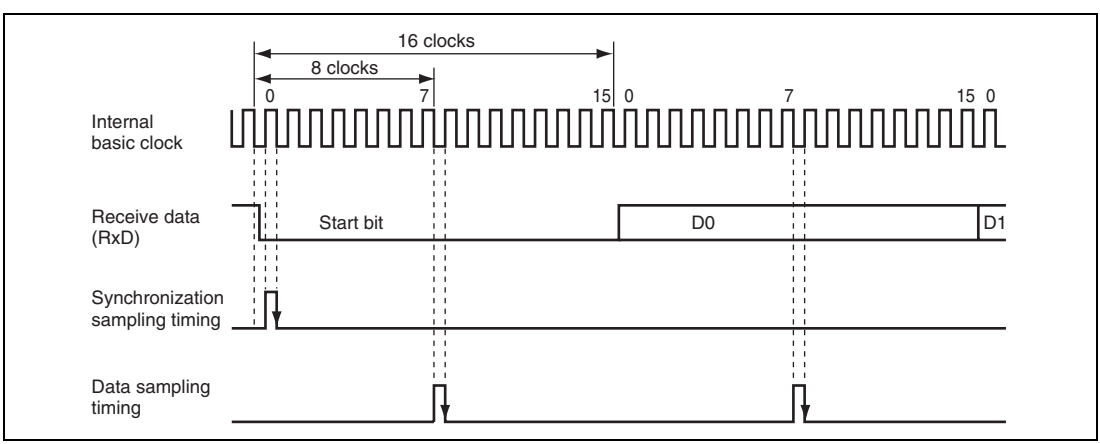


Figure 12.3 Receive Data Sampling Timing in Asynchronous Mode

12.4.3 Clock

Either an internal clock generated by the on-chip baud rate generator or an external clock input to the SCK pin can be selected as the SCI's transfer clock, according to the setting of the C/$\overline{A}$ bit in SMR and the CKE1 and CKE0 bits in SCR. When an external clock is input to the SCK pin, the clock frequency should be 16 times the bit rate used.

When the SCI is operated on an internal clock, the clock can be output from the SCK pin. The frequency of the clock output in this case is equal to the bit rate, and the phase is such that the rising edge of the clock is in the middle of the transmit data, as shown in figure 12.4.

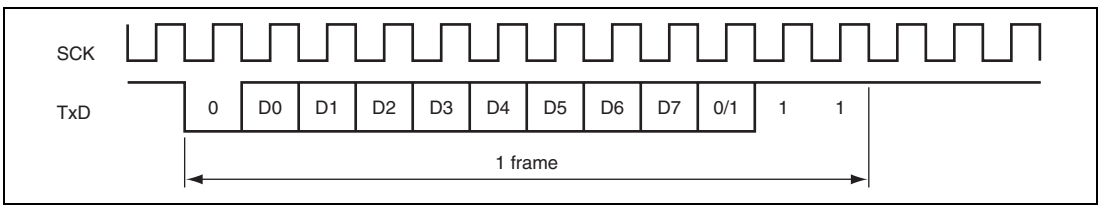
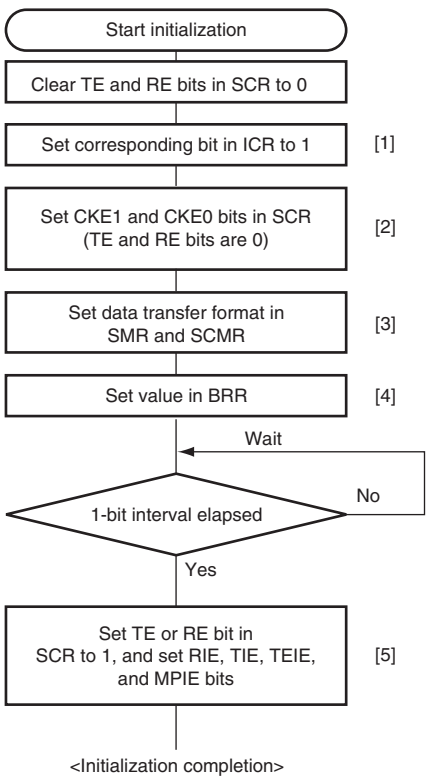


Figure 12.4 Phase Relation between Output Clock and Transmit Data (Asynchronous Mode)

12.4.4 SCI Initialization (Asynchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 12.5. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization.



[1] Set the bit in ICR for the corresponding pin when receiving data or using an external clock.

[2] Set the clock selection in SCR.
Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock output is selected in asynchronous mode, the clock is output immediately after SCR settings are made.

[3] Set the data transfer format in SMR and SCMR.

[4] Write a value corresponding to the bit rate to BRR. This step is not necessary if an external clock is used.

[5] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE, TEIE, and MPIE bits. Setting the TE and RE bits enables the TxD and RxD pins to be used.

Figure 12.5 Sample SCI Initialization Flowchart

12.4.5 Serial Data Transmission (Asynchronous Mode)

Figure 12.6 shows an example of the operation for transmission in asynchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is cleared to 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. Data is sent from the TxD pin in the following order: start bit, transmit data, parity bit or multiprocessor bit (may be omitted depending on the format), and stop bit.
4. The SCI checks the TDRE flag at the timing for sending the stop bit.
5. If the TDRE flag is 0, the next transmit data is transferred from TDR to TSR, the stop bit is sent, and then serial transmission of the next frame is started.
6. If the TDRE flag is 1, the TEND flag in SSR is set to 1, the stop bit is sent, and then the mark state is entered in which 1 is output. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated.

Figure 12.7 shows a sample flowchart for transmission in asynchronous mode.

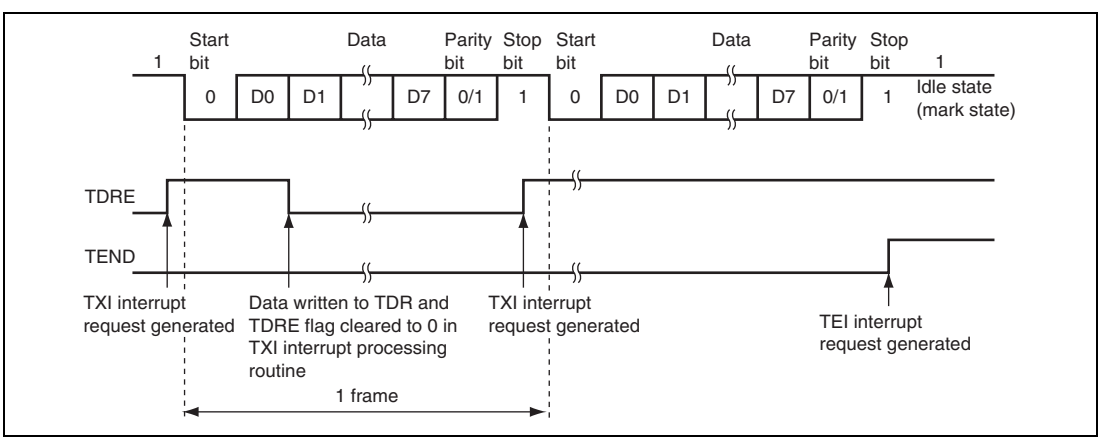
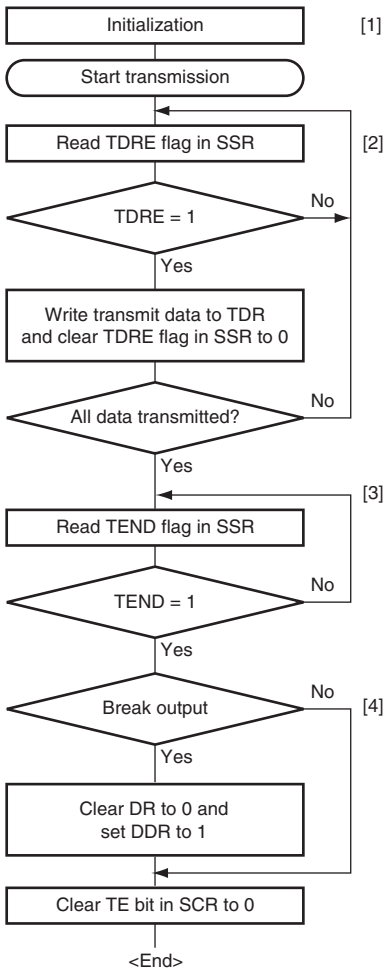


Figure 12.6 Example of Operation for Transmission in Asynchronous Mode (Example with 8-Bit Data, Parity, One Stop Bit)



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin. After the TE bit is set to 1, a 1 is output for a frame, and transmission is enabled.

[2] SCI state check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:

To continue serial transmission, read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port corresponding to the TxD pin to 1, clear DR to 0, then clear the TE bit in SCR to 0.

Figure 12.7 Sample Serial Transmission Flowchart

12.4.6 Serial Data Reception (Asynchronous Mode)

Figure 12.8 shows an example of the operation for reception in asynchronous mode. In serial reception, the SCI operates as described below.

1. The SCI monitors the communication line, and if a start bit is detected, performs internal synchronization, stores receive data in RSR, and checks the parity bit and stop bit.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If a parity error is detected, the PER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
4. If a framing error (when the stop bit is 0) is detected, the FER bit in SSR is set to 1 and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated.
5. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.

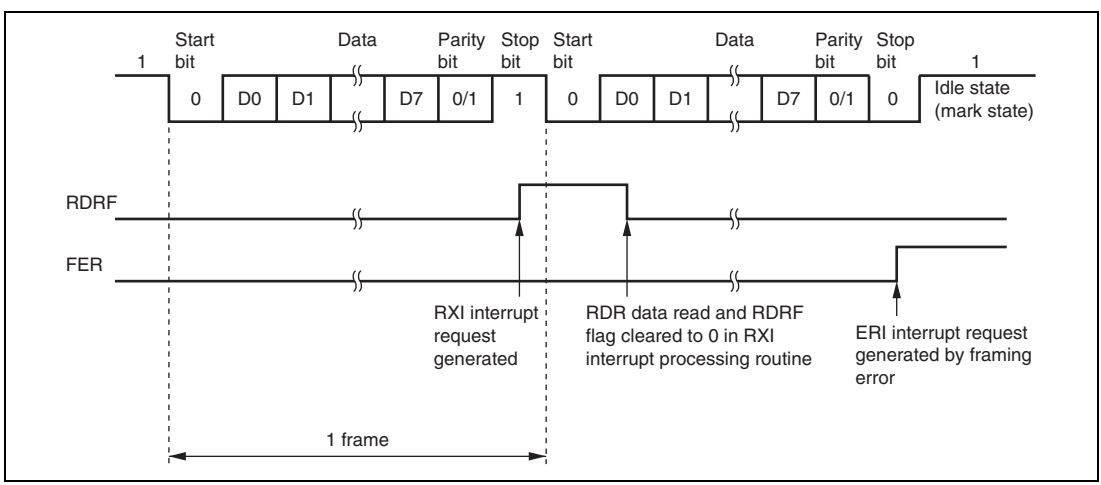


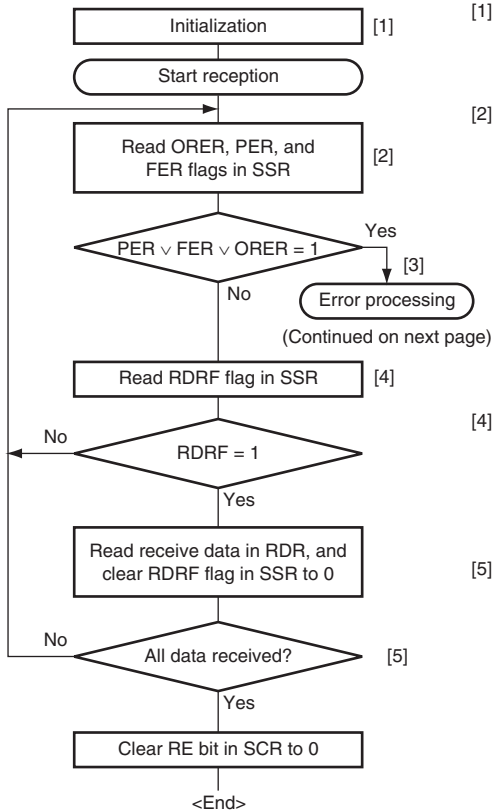
Figure 12.8 Example of SCI Operation for Reception (Example with 8-Bit Data, Parity, One Stop Bit)

Table 12.11 shows the states of the SSR status flags and receive data handling when a receive error is detected. If a receive error is detected, the RDRF flag retains its state before receiving data. Reception cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 12.9 shows a sample flowchart for serial data reception.

Table 12.11 SSR Status Flags and Receive Data Handling

SSR Status Flag					
RDRF*	**ORER**	**FER**	**PER**	**Receive Data**	**Receive Error Type**
1	1	0	0	Lost	Overrun error
0	0	1	0	Transferred to RDR	Framing error
0	0	0	1	Transferred to RDR	Parity error
1	1	1	0	Lost	Overrun error + framing error
1	1	0	1	Lost	Overrun error + parity error
0	0	1	1	Transferred to RDR	Framing error + parity error
1	1	1	1	Lost	Overrun error + framing error + parity error

Note: * The RDRF flag retains the state it had before data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing and break detection:
If a receive error occurs, read the ORER, PER, and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER, PER, and FER flags are all cleared to 0. Reception cannot be resumed if any of these flags are set to 1. In the case of a framing error, a break can be detected by reading the value of the input port corresponding to the RxD pin.

[4] SCI state check and receive data read:
Read SSR and check that RDRF = 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the stop bit for the current frame is received, read the RDRF flag and RDR, and clear the RDRF flag to 0. However, the RDRF flag is cleared automatically when the DMAC is initiated by an RXI interrupt and reads data from RDR.

Figure 12.9 Sample Serial Reception Flowchart (1)

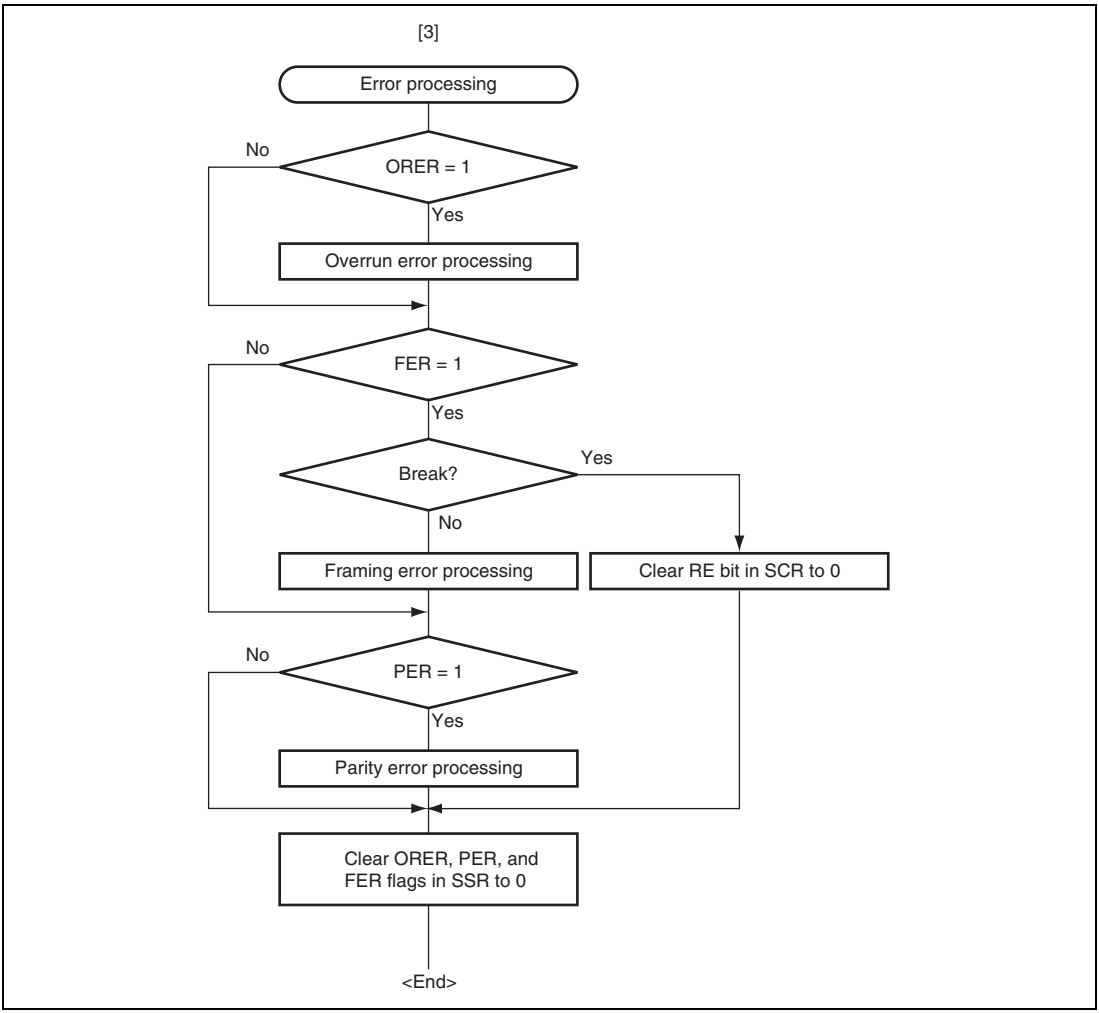


Figure 12.9 Sample Serial Reception Flowchart (2)

12.5 Multiprocessor Communication Function

Use of the multiprocessor communication function enables data transfer to be performed among a number of processors sharing communication lines by means of asynchronous serial communication using the multiprocessor format, in which a multiprocessor bit is added to the transfer data. When multiprocessor communication is carried out, each receiving station is addressed by a unique ID code. The serial communication cycle consists of two component cycles: an ID transmission cycle which specifies the receiving station, and a data transmission cycle for the specified receiving station. The multiprocessor bit is used to differentiate between the ID transmission cycle and the data transmission cycle. If the multiprocessor bit is 1, the cycle is an ID transmission cycle, and if the multiprocessor bit is 0, the cycle is a data transmission cycle. Figure 12.10 shows an example of inter-processor communication using the multiprocessor format. The transmitting station first sends data which includes the ID code of the receiving station and a multiprocessor bit set to 1. It then transmits transmit data added with the multiprocessor bit cleared to 0. The receiving station skips data until data with the multiprocessor bit set to 1 is sent. When data with the multiprocessor bit set to 1 is received, the receiving station compares that data with its own ID. The station whose ID matches then receives the data sent next. Stations whose ID does not match continue to skip data until data with the multiprocessor bit set to 1 is again received.

The SCI uses the MPIE bit in SCR to implement this function. When the MPIE bit is set to 1, transfer of receive data from RSR to RDR, error flag detection, and setting the SSR status flags, RDRF, FER, and ORER in SSR to 1 are prohibited until data with the multiprocessor bit set to 1 is received. On reception of a receive character with the multiprocessor bit set to 1, the MPBR bit in SSR is set to 1 and the MPIE bit is automatically cleared, thus normal reception is resumed. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt is generated.

When the multiprocessor format is selected, the parity bit setting is invalid. All other bit settings are the same as those in normal asynchronous mode. The clock used for multiprocessor communication is the same as that in normal asynchronous mode.

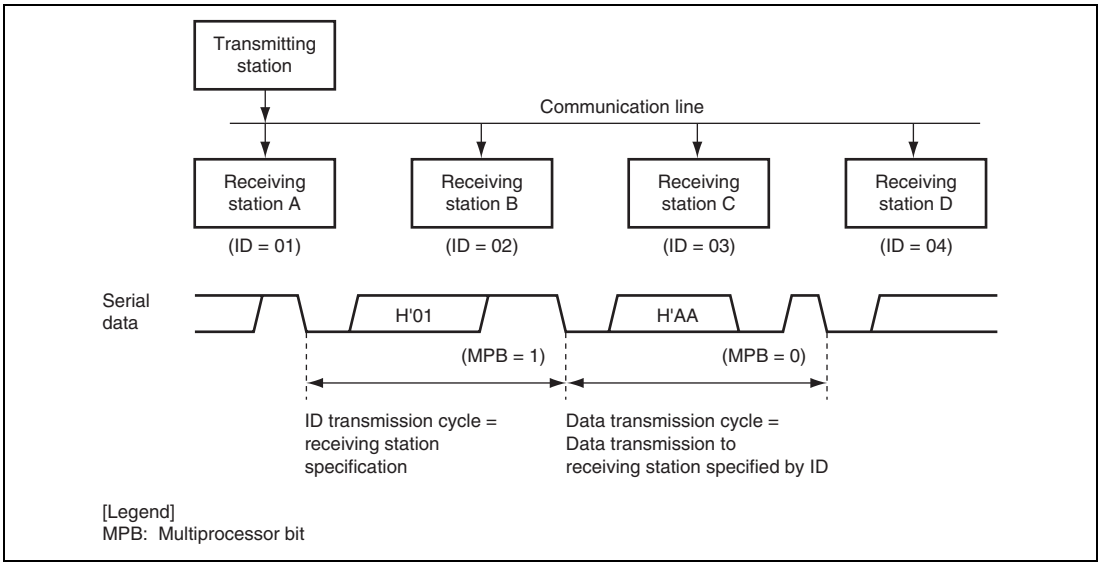
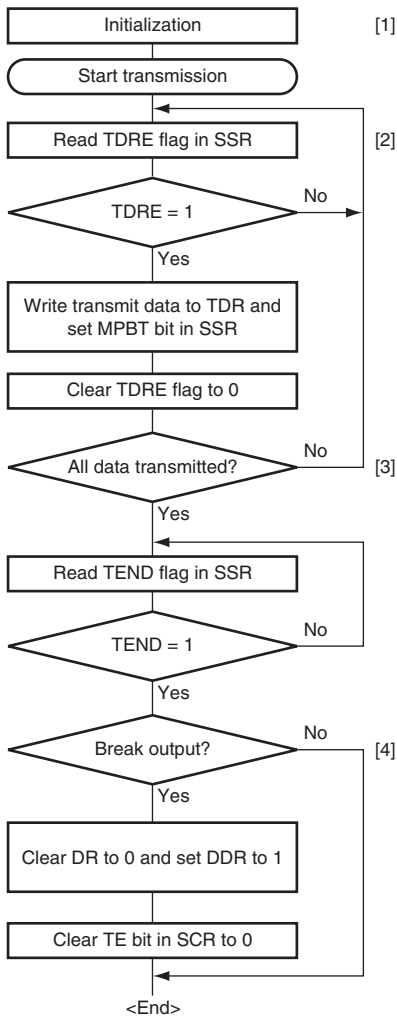


Figure 12.10 Example of Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)

12.5.1 Multiprocessor Serial Data Transmission

Figure 12.11 shows a sample flowchart for multiprocessor serial data transmission. For an ID transmission cycle, set the MPBT bit in SSR to 1 before transmission. For a data transmission cycle, clear the MPBT bit in SSR to 0 before transmission. All other SCI operations are the same as those in asynchronous mode.



[1] SCI initialization:
The TxD pin is automatically designated as the transmit data output pin. After the TE bit is set to 1, a 1 is output for one frame, and transmission is enabled.

[2] SCI status check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR. Set the MPBT bit in SSR to 0 or 1. Finally, clear the TDRE flag to 0.

[3] Serial transmission continuation procedure:
To continue serial transmission, be sure to read 1 from the TDRE flag to confirm that writing is possible, then write data to TDR, and then clear the TDRE flag to 0. However, the TDRE flag is checked and cleared automatically when the DMAC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR.

[4] Break output at the end of serial transmission:
To output a break in serial transmission, set DDR for the port to 1, clear DR to 0, and then clear the TE bit in SCR to 0.

Figure 12.11 Sample Multiprocessor Serial Transmission Flowchart

12.5.2 Multiprocessor Serial Data Reception

Figure 12.13 shows a sample flowchart for multiprocessor serial data reception. If the MPIE bit in SCR is set to 1, data is skipped until data with a 1 multiprocessor bit is sent. On receiving data with a 1 multiprocessor bit, the receive data is transferred to RDR. An RXI interrupt request is generated at this time. All other SCI operations are the same as in asynchronous mode. Figure 12.12 shows an example of SCI operation for multiprocessor format reception.

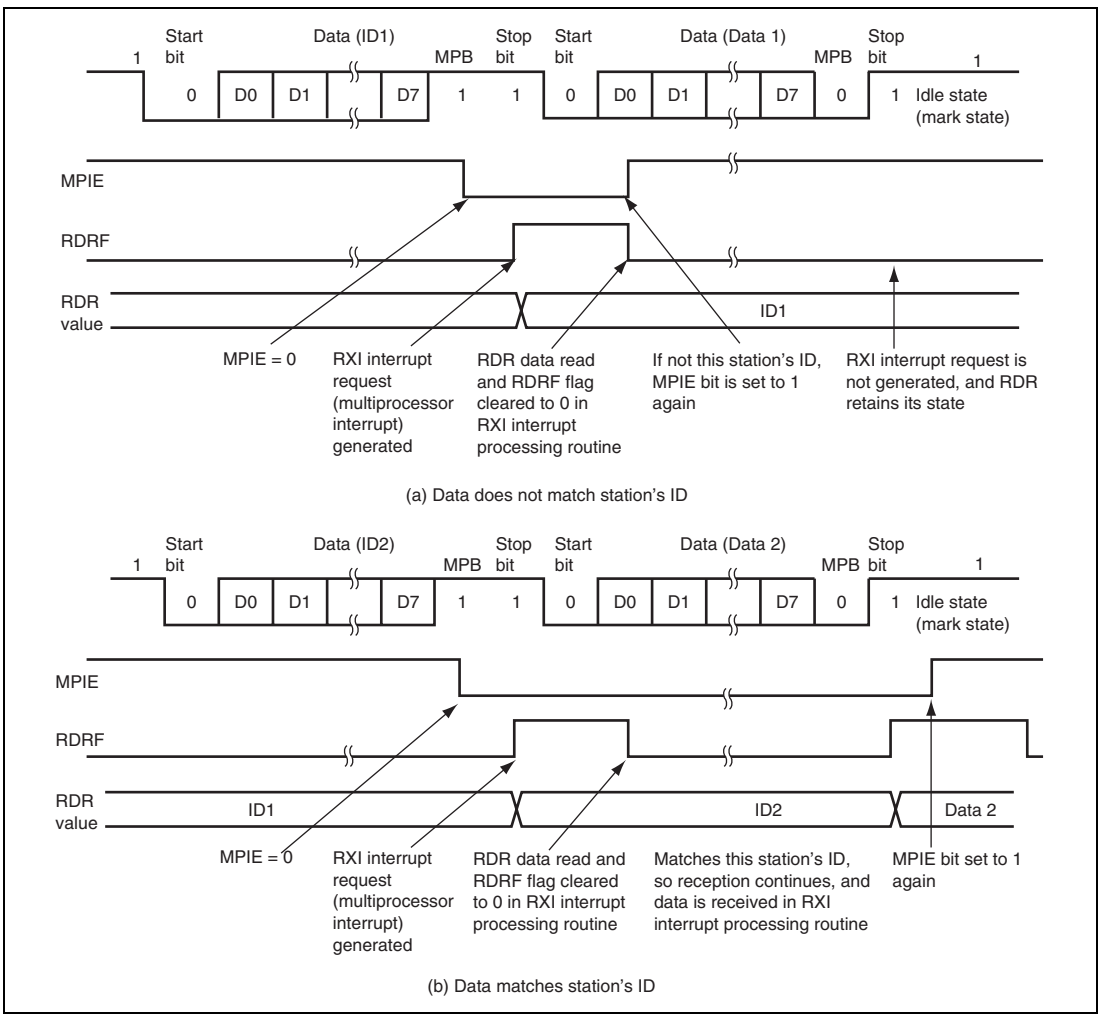
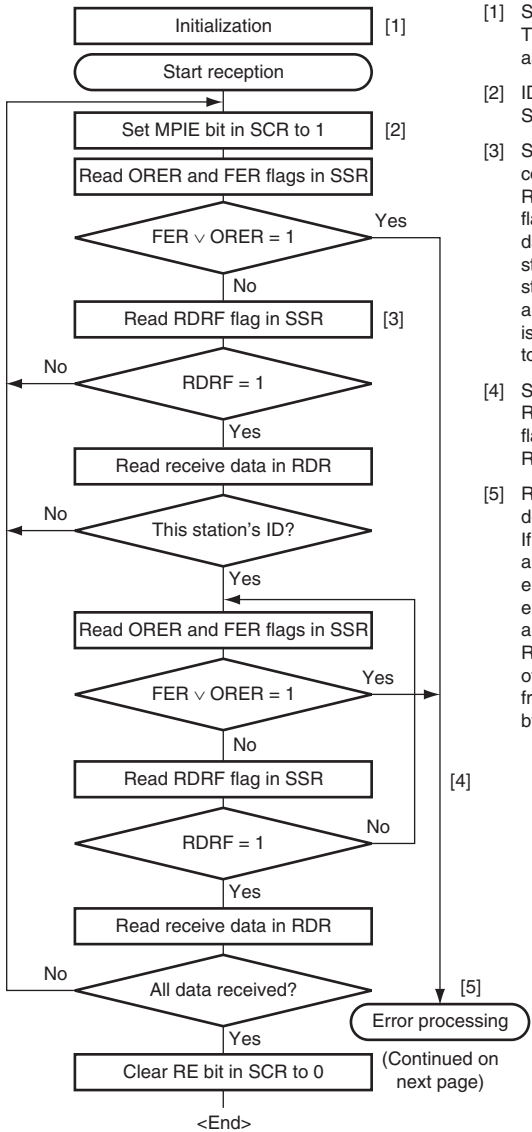


Figure 12.12 Example of SCI Operation for Reception (Example with 8-Bit Data, Multiprocessor Bit, One Stop Bit)



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] ID reception cycle:
Set the MPIE bit in SCR to 1.

[3] SCI state check, ID reception and comparison:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and compare it with this station's ID. If the data is not this station's ID, set the MPIE bit to 1 again, and clear the RDRF flag to 0. If the data is this station's ID, clear the RDRF flag to 0.

[4] SCI state check and data reception:
Read SSR and check that the RDRF flag is set to 1, then read the data in RDR.

[5] Receive error processing and break detection:
If a receive error occurs, read the ORER and FER flags in SSR to identify the error. After performing the appropriate error processing, ensure that the ORER and FER flags are both cleared to 0. Reception cannot be resumed if either of these flags is set to 1. In the case of a framing error, a break can be detected by reading the RxD pin value.

Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (1)

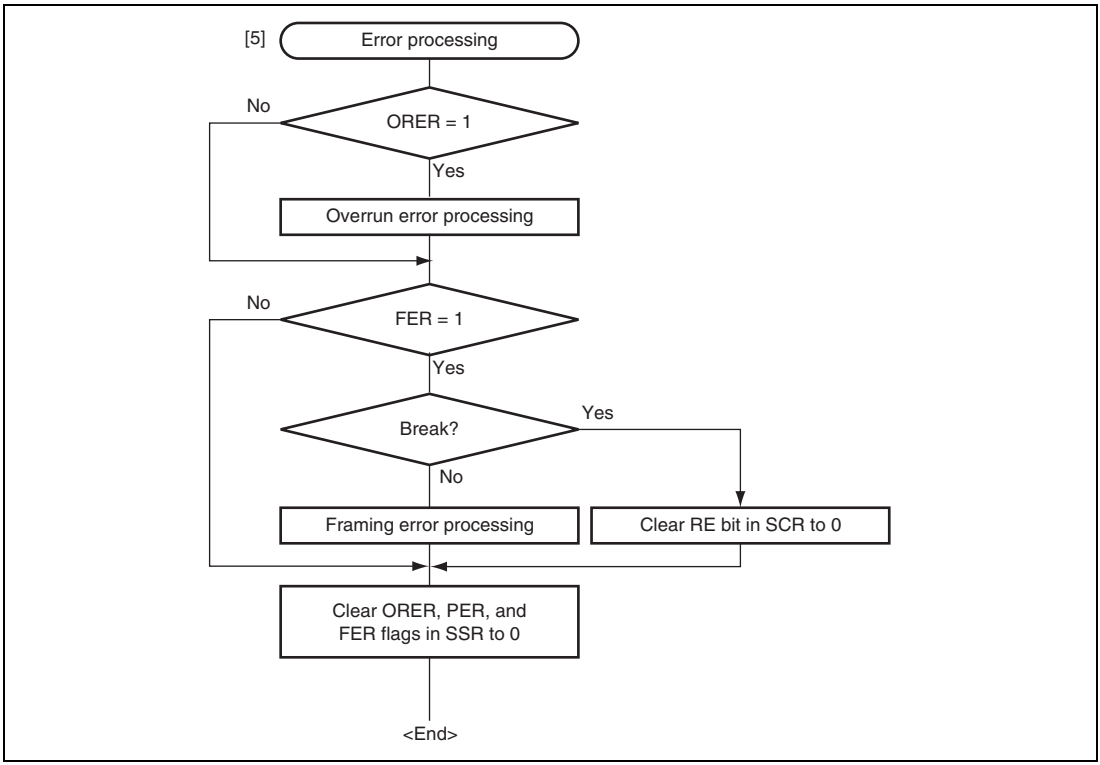
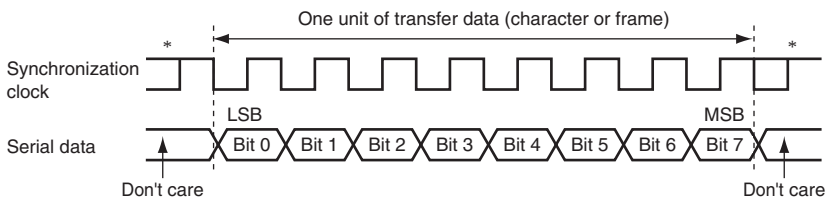


Figure 12.13 Sample Multiprocessor Serial Reception Flowchart (2)

12.6 Operation in Clocked Synchronous Mode

Figure 12.14 shows the general format for clocked synchronous communication. In clocked synchronous mode, data is transmitted or received in synchronization with clock pulses. One character in transfer data consists of 8-bit data. In data transmission, the SCI outputs data from one falling edge of the synchronization clock to the next. In data reception, the SCI receives data in synchronization with the rising edge of the synchronization clock. After 8-bit data is output, the transmission line holds the MSB output state. In clocked synchronous mode, no parity bit or multiprocessor bit is added. Inside the SCI, the transmitter and receiver are independent units, enabling full-duplex communication by use of a common clock. Both the transmitter and the receiver also have a double-buffered structure, so that the next transmit data can be written during transmission or the previous receive data can be read during reception, enabling continuous data transfer.



Note: * Holds a high level except during continuous transfer.

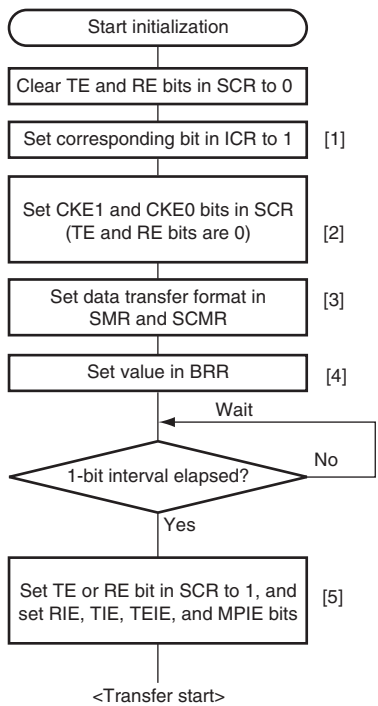
Figure 12.14 Data Format in Clocked Synchronous Communication (LSB-First)

12.6.1 Clock

Either an internal clock generated by the on-chip baud rate generator or an external synchronization clock input at the SCK pin can be selected, according to the setting of the CKE1 and CKE0 bits in SCR. When the SCI is operated on an internal clock, the synchronization clock is output from the SCK pin. Eight synchronization clock pulses are output in the transfer of one character, and when no transfer is performed the clock is fixed high. Note that in the case of reception only, the synchronization clock is output until an overrun error occurs or until the RE bit is cleared to 0.

12.6.2 SCI Initialization (Clocked Synchronous Mode)

Before transmitting and receiving data, first clear the TE and RE bits in SCR to 0, then initialize the SCI as described in a sample flowchart in figure 12.15. When the operating mode, transfer format, etc., is changed, the TE and RE bits must be cleared to 0 before making the change. When the TE bit is cleared to 0, the TDRE flag is set to 1. However, clearing the RE bit to 0 does not initialize the RDRF, PER, FER, and ORER flags, or RDR.



[1] Set the bit in ICR for the corresponding pin when receiving data or using an external clock.

[2] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

[3] Set the data transfer format in SMR and SCMR.

[4] Write a value corresponding to the bit rate to BRR. This step is not necessary if an external clock is used.

[5] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Also set the RIE, TIE TEIE, and MPIE bits. Setting the TE and RE bits enables the TxD and RxD pins to be used.

Note: In simultaneous transmit and receive operations, the TE and RE bits should both be cleared to 0 or set to 1 simultaneously.

Figure 12.15 Sample SCI Initialization Flowchart

12.6.3 Serial Data Transmission (Clocked Synchronous Mode)

Figure 12.16 shows an example of the operation for transmission in clocked synchronous mode. In transmission, the SCI operates as described below.

1. The SCI monitors the TDRE flag in SSR, and if it is 0, recognizes that data has been written to TDR, and transfers the data from TDR to TSR.
2. After transferring data from TDR to TSR, the SCI sets the TDRE flag to 1 and starts transmission. If the TIE bit in SCR is set to 1 at this time, a TXI interrupt request is generated. Because the TXI interrupt processing routine writes the next transmit data to TDR before transmission of the current transmit data has finished, continuous transmission can be enabled.
3. 8-bit data is sent from the TxD pin synchronized with the output clock when clock output mode has been specified and synchronized with the input clock when use of an external clock has been specified.
4. The SCI checks the TDRE flag at the timing for sending the last bit.
5. If the TDRE flag is cleared to 0, the next transmit data is transferred from TDR to TSR, and serial transmission of the next frame is started.
6. If the TDRE flag is set to 1, the TEND flag in SSR is set to 1, and the TxD pin retains the output state of the last bit. If the TEIE bit in SCR is set to 1 at this time, a TEI interrupt request is generated. The SCK pin is fixed high.

Figure 12.17 shows a sample flowchart for serial data transmission. Even if the TDRE flag is cleared to 0, transmission will not start while a receive error flag (ORER, FER, or PER) is set to 1. Make sure to clear the receive error flags to 0 before starting transmission. Note that clearing the RE bit to 0 does not clear the receive error flags.

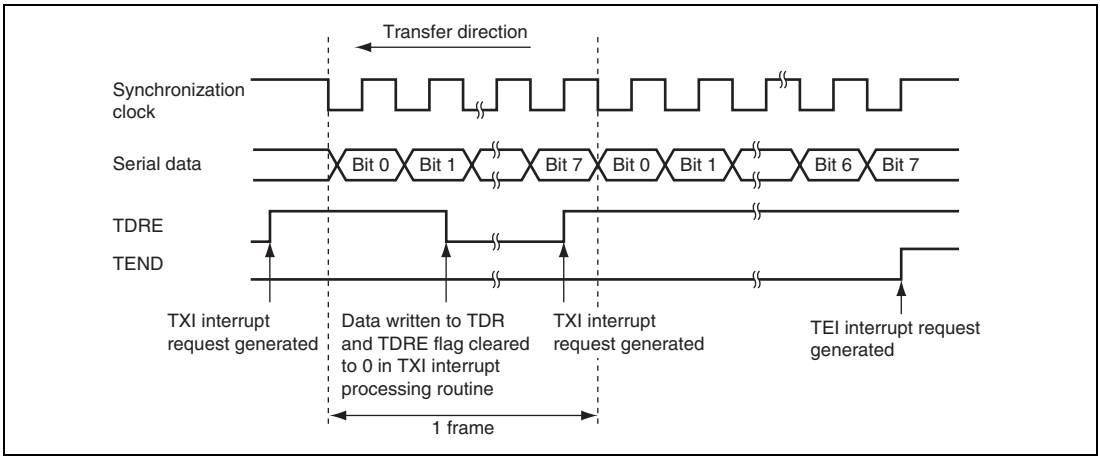


Figure 12.16 Example of Operation for Transmission in Clocked Synchronous Mode

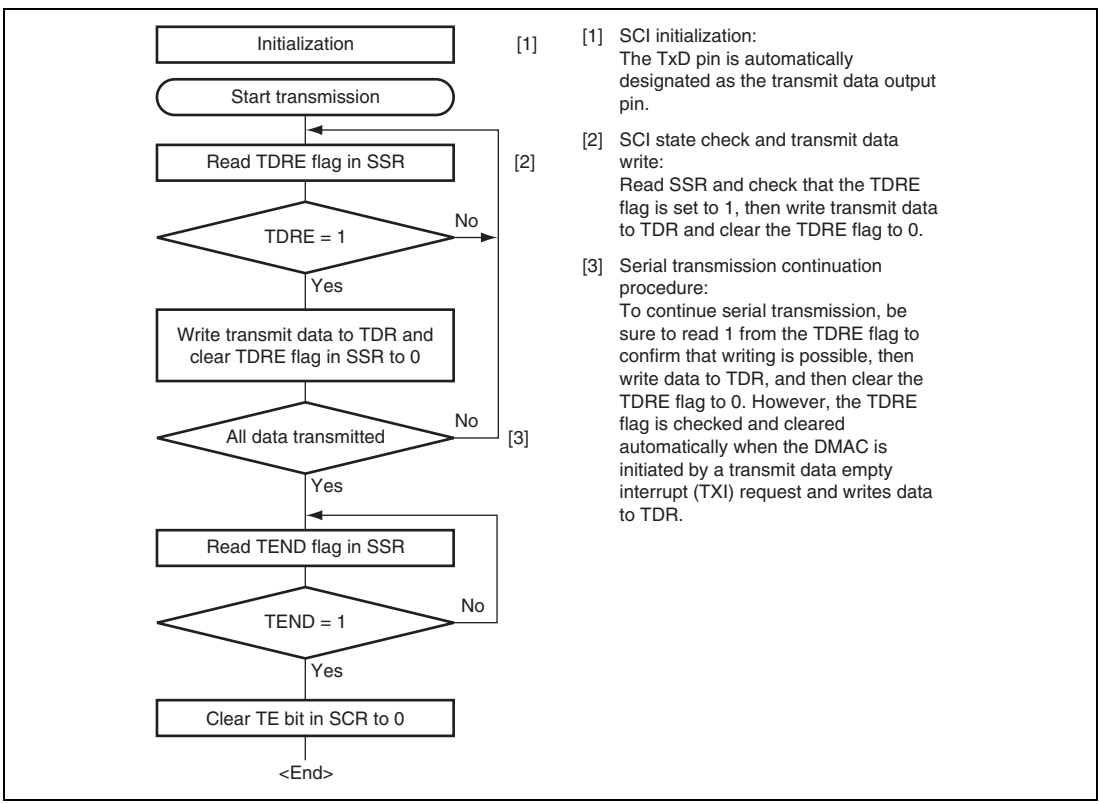


Figure 12.17 Sample Serial Transmission Flowchart

12.6.4 Serial Data Reception (Clocked Synchronous Mode)

Figure 12.18 shows an example of SCI operation for reception in clocked synchronous mode. In serial reception, the SCI operates as described below.

1. The SCI performs internal initialization in synchronization with a synchronization clock input or output, starts receiving data, and stores the receive data in RSR.
2. If an overrun error (when reception of the next data is completed while the RDRF flag in SSR is still set to 1) occurs, the ORER bit in SSR is set to 1. If the RIE bit in SCR is set to 1 at this time, an ERI interrupt request is generated. Receive data is not transferred to RDR. The RDRF flag remains to be set to 1.
3. If reception finishes successfully, the RDRF bit in SSR is set to 1, and receive data is transferred to RDR. If the RIE bit in SCR is set to 1 at this time, an RXI interrupt request is generated. Because the RXI interrupt processing routine reads the receive data transferred to RDR before reception of the next receive data has finished, continuous reception can be enabled.

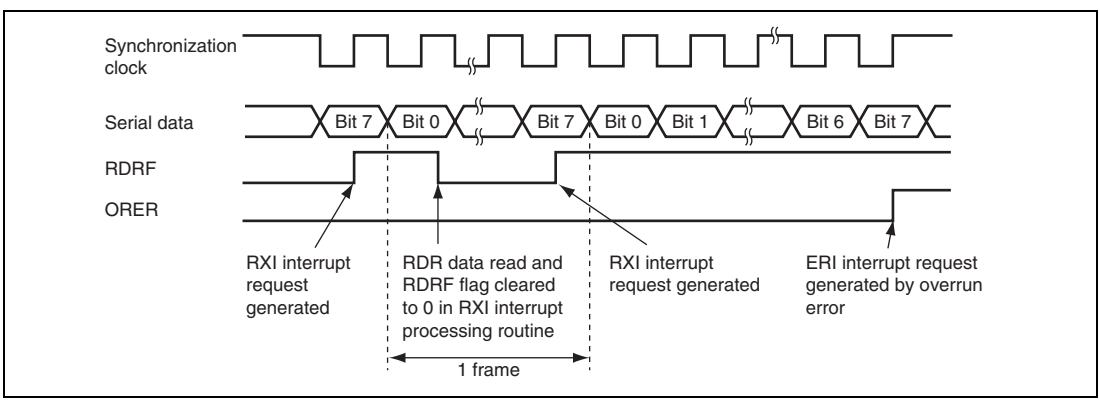
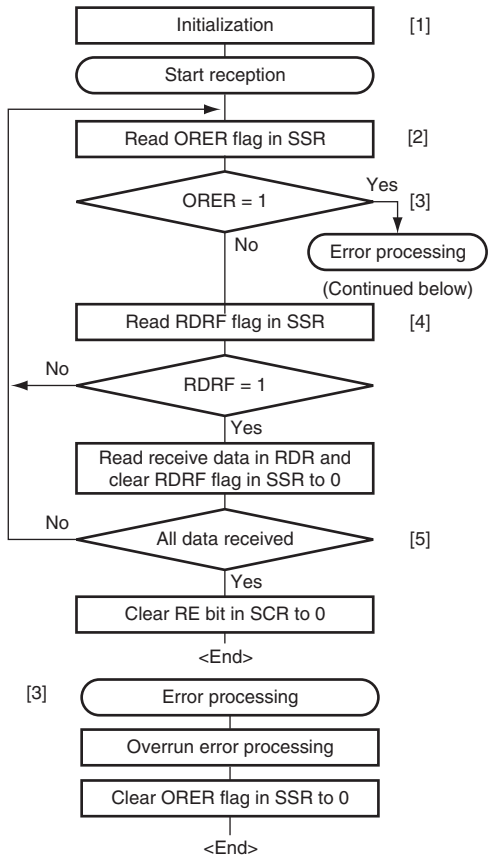


Figure 12.18 Example of Operation for Reception in Clocked Synchronous Mode

Transfer cannot be resumed while a receive error flag is set to 1. Accordingly, clear the ORER, FER, PER, and RDRF bits to 0 before resuming reception. Figure 12.19 shows a sample flowchart for serial data reception.



[1] SCI initialization:
The RxD pin is automatically designated as the receive data input pin.

[2] [3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Reception cannot be resumed if the ORER flag is set to 1.

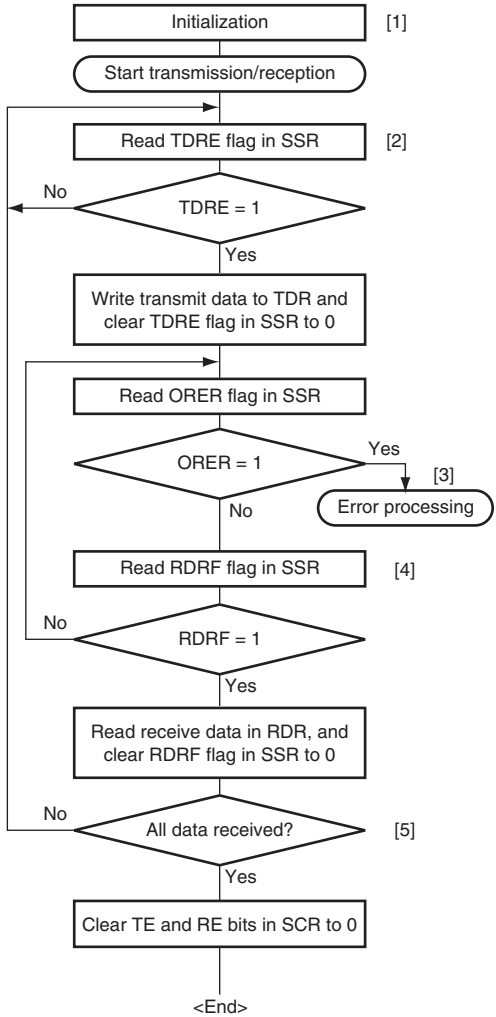
[4] SCI state check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial reception continuation procedure:
To continue serial reception, before the MSB (bit 7) of the current frame is received, reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0 should be finished. However, the RDRF flag is cleared automatically when the DMAC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

Figure 12.19 Sample Serial Reception Flowchart

12.6.5 Simultaneous Serial Data Transmission and Reception (Clocked Synchronous Mode)

Figure 12.20 shows a sample flowchart for simultaneous serial transmit and receive operations. After initializing the SCI, the following procedure should be used for simultaneous serial data transmit and receive operations. To switch from transmit mode to simultaneous transmit and receive mode, after checking that the SCI has finished transmission and the TDRE and TEND flags are set to 1, clear the TE bit to 0. Then simultaneously set both the TE and RE bits to 1 with a single instruction. To switch from receive mode to simultaneous transmit and receive mode, after checking that the SCI has finished reception, clear the RE bit to 0. Then after checking that the RDRF bit and receive error flags (ORER, FER, and PER) are cleared to 0, simultaneously set both the TE and RE bits to 1 with a single instruction.



Note: When switching from transmit or receive operation to simultaneous transmit and receive operations, first clear the TE bit and RE bit to 0, then set both these bits to 1 simultaneously.

[1] SCI initialization:
The TxD pin is designated as the transmit data output pin, and the RxD pin is designated as the receive data input pin, enabling simultaneous transmit and receive operations.

[2] SCI state check and transmit data write:
Read SSR and check that the TDRE flag is set to 1, then write transmit data to TDR and clear the TDRE flag to 0. Transition of the TDRE flag from 0 to 1 can also be identified by a TXI interrupt.

[3] Receive error processing:
If a receive error occurs, read the ORER flag in SSR, and after performing the appropriate error processing, clear the ORER flag to 0. Reception cannot be resumed if the ORER flag is set to 1.

[4] SCI state check and receive data read:
Read SSR and check that the RDRF flag is set to 1, then read the receive data in RDR and clear the RDRF flag to 0. Transition of the RDRF flag from 0 to 1 can also be identified by an RXI interrupt.

[5] Serial transmission/reception continuation procedure:
To continue serial transmission/reception, before the MSB (bit 7) of the current frame is received, finish reading the RDRF flag, reading RDR, and clearing the RDRF flag to 0. Also, before the MSB (bit 7) of the current frame is transmitted, read 1 from the TDRE flag to confirm that writing is possible. Then write data to TDR and clear the TDRE flag to 0.
However, the TDRE flag is checked and cleared automatically when the DMAC is initiated by a transmit data empty interrupt (TXI) request and writes data to TDR. Similarly, the RDRF flag is cleared automatically when the DMAC is initiated by a receive data full interrupt (RXI) and reads data from RDR.

Figure 12.20 Sample Flowchart of Simultaneous Serial Transmission and Reception

12.7 Operation in Smart Card Interface Mode

The SCI supports the IC card (smart card) interface, conforming to ISO/IEC 7816-3 (Identification Card) standard, as an extended serial communication interface function. Smart card interface mode can be selected using the appropriate register.

12.7.1 Sample Connection

Figure 12.21 shows a sample connection between the smart card and this LSI. As in the figure, since this LSI communicates with the IC card using a single transmission line, interconnect the TxD and RxD pins and pull up the data transmission line to V_{CC} using a resistor. Setting the RE and TE bits to 1 with the IC card not connected enables closed transmission/reception allowing self diagnosis. To supply the IC card with the clock pulses generated by the SCI, input the SCK pin output to the CLK pin of the IC card. A reset signal can be supplied via the output port of this LSI.

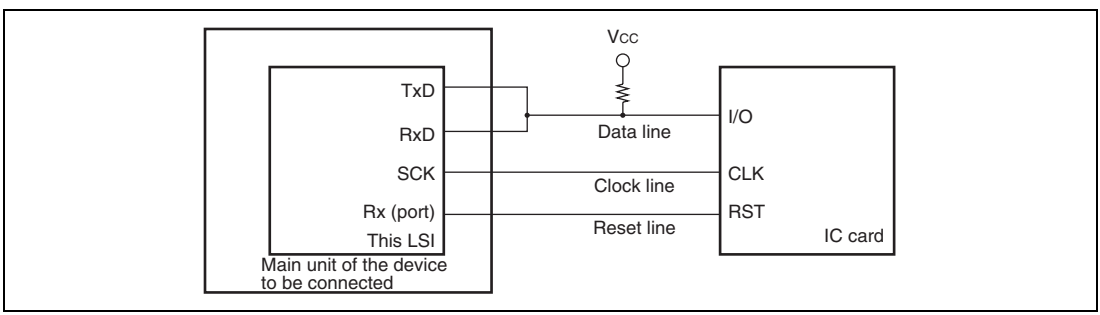


Figure 12.21 Pin Connection for Smart Card Interface

12.7.2 Data Format (Except in Block Transfer Mode)

Figure 12.22 shows the data transfer formats in smart card interface mode.

- One frame contains 8-bit data and a parity bit in asynchronous mode.
- During transmission, at least 2 etu (elementary time unit: time required for transferring one bit) is secured as a guard time after the end of the parity bit before the start of the next frame.
- If a parity error is detected during reception, a low error signal is output for 1 etu after 10.5 etu has passed from the start bit.
- If an error signal is sampled during transmission, the same data is automatically re-transmitted after at least 2 etu.

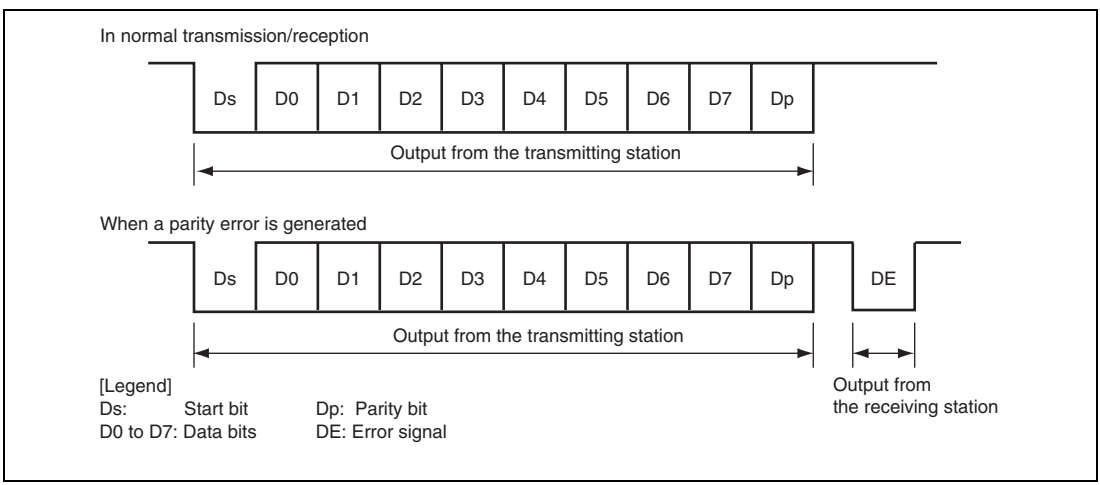


Figure 12.22 Data Formats in Normal Smart Card Interface Mode

For communication with the IC cards of the direct convention and inverse convention types, follow the procedure below.

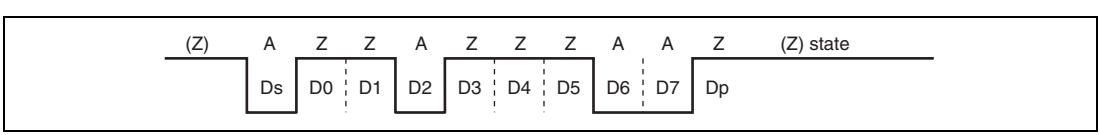


Figure 12.23 Direct Convention (SDIR = SINV = O/$\overline{E}$ = 0)

For the direct convention type, logic levels 1 and 0 correspond to states Z and A, respectively, and data is transferred with LSB-first as the start character, as shown in figure 12.23. Therefore, data in the start character in the figure is H'3B. When using the direct convention type, write 0 to both the SDIR and SINV bits in SCMR. Write 0 to the O/$\overline{E}$ bit in SMR in order to use even parity, which is prescribed by the smart card standard.

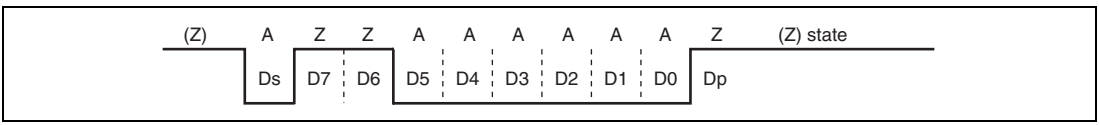


Figure 12.24 Inverse Convention (SDIR = SINV = O/$\overline{E}$ = 1)

For the inverse convention type, logic levels 1 and 0 correspond to states A and Z, respectively and data is transferred with MSB-first as the start character, as shown in figure 12.24. Therefore, data in the start character in the figure is H'3F. When using the inverse convention type, write 1 to both the SDIR and SINV bits in SCMR. The parity bit is logic level 0 to produce even parity, which is prescribed by the smart card standard, and corresponds to state Z. Since the SNIV bit of this LSI only inverts data bits D7 to D0, write 1 to the O/$\overline{E}$ bit in SMR to invert the parity bit in both transmission and reception.

12.7.3 Block Transfer Mode

Block transfer mode is different from normal smart card interface mode in the following respects.

- Even if a parity error is detected during reception, no error signal is output. Since the PER bit in SSR is set by error detection, clear the PER bit before receiving the parity bit of the next frame.
- During transmission, at least 1 etu is secured as a guard time after the end of the parity bit before the start of the next frame.
- Since the same data is not re-transmitted during transmission, the TEND flag is set 11.5 etu after transmission start.
- Although the ERS flag in block transfer mode displays the error signal status as in normal smart card interface mode, the flag is always read as 0 because no error signal is transferred.

12.7.4 Receive Data Sampling Timing and Reception Margin

Only the internal clock generated by the on-chip baud rate generator can be used as a transfer clock in smart card interface mode. In this mode, the SCI can operate on a basic clock with a frequency of 32, 64, 372, or 256 times the bit rate according to the BCP1 and BCP0 bit settings (the frequency is always 16 times the bit rate in normal asynchronous mode). At reception, the falling edge of the start bit is sampled using the basic clock in order to perform internal synchronization. Receive data is sampled on the 16th, 32nd, 186th and 128th rising edges of the basic clock so that it can be latched at the middle of each bit as shown in figure 12.25. The reception margin here is determined by the following formula.

$$M = \left| \left(0.5 - \frac{1}{2N}\right) - (L - 0.5)F - \frac{|D - 0.5|}{N}(1 + F) \right| \times 100\%$$

M: Reception margin (%)
N: Ratio of bit rate to clock (N = 32, 64, 372, 256)
D: Duty cycle of clock (D = 0 to 1.0)
L: Frame length (L = 10)
F: Absolute value of clock frequency deviation

Assuming values of F = 0, D = 0.5, and N = 372 in the above formula, the reception margin is determined by the formula below.

$$M = \left(0.5 - \frac{1}{2 \times 372}\right) \times 100\% = 49.866\%$$

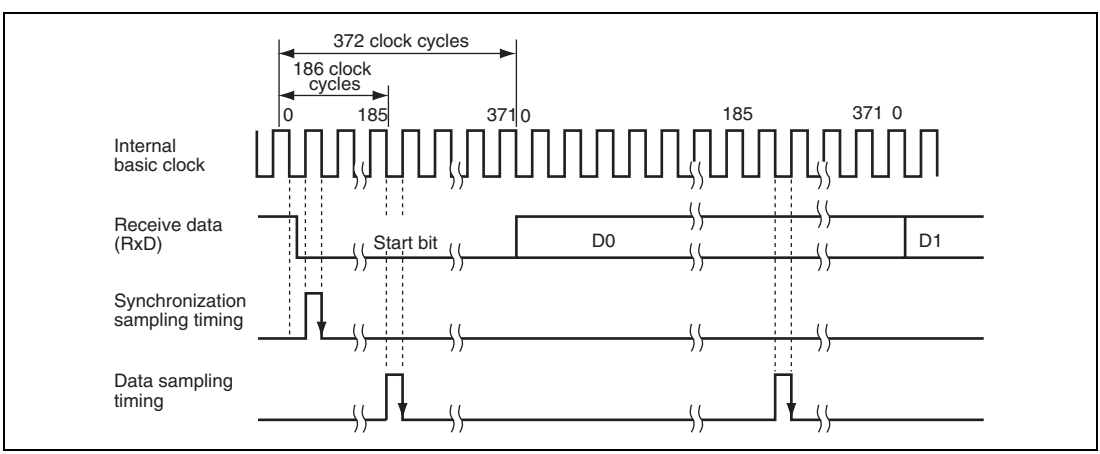


Figure 12.25 Receive Data Sampling Timing in Smart Card Interface Mode (When Clock Frequency is 372 Times the Bit Rate)

12.7.5 Initialization

Before transmitting and receiving data, initialize the SCI using the following procedure. Initialization is also necessary before switching from transmission to reception and vice versa.

1. Clear the TE and RE bits in SCR to 0.
2. Set the ICR bit of the corresponding pin to 1.
3. Clear the error flags ERS, PER, and ORER in SSR to 0.
4. Set the GM, BLK, O/$\overline{E}$, BCP1, BCP0, CKS1, and CKS0 bits in SMR appropriately. Also set the PE bit to 1.
5. Set the SMIF, SDIR, and SINV bits in SCMR appropriately. When the DDR corresponding to the TxD pin is cleared to 0, the TxD and RxD pins are changed from port pins to SCI pins, placing the pins into high impedance state.
6. Set the value corresponding to the bit rate in BRR.
7. Set the CKE1 and CKE0 bits in SCR appropriately. Clear the TIE, RIE, TE, RE, MPIE, and TEIE bits to 0 simultaneously.

   When the CKE0 bit is set to 1, the SCK pin is allowed to output clock pulses.
8. Set the TIE, RIE, TE, and RE bits in SCR appropriately after waiting for at least a 1-bit interval. Setting the TE and RE bits to 1 simultaneously is prohibited except for self diagnosis.

To switch from reception to transmission, first verify that reception has completed, then initialize the SCI. At the end of initialization, RE and TE should be set to 0 and 1, respectively. Reception completion can be verified by reading the RDRF, PER, or ORER flag. To switch from transmission to reception, first verify that transmission has completed, then initialize the SCI. At the end of initialization, TE and RE should be set to 0 and 1, respectively. Transmission completion can be verified by reading the TEND flag.

12.7.6 Data Transmission (Except in Block Transfer Mode)

Data transmission in smart card interface mode (except in block transfer mode) is different from that in normal serial communication interface mode in that an error signal is sampled and data can be re-transmitted. Figure 12.26 shows the data re-transfer operation during transmission.

1. If an error signal from the receiving end is sampled after one frame of data has been transmitted, the ERS bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the ERS bit to 0 before the next parity bit is sampled.
2. For the frame in which an error signal is received, the TEND bit in SSR is not set to 1. Data is re-transferred from TDR to TSR allowing automatic data retransmission.
3. If no error signal is returned from the receiving end, the ERS bit in SSR is not set to 1.
4. In this case, one frame of data is determined to have been transmitted including re-transfer, and the TEND bit in SSR is set to 1. Here, a TXI interrupt request is generated if the TIE bit in SCR is set to 1. Writing transmit data to TDR starts transmission of the next data.

Figure 12.28 shows a sample flowchart for transmission. All the processing steps are automatically performed using a TXI interrupt request to activate the DMAC. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt request if the TIE bit in SCR has been set to 1. This activates the DMAC by a TXI request thus allowing transfer of transmit data if the TXI interrupt request is specified as a source of DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, TEND remains as 0, thus not activating the DMAC. Therefore, the SCI and DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC, be sure to set and enable the DMAC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC).

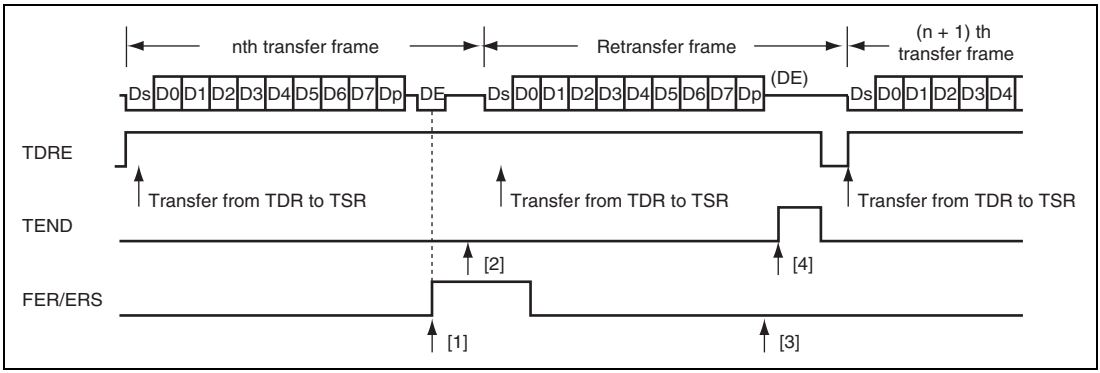


Figure 12.26 Data Re-Transfer Operation in SCI Transmission Mode

Note that the TEND flag is set in different timings depending on the GM bit setting in SMR. Figure 12.27 shows the TEND flag set timing.

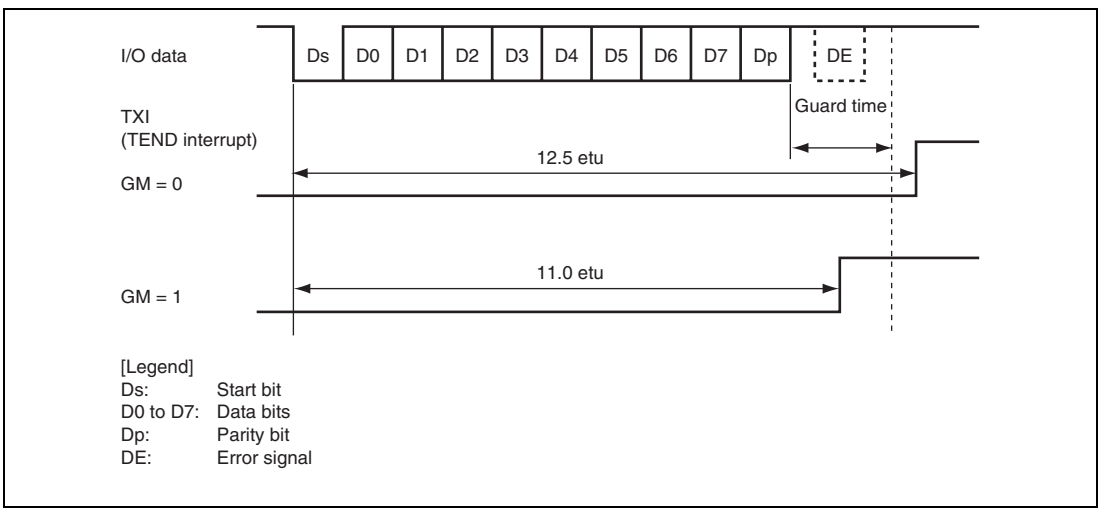


Figure 12.27 TEND Flag Set Timing during Transmission

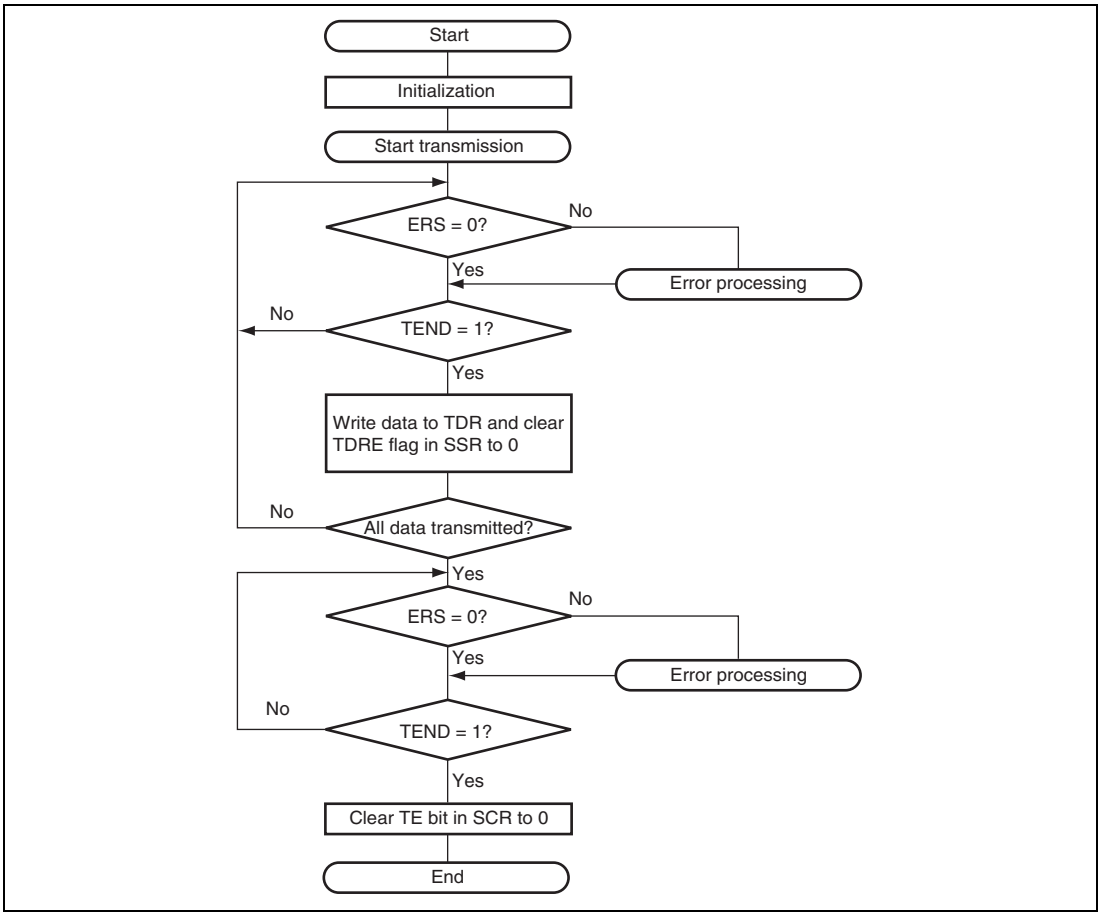


Figure 12.28 Sample Transmission Flowchart

12.7.7 Serial Data Reception (Except in Block Transfer Mode)

Data reception in smart card interface mode is similar to that in normal serial communication interface mode. Figure 12.29 shows the data re-transfer operation during reception.

1. If a parity error is detected in receive data, the PER bit in SSR is set to 1. Here, an ERI interrupt request is generated if the RIE bit in SCR is set to 1. Clear the PER bit to 0 before the next parity bit is sampled.
2. For the frame in which a parity error is detected, the RDRF bin in SSR is not set to 1.
3. If no parity error is detected, the PER bit in SSR is not set to 1.
4. In this case, data is determined to have been received successfully, and the RDRF bit in SSR is set to 1. Here, an RXI interrupt request is generated if the RIE bit in SCR is set to 1.

Figure 12.30 shows a sample flowchart for reception. All the processing steps are automatically performed using an RXI interrupt request to activate the DMAC. In reception, setting the RIE bit to 1 allows an RXI interrupt request to be generated when the RDRF flag is set to 1. This activates the DMAC by an RXI request thus allowing transfer of receive data if the RXI interrupt request is specified as a source of DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC. If an error occurs during reception, i.e., either the ORER or PER flag is set to 1, a transmit/receive error interrupt (ERI) request is generated and the error flag must be cleared. If an error occurs, the DMAC is not activated and receive data is skipped, therefore, the number of bytes of receive data specified in the DMAC is transferred. Even if a parity error occurs and the PER bit is set to 1 in reception, receive data is transferred to RDR, thus allowing the data to be read.

Note: For operations in block transfer mode, see section 12.4, Operation in Asynchronous Mode.

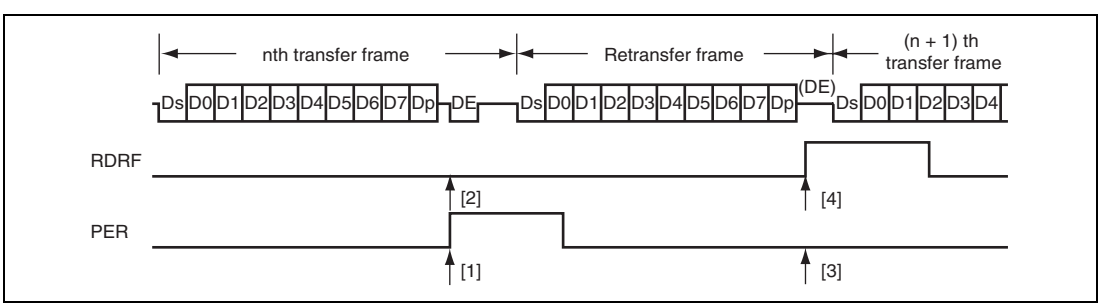


Figure 12.29 Data Re-Transfer Operation in SCI Reception Mode

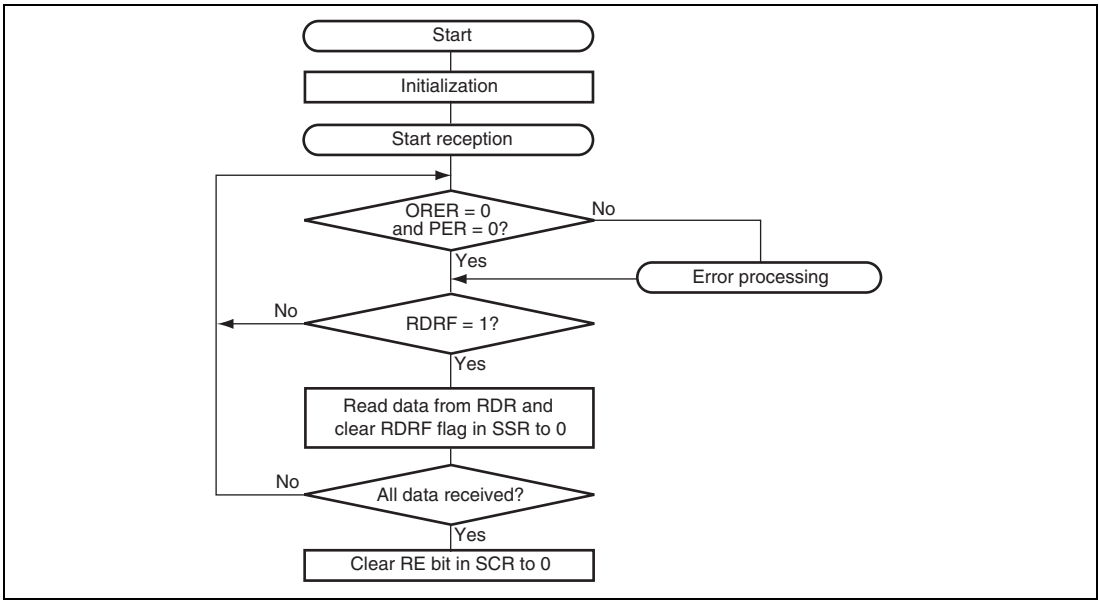


Figure 12.30 Sample Reception Flowchart

12.7.8 Clock Output Control

Clock output can be fixed using the CKE1 and CKE0 bits in SCR when the GM bit in SMR is set to 1. Specifically, the minimum width of a clock pulse can be specified.

Figure 12.31 shows an example of clock output fixing timing when the CKE0 bit is controlled with GM = 1 and CKE1 = 0.

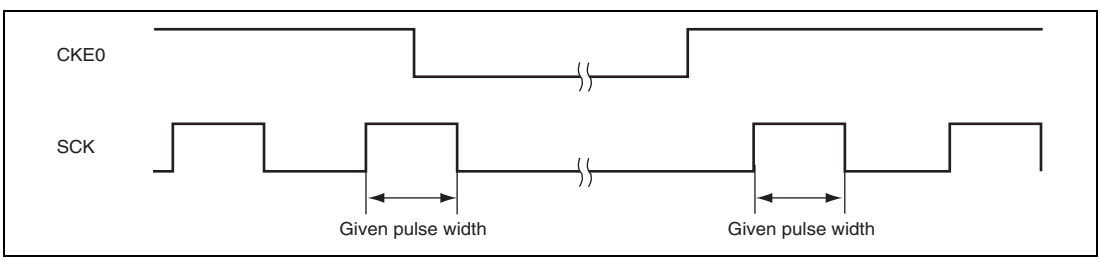


Figure 12.31 Clock Output Fixing Timing

At power-on and transitions to/from software standby mode, use the following procedure to secure the appropriate clock duty cycle.

- At power-on

  To secure the appropriate clock duty cycle simultaneously with power-on, use the following procedure.

  1. Initially, port input is enabled in the high-impedance state. To fix the potential level, use a pull-up or pull-down resistor.
  2. Fix the SCK pin to the specified output using the CKE1 bit in SCR.
  3. Set SMR and SCMR to enable smart card interface mode.

     Set the CKE0 bit in SCR to 1 to start clock output.

- At mode switching

  — At transition from smart card interface mode to software standby mode

  1. Set the data register (DR) and data direction register (DDR) corresponding to the SCK pin to the values for the output fixed state in software standby mode.
  2. Write 0 to the TE and RE bits in SCR to stop transmission/reception. Simultaneously, set the CKE1 bit to the value for the output fixed state in software standby mode.
  3. Write 0 to the CKE0 bit in SCR to stop the clock.
  4. Wait for one cycle of the serial clock. In the mean time, the clock output is fixed to the specified level with the duty cycle retained.
  5. Make the transition to software standby mode.

  — At transition from smart card interface mode to software standby mode

  1. Clear software standby mode.
  2. Write 1 to the CKE0 bit in SCR to start clock output. A clock signal with the appropriate duty cycle is then generated.

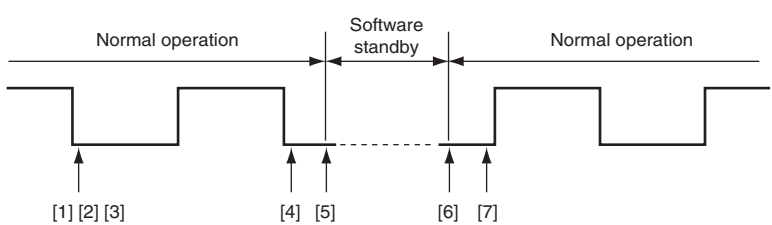


Figure 12.32 Clock Stop and Restart Procedure

12.8 Interrupt Sources

12.8.1 Interrupts in Normal Serial Communication Interface Mode

Table 12.12 shows the interrupt sources in normal serial communication interface mode. A different interrupt vector is assigned to each interrupt source, and individual interrupt sources can be enabled or disabled using the enable bits in SCR.

When the TDRE flag in SSR is set to 1, a TXI interrupt request is generated. When the TEND flag in SSR is set to 1, a TEI interrupt request is generated. A TXI interrupt request can activate the DMAC to allow data transfer. The TDRE flag is automatically cleared to 0 at data transfer by the DMAC.

When the RDRF flag in SSR is set to 1, an RXI interrupt request is generated. When the ORER, PER, or FER flag in SSR is set to 1, an ERI interrupt request is generated. An RXI interrupt can activate the DMAC to allow data transfer. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC.

A TEI interrupt is requested when the TEND flag is set to 1 while the TEIE bit is set to 1. If a TEI interrupt and a TXI interrupt are requested simultaneously, the TXI interrupt has priority for acceptance. However, note that if the TDRE and TEND flags are cleared to 0 simultaneously by the TXI interrupt processing routine, the SCI cannot branch to the TEI interrupt processing routine later.

Table 12.12 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DMAC Activation	Priority
ERI	Receive error	ORER, FER, or PER	Not possible	High
RXI	Receive data full	RDRF	Possible	↑
TXI	Transmit data empty	TDRE	Possible	
TEI	Transmit end	TEND	Not possible	Low

12.8.2 Interrupts in Smart Card Interface Mode

Table 12.13 shows the interrupt sources in smart card interface mode. A transmit end (TEI) interrupt request cannot be used in this mode.

Table 12.13 SCI Interrupt Sources

Name	Interrupt Source	Interrupt Flag	DMAC Activation	Priority
ERI	Receive error or error signal detection	ORER, PER, or ERS	Not possible	High
RXI	Receive data full	RDRF	Possible	↑
TXI	Transmit data empty	TDRE	Possible	Low

Data transmission/reception using the DMAC is also possible in smart card interface mode, similar to in the normal SCI mode. In transmission, the TEND and TDRE flags in SSR are simultaneously set to 1, thus generating a TXI interrupt. This activates the DMAC by a TXI request thus allowing transfer of transmit data if the TXI request is specified as a source of DMAC activation beforehand. The TDRE and TEND flags are automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the SCI automatically re-transmits the same data. During re-transmission, the TEND flag remains as 0, thus not activating the DMAC. Therefore, the SCI and DMAC automatically transmit the specified number of bytes, including re-transmission in the case of error occurrence. However, the ERS flag in SSR, which is set at error occurrence, is not automatically cleared; the ERS flag must be cleared by previously setting the RIE bit in SCR to 1 to enable an ERI interrupt request to be generated at error occurrence.

When transmitting/receiving data using the DMAC, be sure to set and enable the DMAC prior to making SCI settings. For DMAC settings, see section 7, DMA Controller (DMAC).

In reception, an RXI interrupt request is generated when the RDRF flag in SSR is set to 1. This activates the DMAC by an RXI request thus allowing transfer of receive data if the RXI request is specified as a source of DMAC activation beforehand. The RDRF flag is automatically cleared to 0 at data transfer by the DMAC. If an error occurs, the RDRF flag is not set but the error flag is set. Therefore, the DMAC is not activated and an ERI interrupt request is issued to the CPU instead; the error flag must be cleared.

12.9 Usage Notes

12.9.1 Module Stop Mode Setting

Operation of the SCI can be disabled or enabled using the module stop control register. The initial setting is for operation of the SCI to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 19, Power-Down Modes.

12.9.2 Break Detection and Processing

When framing error detection is performed, a break can be detected by reading the RxD pin value directly. In a break, the input from the RxD pin becomes all 0s, and so the FER flag is set, and the PER flag may also be set. Note that, since the SCI continues the receive operation even after receiving a break, even if the FER flag is cleared to 0, it will be set to 1 again.

12.9.3 Mark State and Break Detection

When the TE bit is 0, the TxD pin is used as an I/O port whose direction (input or output) and level are determined by DR and DDR. This can be used to set the TxD pin to mark state (high level) or send a break during serial data transmission. To maintain the communication line in mark state (the state of 1) until TE is set to 1, set both DDR and DR to 1. Since the TE bit is cleared to 0 at this point, the TxD pin becomes an I/O port, and 1 is output from the TxD pin. To send a break during serial transmission, first set DDR to 1 and DR to 0, and then clear the TE bit to 0. When the TE bit is cleared to 0, the transmitter is initialized regardless of the current transmission state, the TxD pin becomes an I/O port, and 0 is output from the TxD pin.

12.9.4 Receive Error Flags and Transmit Operations (Clocked Synchronous Mode Only)

Transmission cannot be started when a receive error flag (ORER, FER, or RER) is set to 1, even if the TDRE flag is cleared to 0. Be sure to clear the receive error flags to 0 before starting transmission. Note also that the receive error flags cannot be cleared to 0 even if the RE bit is cleared to 0.

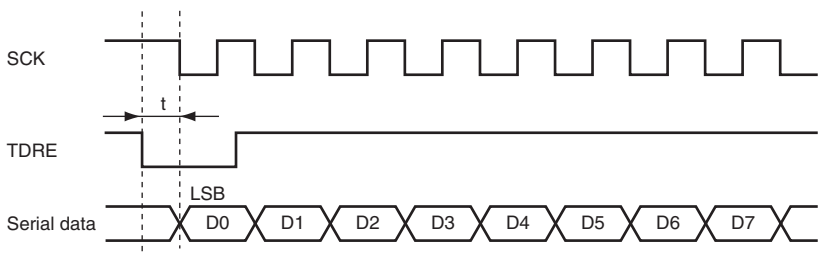
12.9.5 Relation between Writing to TDR and TDRE Flag

The TDRE flag in SSR is a status flag which indicates that transmit data has been transferred from TDR to TSR. When the SCI transfers data from TDR to TSR, the TDRE flag is set to 1.

Data can be written to TDR irrespective of the TDRE flag status. However, if new data is written to TDR when the TDRE flag is 0, that is, when the previous data has not been transferred to TSR yet, the previous data in TDR is lost. Be sure to write transmit data to TDR after verifying that the TDRE flag is set to 1.

12.9.6 Restrictions on Using DMAC

- When the external clock source is used as a synchronization clock, update TDR by the DMAC and wait for at least five Pϕ clock cycles before allowing the transmit clock to be input. If the transmit clock is input within four clock cycles after TDR modification, the SCI may malfunction (figure 12.33).
- When using the DMAC to read RDR, be sure to set the receive end interrupt (RXI) as the DMAC activation source.



Note: When external clock is supplied, t must be more than four clock cycles.

Figure 12.33 Sample Transmission using DMAC in Clocked Synchronous Mode

12.9.7 SCI Operations during Mode Transitions

(1) Transmission

Before making the transition to module stop mode or software standby mode, stop the transmit operations (TE = TIE = TEIE = 0). TSR, TDR, and SSR are reset. The states of the output pins during module stop mode or software standby mode depend on the port settings, and the pins output a high-level signal after mode cancellation. If the transition is made during data transmission, the data being transmitted will be undefined.

To transmit data in the same transmission mode after mode cancellation, set the TE bit to 1, read SSR, write to TDR, clear TDRE in this order, and then start transmission. To transmit data in a different transmission mode, initialize the SCI first.

Figure 12.34 shows a sample flowchart for mode transition during transmission. Figures 12.35 and 12.36 show the port pin states during mode transition.

(2) Reception

Before making the transition to module stop mode or software standby mode, stop the receive operations (RE = 0). RSR, RDR, and SSR are reset. If transition is made during data reception, the data being received will be invalid.

To receive data in the same reception mode after mode cancellation, set the RE bit to 1, and then start reception. To receive data in a different reception mode, initialize the SCI first.

Figure 12.37 shows a sample flowchart for mode transition during reception.

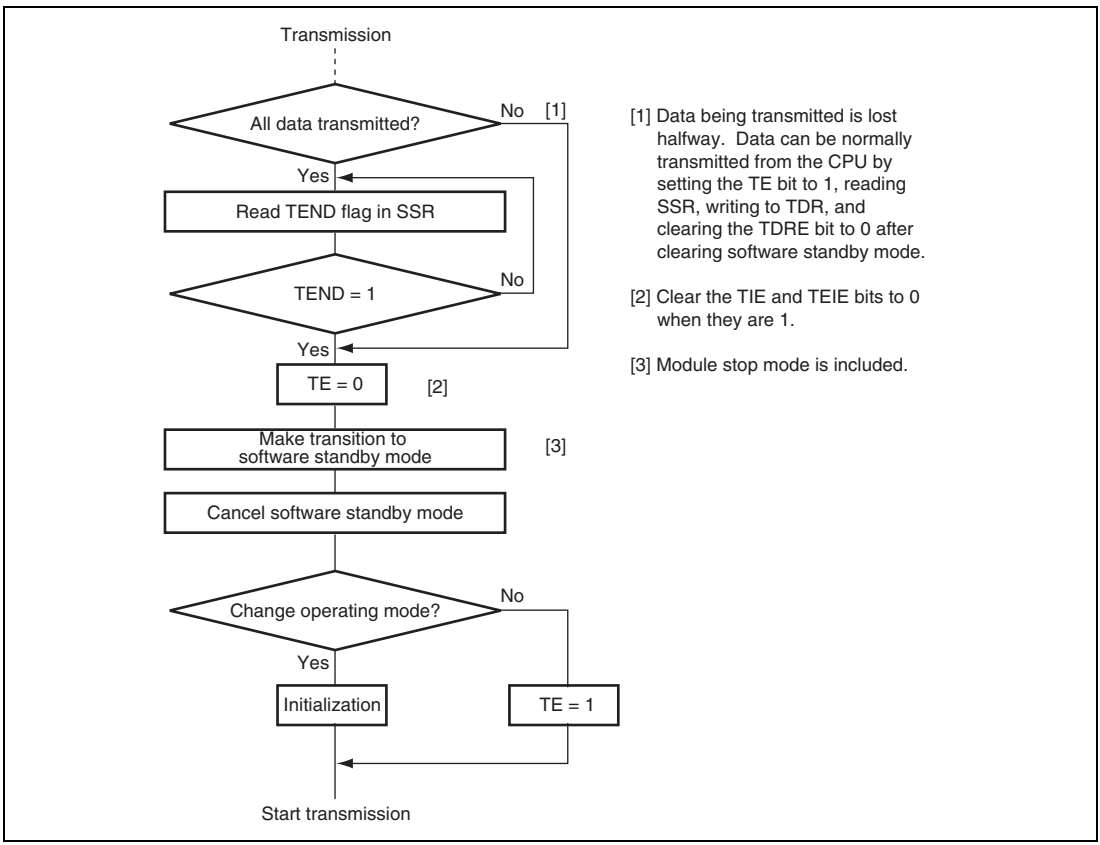


Figure 12.34 Sample Flowchart for Mode Transition during Transmission

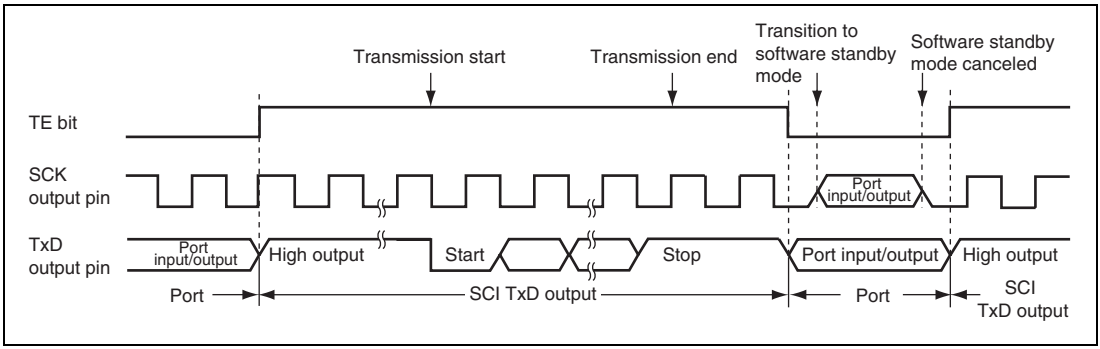


Figure 12.35 Port Pin States during Mode Transition (Internal Clock, Asynchronous Transmission)

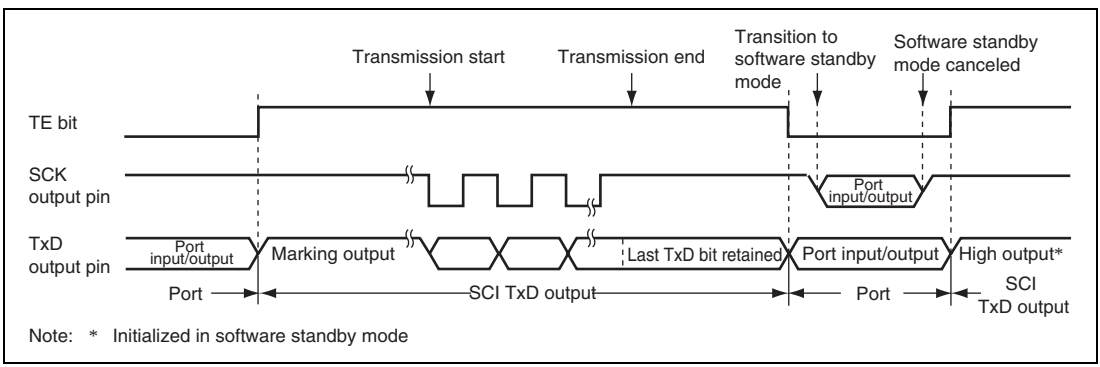


Figure 12.36 Port Pin States during Mode Transition (Internal Clock, Clocked Synchronous Transmission)

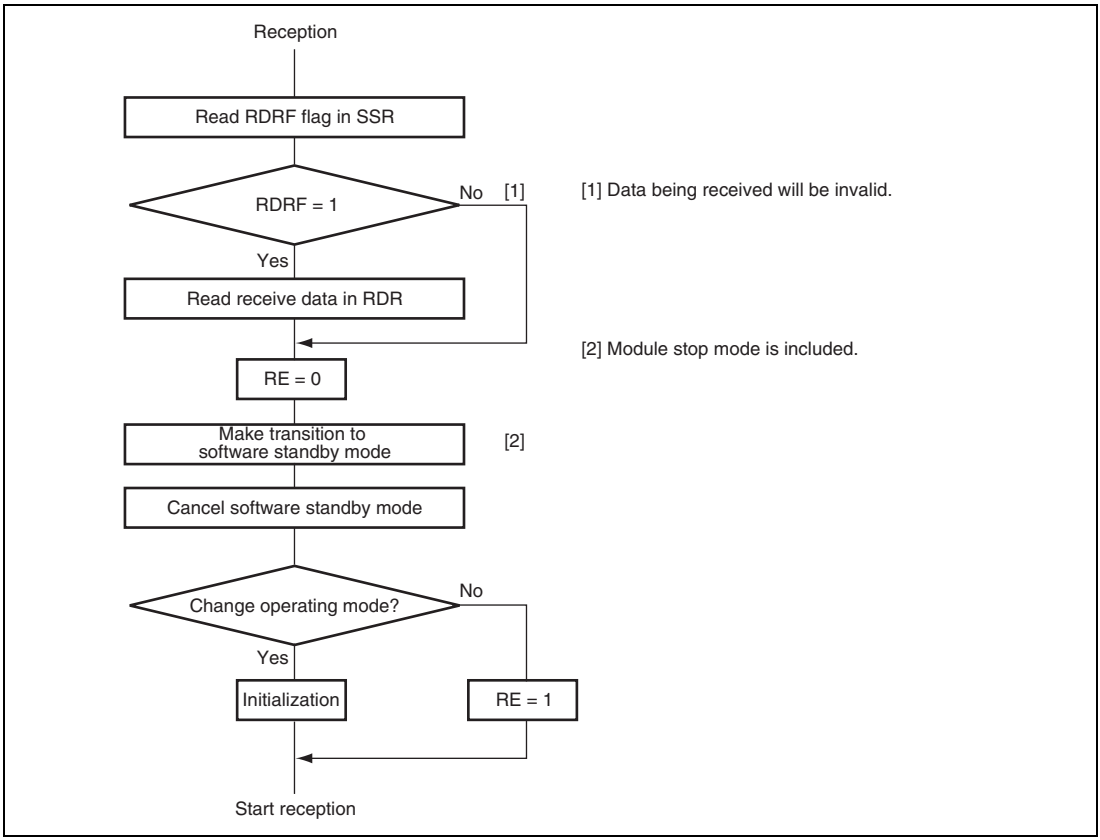


Figure 12.37 Sample Flowchart for Mode Transition during Reception

Section 13 Controller Area Network (RCAN-ET)

13.1 Summary

13.1.1 Overview

This document primarily describes the programming interface for the RCAN-ET module. It serves to facilitate the hardware/software interface so that engineers involved in the RCAN-ET implementation can ensure the design is successful.

13.1.2 Scope

The CAN Data Link Controller function is not described in this document. It is the responsibility of the reader to investigate the CAN Specification Document (see references). The interfaces from the CAN Controller are described, in so far as they pertain to the connection with the User Interface.

The programming model is described in some detail. It is not the intention of this document to describe the implementation of the programming interface, but to simply present the interface to the underlying CAN functionality.

The document places no constraints upon the implementation of the RCAN-ET module in terms of process, packaging or power supply criteria. These issues are resolved where appropriate in implementation specifications.

13.1.3 Audience

In particular this document provides the design reference for software authors who are responsible for creating a CAN application using this module.

In the creation of the RCAN-ET user interface LSI engineers must use this document to understand the hardware requirements.

13.1.4 References

1. CAN Specification Version 2.0 part A, Robert Bosch GmbH, 1991
2. CAN Specification Version 2.0 part B, Robert Bosch GmbH, 1991
3. Implementation Guide for the CAN Protocol, CAN Specification 2.0 Addendum, CAN In Automation, Erlangen, Germany, 1997
4. Road vehicles - Controller area network (CAN): Part 1: Data link layer and physical signalling (ISO-11898-1, 2003)

13.1.5 Features

- Supports CAN specification 2.0B
- Bit timing compliant with ISO-11898-1
- 16 Mailbox version
- 15 programmable Mailboxes for transmit/receive + 1 receive-only mailbox
- Sleep mode for low power consumption and automatic recovery from sleep mode by detecting CAN bus activity
- Programmable receive filter mask (standard and extended identifier) supported by all Mailboxes
- Programmable CAN data rate up to 1MBit/s
- Transmit message queuing with internal priority sorting mechanism against the problem of priority inversion for real-time applications
- Data buffer access without SW handshake requirement in reception
- Flexible micro-controller interface
- Flexible interrupt structure

13.2 Architecture

13.2.1 Block Diagram

The RCAN-ET device offers a flexible and sophisticated way to organize and control CAN frames, providing the compliance to CAN2.0B Active and ISO-11898-1. The module is formed from 5 different functional entities. These are the Micro Processor Interface (MPI), Mailbox, Mailbox Control and CAN Interface. The figure below shows the block diagram of the RCAN-ET Module. The bus interface timing is designed according to the peripheral bus I/F required for each product.

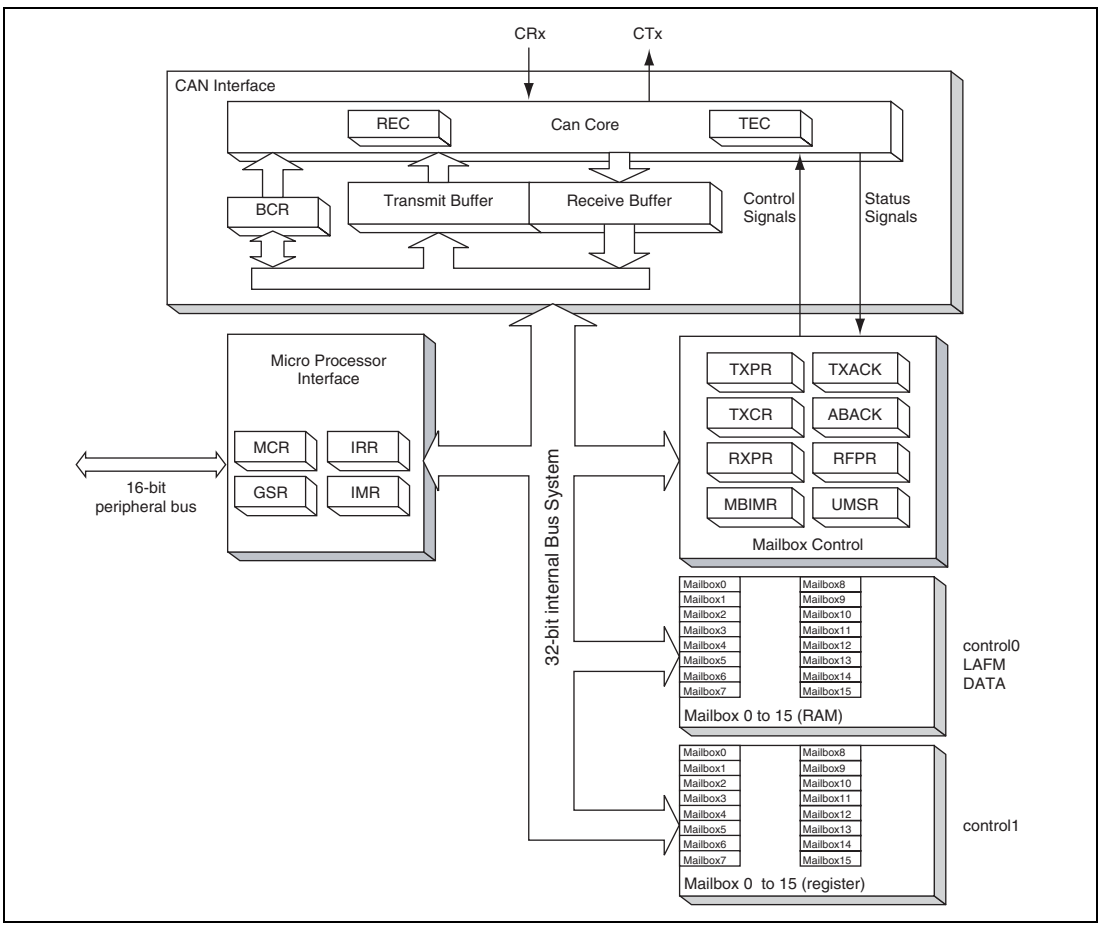


Figure 13.1 RCAN-ET Architecture

13.2.2 Important

Although core of RCAN-ET is designed based on a 32-bit bus system, the whole RCAN-ET including MPI for the CPU has 16-bit bus interface to CPU. In that case, LongWord (32-bit) access must be implemented as 2 consecutive word (16-bit) accesses. In this manual, LongWord access means the two consecutive accesses.

(1) Micro Processor Interface (MPI)

The MPI allows communication between the Renesas CPU and RCAN-ET's registers/mailboxes to control the memory interface. It also contains the Wakeup Control logic that detects the CAN bus activities and notifies the MPI and the other parts of RCAN-ET so that the RCAN-ET can automatically exit the Sleep mode.

It contains registers such as MCR, IRR, GSR and IMR.

(2) Mailbox

The Mailboxes consists of RAM configured as message buffers and registers. There are 16 Mailboxes, and each mailbox has the following information.

<RAM>

- CAN message control (identifier, rtr, ide,etc)
- CAN message data (for CAN Data frames)
- Local Acceptance Filter Mask for reception

<Registers>

- CAN message control (dlc)
- 3-bit wide Mailbox Configuration, Disable Automatic Re-Transmission bit, Auto-Transmission for Remote Request bit, New Message Control bit

(3) Mailbox Control

The Mailbox Control handles the following functions:

- For received messages, compare the IDs and generate appropriate RAM addresses/data to store messages from the CAN Interface into the Mailbox and set/clear appropriate registers accordingly.
- To transmit messages, RCAN-ET will run the internal arbitration to pick the correct priority message, and load the message from the Mailbox into the Tx-buffer of the CAN Interface and set/clear appropriate registers accordingly.
- Arbitrates Mailbox accesses between the CPU and the Mailbox Control.
- Contains registers such as TXPR, TXCR, TXACK, ABACK, RXPR, RFPR, UMSR and MBIMR.

(4) CAN Interface

This block conforms to the requirements for a CAN Bus Data Link Controller which is specified in Ref. [2, 4]. It fulfils all the functions of the Data Link Controller as specified by the OSI model. This functional entity also provides the registers and the logic which are specific to a given CAN bus, which includes the Receive Error Counter, Transmit Error Counter, the Bit Configuration Registers and various useful Test Modes. This block also contains functional entities to hold the data received and the data to be transmitted for the CAN Data Link Controller.

13.2.3 Input/Output Pins

The table below lists the RCAN-ET input and output pins.

Channel	Name	Abbreviation	I/O	Function
0	Transmit data pin	CTx_0	Output	CAN-bus transmit pin
	Receive data pin	CRx_0	Input	CAN-bus receive pin

13.2.4 Memory Map

The diagram of the memory map is shown below.

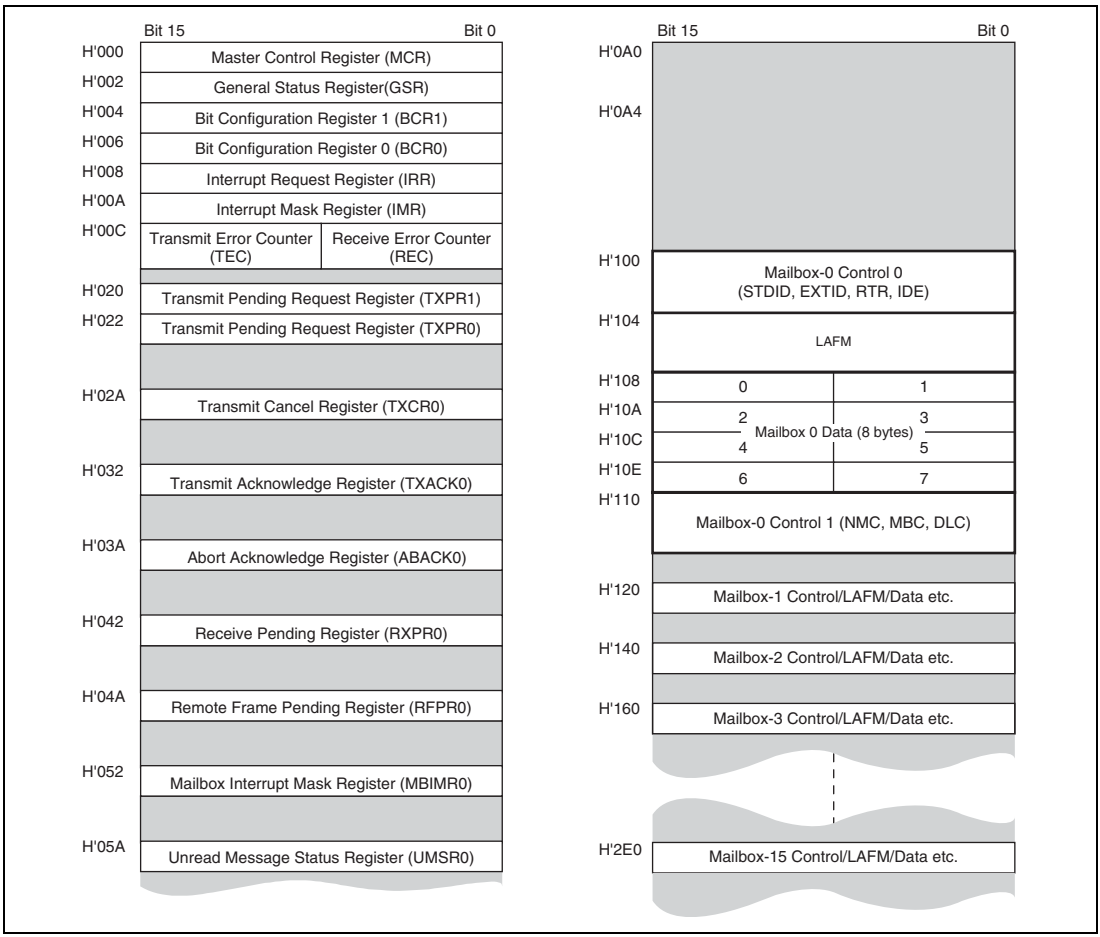


Figure 13.2 RCAN-ET Memory Map

The locations not used (between H'000 and H'2F2) are reserved and cannot be accessed.

13.3 Mailbox

13.3.1 Mailbox Structure

Mailboxes play a role as message buffers to transmit/receive CAN frames. Each Mailbox is comprised of 3 identical storage fields that are 1): Message Control, 2): Local Acceptance Filter Mask, 3): Message Data. The following table shows the address map for the control, LAFM, data and addresses for each mailbox.

Table 13.1 Address map for Each Meailbox

	Address			
	Control0	**LAFM**	**Data**	**Control1**
Mailbox	**4 bytes**	**4 bytes**	**8 bytes**	**2 bytes**
0 (Receive Only)	100 – 103	104– 107	108 – 10F	110 – 111
1	120 – 123	124 – 127	128 – 12F	130 – 131
2	140 – 143	144 – 147	148 – 14F	150 – 151
3	160 – 163	164 - 167	168 – 16F	170 – 171
4	180 – 183	184 – 187	188 – 18F	190 – 191
5	1A0 – 1A3	1A4 – 1A7	1A8 – 1AF	1B0 – 1B1
6	1C0 – 1C3	1C4 – 1C7	1C8 – 1CF	1D0 – 1D1
7	1E0 – 1E3	1E4 – 1E7	1E8 – 1EF	1F0 – 1F1
8	200 – 203	204 – 207	208 – 20F	210 – 211
9	220 – 223	224 – 227	228 – 22F	230 – 231
10	240 – 243	244 – 247	248 – 24F	250 – 251
11	260 – 263	264 – 267	268 – 26F	270 – 271
12	280 – 283	284 – 287	288 – 28F	290 – 291
13	2A0 – 2A3	2A4 – 2A7	2A8 – 2AF	2B0 – 2B1
14	2C0 – 2C3	2C4 – 2C7	2C8 – 2CF	2D0 – 2D1
15	2E0 – 2E3	2E4 – 2E7	2E8 – 2EF	2F0 – 2F1

Mailbox-0 is a receive-only box, and all the other Mailboxes can operate as both receive and transmit boxes, dependant upon the MBC (Mailbox Configuration) bits in the Message Control. The following diagram shows the structure of a Mailbox in detail.

Table 13.2 Roles of Mailboxes

	Tx	Rx
MB15 to MB 1	OK	OK
MB0	—	OK

MB0 (reception MB)

Byte: 8-bit access, Word: 16-bit access, LW (LongWord): 32-bit access

Regiter Name	Address	Data Bus																Access Size	Field Name
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
MB[0].CONTROL0H	H'100 + N*32	IDE	RTR	0	STDID[10:0]											EXTID[17:16]		Word/LW	Control 0
MB[0].CONTROL0L	H'102 + N*32	EXTID[15:0]																Word	
MB[0].LAFMH	H'104 + N*32	IDE_LAFM	0	0	STDID_LAFM[10:0]											EXTID_LAFM[17:16]		Word/LW	LAFM
MB[0].LAFML	H'106 + N*32	EXTID_LAFM[15:0]																Word	
MB[0].MSG_DATA[0][1]	H'108 + N*32	MSG_DATA_0 (first Rx/Tx Byte)							MSG_DATA_1									Byte/Word/LW	Data
MB[0].MSG_DATA[2][3]	H'10A + N*32	MSG_DATA_2							MSG_DATA_3									Byte/Word	
MB[0].MSG_DATA[4][5]	H'10C + N*32	MSG_DATA_4							MSG_DATA_5									Byte/Word/LW	
MB[0].MSG_DATA[6][7]	H'10E + N*32	MSG_DATA_6							MSG_DATA_7									Byte/Word	
MB[0].CONTROL1H, L	H'110 + N*32	0	0	NMC	0	0	MBC[2:0]			0	0	0	0	DLC[3:0]				Byte/Word	Control 1

MB1 to 15 (MB for transmission/reception)

Register Name	Address	Data Bus																Access Size	Field Name
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
MB[x].CONTROL0H	H'100 + N*32	IDE	RTR	0	STDID[10:0]											EXTID[17:16]		Word/LW	Control 0
MB[x].CONTROL0L	H'102 + N*32	EXTID[15:0]																Word	
MB[x].LAFMH	H'104 + N*32	IDE_LAFM	0	0	STDID_LAFM[10:0]											EXTID_LAFM[17:16]		Word/LW	LAFM
MB[x].LAFML	H'106 + N*32	EXTID_LAFM[15:0]																Word	
MB[x].MSG_DATA[0][1]	H'108 + N*32	MSG_DATA_0 (first Rx/Tx Byte)							MSG_DATA_1									Byte/Word/LW	Data
MB[x].MSG_DATA[2][3]	H'10A + N*32	MSG_DATA_2							MSG_DATA_3									Byte/Word	
MB[x].MSG_DATA[4][5]	H'10C + N*32	MSG_DATA_4							MSG_DATA_5									Byte/Word/LW	
MB[x].MSG_DATA[6][7]	H'10E + N*32	MSG_DATA_6							MSG_DATA_7									Byte/Word	
MB[x].CONTROL1H, L	H'110 + N*32	0	0	NMC	ATX	DART	MBC[2:0]			0	0	0	0	DLC[3:0]				Byte/Word	Control 1

Notes: 1. All bits shadowed in grey are reserved and must be written LOW. The value returned by a read may not always be '0' and should not be relied upon.
2. MBC[1] is fixed to "1".
3. ATX and DART are not supported by Mailbox-0, and the MBC setting of Mailbox-0 is limited.
4. Since the initial value of the MCR15 bit is 1, the order of the message control and the bits STDID, RTR, IDE and EXTID in LAFM difffers from that of HCAN2.

Figure 13.3 Mailbox-N Structure

13.3.2 Message Control Field

STDID[10:0]: These bits set the identifier (standard identifier) of data frames and remote frames.

EXTID[17:0]: These bits set the identifier (extended identifier) of data frames and remote frames.

RTR (Remote Transmission Request bit): Used to distinguish between data frames and remote frames. This bit is overwritten by received CAN Frames depending on Data Frames or Remote Frames.

Important: Please note that, when ATX bit is set with the setting MBC=001(bin), the RTR bit will never be set. When a Remote Frame is received, the CPU can be notified by the corresponding RFPR set or IRR[2] (Remote Frame Request Interrupt), however, as RCAN-ET needs to transmit the current message as a Data Frame, the RTR bit remains unchanged.

Important: In order to support automatic answer to remote frame when MBC=001(bin) is used and ATX=1, the RTR flag must be programmed to zero to allow data frame to be transmitted.

Note: When a Mailbox is configured to send a remote frame request, the DLC used for transmission is the one stored into the Mailbox.

RTR	Description
0	Data frame
1	Remote frame

IDE (Identifier Extension bit): Used to distinguish between the standard format and extended format of CAN data frames and remote frames.

IDE	Description
0	Standard format
1	Extended format

- Mailbox-0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	NMC	0	0	MBC[2:0]			0	0	0	0	DLC[3:0]			
Initial value:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R	R	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

Note: MBC[1] of MB0 is always "1".

- Mailbox-15 to 1

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	NMC	ATX	DART	MBC[2:0]			0	0	0	0	DLC[3:0]			
Initial value:	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

NMC (New Message Control): When this bit is set to '0', the Mailbox of which the RXPR or RFPR bit is already set does not store the new message but maintains the old one and sets the UMSR correspondent bit. When this bit is set to '1', the Mailbox of which the RXPR or RFPR bit is already set overwrites with the new message and sets the UMSR correspondent bit.

Important: Please note that if a remote frame is overwritten with a data frame or vice versa could be that both RXPR and RFPR flags (together with UMSR) are set for the same Mailbox. In this case the RTR bit within the Mailbox Control Field should be relied upon.

NMC	Description
0	Overrun mode (Initial value)
1	Overwrite mode

ATX (Automatic Transmission of Data Frame): When this bit is set to '1' and a Remote Frame is received into the Mailbox DLC is stored. Then, a Data Frame is transmitted from the same Mailbox using the current contents of the message data and updated DLC by setting the corresponding TXPR automatically. The scheduling of transmission is still governed by ID priority or Mailbox priority as configured with the Message Transmission Priority control bit (MCR.2). In order to use this function, MBC[2:0] needs to be programmed to be '001' (Bin). When a transmission is performed by this function, the DLC (Data Length Code) to be used is the one that has been received. Application needs to guarantee that the DLC of the remote frame correspond to the DLC of the data frame requested.

Important: When ATX is used and MBC=001 (Bin) the filter for the IDE bit cannot be used since ID of remote frame has to be exactly the same as that of data frame as the reply message.

Important: Please note that, when this function is used, the RTR bit will never be set despite receiving a Remote Frame. When a Remote Frame is received, the CPU will be notified by the corresponding RFPR set, however, as RCAN-ET needs to transmit the current message as a Data Frame, the RTR bit remains unchanged.

Important: Please note that in case of overrun condition (UMSR flag set when the Mailbox has its NMC=0), the message received is discarded. In case a remote frame is causing overrun into a Mailbox configured with ATX=1, a request to automatically transmit the corresponding message may be accepted.

ATX	Description
0	Automatic Transmission of Data Frame disabled (Initial value)
1	Automatic Transmission of Data Frame enabled

DART (Disable Automatic Re-Transmission): When this bit is set, it disables the automatic re-transmission of a message in the event of an error on the CAN bus or an arbitration lost on the CAN bus. In effect, when this function is used, the corresponding TXCR bit is automatically set at the start of transmission. When this bit is set to '0', RCAN-ET tries to transmit the message as many times as required until it is successfully transmitted or it is cancelled by the TXCR.

DART	Description
0	Re-transmission enabled (Initial value)
1	Re-Transmission disabled

MBC[2:0] (Mailbox Configuration): These bits configure the nature of each Mailbox as follows. When MBC=111 (Bin), the Mailbox is inactive, i.e., it does not receive or transmit a message regardless of TXPR or other settings. The MBC='110', '101' and '100' settings are prohibited. When the MBC is set to any other value, the LAFM field becomes available. Please don't set TXPR when MBC is set as reception. There is no hardware protection, and TXPR remains set. MBC[1] of Mailbox-0 is fixed to "1" by hardware.

Table 13.3 Mailbox Function Setting

MBC[2]	MBC[1]	MBC[0]	Data Frame Transmit	Remote Frame Transmit	Data Frame Receive	Remote Frame Receive	Remarks
0	0	0	Yes	Yes	No	No	• Not allowed for Mailbox-0
0	0	1	Yes	Yes	No	Yes	• Can be used with ATX* • Not allowed for Mailbox-0 • LAFM can be used
0	1	0	No	No	Yes	Yes	• Allowed for Mailbox-0 • LAFM can be used
0	1	1	No	No	Yes	No	• Allowed for Mailbox-0 • LAFM can be used
1	0	0	Setting prohibited				
1	0	1	Setting prohibited				
1	1	0	Setting prohibited				
1	1	1	Mailbox inactive (Initial value)				

Notes: * In order to support automatic retransmission, RTR shall be "0" when MBC=001 (bin) and ATX=1.
When ATX=1 is used, the filter for IDE must not be used

DLC[3:0] (Data Length Code): These bits encode the number of data bytes from 0, 1, 2, … 8 that will be transmitted in a data frame. Please note that when a remote frame request is transmitted the DLC value to be used must be the same as the DLC of the data frame that is requested.

DLC[3]	DLC[2]	DLC[1]	DLC[0]	Description
0	0	0	0	Data Length = 0 bytes (Initial value)
0	0	0	1	Data Length = 1 byte
0	0	1	0	Data Length = 2 bytes
0	0	1	1	Data Length = 3 bytes
0	1	0	0	Data Length = 4 bytes
0	1	0	1	Data Length = 5 bytes
0	1	1	0	Data Length = 6 bytes
0	1	1	1	Data Length = 7 bytes
1	x	x	x	Data Length = 8 bytes

13.3.3 Local Acceptance Filter Mask (LAFM)

When MBC is set to 001, 010, 011 (Bin), this field is used as LAFM Field. The LAFM allows a Mailbox to accept more than one identifier. The LAFM is comprised of two 16-bit read/write areas as shown figure 13.4.

Register Name	Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Acces Size	Feld Name
MB[0].LAFMH	H'104 + N*32	IDE_ LAFM	0	0	STDID_LAFM[10:0]											EXTID_ LAFM[17:16]		Word/LW	LAFM Field
MB[0].LAFML	H'106 + N*32	EXTID_LAFM[15:0]																Word	

Figure 13.4 Acceptance Filter

If a bit is set in the LAFM, then the corresponding bit of a received CAN identifier is ignored when the RCAN-ET searches a Mailbox with the matching CAN identifier. If the bit is cleared, then the corresponding bit of a received CAN identifier must match to the STDID/IDE/EXTID set in the mailbox to be stored. The structure of the LAFM is same as the message control in a Mailbox. If this function is not required, it must be filled with '0'.

Important: RCAN-ET starts to find a matching identifier from Mailbox-15 down to Mailbox-0. As soon as RCAN-ET finds one matching, it stops the search. The message will be stored or not depending on the NMC and RXPR/RFPR flags. This means that, even using LAFM, a received message can only be stored into 1 Mailbox.

Important: When a message is received and a matching Mailbox is found, the whole message is stored into the Mailbox. This means that, if the LAFM is used, the STDID, RTR, IDE and EXTID may differ to the ones originally set as they are updated with the STDID, RTR, IDE and EXTID of the received message.

STD_LAFM[10:0] — Filter mask bits for the CAN base identifier [10:0] bits.

STD_LAFM[10:0]	Description
0	Corresponding STD_ID bit is cared
1	Corresponding STD_ID bit is "don't cared"

EXT_LAFM[17:0] — Filter mask bits for the CAN Extended identifier [17:0] bits.

EXT_LAFM[17:0]	Description
0	Corresponding EXT_ID bit is cared
1	Corresponding EXT_ID bit is "don't cared"

IDE_LAFM — Filter mask bit for the CAN IDE bit.

IDE_LAFM	Description
0	Corresponding IDE_ID bit is cared
1	Corresponding IDE_ID bit is "don't cared"

13.3.4 Message Data Fields

Storage for the CAN message data that is transmitted or received. MSG_DATA[0] corresponds to the first data byte that is transmitted or received. The bit order on the CAN bus is bit 7 through to bit 0.

13.4 RCAN-ET Control Registers

The following sections describe RCAN-ET control registers. The address is mapped as follow.

Important: These registers can only be accessed in Word size (16-bit).

Table 13.4 RCAN-ET Control Registers Configuration

Description	Address	Name	Access Size (bits)
Master Control Register	000	MCR	Word
General Status Register	002	GSR	Word
Bit Configuration Register 1	004	BCR1	Word
Bit Configuration Register 0	006	BCR0	Word
Interrupt Request Register	008	IRR	Word
Interrupt Mask Register	00A	IMR	Word
Transmit Error Counter/ Receive Error Counter	00C	TEC/REC	Word

13.4.1 Master Control Register (MCR)

The Master Control Register (MCR) is a 16-bit read/write register that controls RCAN-ET.

- MCR (Address = H'000)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MCR15	MCR14	—	—	—	TST[2:0]			MCR7	MCR6	MCR5	—	—	MCR2	MCR1	MCR0
Initial value:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R/W:	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W

Bit 15 — ID Reorder (MCR15): This bit changes the order of STDID, RTR, IDE and EXTID of both message control and LAFM.

Bit15: MCR15	Description
0	RCAN-ET is the same as HCAN2
1	RCAN-ET is not the same as HCAN2 (Initial value)

MCR15 (ID Reorder) = 0

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Feld Name
H'100 + N*32	0	STDID[10:0]											RTR	IDE	EXTID[17:16]		Word/LW	Control 0
H'102 + N*32	EXTID[15:0]																Word	
H'104 + N*32	0	STDID_LAFM[10:0]											0	IDE_ LAFM	EXTID_LAFM [17:16]		Word/LW	LAFM Field
H'106 + N*32	EXTID_LAFM[15:0]																Word	

MCR15 (ID Reorder) = 1

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Access Size	Feld Name
H'100 + N*32	IDE	RTR	0	STDID[10:0]											EXTID[17:16]		Word/LW	Control 0
H'102 + N*32	EXTID[15:0]																Word	
H'104 + N*32	IDE_ LAFM	0	0	STDID_LAFM[10:0]											EXTID_LAFM [17:16]		Word/LW	LAFM Field
H'106 + N*32	EXTID_LAFM[15:0]																Word	

Figure 13.5 ID Reorder

This bit can be modified only in reset mode.

Bit 14 — Auto Halt Bus Off (MCR14): If both this bit and MCR6 are set, MCR1 is automatically set as soon as RCAN-ET enters Bus Off.

Bit14: MCR14	Description
0	RCAN-ET remains in Bus Off for normal recovery sequence (128 x 11 Recessive Bits) (Initial value)
1	RCAN-ET moves directly into Halt Mode after it enters Bus Off if MCR6 is set.

This bit can be modified only in reset mode.

Bit 13 — Reserved. The written value should always be '0' and the returned value is '0'.

Bit 12 — Reserved. The written value should always be '0' and the returned value is '0'.

Bit 11 — Reserved. The written value should always be '0' and the returned value is '0'.

Bit 10 - 8 — Test Mode (TST[2:0]): This bit enables/disables the test modes. Please note that before activating the Test Mode it is requested to move RCAN-ET into Halt mode or Reset mode. This is to avoid that the transition to Test Mode could affect a transmission/reception in progress. For details, please refer to section 13.6.2, Test Mode Settings.

Please note that the test modes are allowed only for diagnosis and tests and not when RCAN-ET is used in normal operation.

Bit10: TST2	Bit9: TST1	Bit8: TST0	Description
0	0	0	Normal Mode (initial value)
0	0	1	Listen-Only Mode (Receive-Only Mode)
0	1	0	Self Test Mode 1 (External)
0	1	1	Self Test Mode 2 (Internal)
1	0	0	Write Error Counter
1	0	1	Error Passive Mode
1	1	0	Setting prohibited
1	1	1	Setting prohibited

Bit 7 — Auto-wake Mode (MCR7): MCR7 enables or disables the Auto-wake mode. If this bit is set, the RCAN-ET automatically cancels the sleep mode (MCR5) by detecting CAN bus activity (dominant bit). If MCR7 is cleared the RCAN-ET does not automatically cancel the sleep mode.

RCAN-ET cannot store the message that wakes it up.

Note: MCR7 cannot be modified while in sleep mode.

Bit7: MCR7	Description
0	Auto-wake by CAN bus activity disabled (Initial value)
1	Auto-wake by CAN bus activity enabled

Bit 6 — Halt during Bus Off (MCR6): MCR6 enables or disables entering Halt mode immediately when MCR1 is set during Bus Off. This bit can be modified only in Reset or Halt mode. Please note that when Halt is entered in Bus Off the CAN engine is also recovering immediately to Error Active mode.

Bit6: MCR6	Description
0	If MCR[1] is set, RCAN-ET will not enter Halt mode during Bus Off but wait up to end of recovery sequence. (initial value)
1	Enter Halt mode immediately during Bus Off if MCR[1] or MCR[14] are asserted.

Bit 5 — Sleep Mode (MCR5): Enables or disables Sleep mode transition. If this bit is set, while RCAN-ET is in halt mode, the transition to sleep mode is enabled. Setting MCR5 is allowed after entering Halt mode. The two Error Counters (REC, TEC) will remain the same during Sleep mode. This mode will be exited in two ways:

1. by writing a '0' to this bit position,
2. or, if MCR[7] is enabled, after detecting a dominant bit on the CAN bus.

If Auto wake up mode is disabled, RCAN-ET will ignore all CAN bus activities until the sleep mode is terminated. When leaving this mode the RCAN-ET will synchronise to the CAN bus (by checking for 11 recessive bits) before joining CAN Bus activity. This means that, when the No.2 method is used, RCAN-ET will miss the first message to receive. CAN transceivers stand-by mode will also be unable to cope with the first message when exiting stand by mode, and the S/W needs to be designed in this manner.

In sleep mode only the following registers can be accessed: MCR, GSR, IRR and IMR.

Important: RCAN-ET is required to be in Halt mode before requesting to enter in Sleep mode. That allows the CPU to clear all pending interrupts before entering sleep mode. Once all interrupts are cleared RCAN-ET must leave the Halt mode and enter Sleep mode simultaneously (by writing MCR[5]=1 and MCR[1]=0 at the same time).

Bit 5: MCR5	Description
0	RCAN-ET sleep mode released (Initial value)
1	Transition to RCAN-ET sleep mode enabled

Bit 4 — Reserved. The written value should always be '0' and the returned value is '0'.

Bit 3 — Reserved. The written value should always be '0' and the returned value is '0'.

Bit 2 — Message Transmission Priority (MCR2): MCR2 selects the order of transmission for pending transmit data. If this bit is set, pending transmit data are sent in order of the bit position in the Transmission Pending Register (TXPR). The order of transmission starts from Mailbox-15 as the highest priority, and then down to Mailbox-1 (if those mailboxes are configured for transmission).

If MCR2 is cleared, all messages for transmission are queued with respect to their priority (by running internal arbitration). The highest priority message has the Arbitration Field (STDID + IDE bit + EXTID (if IDE=1) + RTR bit) with the lowest digital value and is transmitted first. The internal arbitration includes the RTR bit and the IDE bit (internal arbitration works in the same way as the arbitration on the CAN Bus between two CAN nodes starting transmission at the same time).

This bit can be modified only in Reset or Halt mode.

Bit 2: MCR2	Description
0	Transmission order determined by message identifier priority (Initial value)
1	Transmission order determined by mailbox number priority (Mailbox-15 → Mailbox-1)

Bit 1—Halt Request (MCR1): Setting the MCR1 bit causes the CAN controller to complete its current operation and then enter Halt mode (where it is cut off from the CAN bus). The RCAN-ET remains in Halt Mode until the MCR1 is cleared. During the Halt mode, the CAN Interface does not join the CAN bus activity and does not store messages or transmit messages. All the user registers (including Mailbox contents and TEC/REC) remain unchanged with the exception of IRR0 and GSR4 which are used to notify the halt status itself. If the CAN bus is in idle or intermission state regardless of MCR6, RCAN-ET will enter Halt Mode within one Bit Time. If MCR6 is set, a halt request during Bus Off will be also processed within one Bit Time. Otherwise the full Bus Off recovery sequence will be performed beforehand. Entering the Halt Mode can be notified by IRR0 and GSR4.

If both MCR14 and MCR6 are set, MCR1 is automatically set as soon as RCAN-ET enters Bus Off.

In the Halt mode, the RCAN-ET configuration can be modified with the exception of the Bit Timing setting, as it does not join the bus activity. MCR[1] has to be cleared by writing a '0' in order to re-join the CAN bus. After this bit has been cleared, RCAN-ET waits until it detects 11 recessive bits, and then joins the CAN bus.

Note: After issuing a Halt request the CPU is not allowed to set TXPR or TXCR or clear MCR1 until the transition to Halt mode is completed (notified by IRR0 and GSR4). After MCR1 is set this can be cleared only after entering Halt mode or through a reset operation (SW or HW).

Note: Transition into or recovery from HALT mode, is only possible if the BCR1 and BCR0 registers are configured to a proper Baud Rate.

Bit 1: MCR1	Description
0	Clear Halt request (Initial value)
1	Halt mode transition request

Bit 0 — Reset Request (MCR0): Controls resetting of the RCAN-ET module. When this bit is changed from '0' to '1' the RCAN-ET controller enters its reset routine, re-initialising the internal logic, which then sets GSR3 and IRR0 to notify the reset mode. All user registers are initialised.

RCAN-ET can be re-configured while this bit is set. This bit has to be cleared by writing a '0' to join the CAN bus. After this bit is cleared, the RCAN-ET module waits until it detects 11 recessive bits, and then joins the CAN bus. The Baud Rate needs to be set up to a proper value in order to sample the value on the CAN Bus.

After Power On Reset, this bit and GSR3 are always set. This means that a reset request has been made and RCAN-ET needs to be configured.

The Reset Request is equivalent to a Power On Reset but controlled by Software.

Bit 0: MCR0	Description
0	Clear Reset request
1	CAN Interface reset mode transition request (Initial value)

13.4.2 General Status Register (GSR)

The General Status Register (GSR) is a 16-bit read-only register that indicates the status of RCAN-ET.

- GSR (Address = H'002)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	GSR5	GSR4	GSR3	GSR2	GSR1	GSR0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Bits 15 to 6: Reserved. The written value should always be '0' and the returned value is '0'.

Bit 5 — Error Passive Status Bit (GSR5): Indicates whether the CAN Interface is in Error Passive or not. This bit will be set high as soon as the RCAN-ET enters the Error Passive state and is cleared when the module enters again the Error Active state (this means the GSR5 will stay high during Error Passive and during Bus Off). Consequently to find out the correct state both GSR5 and GSR0 must be considered.

Bit 5: GSR5	Description
0	RCAN-ET is not in Error Passive or in Bus Off status (Initial value) [Reset condition] RCAN-ET is in Error Active state
1	RCAN-ET is in Error Passive (if GSR0=0) or Bus Off (if GSR0=1) [Setting condition] When TEC ≥ 128 or REC ≥ 128, or error passive test mode is selected

Bit 4 — Halt/Sleep Status Bit (GSR4): Indicates whether the CAN engine is in the halt/sleep state or not. Please note that the clearing time of this flag is not the same as the setting time of IRR12.

Please note that this flag reflects the status of the CAN engine and not of the full RCAN-ET IP. RCAN-ET exits sleep mode and can be accessed once MCR5 is cleared. The CAN engine exits sleep mode only after two additional transmission clocks on the CAN Bus.

Bit 4: GSR4	Description
0	RCAN-ET is not in the Halt state or Sleep state (Initial value)
1	Halt mode (if MCR1=1) or Sleep mode (if MCR5=1) [Setting condition] If MCR1 is set and the CAN bus is either in intermission or idle or MCR5 is set and RCAN-ET is in the halt mode or RCAN-ET is moving to Bus Off when MCR14 and MCR6 are both set

Bit 3 — Reset Status Bit (GSR3): Indicates whether the RCAN-ET is in the reset state or not.

Bit 3: GSR3	Description
0	RCAN-ET is not in the reset state
1	Reset state (Initial value) [Setting condition] After an RCAN-ET internal reset (due to SW or HW reset)

Bit 2 — Message Transmission in progress Flag (GSR2): Flag that indicates to the CPU if the RCAN-ET is in Bus Off or transmitting a message or an error/overload flag due to error detected during transmission. The timing to set TXACK is different from the time to clear GSR2. TXACK is set at the 7th bit of End Of Frame. GSR2 is set at the 3rd bit of intermission if there are no more messages ready to be transmitted. It is also set by arbitration lost, bus idle, reception, reset or halt transition.

Bit 2: GSR2	Description
0	RCAN-ET is in Bus Off or a transmission is in progress
1	[Setting condition] Not in Bus Off and no transmission in progress (Initial value)

Bit 1—Transmit/Receive Warning Flag (GSR1): Flag that indicates an error warning.

Bit 1: GSR1	Description
0	[Reset condition] When (TEC < 96 and REC < 96) or Bus Off (Initial value)
1	[Setting condition] When 96 ≤ TEC < 256 or 96 ≤ REC < 256

Note: REC is incremented during Bus Off to count the recurrences of 11 recessive bits as requested by the Bus Off recovery sequence. However the flag GSR1 is not set in Bus Off.

Bit 0—Bus Off Flag (GSR0): Flag that indicates that RCAN-ET is in the bus off state.

Bit 0: GSR0	Description
0	[Reset condition] Recovery from bus off state or after a HW or SW reset (Initial value)
1	[Setting condition] When TEC ≥ 256 (bus off state)

Note: Only the lower 8 bits of TEC are accessible from the user interface. The 9th bit is equivalent to GSR0.

13.4.3 Bit Configuration Register (BCR0, BCR1)

The bit configuration registers (BCR0 and BCR1) are 2 X 16-bit read/write register that are used to set CAN bit timing parameters and the baud rate pre-scaler for the CAN Interface.

The Time quanta is defined as:

$$\text{Time quanta} = \frac{2 * \text{BRP}}{f_{clk}}$$

Where: BRP (Baud Rate Pre-scaler) is the value stored in BCR0 incremented by 1 and fclk is the used peripheral bus frequency.

- BCR1 (Address = H'004)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSG1[3:0]				—	TSG2[2:0]			—	—	SJW[1:0]		—	—	—	BSP
Initial value:	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	R/W	R/W	R/W	R	R	R/W	R/W	R	R	R	R/W

Please refer to the table below for TSG1 and TSG2 setting.

Bits 15 to 12 — Time Segment 1 (TSG1[3:0] = BCR1[15:12]): These bits are used to set the segment TSEG1 (=PRSEG + PHSEG1) to compensate for edges on the CAN Bus with a positive phase error. A value from 4 to 16 time quanta can be set.

Bit 15: TSG1[3]	Bit 14: TSG1[2]	Bit 13: TSG1[1]	Bit 12: TSG1[0]	Description
0	0	0	0	Setting prohibited (Initial value)
0	0	0	1	Setting prohibited
0	0	1	0	Setting prohibited
0	0	1	1	PRSEG + PHSEG1 = 4 time quanta
0	1	0	0	PRSEG + PHSEG1 = 5 time quanta
: :	: :	: :	: :	: :
1	1	1	1	PRSEG + PHSEG1 = 16 time quanta

Bit 11: Reserved. The written value should always be '0' and the returned value is '0'.

Bits 10 to 8 — Time Segment 2 (TSG2[2:0] = BCR1[10:8]): These bits are used to set the segment TSEG2 (=PHSEG2) to compensate for edges on the CAN Bus with a negative phase error. A value from 2 to 8 time quanta can be set as shown below.

Bit 10: TSG2[2]	Bit 9: TSG2[1]	Bit 8: TSG2[0]	Description
0	0	0	Setting prohibited (Initial value)
0	0	1	PHSEG2 = 2 time quanta (conditionally prohibited) See the table below for TSG1 and TSG2 setting.
0	1	0	PHSEG2 = 3 time quanta
0	1	1	PHSEG2 = 4 time quanta
1	0	0	PHSEG2 = 5 time quanta
1	0	1	PHSEG2 = 6 time quanta
1	1	0	PHSEG2 = 7 time quanta
1	1	1	PHSEG2 = 8 time quanta

Bits 7 and 6: Reserved. The written value should always be '0' and the returned value is '0'.

Bits 5 and 4 - ReSynchronisation Jump Width (SJW[1:0] = BCR0[5:4]): These bits set the synchronisation jump width.

Bit 5: SJW[1]	Bit 4: SJW[0]	Description
0	0	Synchronisation Jump width = 1 time quantum (Initial value)
0	1	Synchronisation Jump width = 2 time quanta
1	0	Synchronisation Jump width = 3 time quanta
1	1	Synchronisation Jump width = 4 time quanta

Bits 3 to 1 Reserved. The written value should always be '0' and the returned value is '0'.

Bit 0 — Bit Sample Point (BSP = BCR1[0]): Sets the point at which data is sampled.

Bit 0: BSP	Description
0	Bit sampling at one point (end of time segment 1) (Initial value)
1	Bit sampling at three points (rising edge of the last three clock cycles of PHSEG1)

- BCR0 (Address = H'006)

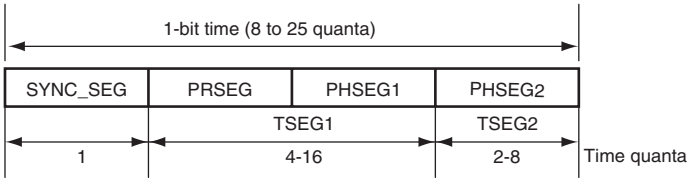
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	BRP[7:0]							
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bits 8 to 15 : Reserved. The written value should always be '0' and the returned value is '0'.

Bits 7 to 0—Baud Rate Pre-scale (BRP[7:0] = BCR0 [7:0]): These bits are used to define the peripheral bus clock periods contained in a Time Quantum.

Bit 7: BRP[7]	Bit 6: BRP[6]	Bit 5: BRP[5]	Bit 4: BRP[4]	Bit 3: BRP[3]	Bit 2: BRP[2]	Bit 1: BRP[1]	Bit 0: BRP[0]	Description
0	0	0	0	0	0	0	0	2 X peripheral bus clock (Initial value)
0	0	0	0	0	0	0	1	4 X peripheral bus clock
0	0	0	0	0	0	1	0	6 X peripheral bus clock
: :	: :	: :	: :	: :	: :	: :	: :	2*(register value+1) X peripheral bus clock
1	1	1	1	1	1	1	1	512 X peripheral bus clock

- Requirements of Bit Configuration Register



SYNC_SEG: Segment for establishing synchronisation of nodes on the CAN bus. (Normal bit edge transitions occur in this segment.)

PRSEG: Segment for compensating for physical delay between networks.

PHSEG1: Buffer segment for correcting phase drift (positive). (This segment is extended when synchronisation (resynchronisation) is established.)

PHSEG2: Buffer segment for correcting phase drift (negative). (This segment is shortened when synchronisation (resynchronisation) is established)

TSEG1: TSG1 + 1

TSEG2: TSG2 + 1

The RCAN-ET Bit Rate Calculation is:

$$\text{Bit Rate} = \text{fclk} / \{2 \times (\text{BRP} + 1) \times (\text{TSEG1} + \text{TSEG2} + 1)\}$$

where BRP is given by 2 × register value + 1, TSEG1 and TSEG2 are derived values from the descriptions of the following table. The time segment '+ 1' in the above formula is for the Sync-Seg whose duration is 1 time quanta.

f_{CLK} = Peripheral Clock

BCR Setting Constraints

$\text{TSEG1min} > \text{TSEG2} \geq \text{SJWmax}$ (SJW = 1 to 4)

$8 \leq \text{TSEG1} + \text{TSEG2} + 1 \leq 25$ time quanta (TSEG1 + TSEG2 + 1 = 7 is not allowed)

$\text{TSEG2} \geq 2$

These constraints allow the setting range shown in the table below for TSEG1 and TSEG2 in the Bit Configuration Register. The number in the table shows possible setting of SJW. "No" shows that there is no allowed combination of TSEG1 and TSEG2.

Table 13.5 TSG and TSEG Setting

		001	010	011	100	101	110	111	TSG2
		2	3	4	5	6	7	8	TSEG2
TSG1	TSEG1								
0011	4	No	1-3	No	No	No	No	No	
0100	5	1-2	1-3	1-4	No	No	No	No	
0101	6	1-2	1-3	1-4	1-4	No	No	No	
0110	7	1-2	1-3	1-4	1-4	1-4	No	No	
0111	8	1-2	1-3	1-4	1-4	1-4	1-4	No	
1000	9	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1001	10	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1010	11	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1011	12	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1100	13	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1101	14	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1110	15	1-2	1-3	1-4	1-4	1-4	1-4	1-4	
1111	16	1-2	1-3	1-4	1-4	1-4	1-4	1-4	

Example: To have a Bit rate of 250 Kbps with a frequency of fclk = 25 MHz, it is possible to set: BRP = 4, TSEG1 = 5, TSEG2 = 4. Then the configuration to write is BCR1 = 4300 and BCR0 = 0004.

Example: To have a Bit rate of 500 Kbps with a frequency of fclk = 20 MHz, it is possible to set: BRP = 1, TSEG1 = 6, TSEG2 = 3. Then the configuration to write is BCR1 = 5200 and BCR0 = 0001.

13.4.4 Interrupt Request Register (IRR)

The interrupt register (IRR) is a 16-bit read/write-clearable register containing status flags for the various interrupt sources.

- IRR (Address = H'008)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	IRR13	IRR12	—	—	IRR9	IRR8	IRR7	IRR6	IRR5	IRR4	IRR3	IRR2	IRR1	IRR0
Initial value:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R/W:	R	R	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R/W

Bits 15 to 14: Reserved.

Bit 13 - Message Error Interrupt (IRR13): this interrupt indicates that:

- A message error has occurred when in test mode.
- Note: If a Message Overload condition occurs when in Test Mode, then this bit will not be set. When not in test mode this interrupt is inactive.

Bit 13: IRR13	Description
0	Message error has not occurred in test mode (Initial value) [Clearing condition] Writing 1 (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	[Setting condition] message error has occurred in test mode

Bit 12 – Bus Activity while in Sleep Mode (IRR12): IRR12 indicates that a CAN bus activity is present. While the RCAN-ET is in sleep mode and a dominant bit is detected on the CAN bus, this bit is set. This interrupt is cleared by writing a '1' to this bit position. Writing a '0' has no effect. If auto wakeup is not used and this interrupt is not requested it needs to be disabled by the related interrupt mask register. If auto wake up is not used and this interrupt is requested it should be cleared only after recovering from sleep mode. This is to avoid that a new falling edge of the reception line causes the interrupt to get set again.

Please note that the setting time of this interrupt is different from the clearing time of GSR4.

Bit 12: IRR12	Description
0	bus idle state (Initial value) [Clearing condition] Writing 1 (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	[Setting condition] dominant bit level detection on the Rx line while in sleep mode

Bits 11 to 10: Reserved

Bit 9 – Message Overrun/Overwrite Interrupt Flag (IRR9): Flag indicating that a message has been received but the existing message in the matching Mailbox has not been read as the corresponding RXPR or RFPR is already set to '1' and not yet cleared by the CPU. Thus the received message is either abandoned (overrun) or overwritten dependant upon the NMC (New Message Control) bit. This bit is cleared when all bit in UMSR (Unread Message Status Register) are cleared (by writing '1') or by setting MBIMR (MailBox interrupt Mast Register) for all UMSR flag set . It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit position has no effect.

Bit 9: IRR9	Description
0	No pending notification of message overrun/overwrite [Clearing condition] Clearing of all bit in UMSR/setting MBIMR for all UMSR set (initial value)
1	A receive message has been discarded due to overrun condition or a message has been overwritten [Setting condition] Message is received while the corresponding RXPR and/or RFPR =1 and MBIMR =0

Bit 8 - Mailbox Empty Interrupt Flag (IRR8): This bit is set when one of the messages set for transmission has been successfully sent (corresponding TXACK flag is set) or has been successfully aborted (ABACK flag corresponding to the message whose transmission cancel has been executed is set). The related TXPR is also cleared and this mailbox is now ready to accept a new message data for the next transmission. In effect, this bit is set by an OR'ed signal of the TXACK and ABACK bits not masked by the corresponding MBIMR flag. Therefore, this bit is automatically cleared when all the TXACK and ABACK bits are cleared. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit position has no effect.

Bit 8: IRR8	Description
0	Messages set for transmission or transmission cancellation request NOT progressed. (Initial value) [Clearing Condition] All the TXACK and ABACK bits are cleared/setting MBIMR for all TXACK and ABACK set
1	Message has been transmitted or aborted (cancelled), and new message can be stored [Setting condition] When a TXACK or ABACK bit is set (if MBIMR=0).

Bit 7 - Overload Frame (IRR7): Flag indicating that the RCAN-ET has detected the transmission of an overload frame. IRR7 is cleared by writing a '1' to this bit position; writing a '0' has no effect.

Bit 7: IRR7	Description
0	[Clearing condition] Writing 1 (Initial value) (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	[Setting conditions] Overload condition detected

Bit 6 - Bus Off Interrupt Flag (IRR6): This bit is set when RCAN-ET enters the Bus-off state or when RCAN-ET leaves Bus-off and returns to Error-Active. The cause therefore is the existing condition TEC ≥ 256 at the node or the end of the Bus-off recovery sequence (128X11 consecutive recessive bits) or the transition from Bus Off to Halt (automatic or manual). This bit remains set even if the RCAN-ET node leaves the bus-off condition, and needs to be explicitly cleared by S/W. The S/W is expected to read the GSR0 to judge whether RCAN-ET is in the bus-off or error active status. It is cleared by writing a '1' to this bit position even if the node is still bus-off. Writing a '0' has no effect.

Bit 6: IRR6	Description
0	[Clearing condition] Writing 1 (Initial value) (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Enter Bus off state caused by transmit error or Error Active state returning from Bus-off [Setting condition] When TEC becomes ≥ 256 or End of Bus-off after 128X11 consecutive recessive bits or transition from Bus Off to Halt

Bit 5 - Error Passive Interrupt Flag (IRR5): Interrupt flag indicating the error passive state caused by the transmit or receive error counter or by Error Passive forced by test mode. This bit is reset by writing a '1' to this bit position, writing a '0' has no effect. If this bit is cleared the node may still be error passive. Please note that the SW needs to check GSR0 and GSR5 to judge whether RCAN-ET is in Error Passive or Bus Off status.

Bit 5: IRR5	Description
0	[Clearing condition] Writing 1 (Initial value) (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error passive state caused by transmit/receive error [Setting condition] When TEC ≥ 128 or REC ≥ 128 or Error Passive test mode is used

Bit 4 - Receive Error Counter Warning Interrupt Flag (IRR4): This bit becomes set if the receive error counter (REC) reaches a value greater than 95 when RCAN-ET is not in the Bus Off status. The interrupt is reset by writing a '1' to this bit position, writing '0' has no effect.

Bit 4: IRR4	Description
0	[Clearing condition] Writing 1 (Initial value) (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error warning state caused by receive error [Setting condition] When REC ≥ 96 and RCAN-ET is not in Bus Off

Bit 3 - Transmit Error Counter Warning Interrupt Flag (IRR3): This bit becomes set if the transmit error counter (TEC) reaches a value greater than 95. The interrupt is reset by writing a '1' to this bit position, writing '0' has no effect.

Bit 3: IRR3	Description
0	[Clearing condition] Writing 1 (Initial value) (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Error warning state caused by transmit error [Setting condition] When TEC ≥ 96

Bit 2 - Remote Frame Request Interrupt Flag (IRR2): flag indicating that a remote frame has been received in a mailbox. This bit is set if at least one receive mailbox, with related MBIMR not set, contains a remote frame transmission request. This bit is automatically cleared when all bits in the Remote Frame Receive Pending Register (RFPR), are cleared. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit has no effect.

Bit 2: IRR2	Description
0	[Clearing condition] Clearing of all bits in RFPR (Initial value)
1	at least one remote request is pending [Setting condition] When remote frame is received and the corresponding MBIMR = 0

Bit 1 – Data Frame Received Interrupt Flag (IRR1): IRR1 indicates that there are pending Data Frames received. If this bit is set at least one receive mailbox contains a pending message. This bit is cleared when all bits in the Data Frame Receive Pending Register (RXPR) are cleared, i.e. there is no pending message in any receiving mailbox. It is in effect a logical OR of the RXPR flags from each configured receive mailbox with related MBIMR not set. It is also cleared by writing a '1' to all the correspondent bit position in MBIMR. Writing to this bit has no effect.

Bit 1: IRR1	Description
0	[Clearing condition] Clearing of all bits in RXPR (Initial value)
1	Data frame received and stored in Mailbox [Setting condition] When data is received and the corresponding MBIMR = 0

Bit 0 – Reset/Halt/Sleep Interrupt Flag (IRR0): This flag can get set for three different reasons. It can indicate that:

1. Reset mode has been entered after a SW (MCR0) or HW reset
2. Halt mode has been entered after a Halt request (MCR1)
3. Sleep mode has been entered after a sleep request (MCR5) has been made while in Halt mode.

The GSR may be read after this bit is set to determine which state RCAN-ET is in.

Important: When a Sleep mode request needs to be made, the Halt mode must be used beforehand. Please refer to the MCR5 description and Figure 13.8 - Halt Mode/Sleep Mode.

IRR0 is set by the transition from "0" to "1" of GSR3 or GSR4 or by transition from Halt mode to Sleep mode. So, IRR0 is not set if RCAN-ET enters Halt mode again right after exiting from Halt mode, without GSR4 being cleared. Similarly, IRR0 is not set by direct transition from Sleep mode to Halt Request. At the transition from Halt/Sleep mode to Transition/Reception, clearing GSR4 needs (one-bit time - TSEG2) to (one-bit time * 2 - TSEG2).

In the case of Reset mode, IRR0 is set, however, the interrupt to the CPU is not asserted since IMR0 is automatically set by initialization.

Bit 0: IRR0	Description
0	[Clearing condition] Writing 1 (When the CPU is used to clear this flag by writing 1 while the corresponding interrupt is enabled, be sure to read the flag after writing 1 to it.)
1	Transition to S/W reset mode or transition to halt mode or transition to sleep mode (Initial value) [Setting condition] When reset/halt/sleep transition is completed after a reset (MCR0 or HW) or Halt mode (MCR1) or Sleep mode (MCR5) is requested

13.4.5 Interrupt Mask Register (IMR)

The interrupt mask register (IMR) is a 16 bit register that protects all corresponding interrupts in the Interrupt Request Register (IRR) from generating an output signal on the IRQ. An interrupt request is masked if the corresponding bit position is set to '1'. This register can be read or written at any time. The IMR directly controls the generation of IRQ, but does not prevent the setting of the corresponding bit in the IRR.

- IMR (Address = H'00A)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0
Initial value:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15 to 0: Maskable interrupt sources corresponding to IRR[15:0] respectively. When a bit is set, the interrupt signal is not generated, although setting the corresponding IRR bit is still performed.

Bit[15:0]: IMRn	Description
0	Corresponding IRR is not masked (IRQ is generated for interrupt conditions)
1	Corresponding interrupt of IRR is masked (Initial value)

13.4.6 Transmit Error Counter (TEC) and Receive Error Counter (REC)

The Transmit Error Counter (TEC) and Receive Error Counter (REC) is a 16-bit read/(write) register that functions as a counter indicating the number of transmit/receive message errors on the CAN Interface. The count value is stipulated in the CAN protocol specification Refs. [1], [2], [3] and [4]. In modes other than Write Error Counter test mode, this register is read only, and can only be modified by the CAN Interface. This register can be cleared by a Reset request (MCR0) or entering to bus off.

In Write Error Counter test mode (i.e. TST[2:0] = 3'b100), it is possible to write to this register. The same value can only be written to TEC/REC, and the value written into TEC is set to TEC and REC. When writing to this register, RCAN-ET needs to be put into Halt Mode. This feature is only intended for test purposes.

- TEC/REC (Address = H'00C)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Initial value:	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*	R/W*

Note: * It is only possible to write the value in test mode when TST[2:0] in MCR is 3'b100.
REC is incremented during Bus Off to count the recurrences of 11 recessive bits as requested by the Bus Off recovery sequence.

13.5 RCAN-ET Mailbox Registers

The following sections describe RCAN-ET Mailbox registers that control / flag individual Mailboxes. The address is mapped as follows.

Important: LongWord access is carried out as two consecutive Word accesses.

Table 13.6 RCAN-ET Mailbox Registers

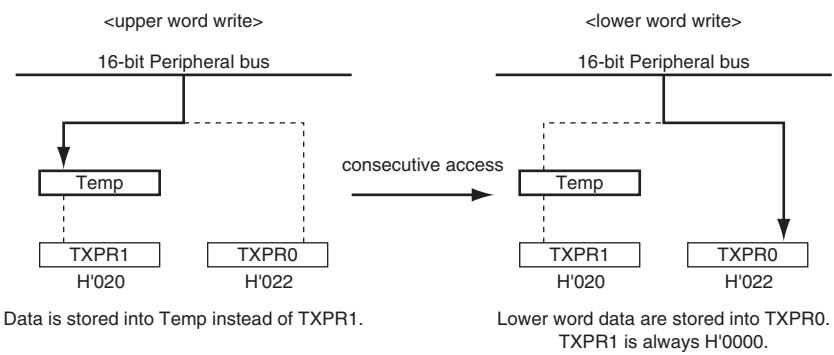
Description	Address	Name	Access Size (bits)
Transmit Pending 1	H'020	TXPR1	LW
Transmit Pending 0	H'022	TXPR0	—
	H'024		
	H'026		
	H'028		
Transmit Cancel 0	H'02A	TXCR0	
	H'02C		
	H'02E		
	H'030		
Transmit Acknowledge 0	H'032	TXACK0	Word
	H'034		
	H'036		
	H'038		
Abort Acknowledge 0	H'03A	ABACK0	Word
	H'03C		
	H'03E		
	H'040		
Data Frame Receive Pending 0	H'042	RXPR0	Word
	H'044		
	H'046		
	H'048		
Remote Frame Receive Pending 0	H'04A	RFPR0	Word
	H'04C		
	H'04E		
	H'050		

Description	Address	Name	Access Size (bits)
Mailbox Interrupt Mask Register 0	H'052	MBIMR0	Word
	H'054		
	H'056		
	H'058		
Unread Message Status Register 0	H'05A	UMSR0	Word
	H'05C		
	H'05E		

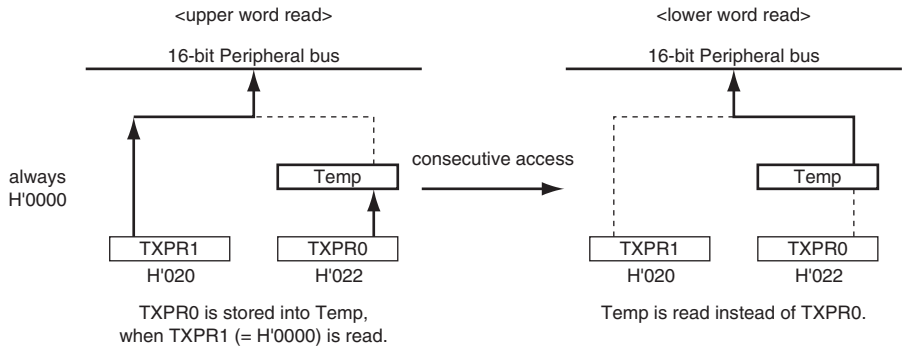
13.5.1 Transmit Pending Register (TXPR0, TXPR1)

The concatenation of TXPR0 and TXPR1 is a 32-bit register that contains any transmit pending flags for the CAN module. In the case of 16-bit bus interface, Long Word access is carried out as two consecutive word accesses.

<Longword Write Operation>



<Longword Read Operation>



The TXPR1 register cannot be modified and it is always fixed to '0'. The TXPR0 controls Mailbox-15 to Mailbox-1. The CPU may set the TXPR bits to affect any message being considered for transmission by writing a '1' to the corresponding bit location. Writing a '0' has no effect, and TXPR cannot be cleared by writing a '0' and must be cleared by setting the corresponding TXCR bits. TXPR may be read by the CPU to determine which, if any, transmissions are pending or in progress. In effect there is a transmit pending bit for all Mailboxes except for the Mailbox-0. Writing a '1' to a bit location when the mailbox is not configured to transmit is not allowed.

The RCAN-ET will clear a transmit pending flag after successful transmission of its corresponding message or when a transmission abort is requested successfully from the TXCR. The TXPR flag is not cleared if the message is not transmitted due to the CAN node losing the arbitration process or due to errors on the CAN bus, and RCAN-ET automatically tries to transmit it again unless its DART bit (Disable Automatic Re-Transmission) is set in the Message-Control of the corresponding Mailbox. In such case (DART set), the transmission is cleared and notified through Mailbox Empty Interrupt Flag (IRR8) and the correspondent bit within the Abort Acknowledgement Register (ABACK).

If the status of the TXPR changes, the RCAN-ET shall ensure that in the identifier priority scheme (MCR2=0), the highest priority message is always presented for transmission in an intelligent way even under circumstances such as bus arbitration losses or errors on the CAN bus. Please refer to section 13.6, Application Note, for details.

When the RCAN-ET changes the state of any TXPR bit position to a '0', an empty slot interrupt (IRR8) may be generated. This indicates that either a successful or an aborted mailbox transmission has just been made. If a message transmission is successful it is signalled in the TXACK register, and if a message transmission abortion is successful it is signalled in the ABACK register. By checking these registers, the contents of the Message of the corresponding Mailbox may be modified to prepare for the next transmission.

- TXPR1

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXPR1[15:0]															
Initial value:	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*	R/W*

Note: * Read value is always H'0000. Long word access is mandatory when reading or writing TXPR1/TXPR0. Write operation to TXPR1 has no effect.

- TXPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXPR0[15:1]															0
Initial value:	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*	—

Note: * It is possible only to write a '1' for a Mailbox configured as transmitter. Read value is always H'0000. Long word access is mandatory when reading or writing TXPR1/TXPR0. Write operation to bit 0 of TXPR0 has no effect.

Bit 15 to 1 — indicates that the corresponding Mailbox is requested to transmit a CAN Frame. The bit 15 to 1 corresponds to Mailbox-15 to 1 respectively. When multiple bits are set, the order of the transmissions is governed by the MCR2 – CAN-ID or Mailbox number.

Bit[15:1]:TXPR0	Description
0	Transmit message idle state in corresponding mailbox (Initial value) [Clearing Condition] Completion of message transmission or message transmission abortion (automatically cleared)
1	Transmission request made for corresponding mailbox

Bit 0— Reserved: This bit is always '0' as this is a receive-only Mailbox. Writing a '1' to this bit position has no effect. The returned value is '0'.

13.5.2 Transmit Cancel Register (TXCR0)

The TXCR0 is a 16-bit read/conditionally-write register. The TXCR0 controls Mailbox-15 to Mailbox-1.This register is used by the CPU to request the pending transmission requests in the TXPR to be cancelled. To clear the corresponding bit in the TXPR the CPU must write a '1' to the bit position in the TXCR. Writing a '0' has no effect.

When an abort has succeeded the CAN controller clears the corresponding TXPR + TXCR bits, and sets the corresponding ABACK bit. However, once a Mailbox has started a transmission, it cannot be cancelled by this bit. In such a case, if the transmission finishes in success, the CAN controller clears the corresponding TXPR + TXCR bit, and sets the corresponding TXACK bit, however, if the transmission fails due to a bus arbitration loss or an error on the bus, the CAN controller clears the corresponding TXPR + TXCR bit, and sets the corresponding ABACK bit. If an attempt is made by the CPU to clear a mailbox transmission that is not transmit-pending it has no effect. In this case the CPU will be not able at all to set the TXCR flag.

- TXCR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXCR0[15:1]															0
Initial value:	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*	—

Note: * Only writing a '1' to a Mailbox that is requested for transmission and is configured as transmit.

Bit 15 to 1 — requests the corresponding Mailbox, that is in the queue for transmission, to cancel its transmission. The bit 15 to 1 corresponds to Mailbox-15 to 1 (and TXPR0[15:1]) respectively.

Bit[15:1]:TXCR0	Description
0	Transmit message cancellation idle state in corresponding mailbox (Initial value) [Clearing Condition] Completion of transmit message cancellation (automatically cleared)
1	Transmission cancellation request made for corresponding mailbox

Bit 0 — This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

13.5.3 Transmit Acknowledge Register0 (TXACK0)

The TXACK0 is a 16-bit read/conditionally-write register. This register is used to signal to the CPU that a mailbox transmission has been successfully made. When a transmission has succeeded the RCAN-ET sets the corresponding bit in the TXACK register. The CPU may clear a TXACK bit by writing a '1' to the corresponding bit location. Writing a '0' has no effect.

- TXACK0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TXACK0[15:1]															0
Initial value:	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*	—

Note: * Only when writing a '1' to clear.

Bit 15 to 1 — notifies that the requested transmission of the corresponding Mailbox has been finished successfully. The bit 15 to 1 corresponds to Mailbox-15 to 1 respectively.

Bit[15:1]:TXACK0	Description
0	[Clearing Condition] Writing '1' (Initial value)
1	Corresponding Mailbox has successfully transmitted message (Data or Remote Frame) [Setting Condition] Completion of message transmission for corresponding mailbox

Bit 0 — This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

13.5.4 Abort Acknowledge Register0 (ABACK0)

The ABACK0 is a 16-bit read/conditionally-write register. This register is used to signal to the CPU that a mailbox transmission has been aborted as per its request. When an abort has succeeded the RCAN-ET sets the corresponding bit in the ABACK register. The CPU may clear the Abort Acknowledge bit by writing a '1' to the corresponding bit location. Writing a '0' has no effect. An ABACK bit position is set by the RCAN-ET to acknowledge that a TXPR bit has been cleared by the corresponding TXCR bit.

- ABACK0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ABACK0[15:1]															0
Initial value:	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*	—

Note: * Only when writing a '1' to clear.

Bit 15 to 1 — notifies that the requested transmission cancellation of the corresponding Mailbox has been performed successfully. The bit 15 to 1 corresponds to Mailbox-15 to 1 respectively.

Bit[15:1]:ABACK0	Description
0	[Clearing Condition] Writing '1' (Initial value)
1	Corresponding Mailbox has cancelled transmission of message (Data or Remote Frame) [Setting Condition] Completion of transmission cancellation for corresponding mailbox

Bit 0 — This bit is always '0' as this is a receive-only mailbox. Writing a '1' to this bit position has no effect and always read back as a '0'.

13.5.5 Data Frame Receive Pending Register0 (RXPR0)

The RXPR0 is a 16-bit read/conditionally-write register. The RXPR is a register that contains the received Data Frames pending flags associated with the configured Receive Mailboxes. When a CAN Data Frame is successfully stored in a receive mailbox the corresponding bit is set in the RXPR. The bit may be cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. However, the bit may only be set if the mailbox is configured by its MBC (Mailbox Configuration) to receive Data Frames. When a RXPR bit is set, it also sets IRR1 (Data Frame Received Interrupt Flag) if its MBIMR (Mailbox Interrupt Mask Register) is not set, and the interrupt signal is generated if IMR1 is not set. Please note that these bits are only set by receiving Data Frames and not by receiving Remote frames.

- RXPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RXPR0[15:0]															
Initial value:	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*	R/W*

Note: * Only when writing a '1' to clear.

Bit 15 to 0 — Configurable receive mailbox locations corresponding to each mailbox position from 15 to 0 respectively.

Bit[15:0]: RXPR0	Description
0	[Clearing Condition] Writing '1' (Initial value)
1	Corresponding Mailbox received a CAN Data Frame [Setting Condition] Completion of Data Frame receive on corresponding mailbox

13.5.6 Remote Frame Receive Pending Register0 (RFPR0)

The RFPR0 is a 16-bit read/conditionally-write register. The RFPR is a register that contains the received Remote Frame pending flags associated with the configured Receive Mailboxes. When a CAN Remote Frame is successfully stored in a receive mailbox the corresponding bit is set in the RFPR. The bit may be cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. In effect there is a bit position for all mailboxes. However, the bit may only be set if the mailbox is configured by its MBC (Mailbox Configuration) to receive Remote Frames. When a RFPR bit is set, it also sets IRR2 (Remote Frame Request Interrupt Flag) if its MBIMR (Mailbox Interrupt Mask Register) is not set, and the interrupt signal is generated if IMR2 is not set. Please note that these bits are only set by receiving Remote Frames and not by receiving Data frames.

- RFPR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RFPR0[15:0]															
Initial value:	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*	R/W*

Note: * Only when writing a '1' to clear.

Bit 15 to 0 — Remote Request pending flags for mailboxes 15 to 0 respectively.

Bit[15:0]: RFPR0	Description
0	[Clearing Condition] Writing '1' (Initial value)
1	Corresponding Mailbox received Remote Frame [Setting Condition] Completion of remote frame receive in corresponding mailbox

13.5.7 Mailbox Interrupt Mask Register0 (MBIMR0)

The MBIMR0 is a 16-bit read/write register. The MBIMR only prevents the setting of IRR related to the Mailbox activities, that are IRR[1] – Data Frame Received Interrupt, IRR[2] – Remote Frame Request Interrupt, IRR[8] – Mailbox Empty Interrupt, and IRR[9] – Message OverRun/OverWrite Interrupt. If a mailbox is configured as receive, a mask at the corresponding bit position prevents the generation of a receive interrupt (IRR[1] and IRR[2] and IRR[9]) but does not prevent the setting of the corresponding bit in the RXPR or RFPR or UMSR. Similarly when a mailbox has been configured for transmission, a mask prevents the generation of an Interrupt signal and setting of an Mailbox Empty Interrupt due to successful transmission or abortion of transmission (IRR[8]), however, it does not prevent the RCAN-ET from clearing the corresponding TXPR/TXCR bit + setting the TXACK bit for successful transmission, and it does not prevent the RCAN-ET from clearing the corresponding TXPR/TXCR bit + setting the ABACK bit for abortion of the transmission.

A mask is set by writing a '1' to the corresponding bit position for the mailbox activity to be masked. At reset all mailbox interrupts are masked.

- MBIMR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MBIMR0[15:0]															
Initial value:	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 15 to 0 — Enable or disable interrupt requests from individual Mailbox-15 to Mailbox-0 respectively.

Bit[15:0]: MBIMR0	Description
0	Interrupt Request from IRR1/IRR2/IRR8/IRR9 enabled
1	Interrupt Request from IRR1/IRR2/IRR8/IRR9 disabled (initial value)

13.5.8 Unread Message Status Register0 (UMSR0)

The UMAR0 is a 16-bit read/conditionally write register and it records the mailboxes whose contents have not been accessed by the CPU prior to a new message being received. If the CPU has not cleared the corresponding bit in the RXPR or RFPR when a new message for that mailbox is received, the corresponding UMSR bit is set to '1'. This bit may be cleared by writing a '1' to the corresponding bit location in the UMSR. Writing a '0' has no effect.

If a mailbox is configured as transmit box, the corresponding UMSR will not be set.

- UMSR0

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	UMSR0[15:0]															
Initial value:	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*	R/W*

Bit 15 to 0 — Indicate that an unread received message has been overwritten or overrun condition has occurred for Mailboxes 15 to 0.

Bit[15:0]: UMSR0	Description
0	[Clearing Condition] Writing '1' (initial value)
1	Unread received message is overwritten by a new message or overrun condition [Setting Condition] When a new message is received before RXPR or RFPR is cleared

13.5.9 RCAN Monitor Register (RCANMON)

RCANMON controls the transmission stop for CTx pin, enables/disables RCAN-ET transmission/reception pins, and monitors the status of RCAN-ET pins. This register is not initialized by the software reset (MCR0).

Bit	7	6	5	4	3	2	1	0
Bit Name	—	CTxSTP	RCANE	—	—	—	CTxD	CRxD
Initial Value	0	0	0	Undefined	Undefined	Undefined	Undefined	Undefined
R/W	—	R/W	R/W	—	—	—	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	—	Undefined	R/W	Reserved Reading/writing to this bit is invalid.
6	CTxSTP	0	R/W	CTx Transmit Stop Setting this bit to 1 forces CTx pin to be set to 1, whether or not the RCAN-ET is transmitting data.
5	RCANE	0	R/W	RCAN-ET Transmit/Receive Pin Enable Setting this bit to 1 enables CTx and CRx pins of the RCAN-ET are enabled.
4 to 2	—	All undefined	—	Reserved Reading/writing to these bits is invalid.
1	CTxD	Undefined	R	CTx Transmit Data Monitor The status of CTx pin is acquired when this bit is read. Writing to this bit is invalid.
0	CRxD	Undefined	R	CRx Receive Data Monitor The status of CRx pin is acquired when this bit is read. Writing to this bit is invalid.

Figure 13.6 shows a diagram to illustrate how those bits of RCANMON are connected together to form a port interface.

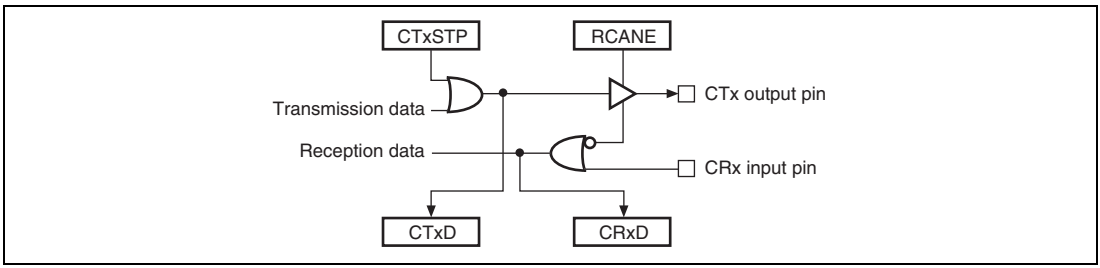


Figure 13.6 Connection Diagram of Port Interface

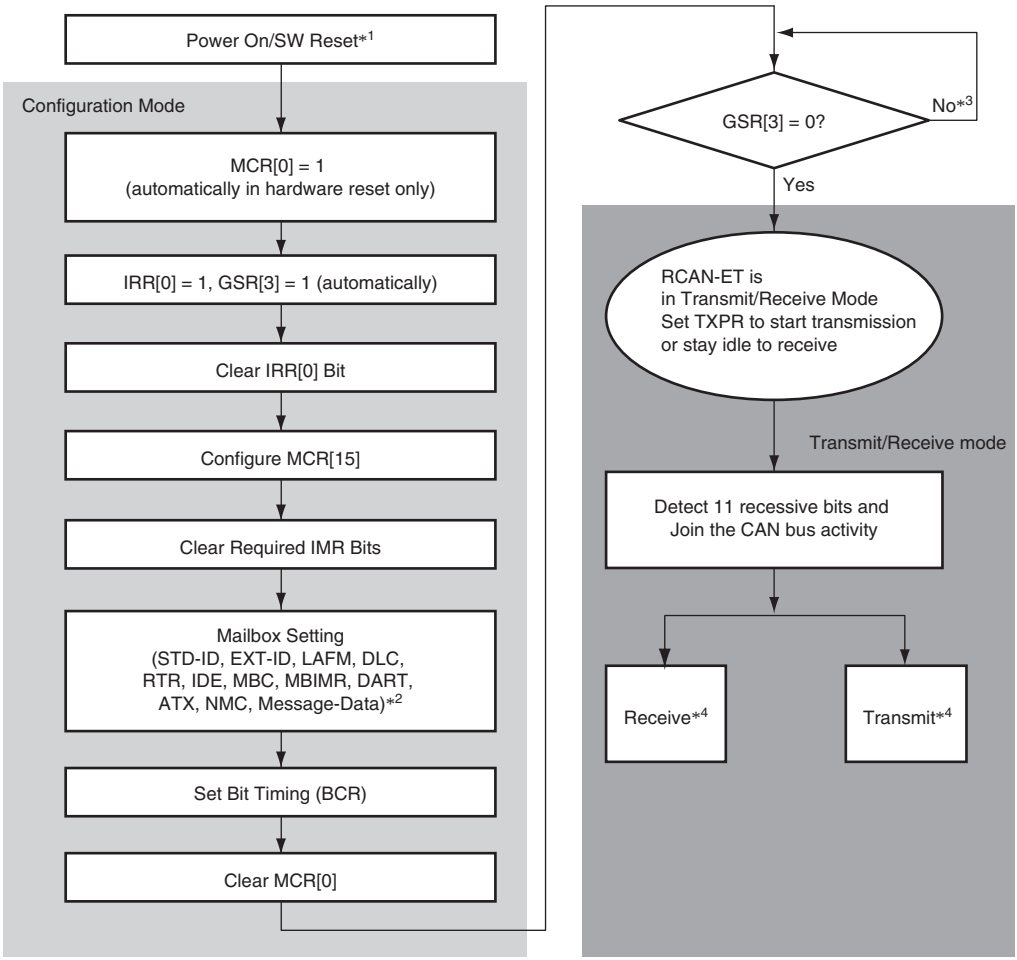
13.6 Application Note

13.6.1 Configuration of RCAN-ET

RCAN-ET is considered in configuration mode or after a H/W (Power On Reset)/ S/W (MCR[0]) reset or when in Halt mode. In both conditions RCAN-ET cannot join the CAN Bus activity and configuration changes have no impact on the traffic on the CAN Bus.

(1) After a reset request

The following sequence is an example of how to configure the RCAN-ET after (S/W or H/W) reset. After reset, all the registers are initialised, therefore, RCAN-ET needs to be configured before joining the CAN bus activity. Please read the notes carefully.



Notes: 1. SW reset could be performed at any time by setting MCR[0] = 1.
2. Mailboxes are comprised of RAMs, therefore, please initialize all the mailboxes enabled by MBC.
3. It takes approximately 25 peripheral bus cycles for GSR[3] to be cleared to 0.
4. If there is no TXPR set, RCAN-ET will receive the next incoming message.
If there is a TXPR(s) set, RCAN-ET will start transmission of the message and will be arbitrated by the CAN bus.
If it loses the arbitration, it will become a receiver.

Figure 13.7 Reset Sequence

(2) Halt mode

When RCAN-ET is in Halt mode, it cannot take part to the CAN bus activity. Consequently the user can modify all the requested registers without influencing existing traffic on the CAN Bus. It is important for this that the user waits for the RCAN-ET to be in halt mode before to modify the requested registers - note that the transition to Halt Mode is not always immediate (transition will occurs when the CAN Bus is idle or in intermission). After RCAN-ET transit to Halt Mode, GSR4 is set.

Once the configuration is completed the Halt request needs to be released. RCAN-ET will join CAN Bus activity after the detection of 11 recessive bits on the CAN Bus.

(3) Sleep mode

When RCAN-ET is in sleep mode the clock for the main blocks of the IP is stopped in order to reduce power consumption. Only the following user registers are clocked and can be accessed: MCR, GSR, IRR and IMR. Interrupt related to transmission (TXACK and ABACK) and reception (RXPR and RFPR) cannot be cleared when in sleep mode (as TXACK, ABACK, RXPR and RFPR are not accessible) and must to be cleared beforehand.

The following diagram shows the flow to follow to move RCAN-ET into sleep mode.

(4) Can sleep mode

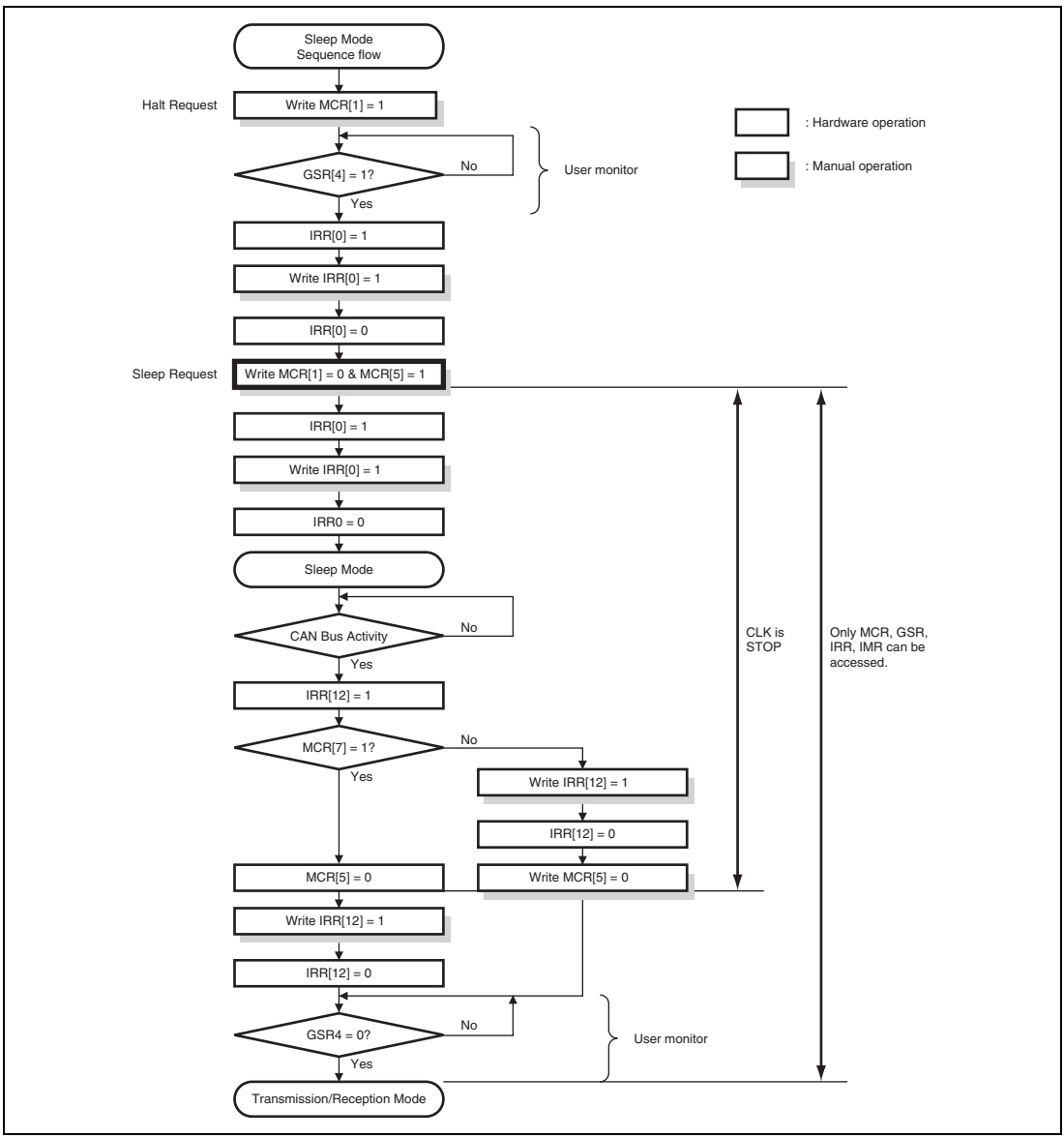
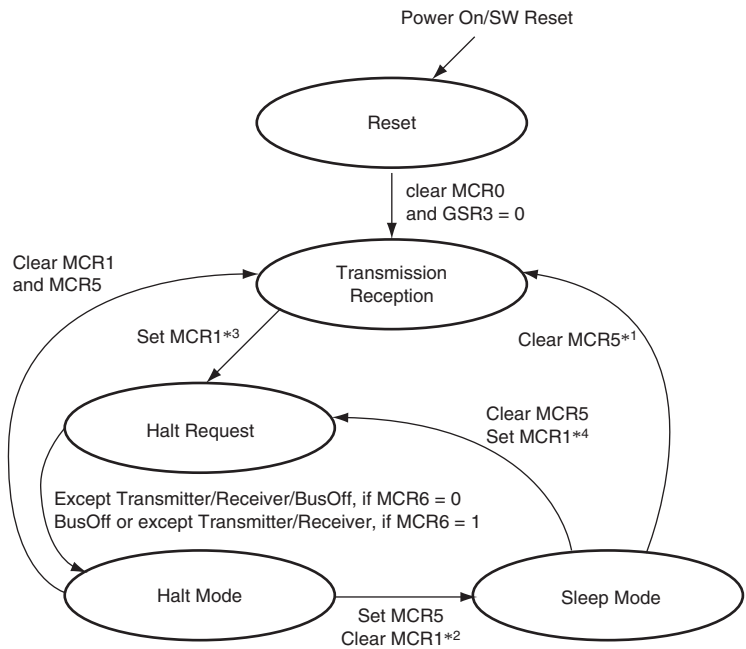


Figure 13.8 Halt Mode/Sleep Mode

Figure 13.9 - Halt Mode/Sleep Mode shows allowed state transition.

- Please don't set MCR5 (Sleep Mode) without entering Halt Mode.
- After MCR1 is set, please don't clear it before GSR4 is set and RCAN-ET enters Halt Mode.



Notes:
1. MCR5 can be cleared by automatically by detecting a dominant bit on the CAN Bus if MCR7 is set or by writing "0".
2. MCR1 is cleared in SW. Clearing MCR1 and setting MCR5 have to be carried out by the same instruction.
3. MCR1 must not be cleared in SW, before GSR4 is set. MCR1 can be set automatically in HW when RCAN-ET moves to Bus Off and MCR14 and MCR6 are both set.
4. When MCR5 is cleared and MCR1 is set at the same time, RCAN-ET moves to Halt Request. Right after that, it moves to Halt Mode with no reception/transmission.

Figure 13.9 Halt Mode/Sleep Mode

The following table shows conditions to access registers.

Table 13.7 Conditions to access registers

	RCAN-ET Registers								
Status Mode	**MCR GSR**	**IRR IMR**	**BCR**	**MBIMR**	**Flag_register**	**mailbox (ctrl0, LAFM)**		**mailbox (data)**	**mailbox (ctrl1)**	
Reset	yes	yes	yes	yes	yes	yes		yes	yes	
Transmission Reception Halt Request	yes	yes	no*1	yes	yes	no*1	yes*2	yes*2	no*1	yes*2
Halt	yes	yes	no*1	yes	yes	yes		yes	yes	
Sleep	yes	yes	no	no	no	no		no	no	

Notes: 1. No hardware protection
2. When TXPR is not set.

13.6.2 Test Mode Settings

The RCAN-ET has various test modes. The register TST[2:0] (MCR[10:8]) is used to select the RCAN-ET test mode. The default (initialised) settings allow RCAN-ET to operate in Normal mode. The following table is examples for test modes.

Test Mode can be selected only while in configuration mode. The user must then exit the configuration mode (ensuring BCR0/BCR1 is set) in order to run the selected test mode.

Table.13.8 Test Mode Settings

Bit10: TST2	Bit9: TST1	Bit8: TST0	Description
0	0	0	Normal Mode (initial value)
0	0	1	Listen-Only Mode (Receive-Only Mode)
0	1	0	Self Test Mode 1 (External)
0	1	1	Self Test Mode 2 (Internal)
1	0	0	Write Error Counter
1	0	1	Error Passive Mode
1	1	0	Setting prohibited
1	1	1	Setting prohibited

- Normal Mode

  RCAN-ET operates in the normal mode.

- Listen-Only Mode:

  ISO-11898 requires this mode for baud rate detection. The Error Counters are cleared and disabled so that the TEC/REC does not increase the values, and the CTx Output is disabled so that RCAN-ET does not generate error frames or acknowledgment bits. IRR13 is set when a message error occurs.

- Self Test Mode 1

  RCAN-ET generates its own Acknowledge bit, and can store its own messages into a reception mailbox (if required). The CRx/CTx pins must be connected to the CAN bus.

- Self Test Mode 2

  RCAN-ET generates its own Acknowledge bit, and can store its own messages into a reception mailbox (if required). The CRx/CTx pins do not need to be connected to the CAN bus or any external devices, as the internal CTx is looped back to the internal CRx. CTx pin outputs only recessive bits and CRx pin is disabled.

- Write Error Counter

  TEC/REC can be written in this mode. RCAN-ET can be forced to become an Error Passive mode by writing a value greater than 127 into the Error Counters. The value written into TEC is used to write into REC, so only the same value can be set to these registers. Similarly, RCAN-ET can be forced to become an Error Warning by writing a value greater than 95 into them.

- Error Passive mode

  RCAN-ET needs to be in Halt Mode when writing into TEC/REC (MCR1 must be "1" when writing to the Error Counter). Furthermore this test mode needs to be exited prior to leaving Halt mode.Error Passive Mode: RCAN-ET can be forced to enter Error Passive mode.

Note: the REC will not be modified by implementing this Mode. However, once running in Error Passive Mode, the REC will increase normally should errors be received. In this Mode, RCAN-ET will enter Bus Off if TEC reaches 256 (Dec). However when this mode is used RCAN-ET will not be able to become Error Active. Consequently, at the end of the Bus Off recovery sequence, RCAN-ET will move to Error Passive and not to Error Active

When message error occurs, IRR13 is set in all test modes.

13.6.3 Message Transmission Sequence

(1) Message Transmission Request

The sequence to transmit a CAN frame onto the bus is shown in figure 13.10. As described in the previous register section, please note that IRR8 is set when one of the TXACK or ABACK bits is set, meaning one of the Mailboxes has completed its transmission or transmission abortion and is now ready to be updated for the next transmission, whereas, the GSR2 means that there is currently no transmission request made (No TXPR flags set).

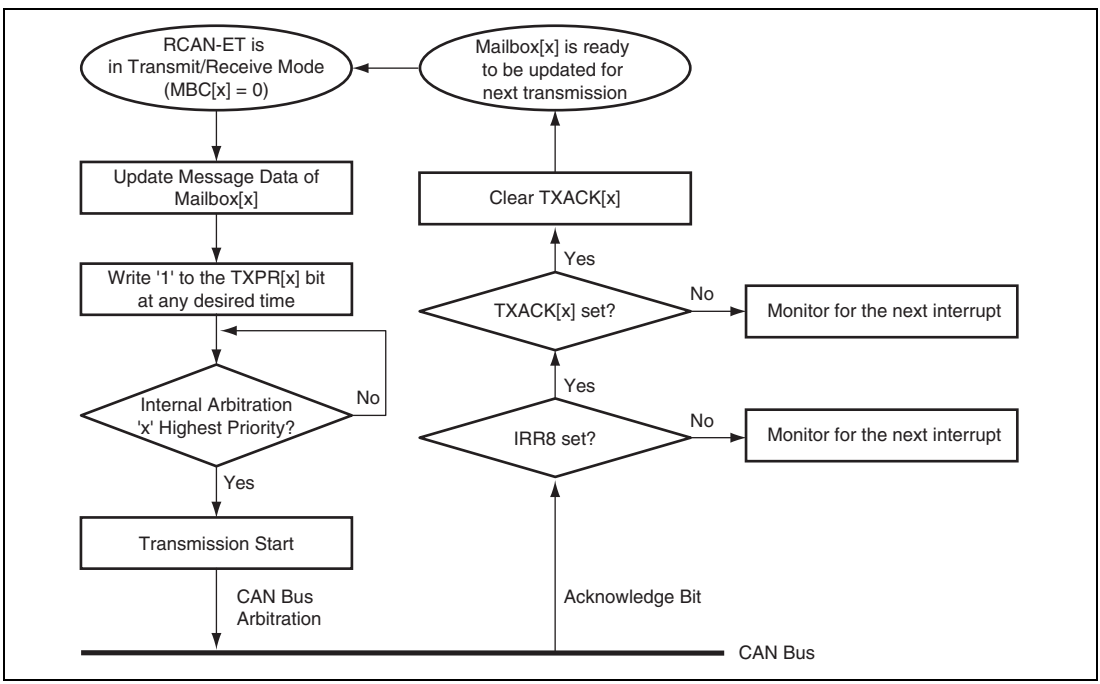


Figure 13.10 Transmission Request

(2) Internal Arbitration for Transmission

The following diagram explains how RCAN-ET manages to schedule transmission-requested messages in the correct order based on the CAN identifier. 'Internal arbitration' picks up the highest priority message amongst transmit-requested messages.

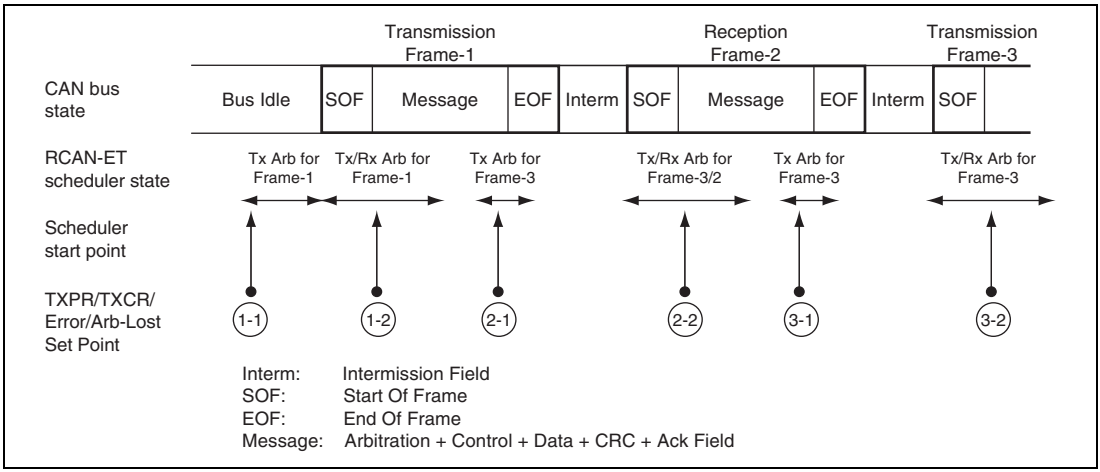


Figure 13.11 Internal Arbitration for transmission

The RCAN-ET has two state machines. One is for transmission, and the other is for reception.

1-1: When a TXPR bit(s) is set while the CAN bus is idle, the internal arbitration starts running immediately and the transmission is started.

1-2: Operations for both transmission and reception starts at SOF. Since there is no reception frame, RCAN-ET becomes transmitter.

2-1: At crc delimiter, internal arbitration to search next message transmitted starts.

2-2: Operations for both transmission and reception starts at SOF. Because of a reception frame with higher priority, RCAN-ET becomes receiver. Therefore, Reception is carried out instead of transmitting Frame-3.

3-1: At crc delimiter, internal arbitration to search next message transmitted starts.

3-2: Operations for both transmission and reception starts at SOF. Since a transmission frame has higher priority than reception one, RCAN-ET becomes transmitter.

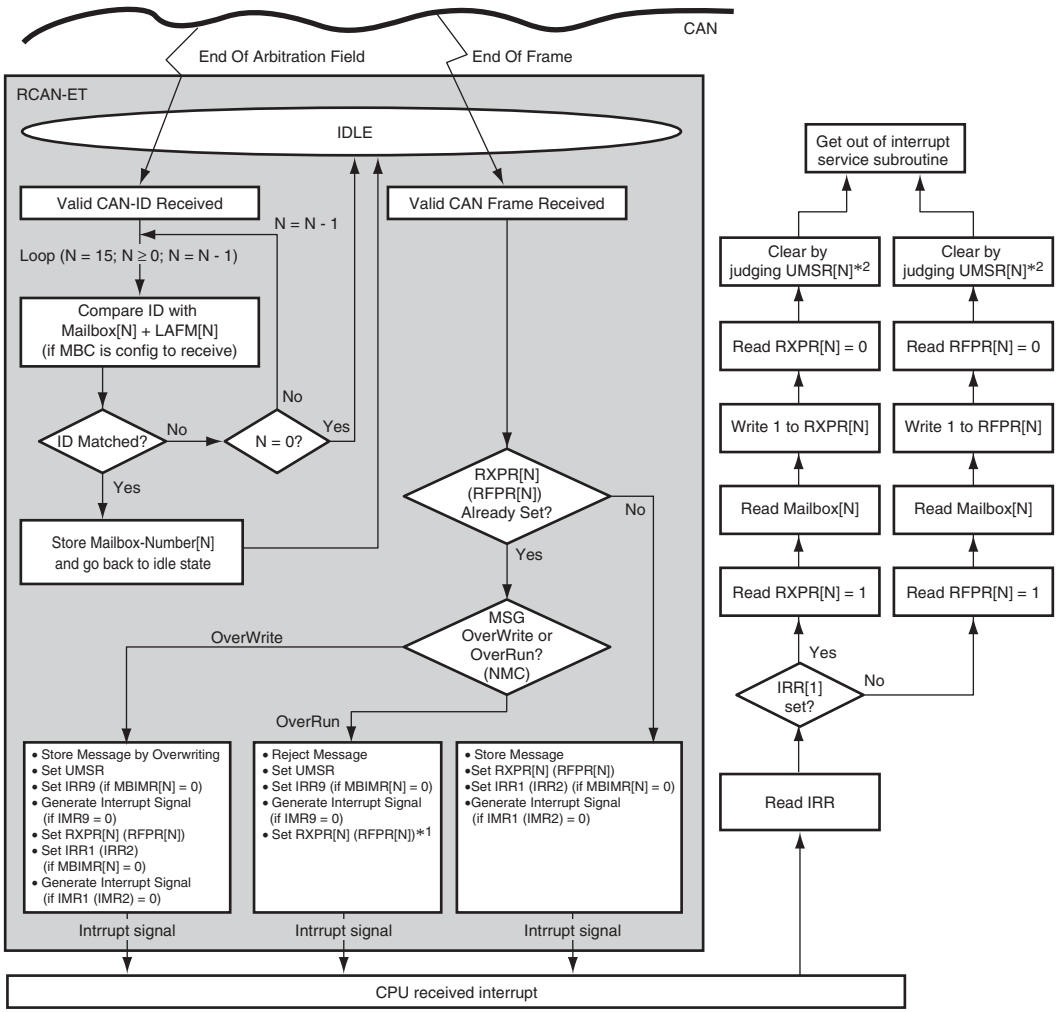
Internal arbitration for the next transmission is also performed at the beginning of each error delimiter in case of an error is detected on the CAN Bus. It is also performed at the beginning of error delimiters following overload frame.

As the arbitration for transmission is performed at CRC delimiter, in case a remote frame request is received into a Mailbox with ATX=1 the answer can join the arbitration for transmission only at the following Bus Idle, CRC delimiter or Error Delimiter.

Depending on the status of the CAN bus, following the assertion of the TXCR, the corresponding Message abortion can be handled with a delay of maximum 1 CAN Frame.

13.6.4 Message Receive Sequence

The diagram below shows the message receive sequence.



Notes: 1. Only if CPU clears RXPR[N]/RFPR[N] at the same time that UMSR is set in overrun, RXPR[N]/RFPR[N] may be set again even though the message has not been updated.
2. In case overwrite configuration (NMC = 1) is used for the Mailbox N the message must be discarded when UMSR[N] = 1, UMSR[N] cleared and the full Interrupt Service Routine started again. In case of overrun configuration (NMC = 0) is used clear again RXPR[N]/RFPR[N]/UMSR[N] when UMSR[N] = 1 and consider the message obsolete.

Figure 13.12 Message receive sequence

When RCAN-ET recognises the end of the Arbitration field while receiving a message, it starts comparing the received identifier to the identifiers set in the Mailboxes, starting from Mailbox-15 down to Mailbox-0. It first checks the MBC if it is configured as a receive box, and reads LAFM, and reads the CAN-ID of Mailbox-15 (if configured as receive) to finally compare them to the received ID. If it does not match, the same check takes place at Mailbox-14 (if configured as receive). Once RCAN-ET finds a matching identifier, it stores the number of Mailbox-[N] into an internal buffer, stops the search, and goes back to idle state, waiting for the EndOfFrame (EOF) to come. When the 6th bit of EOF is notified by the CAN Interface logic, the received message is written or abandoned, depending on the NMC bit. No modification of configuration during communication is allowed. Entering Halt Mode is one of ways to modify configuration. If it is written into the corresponding Mailbox, including the CAN-ID, i.e., there is a possibility that the CAN-ID is overwritten by a different CAN-ID of the received message due to the LAFM used. This also implies that, if the identifier of a received message matches to ID + LAFM of 2 or more Mailboxes, the higher numbered Mailbox will always store the relevant messages and the lower numbered Mailbox will never receive messages. Therefore, the settings of the identifiers and LAFMs need to be carefully selected.

With regards to the reception of data and remote frames described in the above flow diagram the clearing of the UMSR flag after the reading of IRR is to detect situations where a message is overwritten by a new incoming message stored in the same mailbox while the interrupt service routine is running. If during the final check of UMSR a overwrite condition is detected the message needs to be discarded and read again.

In case UMSR is set and the Mailbox is configured for overrun (NMC = 0), the message is still valid, however it is obsolete as it is not reflecting the latest message monitored on the CAN bus.

Please access the full Mailbox content (area of Mailbox[N]) before clearing the related RXPR/RFPR flag.

Please note that in the case a received remote frame is overwritten by a data frame, both the remote frame request interrupt (IRR2) and data frame received interrupt (IRR1) and also the Receive Flags (RXPR and RFPR) are set. In an analogous way, the overwriting of a data frame by a remote frame, leads to setting both IRR2 and IRR1.

In the Overrun Mode (NMC = '0'), only the first Mailbox will cause the flags to be asserted. So, if a Data Frame is initially received, then RXPR and IPR1 are both asserted. If a Remote Frame is then received before the Data Frame has been read, then RFPR and IRR2 are NOT set. In this case UMSR of the corresponding Mailbox will still be set.

13.6.5 Reconfiguration of Mailbox

When re-configuration of Mailboxes is required, the following procedures should be taken.

(1) Change configuration of transmit box

Two cases are possible.

- Change of ID, RTR, IDE, LAFM, Data, DLC, NMC, ATX, DART
  This change is possible only when MBC=3'b000. Confirm that the corresponding TXPR is not set. The configuration (except MBC bit) can be changed at any time.
- Change from transmit to receive configuration (MBC)
  Confirm that the corresponding TXPR is not set. The configuration can be changed only in Halt or reset state. Please note that it might take longer for RCAN-ET to transit to halt state if it is receiving or transmitting a message (as the transition to the halt state is delayed until the end of the reception/transmission), and also RCAN-ET will not be able to receive/transmit messages during the Halt state.
  In case RCAN-ET is in the Bus Off state the transition to halt state depends on the configuration of the bit 6 of MCR and also bit and 14 of MCR.

(2) Change configuration (ID, RTR, IDE, LAFM, Data, DLC, NMC, ATX, DART, MBC) of receive box or Change receive box to transmit box

The configuration can be changed only in Halt Mode.

RCAN-ET will not lose a message if the message is currently on the CAN bus and RCAN-ET is a receiver. RCAN-ET will be moving into Halt Mode after completing the current reception. Please note that it might take longer if RCAN-ET is receiving or transmitting a message (as the transition to the halt state is delayed until the end of the reception/transmission), and also RCAN-ET will not be able to receive/transmit messages during the Halt Mode.

In case RCAN-ET is in the Bus Off state the transition to halt mode depends on the configuration of the bit 6 and 14 of MCR.

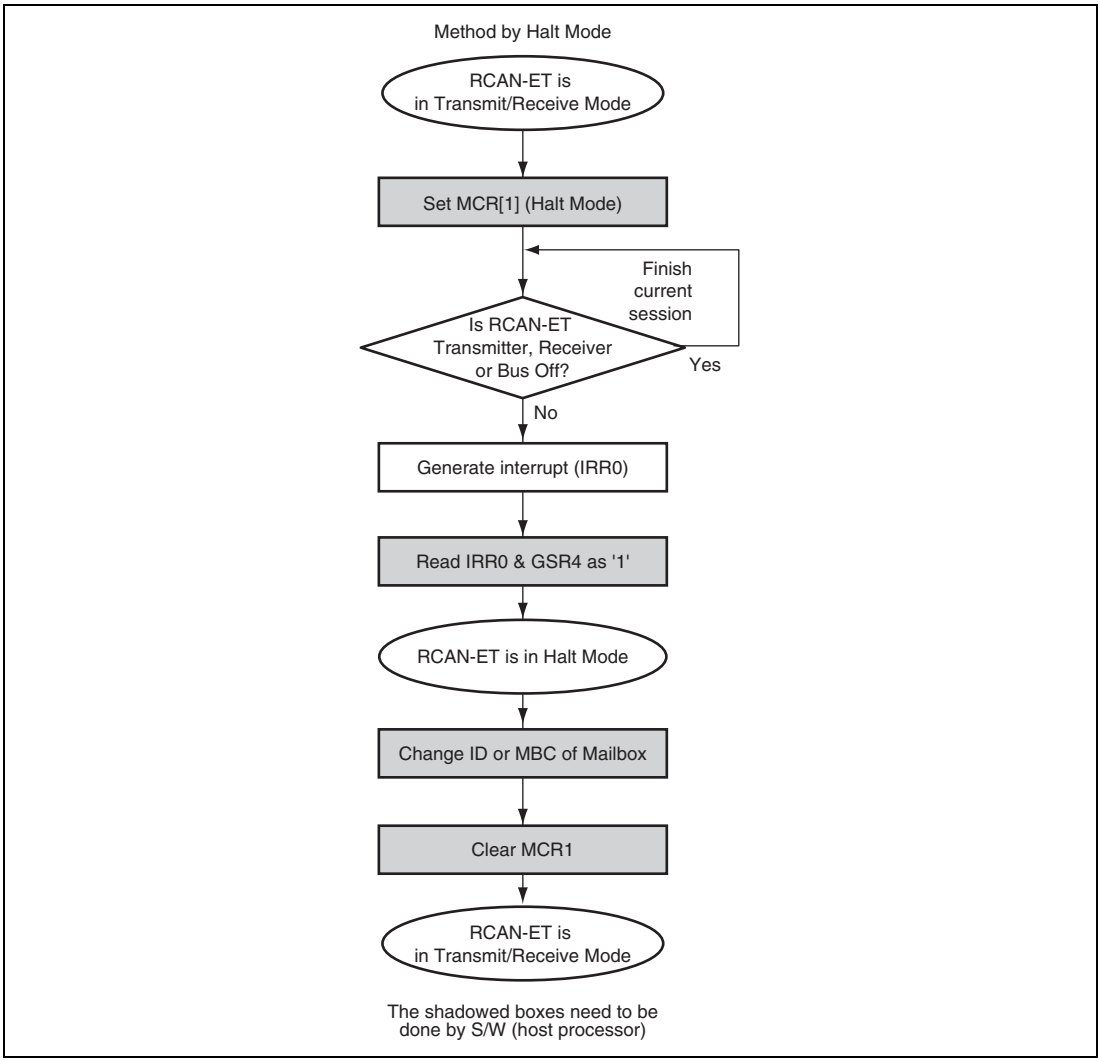


Figure 13.13 Change ID of Receive Box or Change Receive Box to Transmit Box

13.7 Interrupt Sources

Table 13.9 lists the RCAN-ET interrupt sources. With the exception of the reset processing interrupt (IRR0) by a power-on reset, these sources can be masked. Masking is implemented using the mailbox interrupt mask register 0 (MBIMR0) and interrupt mask register (IMR). For details on the interrupt vector of each interrupt source, see section 5, Interrupt Controller.

Table 13.9 RCAN-ET Interrupt Sources

Channel	Interrupt	Description	Interrupt Flag	DMAC Activation
0	ERS_0	Error Passive Mode (TEC ≥ 128 or REC ≥ 128)	IRR5	Not possible
		Bus Off (TEC ≥ 256)/Bus Off recovery	IRR6	
		Error warning (TEC ≥ 96)	IRR3	
		Error warning (REC ≥ 96)	IRR4	
	OVR_0	Message error detection	IRR13*[1]	
		Reset/halt/CAN sleep transition	IRR0	
		Overload frame transmission	IRR7	
		Unread message overwrite (overrun)	IRR9	
		Detection of CAN bus operation in CAN sleep mode	IRR12	
	SLE_0	Message transmission/transmission disabled (slot empty)	IRR8	
	RM1_0*[2]	Data frame reception/	IRR1*[3]	
	RM0_0*[2]	Remote frame reception	IRR2*[3]	Possible*[4]

Notes: 1. Available only in Test Mode.

2. RM0 is an interrupt generated by the remote request pending flag for mailbox 0 (RFPR0[0]) or the data frame receive flag for mailbox 0 (RXPR0[0]). RM1 is an interrupt generated by the remote request pending flag for mailbox n (RFPR0[n]) or the data frame receive flag for mailbox n (RXPR0[n]) (n = 1 to 15).

3. IRR1 is a data frame received interrupt flag for mailboxes 0 to 15, and IRR2 is a remote frame request interrupt flag for mailboxes 0 to 15.

4. Only a RM0_0 interrupt enables the DMAC activation.

13.8 DMAC Interface

The DMAC can be activated by the reception of a message in RCAN-ET mailbox 0. When the DMAC activation is set and DMAC transfer ends, the flags in RXPR0 and RFPR0 are automatically cleared. At this point, an interrupt request is not sent to the CPU by a reception interrupt from the RCAN-ET. Figure 13.14 shows a DMAC transfer flowchart.

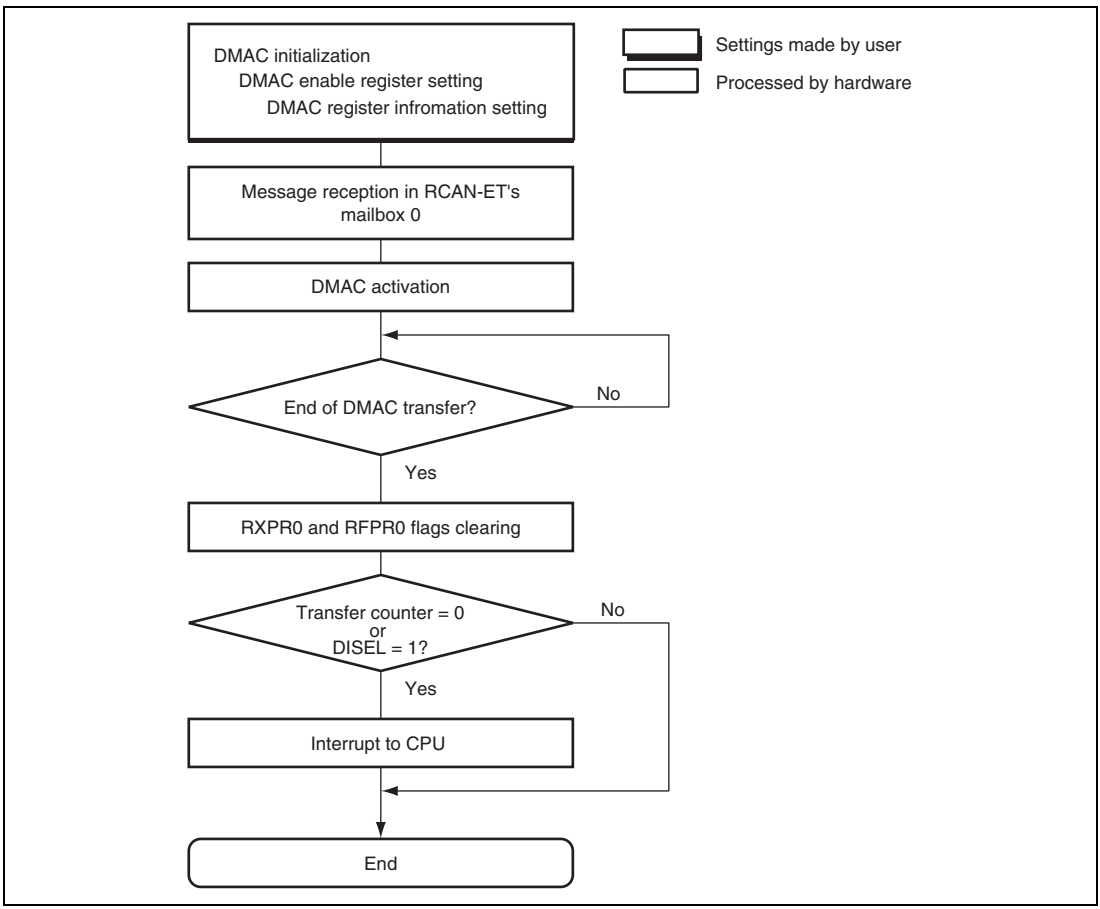


Figure 13.14 DMAC Transfer Flowchart

13.9 CAN Bus Interface

A bus transceiver IC is necessary to connect this LSI to a CAN bus. A Renesas HA13721 transceiver IC and its compatible products are recommended. Figure 13.15 shows a sample connection diagram.

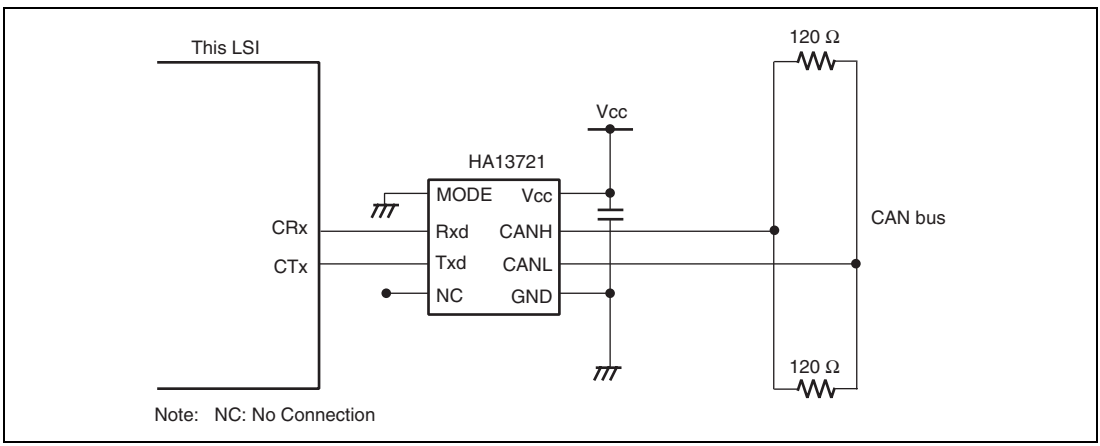


Figure 13.15 High-Speed Interface Using HA13721

13.10 Usage Notes

13.10.1 Module Stop Mode

The module stop control register (MSTPCRC) controls the supply of clocks to RCAN-ET. As an initial value, the clock to RCAN-ET is halted. Registers except for the RCAN-ET monitor register (RCANMON) should be accessed after the module stop mode is released.

13.10.2 Reset

Two types of resets are supported for RCAN-ET.

- Hardware reset
  RCAN-ET is initialized by a power-on reset, or hardware standby, module stop, or software standby mode.
- Software reset
  The MCR0 bit in the master control register (MCR) initializes registers other than MCR and CAN communication functions. RCAN monitor register (RCANMON) is not initialized by this.

As the IRR0 bit in the interrupt request register (IRR) is initialized and set to 1 at a reset, it should be cleared to 0 in the configuration mode shown in the reset sequence diagram.

The area except for the message control field 1 (CONTROL1) of Mailbox is consisted of RAM, and not initialized at a reset. After a power-on reset, all the Mailboxes should be initialized in the configuration mode shown in the reset sequence diagram.

13.10.3 CAN Sleep Mode

The supply of main clocks in the modules is stopped in CAN sleep mode. Therefore, registers other than MCR, GSR, IRR, and IMR should not be accessed in CAN sleep mode.

13.10.4 Register Access

When the CAN bus receive frame is being stored in the Mailbox with the CAN communication functions of RCAN-ET, accessing the Mailbox area generates 0 to 5 peripheral bus cycles as a wait.

13.10.5 Interrupts

As shown in table 13.9, the Mailbox 0 receive interrupt enables the DMAC activation. When an interrupt is specified as to be activated by the Mailbox 0 receive interrupt and cleared by the interrupt source at the DMA transfer, up to the message control field 1 (CONTROL1) of Mailbox 0 should be read using the block transfer mode.

Section 14 Synchronous Serial Communication Unit (SSU)

This LSI has three independent synchronous serial communication unit (SSU) channels. The SSU has master mode in which this LSI outputs clocks as a master device for synchronous serial communication and slave mode in which clocks are input from an external device for synchronous serial communication. Synchronous serial communication can be performed with devices having different clock polarity and clock phase. Figure 14.1 is a block diagram of the SSU.

14.1 Features

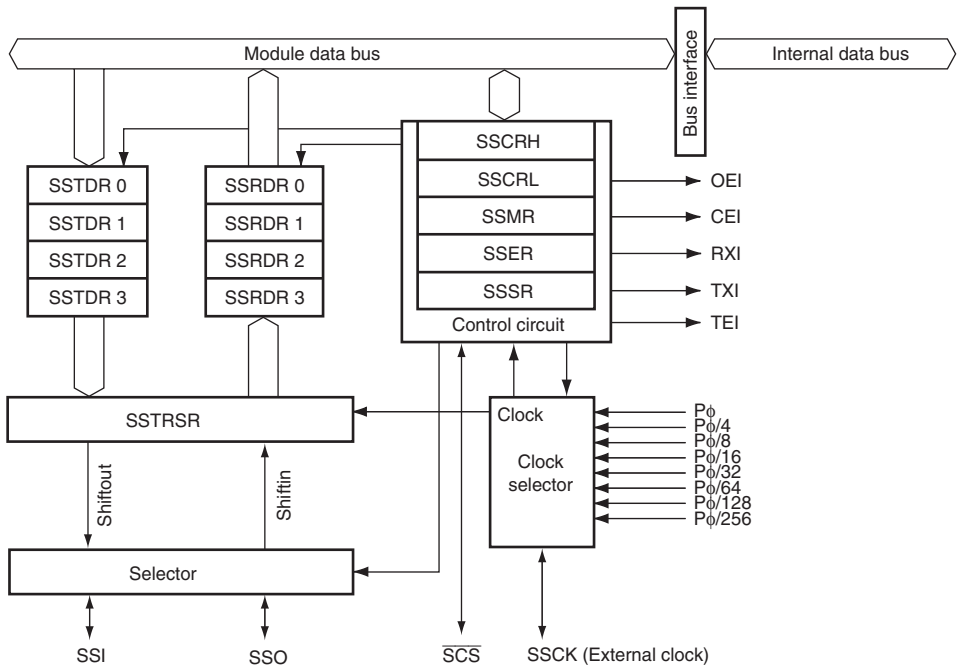
- Choice of SSU mode and clock synchronous mode
- Choice of master mode and slave mode
- Choice of standard mode and bidirectional mode
- Synchronous serial communication with devices with different clock polarity and clock phase
- Choice of 8/16/24/32-bit width of transmit/receive data
- Full-duplex communication capability

  The shift register is incorporated, enabling transmission and reception to be executed simultaneously.
- Consecutive serial communication
- Choice of LSB-first or MSB-first transfer
- Choice of a clock source

  Pϕ/4, Pϕ/8, Pϕ/16, Pϕ/32, Pϕ/64, Pϕ/128, Pϕ/256, or an external clock
- Five interrupt sources

  Transmit-end, transmit-data-register-empty, receive-data-full, overrun-error, and conflict error
- Module stop mode can be set

Figure 14.1 shows a block diagram of the SSU.



[Legend]

SSCRH:	SS control register H
SSCRL:	SS control register L
SSCR2:	SS control register 2
SSMR:	SS mode register
SSER:	SS enable register
SSSR:	SS status register
SSTDR0 to SSTDR3:	SS transmit data registers 0 to 3
SSRDR0 to SSRDR3:	SS receive data registers 0 to 3
SSTRSR:	SS shift register

Figure 14.1 Block Diagram of SSU

14.2 Input/Output Pins

Table 14.1 shows the SSU pin configuration.

Table 14.1 Pin Configuration

Channel	Abbr.*	I/O	Function
0	SSCK0	I/O	Channel 0 SSU clock input/output
	SSI0	I/O	Channel 0 SSU data input/output
	SSO0	I/O	Channel 0 SSU data input/output
	$\overline{\text{SCS0}}$	I/O	Channel 0 SSU chip select input/output
1	SSCK1	I/O	Channel 1 SSU clock input/output
	SSI1	I/O	Channel 1 SSU data input/output
	SSO1	I/O	Channel 1 SSU data input/output
	$\overline{\text{SCS1}}$	I/O	Channel 1 SSU chip select input/output
2	SSCK2	I/O	Channel 2 SSU clock input/output
	SSI2	I/O	Channel 2 SSU data input/output
	SSO2	I/O	Channel 2 SSU data input/output
	$\overline{\text{SCS2}}$	I/O	Channel 2 SSU chip select input/output

Note: * Because channel numbers are omitted in later descriptions, these are shown SSCK, SSI, SSO, and $\overline{\text{SCS}}$.

14.3 Register Descriptions

The SSU has the following registers.

(1) Channel 0

- SS control register H_0 (SSCRH_0)
- SS control register L_0 (SSCRL_0)
- SS mode register_0 (SSMR_0)
- SS enable register_0 (SSER_0)
- SS status register_0 (SSSR_0)
- SS control register 2_0 (SSCR2_0)
- SS transmit data register 0_0 (SSTDR0_0)
- SS transmit data register 1_0 (SSTDR1_0)
- SS transmit data register 2_0 (SSTDR2_0)
- SS transmit data register 3_0 (SSTDR3_0)
- SS receive data register 0_0 (SSRDR0_0)
- SS receive data register 1_0 (SSRDR1_0)
- SS receive data register 2_0 (SSRDR2_0)
- SS receive data register 3_0 (SSRDR3_0)
- SS shift register_0 (SSTRSR_0)

(2) Channel 1

- SS control register H_1 (SSCRH_1)
- SS control register L_1 (SSCRL_1)
- SS mode register_1 (SSMR_1)
- SS enable register_1 (SSER_1)
- SS status register_1 (SSSR_1)
- SS control register 2_1 (SSCR2_1)
- SS transmit data register 0_1 (SSTDR0_1)
- SS transmit data register 1_1 (SSTDR1_1)
- SS transmit data register 2_1 (SSTDR2_1)
- SS transmit data register 3_1 (SSTDR3_1)
- SS receive data register 0_1 (SSRDR0_1)
- SS receive data register 1_1 (SSRDR1_1)
- SS receive data register 2_1 (SSRDR2_1)
- SS receive data register 3_1 (SSRDR3_1)
- SS shift register_1 (SSTRSR_1)

(3) Channel 2

- SS control register H_2 (SSCRH_2)
- SS control register L_2 (SSCRL_2)
- SS mode register_2 (SSMR_2)
- SS enable register_2 (SSER_2)
- SS status register_2 (SSSR_2)
- SS control register 2_2 (SSCR2_2)
- SS transmit data register 0_2 (SSTDR0_2)
- SS transmit data register 1_2 (SSTDR1_2)
- SS transmit data register 2_2 (SSTDR2_2)
- SS transmit data register 3_2 (SSTDR3_2)
- SS receive data register 0_2 (SSRDR0_2)
- SS receive data register 1_2 (SSRDR1_2)
- SS receive data register 2_2 (SSRDR2_2)
- SS receive data register 3_2 (SSRDR3_2)
- SS shift register_2 (SSTRSR_2)

14.3.1 SS Control Register H (SSCRH)

SSCRH specifies master/slave device selection, bidirectional mode enable, SSO pin output value selection, SSCK pin selection, and $\overline{\text{SCS}}$ pin selection.

Bit	7	6	5	4	3	2	1	0
Bit Name	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0
Initial Value	0	0	0	0	1	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	MSS	0	R/W	Master/Slave Device Select Selects that this module is used in master mode or slave mode. When master mode is selected, transfer clocks are output from the SSCK pin. When the CE bit in SSSR is set, this bit is automatically cleared. 0: Slave mode is selected. 1: Master mode is selected.
6	BIDE	0	R/W	Bidirectional Mode Enable Selects that both serial data input pin and output pin are used or one of them is used. However, transmission and reception are not performed simultaneously when bidirectional mode is selected. For details, section 14.4.3, Relationship between Data Input/Output Pins and Shift Register. 0: Standard mode (two pins are used for data input and output) 1: Bidirectional mode (one pin is used for data input and output)
5	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
4	SOL	0	R/W	Serial Data Output Value Select The serial data output retains its level of the last bit after completion of transmission. The output level before or after transmission can be specified by setting this bit. When specifying the output level, use the MOV instruction after clearing the SOLP bit to 0. Since writing to this bit during data transmission causes malfunctions, this bit should not be changed. 0: Serial data output is changed to low. 1: Serial data output is changed to high.
3	SOLP	1	R/W	SOL Bit Write Protect When changing the output level of serial data, set the SOL bit to 1 or clear the SOL bit to 0 after clearing the SOLP bit to 0 using the MOV instruction. 0: Output level can be changed by the SOL bit 1: Output level cannot be changed by the SOL bit. This bit is always read as 1.
2	SCKS	0	R/W	SSCK Pin Select Selects that the SSCK pin functions as a port or a serial clock pin. When the SSCK pin is used as a serial clock pin, this bit must be set to 1. 0: Functions as an I/O port. 1: Functions as a serial clock.
1	CSS1	0	R/W	$\overline{\text{SCS}}$ Pin Select
0	CSS0	0	R/W	Select that the $\overline{\text{SCS}}$ pin functions as a port or $\overline{\text{SCS}}$ input or output. However, when MSS = 0, the $\overline{\text{SCS}}$ pin functions as an input pin regardless of the CSS1 and CSS0 settings. 00: I/O port 01: Function as $\overline{\text{SCS}}$ input 10: Function as $\overline{\text{SCS}}$ automatic input/output (function as $\overline{\text{SCS}}$ input before and after transfer and output a low level during transfer) 11: Function as $\overline{\text{SCS}}$ automatic output (outputs a high level before and after transfer and outputs a low level during transfer)

14.3.2 SS Control Register L (SSCRL)

SSCRL selects operating mode, software reset, and transmit/receive data length.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	SSUMS	SRES	—	—	—	DATS1	DATS0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
6	SSUMS	0	R/W	Selects transfer mode from SSU mode and clock synchronous mode. 0: SSU mode 1: Clock synchronous mode
5	SRES	0	R/W	Software Reset Setting this bit to 1 forcibly resets the SSU internal sequencer. After that, this bit is automatically cleared. The ORER, TEND, TDRE, RDRF, and CE bits in SSSR and the TE and RE bits in SSER are also initialized. Values of other bits for SSU registers are held. To stop transfer, set this bit to 1 to reset the SSU internal sequencer.
4 to 2	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
1	DATS1	0	R/W	Transmit/Receive Data Length Select
0	DATS0	0	R/W	Select serial data length. 00: 8 bits 01: 16 bits 10: 32 bits 11: 24 bits

14.3.3 SS Mode Register (SSMR)

SSMR selects the MSB first/LSB first, clock polarity, clock phase, and clock rate of synchronous serial communication.

Bit	7	6	5	4	3	2	1	0
Bit Name	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	MLS	0	R/W	MSB First/LSB First Select Selects that the serial data is transmitted in MSB first or LSB first. 0: LSB first 1: MSB first
6	CPOS	0	R/W	Clock Polarity Select Selects the SSCK clock polarity. 0: High output in idle mode, and low output in active mode 1: Low output in idle mode, and high output in active mode
5	CPHS	0	R/W	Clock Phase Select (Only for SSU Mode) Selects the SSCK clock phase. 0: Data changes at the first edge. 1: Data is latched at the first edge.
4, 3	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
2 1 0	CKS2 CKS1 CKS0	0 0 0	R/W R/W R/W	Transfer Clock Rate Select Select the transfer clock rate (prescaler division rate) when an internal clock is selected. 000: Reserved 100: Pϕ/32 001: Pϕ/4 101: Pϕ/64 010: Pϕ/8 110: Pϕ/128 011: Pϕ/16 111: Pϕ/256

14.3.4 SS Enable Register (SSER)

SSER performs transfer/receive control of synchronous serial communication and setting of interrupt enable.

Bit	7	6	5	4	3	2	1	0
Bit Name	TE	RE	—	—	TEIE	TIE	RIE	CEIE
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	TE	0	R/W	Transmit Enable When this bit is set to 1, transmission is enabled.
6	RE	0	R/W	Receive Enable When this bit is set to 1, reception is enabled.
5, 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
3	TEIE	0	R/W	Transmit End Interrupt Enable When this bit is set to 1, a TEI interrupt request is enabled.
2	TIE	0	R/W	Transmit Interrupt Enable When this bit is set to 1, a TXI interrupt request is enabled.
1	RIE	0	R/W	Receive Interrupt Enable When this bit is set to 1, an RXI interrupt request and an OEI interrupt request are enabled.
0	CEIE	0	R/W	Conflict Error Interrupt Enable When this bit is set to 1, a CEI interrupt request is enabled.

14.3.5 SS Status Register (SSSR)

SSSR is a status flag register for interrupts.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	ORER	—	—	TEND	TDRE	RDRF	CE
Initial Value	0	0	0	0	0	1	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved This bit is always read as 0. The write value should always be 0.
6	ORER	0	R/W	Overrun Error If the next data is received while RDRF = 1, an overrun error occurs, indicating abnormal termination. SSRDR stores 1-frame receive data before an overrun error occurs and loses data to be received later. While ORER = 1, consecutive serial reception cannot be continued. Serial transmission cannot be continued, either. [Setting condition] When one byte of the next reception is completed with RDRF = 1 [Clearing condition] When writing 0 after reading ORER = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)
5, 4	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
3	TEND	1	R	Transmit End [Setting condition] • When the last bit of transmit data is transmitted while the TENDSTS bit in SSCR2 is cleared to 0 and the TDRE bit is set to 1 • After the last bit of transmit data is transmitted while the TENDSTS bit in SSCR2 is set to 1 and the TDRE bit is set to 1 [Clearing conditions] • When writing 0 after reading TEND = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When writing data to SSTDR
2	TDRE	1	R/W	Transmit Data Empty Indicates whether or not SSTDR contains transmit data. [Setting conditions] • When the TE bit in SSER is 0 • When data is transferred from SSTDR to SSTRSR and SSTDR is ready to be written to. [Clearing conditions] • When writing 0 after reading TDRE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When writing data to SSTDR with TE = 1
1	RDRF	0	R/W	Receive Data Register Full Indicates whether or not SSRDR contains receive data. [Setting condition] • When receive data is transferred from SSTRSR to SSRDR after successful serial data reception [Clearing conditions] • When writing 0 after reading RDRF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When reading receive data from SSRDR

Bit	Bit Name	Initial Value	R/W	Description
0	CE	0	R/W	Conflict/Incomplete Error Indicates that a conflict error has occurred when 0 is externally input to the $\overline{SCS}$ pin with SSUMS = 0 (SSU mode) and MSS = 1 (master mode). If the $\overline{SCS}$ pin level changes to 1 with SSUMS = 0 (SSU mode) and MSS = 0 (slave mode), an incomplete error occurs because it is determined that a master device has terminated the transfer. Data reception does not continue while the CE bit is set to 1. Serial transmission also does not continue. Reset the SSU internal sequencer by setting the SRES bit in SSCRL to 1 before resuming transfer after incomplete error. [Setting condition] • When a low level is input to the $\overline{SCS}$ pin in master mode (the MSS bit in SSCRH is set to 1) • When the $\overline{SCS}$ pin is changed to 1 during transfer in slave mode (the MSS bit in SSCRH is cleared to 0) [Clearing condition] • When writing 0 after reading CE = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.)

14.3.6 SS Control Register 2 (SSCR2)

SSCR2 is a register that enables/disables the open-drain outputs of the SSO, SSI, SSCK, and $\overline{\text{SCS}}$ pins, selects the assert timing of the $\overline{\text{SCS}}$ pin, data output timing of the SSO pin, and set timing of the TEND bit.

Bit	7	6	5	4	3	2	1	0
Bit Name	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	SDOS	0	R/W	Serial Data Pin Open Drain Select Selects whether the serial data output pin is used as a CMOS or an NMOS open drain output. Pins to output serial data differ according to the register setting. For details, 14.4.3, Relationship between Data Input/Output Pins and Shift Register. 0: CMOS output 1: NMOS open drain output
6	SSCKOS	0	R/W	SSCK Pin Open Drain Select Selects whether the SSCK pin is used as a CMOS or an NMOS open drain output. 0: CMOS output 1: NMOS open drain output
5	SCSOS	0	R/W	$\overline{\text{SCS}}$ Pin Open Drain Select Selects whether the $\overline{\text{SCS}}$ pin is used as a CMOS or an NMOS open drain output. 0: CMOS output 1: NMOS open drain output
4	TENDSTS	0	R/W	Selects the timing of setting the TEND bit (valid in SSU and master mode). 0: Sets the TEND bit when the last bit is being transmitted 1: Sets the TEND bit after the last bit is transmitted

Bit	Bit Name	Initial Value	R/W	Description
3	SCSATS	0	R/W	Selects the assertion timing of the $\overline{SCS}$ pin (valid in SSU and master mode). 0: Min. values of t_{LEAD} and t_{LAG} are $1/2 \times t_{SUcyc}$ 1: Min. values of t_{LEAD} and t_{LAG} are $3/2 \times t_{SUcyc}$
2	SSODTS	0	R/W	Selects the data output timing of the SSO pin (valid in SSU and master mode) 0: While BIDE = 0, MSS = 1, and TE = 1 or while BIDE = 1, TE = 1, and RE = 0, the SSO pin outputs data 1: While BIDE = 0, MSS = 1, and TE = 1 or while BIDE = 1, TE = 1, and RE = 0, the SSO pin outputs data while the SCS pin is driven low
1, 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

14.3.7 SS Transmit Data Registers 0 to 3 (SSTDR0 to SSTDR3)

SSTDR is an 8-bit register that stores transmit data. When 8-bit data length is selected by bits DATS1 and DATS0 in SSCRL, SSTDR0 is valid. When 16-bit data length is selected, SSTDR0 and SSTDR1 are valid. When 24-bit data length is selected, SSTDR0, SSTDR1, and SSTDR2 are valid. When 32-bit data length is selected, SSTDR0 to SSTDR3 are valid. Be sure not to access to invalid SSTDRs.

When the SSU detects that SSTRSR is empty, it transfers the transmit data written in SSTDR to SSTRSR and starts serial transmission. If the next transmit data has already been written to SSTDR during serial transmission, the SSU performs consecutive serial transmission.

Although SSTDR can always be read from or written to by the CPU and DMAC, to achieve reliable serial transmission, write transmit data to SSTDR after confirming that the TDRE bit in SSSR is set to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	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

Table 14.2 Correspondence Between DATS Bit Setting and SSTDR

	DATS[1:0] (SSCRL[1:0])			
SSTDR	**00**	**01**	**10**	**11 (Setting Invalid)**
0	Valid	Valid	Valid	Valid
1	Invalid	Valid	Valid	Valid
2	Invalid	Invalid	Valid	Valid
3	Invalid	Invalid	Valid	Invalid

14.3.8 SS Receive Data Registers 0 to 3 (SSRDR0 to SSRDR3)

SSRDR is an 8-bit register that stores receive data. When 8-bit data length is selected by bits DATS1 and DATS0 in SSCRL, SSRDR0 is valid. When 16-bit data length is selected, SSRDR0 and SSRDR1 are valid. When 24-bit data length is selected, SSRDR0, SSRDR1, and SSRDR2 are valid. When 32-bit data length is selected, SSRDR0 to SSRDR3 are valid. Be sure not to access to invalid SSRDR.

When the SSU has received 1-byte data, it transfers the received serial data from SSTRSR to SSRDR where it is stored. After this, SSTRSR is ready for reception. Since SSTRSR and SSRDR function as a double buffer in this way, consecutive receive operations can be performed.

Read SSRDR after confirming that the RDRF bit in SSSR is set to 1.

SSRDR is a read-only register, therefore, cannot be written to by the CPU.

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R

Table 14.3 Correspondence Between DATS Bit Setting and SSRDR

	DATS[1:0] (SSCRL[1:0])			
SSRDR	**00**	**01**	**10**	**11 (Setting Invalid)**
0	Valid	Valid	Valid	Valid
1	Invalid	Valid	Valid	Valid
2	Invalid	Invalid	Valid	Valid
3	Invalid	Invalid	Valid	Invalid

14.3.9 SS Shift Register (SSTRSR)

SSTRSR is a shift register that transmits and receives serial data.

When data is transferred from SSTDR to SSTRSR, bit 0 of transmit data is bit 0 in the SSTDR contents (MLS = 0: LSB first communication) and is bit 7 in the SSTDR contents (MLS = 1: MSB first communication). The SSU transfers data from the LSB (bit 0) in SSTRSR to the SSO pin to perform serial data transmission.

In reception, the SSU sets serial data that has been input via the SSI pin in SSTRSR from the LSB (bit 0). When 1-byte data has been received, the SSTRSR contents are automatically transferred to SSRDR. SSTRSR cannot be directly accessed by the CPU.

14.4 Operation

14.4.1 Transfer Clock

A transfer clock can be selected from eight internal clocks and an external clock. When using this module, set the SCKS bit in SSCRH to 1 to select the SSCK pin as a serial clock. When the MSS bit in SSCRH is 1, an internal clock is selected and the SSCK pin is used as an output pin. When transfer is started, the clock with the transfer rate set by bits CKS2 to CKS0 in SSMR is output from the SSCK pin. When MSS = 0, an external clock is selected and the SSCK pin is used as an input pin.

14.4.2 Relationship of Clock Phase, Polarity, and Data

The relationship of clock phase, polarity, and transfer data depends on the combination of the CPOS and CPHS bits in SSMR. Figure 14.2 shows the relationship. When SSUMS = 1, the CPHS setting is invalid although the CPOS setting is valid.

Setting the MLS bit in SSMR selects that MSB or LSB first communication. When MLS = 0, data is transferred from the LSB to the MSB. When MLS = 1, data is transferred from the MSB to the LSB.

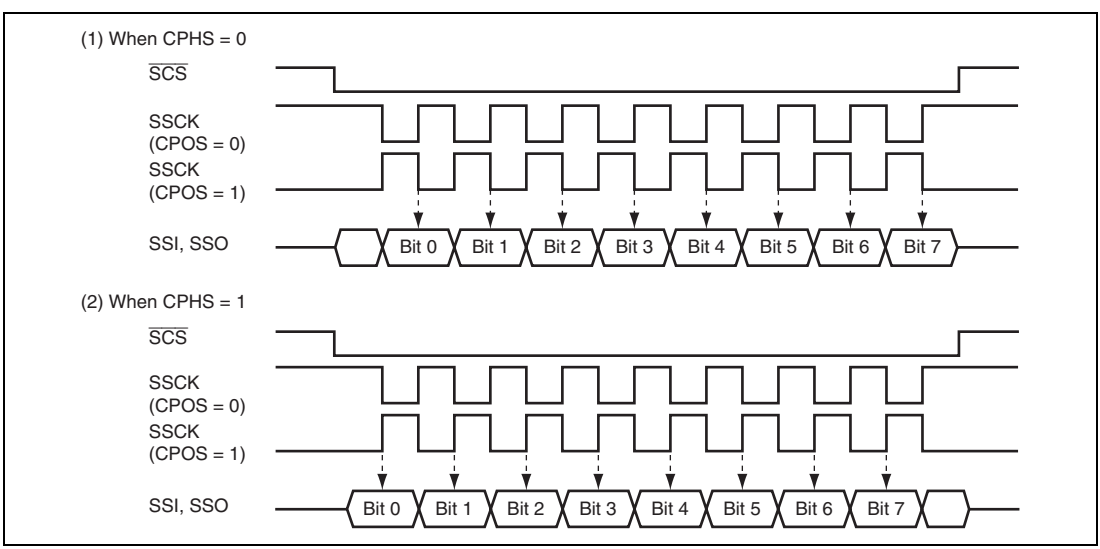


Figure 14.2 Relationship of Clock Phase, Polarity, and Data

14.4.3 Relationship between Data Input/Output Pins and Shift Register

The connection between data input/output pins and the SS shift register (SSTRSR) depends on the combination of the MSS and BIDE bits in SSCRH and the SSUMS bit in SSCRL. Figure 14.3 shows the relationship.

The SSU transmits serial data from the SSO pin and receives serial data from the SSI pin when operating with BIDE = 0 and MSS = 1 (standard, master mode) (see figure 14.3 (1)). The SSU transmits serial data from the SSI pin and receives serial data from the SSO pin when operating with BIDE = 0 and MSS = 0 (standard, slave mode) (see figure 14.3 (2)).

The SSU transmits and receives serial data from the SSO pin regardless of master or slave mode when operating with BIDE = 1 (bidirectional mode) (see figures 14.3 (3) and (4)).

However, even if both the TE and RE bits are set to 1, transmission and reception are not performed simultaneously. Either the TE or RE bit must be selected.

The SSU transmits serial data from the SSO pin and receives serial data from the SSI pin when operating with SSUMS = 1. The SSCK pin outputs the internal clock when MSS = 1 and function as an input pin when MSS = 0 (see figures 14.3 (5) and (6)).

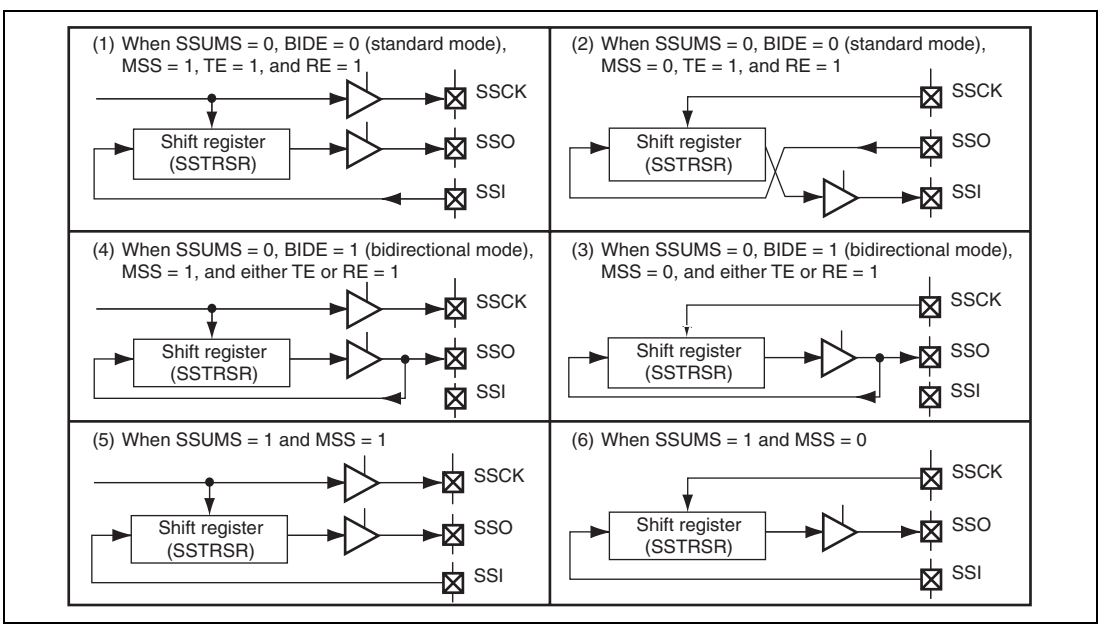


Figure 14.3 Relationship between Data Input/Output Pins and the Shift Register

14.4.4 Communication Modes and Pin Functions

The SSU switches the input/output pin (SSI, SSO, SSCK, and $\overline{\text{SCS}}$) functions according to the communication modes and register settings. When a pin is used as an input pin, set the corresponding bit in the input buffer control register (ICR) to 1. The relationship of communication modes and input/output pin functions are shown in tables 14.4 to 14.6.

Table 14.4 Communication Modes and Pin States of SSI and SSO Pins

Communication Mode	Register Setting					Pin State	
	SSUMS	BIDE	MSS	TE	RE	SSI	SSO
SSU communication mode	0	0	0	0	1	—	Input
				1	0	Output	—
					1	Output	Input
			1	0	1	Input	—
				1	0	—	Output
					1	Input	Output
SSU (bidirectional) communication mode	0	1	0	0	1	—	Input
				1	0	—	Output
			1	0	1	—	Input
				1	0	—	Output
Clock synchronous communication mode	1	0	0	0	1	Input	—
				1	0	—	Output
					1	Input	Output
			1	0	1	Input	—
				1	0	—	Output
					1	Input	Output

[Legend]
—: Not used as SSU pin

Table 14.5 Communication Modes and Pin States of SSCK Pin

Communication Mode	Register Setting			Pin State
	SSUMS	MSS	SCKS	SSCK
SSU communication mode	0	0	0	—
			1	Input
		1	0	—
			1	Output
Clock synchronous communication mode	1	0	0	—
			1	Input
		1	0	—
			1	Output

[Legend]

—: Not used as SSU pin

Table 14.6 Communication Modes and Pin States of $\overline{\text{SCS}}$ Pin

Communication Mode	Register Setting				Pin State
	SSUMS	MSS	CSS1	CSS0	$\overline{\text{SCS}}$
SSU communication mode	0	0	×	×	Input
		1	0	0	—
			0	1	Input
			1	0	Automatic input/output
			1	1	Output
Clock synchronous communication mode	1	×	×	×	—

[Legend]

×: Don't care

—: Not used as SSU pin

14.4.5 SSU Mode

In SSU mode, data communications are performed via four lines: clock line (SSCK), data input line (SSI or SSO), data output line (SSI or SSO), and chip select line ($\overline{\text{SCS}}$).

In addition, the SSU supports bidirectional mode in which a single pin functions as data input and data output lines.

(1) Initial Settings in SSU Mode

Figure 14.4 shows an example of the initial settings in SSU mode. Before data transfer, clear both the TE and RE bits in SSER to 0 to set the initial values.

Note: Before changing operating modes and communications formats, clear both the TE and RE bits to 0. Although clearing the TE bit to 0 sets the TDRE bit to 1, clearing the RE bit to 0 does not change the values of the RDRF and ORER bits and SSRDR. Those bits retain the previous values.

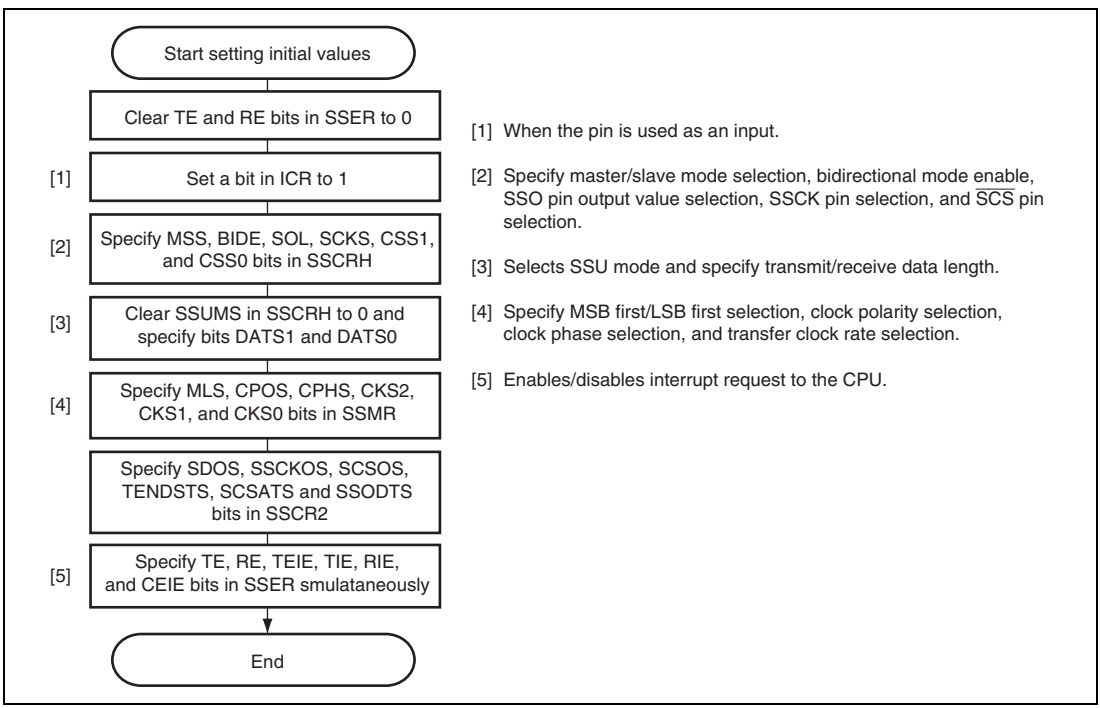


Figure 14.4 Example of Initial Settings in SSU Mode

(2) Data Transmission

Figure 14.5 shows an example of transmission operation, and figure 14.6 shows a flowchart example of data transmission.

When transmitting data, the SSU operates as shown below.

In master mode, the SSU outputs a transfer clock and data. In slave mode, when a low level signal is input to the $\overline{\text{SCS}}$ pin and a transfer clock is input to the SSCK pin, the SSU outputs data in synchronization with the transfer clock.

Writing transmit data to SSTDR after the TE bit is set to 1 clears the TDRE bit in SSSR to 0, and the SSTDR contents are transferred to SSTRSR. After that, the SSU sets the TDRE bit to 1 and starts transmission. At this time, if the TIE bit in SSER is set to 1, a TXI interrupt is generated.

When 1-frame data has been transferred with TDRE = 0, the SSTDR contents are transferred to SSTRSR to start the next frame transmission. When the 8th bit of transmit data has been transferred with TDRE = 1, the TEND bit in SSSR is set to 1 and the state is retained. At this time, if the TEIE bit is set to 1, a TEI interrupt is generated. After transmission, the output level of the SSCK pin is fixed high when CPOS = 0 and low when CPOS = 1.

While the ORER bit in SSSR is set to 1, transmission is not performed. Check that the ORER bit is cleared to 0.

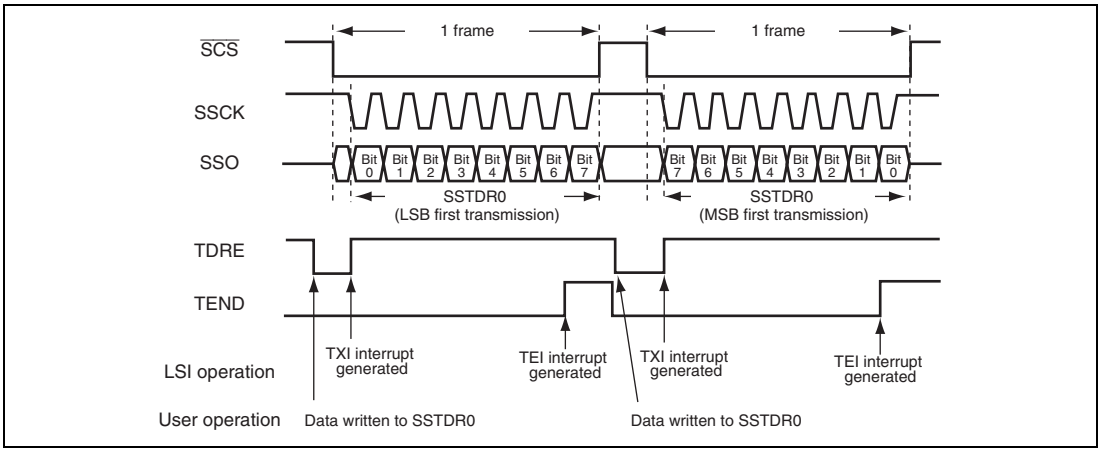


Figure 14.5 Example of Transmission Operation (SSU Mode)
When 8-bit data length is selected (SSTDR0 is valid) with CPOS = 0 and CPHS = 0 (1)

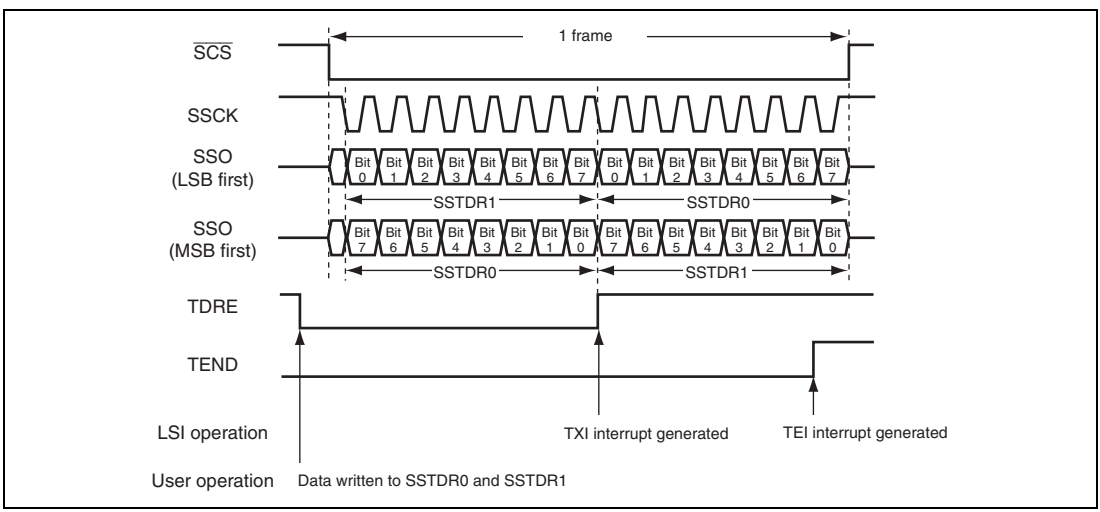


Figure 14.5 Example of Transmission Operation (SSU Mode) When 16-bit data length is selected (SSTDR0 and SSTDR1 are valid) with CPOS = 0 and CPHS = 0 (2)

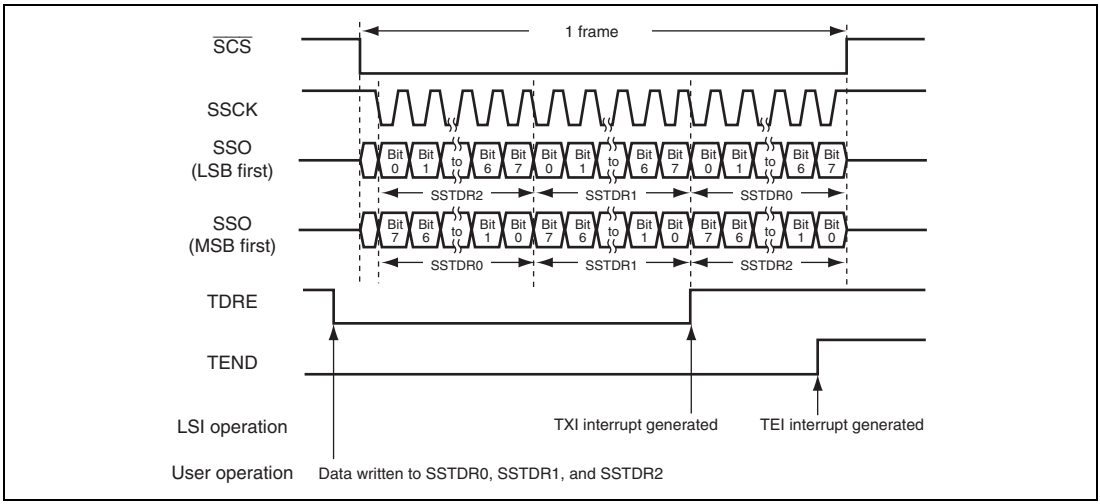


Figure 14.5 Example of Transmission Operation (SSU Mode) When 24-bit data length is selected (SSTDR0, SSTDR1, and SSTDR2 are valid) with CPOS = 0 and CPHS = 0 (3)

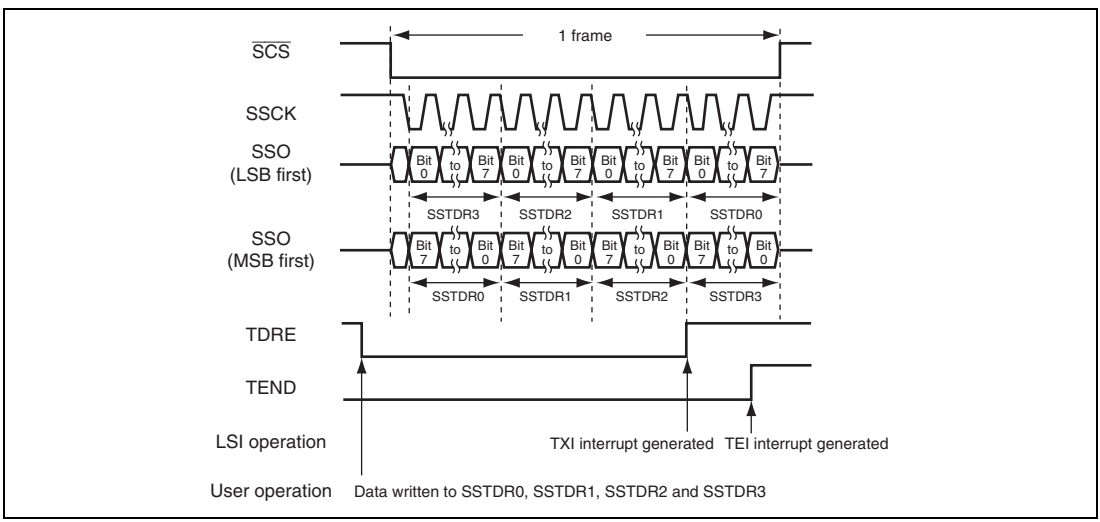
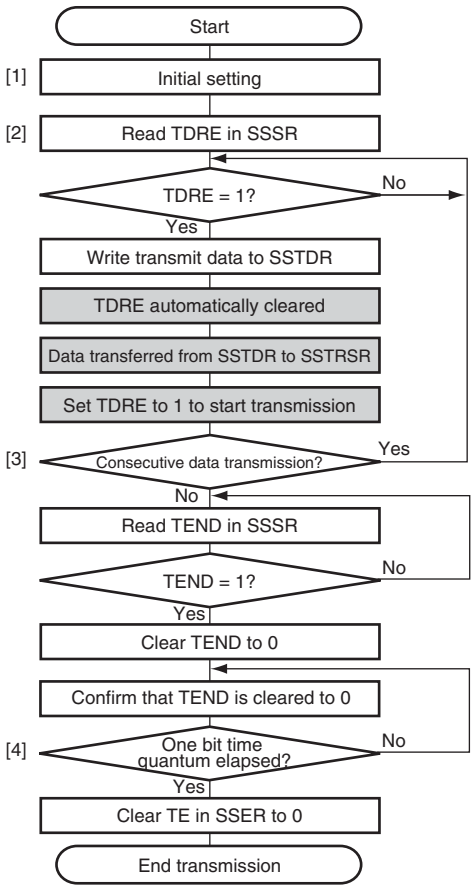


Figure 14.5 Example of Transmission Operation (SSU Mode)
When 32-bit data length is selected (SSTDR0, SSTDR1, SSTDR2 and SSTDR3 are valid) with CPOS = 0 and CPHS = 0 (4)



[1] Initial setting:
Specify the transmit data format.

[2] Check that the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit is 1. The TDRE bit is automatically cleared to 0 and transmission is started by writing data to SSTDR.

[3] Procedure for consecutive data transmission:
To continue data transmission, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

[4] Procedure for data transmission end:
To end data transmission, confirm that the TEND bit is cleared to 0. After completion of transmitting the last bit, clear the TE bit to 0.

Note: Hatching boxes represent SSU internal operations.

Figure 14.6 Flowchart Example of Data Transmission (SSU Mode)

(3) Data Reception

Figure 14.7 shows an example of reception operation, and figure 14.8 shows a flowchart example of data reception. When receiving data, the SSU operates as shown below.

After setting the RE bit to 1 and dummy-reading SSRDR, the SSU starts data reception.

In master mode, the SSU outputs a transfer clock and receives data. In slave mode, when a low level signal is input to the $\overline{SCS}$ pin and a transfer clock is input to the SSCK pin, the SSU receives data in synchronization with the transfer clock.

When 1-frame data has been received, the RDRF bit in SSSR is set to 1 and the receive data is stored in SSRDR. At this time, if the RIE bit in SSER is set to 1, an RXI interrupt is generated. The RDRF bit is automatically cleared to 0 by reading SSRDR.

When the RDRF bit has been set to 1 at the 8th rising edge of the transfer clock, the ORER bit in SSSR is set to 1. This indicates that an overrun error (OEI) has occurred. At this time, data reception is stopped. While the ORER bit in SSSR is set to 1, reception is not performed. To resume the reception, clear the ORER bit to 0.

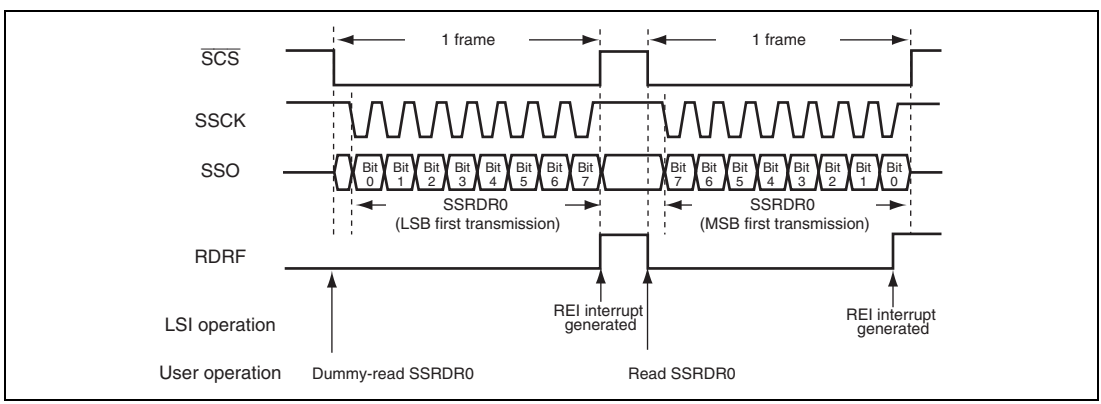


Figure 14.7 Example of Reception Operation (SSU Mode)
When 8-bit data length is selected (SSRDR0 is valid) with CPOS = 0 and CPHS = 0 (1)

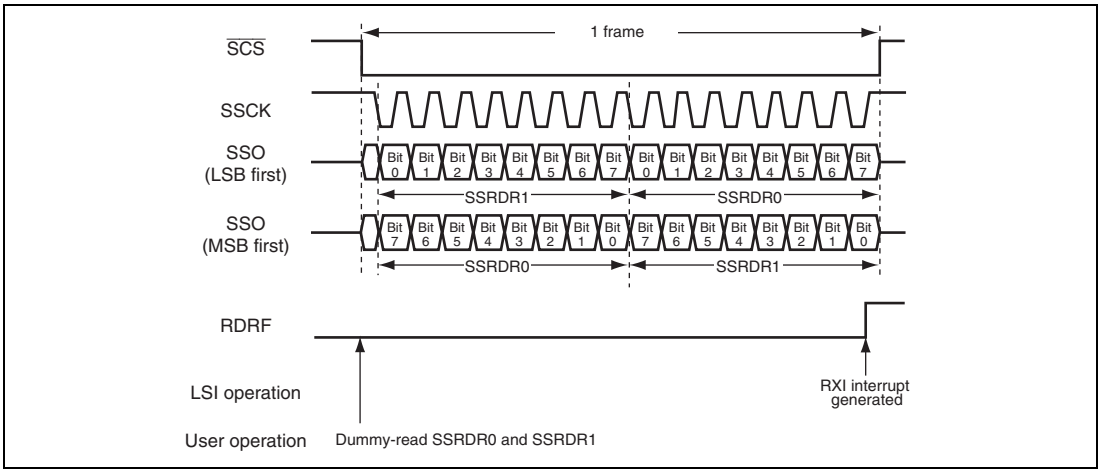


Figure 14.7 Example of Reception Operation (SSU Mode) When 16-bit data length is selected (SSRDR0 and SSRDR1 are valid) with CPOS = 0 and CPHS = 0 (2)

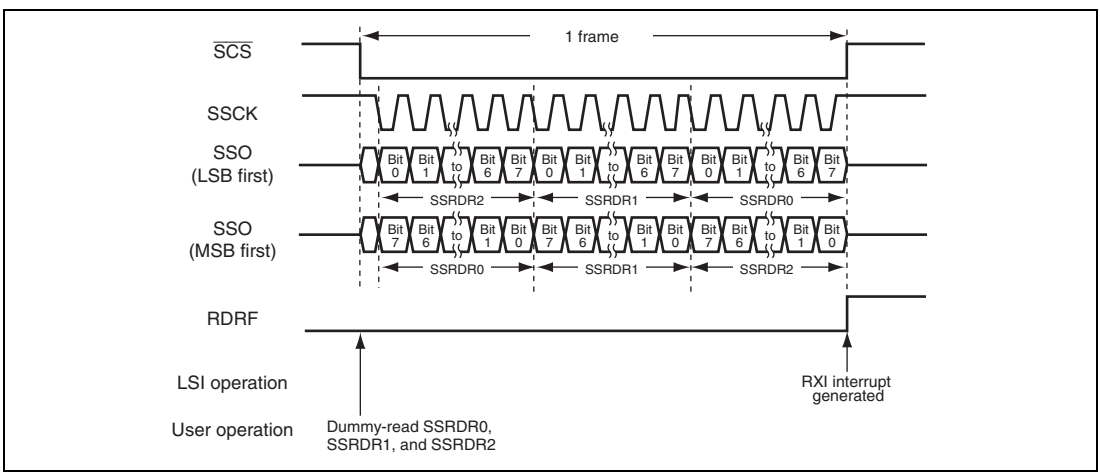


Figure 14.7 Example of Reception Operation (SSU Mode) When 24-bit data length is selected (SSRDR0, SSRDR1, and SSRDR2 are valid) with CPOS = 0 and CPHS = 0 (3)

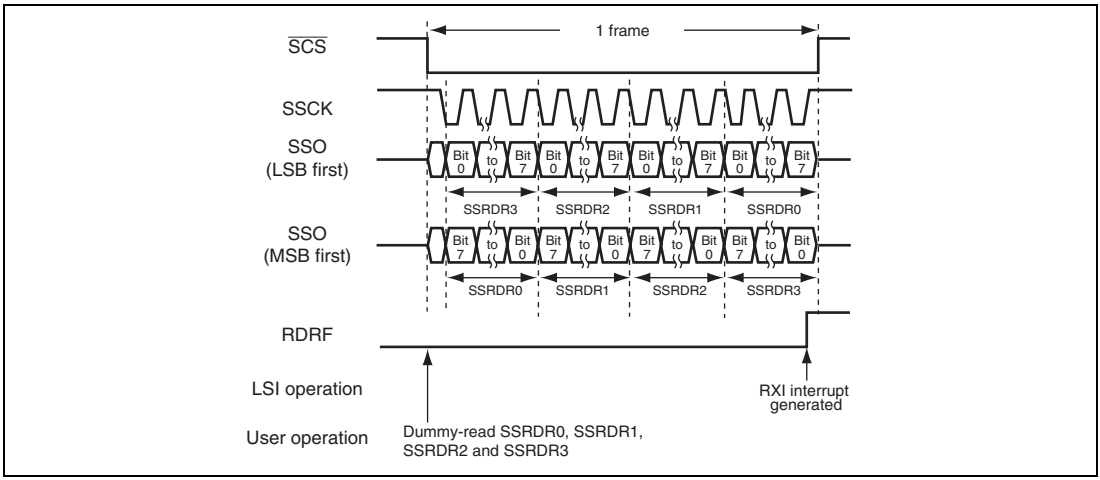
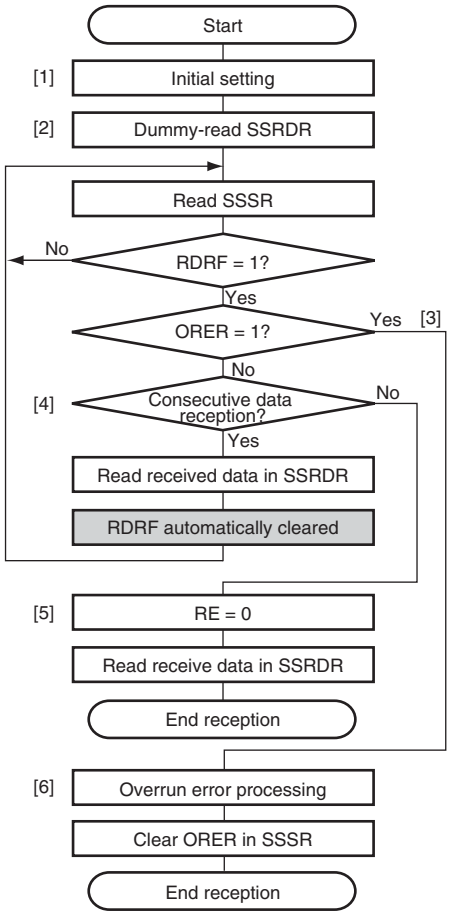


Figure 14.7 Example of Reception Operation (SSU Mode)
When 32-bit data length is selected (SSRDR0, SSRDR1, SSRDR2 and SSRDR3 are valid)
with CPOS = 0 and CPHS = 0 (4)



[1] Initial setting:
Specify the receive data format.

[2] Start reception:
When SSRDR is dummy-read with RE = 1, reception is started.

[3], [6] Receive error processing:
When a receive error occurs, execute the designated error processing after reading the ORER bit in SSSR. After that, clear the ORER bit to 0. While the ORER bit is set to 1, transmission or reception is not resumed.

[4] To continue single reception:
When continuing single reception, wait for time of t_{SUcyc} while the RDRF flag is set to 1 and then read receive data in SSRDR.
The next single reception starts after reading receive data in SSRDR.

[5] To complete reception:
To complete reception, read receive data after clearing the RE bit to 0. When reading SSRDR without clearing the RE bit, reception is resumed.

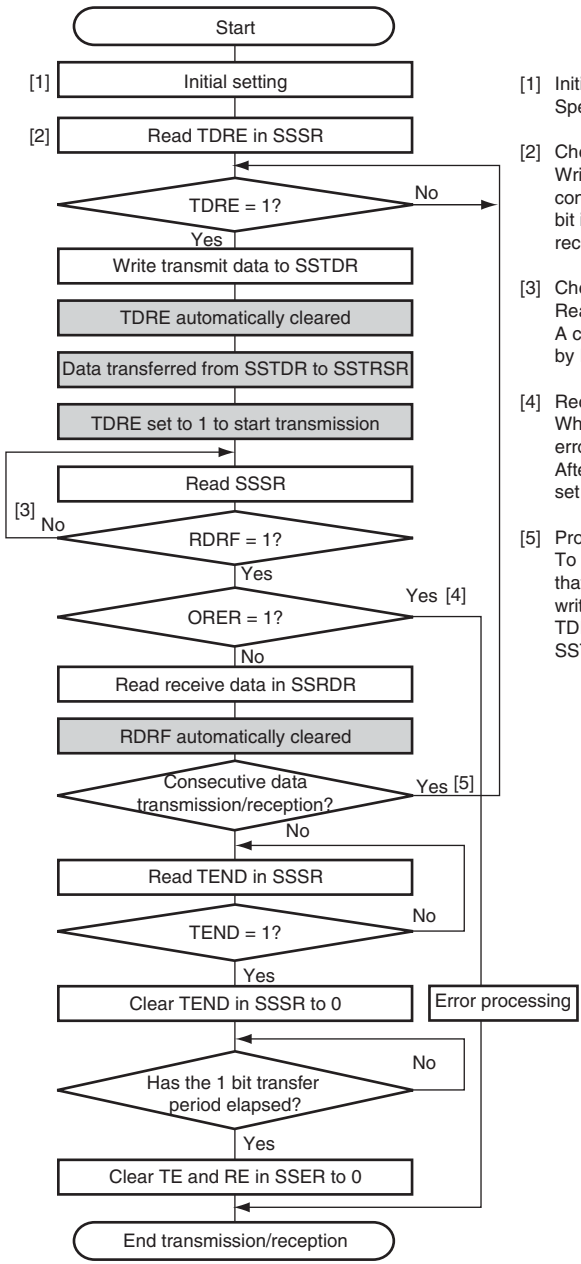
Note: Hatching boxes represent SSU internal operations.

Figure 14.8 Flowchart Example of Data Reception (SSU Mode)

(4) Data Transmission/Reception

Figure 14.9 shows a flowchart example of simultaneous transmission/reception. The data transmission/reception is performed combining the data transmission and data reception as mentioned above. The data transmission/reception is started by writing transmit data to SSTDR with TE = RE = 1.

Before switching transmission mode (TE = 1) or reception mode (RE = 1) to transmission/reception mode (TE = RE = 1), clear the TE and RE bits to 0. When starting the transfer, confirm that the TEND, RDRF, and ORER bits are cleared to 0 before setting the TE or RE bit to 1.



Note: Hatching boxes represent SSU internal operations.

[1] Initial setting:
Specify the transmit/receive data format.

[2] Check the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. The TDRE bit is automatically cleared to 0 and transmission/reception is started by writing data to SSTDR.

[3] Check the SSU state:
Read SSSR confirming that the RDRF bit is 1.
A change of the RDRF bit (from 0 to 1) can be notified by RXI interrupt.

[4] Receive error processing:
When a receive error occurs, execute the designated error processing after reading the ORER bit in SSSR. After that, clear the ORER bit to 0. While the ORER bit is set to 1, transmission or reception is not resumed.

[5] Procedure for consecutive data transmission/reception:
To continue serial data transmission/reception, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

Figure 14.9 Flowchart Example of Simultaneous Transmission/Reception (SSU Mode)

14.4.6 $\overline{\text{SCS}}$ Pin Control and Conflict Error

When bits CSS1 and CSS0 in SSCRH are specified to B'10 and the SSUMS bit in SSCRL is cleared to 0, the $\overline{\text{SCS}}$ pin functions as an input (Hi-Z) to detect conflict error. The conflict detection period is from setting the MSS bit in SSCRH to 1 to starting serial transfer and after transfer ends. When a low level signal is input to the $\overline{\text{SCS}}$ pin within the period, a conflict error occurs. At this time, the CE bit in SSSR is set to 1 and the MSS bit is cleared to 0.

Note: While the CE bit is set to 1, transmission or reception is not resumed. Clear the CE bit to 0 before resuming the transmission or reception.

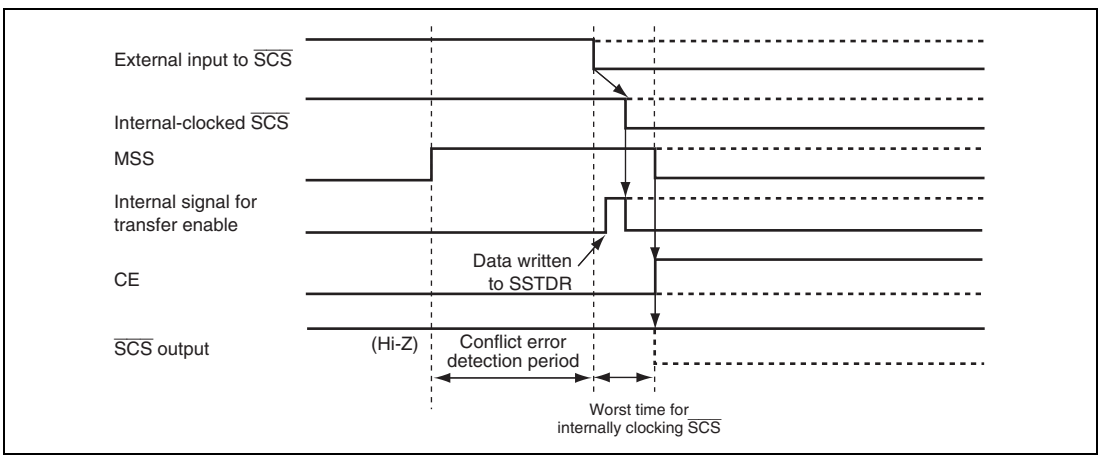


Figure 14.10 Conflict Error Detection Timing (Before Transfer)

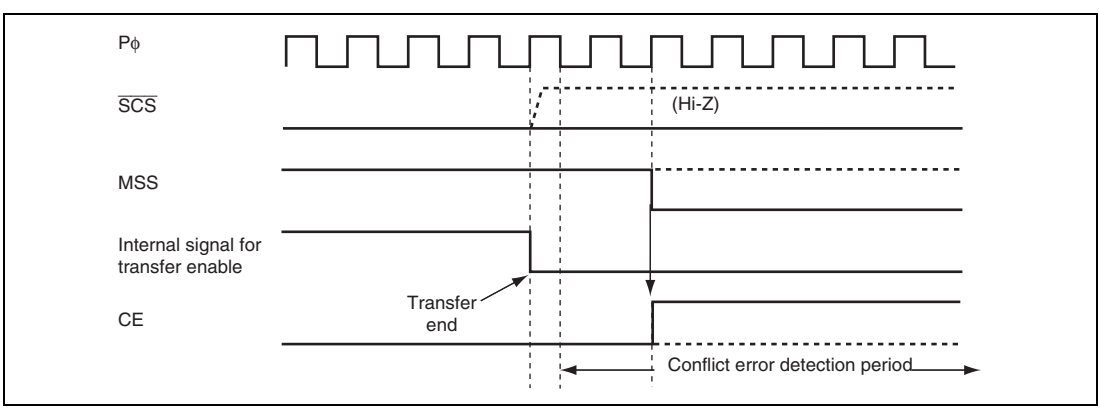


Figure 14.11 Conflict Error Detection Timing (After Transfer End)

14.4.7 Clock Synchronous Communication Mode

In clock synchronous communication mode, data communications are performed via three lines: clock line (SSCK), data input line (SSI), and data output line (SSO).

(1) Initial Settings in Clock Synchronous Communication Mode

Figure 14.12 shows an example of the initial settings in clock synchronous communication mode. Before data transfer, clear both the TE and RE bits in SSER to 0 to set the initial values.

Note: Before changing operating modes and communications formats, clear both the TE and RE bits to 0. Although clearing the TE bit to 0 sets the TDRE bit to 1, clearing the RE bit to 0 does not change the values of the RDRF and ORER bits and SSRDR. Those bits retain the previous values.

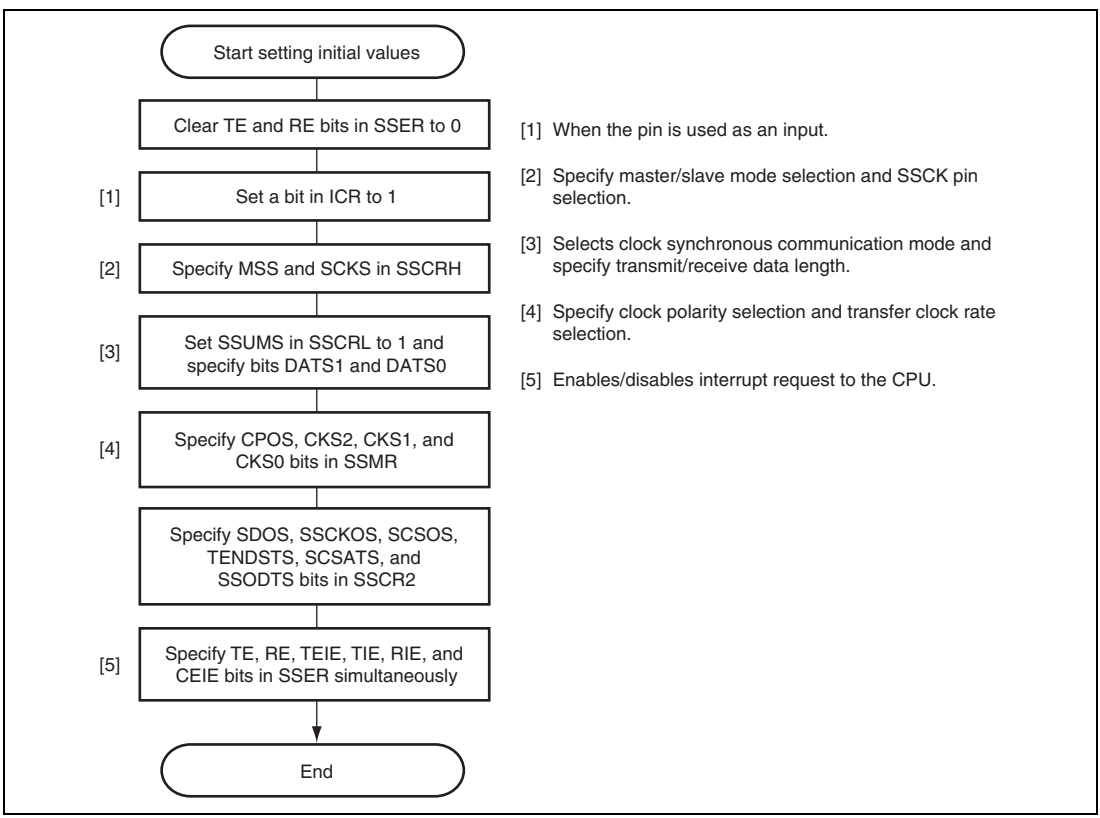


Figure 14.12 Example of Initial Settings in Clock Synchronous Communication Mode

(2) Data Transmission

Figure 14.13 shows an example of transmission operation, and figure 14.14 shows a flowchart example of data transmission. When transmitting data in clock synchronous communication mode, the SSU operates as shown below.

In master mode, the SSU outputs a transfer clock and data. In slave mode, when a transfer clock is input to the SSCK pin, the SSU outputs data in synchronization with the transfer clock.

Writing transmit data to SSTDR after the TE bit is set to 1 clears the TDRE bit in SSSR to 0, and the SSTDR contents are transferred to SSTRSR. After that, the SSU sets the TDRE bit to 1 and starts transmission. At this time, if the TIE bit in SSER is set to 1, a TXI interrupt is generated.

When 1-frame data has been transferred with TDRE = 0, the SSTDR contents are transferred to SSTRSR to start the next frame transmission. When the 8th bit of transmit data has been transferred with TDRE = 1, the TEND bit in SSSR is set to 1 and the state is retained. At this time, if the TEIE bit is set to 1, a TEI interrupt is generated.

While the ORER bit in SSSR is set to 1, transmission is not performed. Check that the ORER bit is cleared to 0.

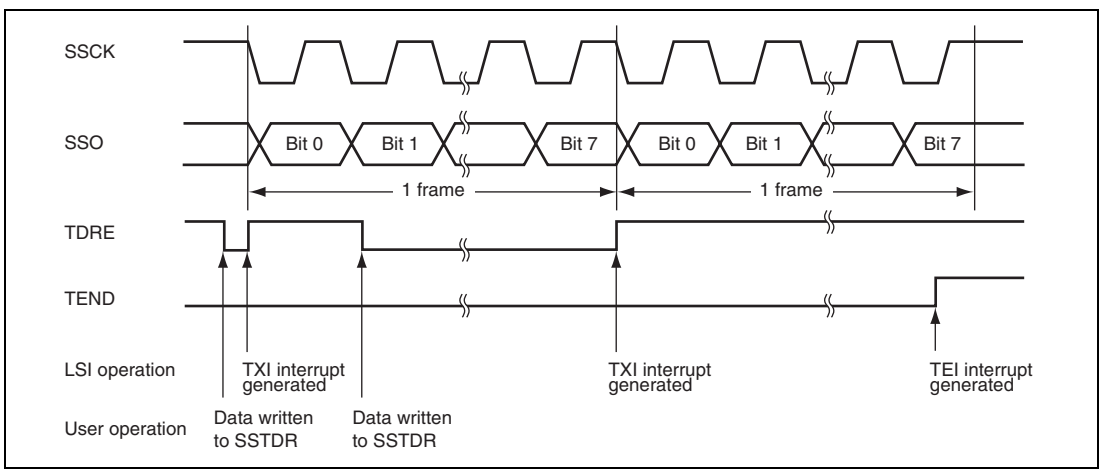
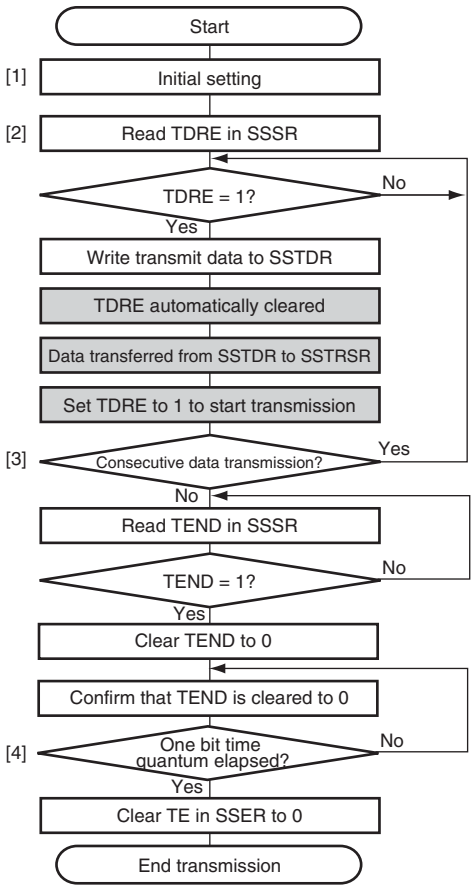


Figure 14.13 Example of Transmission Operation (Clock Synchronous Communication Mode)



[4][1] Initial setting:
Specify the transmit data format.

[2] Check that the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit is 1. The TDRE bit is automatically cleared to 0 and transmission is started by writing data to SSTDR.

[3] Procedure for consecutive data transmission:
To continue data transmission, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

[4] Procedure for data transmission end:
To end data transmission, confirm that the TEND bit is cleared to 0. After completion of transmitting the last bit, clear the TE bit to 0.

Note: Hatching boxes represent SSU internal operations.

Figure 14.14 Flowchart Example of Transmission Operation (Clock Synchronous Communication Mode)

(3) Data Reception

Figure 14.15 shows an example of reception operation, and figure 14.16 shows a flowchart example of data reception. When receiving data, the SSU operates as shown below.

After setting the RE bit in SSER to 1, the SSU starts data reception.

In master mode, the SSU outputs a transfer clock and receives data. In slave mode, when a transfer clock is input to the SSCK pin, the SSU receives data in synchronization with the transfer clock.

When 1-frame data has been received, the RDRF bit in SSSR is set to 1 and the receive data is stored in SSRDR. At this time, if the RIE bit is set to 1, an RXI interrupt is generated. The RDRF bit is automatically cleared to 0 by reading SSRDR.

When the RDRF bit has been set to 1 at the 8th rising edge of the transfer clock, the ORER bit in SSSR is set to 1. This indicates that an overrun error (OEI) has occurred. At this time, data reception is stopped. While the ORER bit in SSSR is set to 1, reception is not performed. To resume the reception, clear the ORER bit to 0.

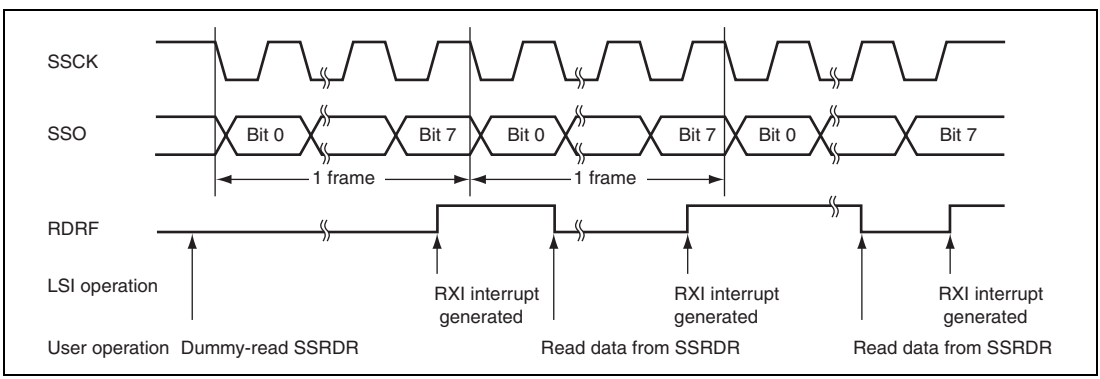
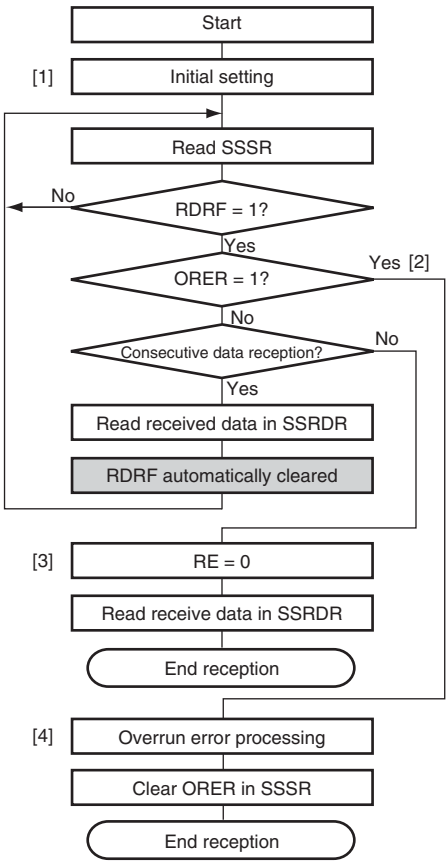


Figure 14.15 Example of Reception Operation (Clock Synchronous Communication Mode)



[1] Initial setting:
Specify the receive data format.

[2], [4] Receive error processing:
When a receive error occurs, execute the designated error processing after reading the ORER bit in SSSR. After that, clear the ORER bit to 0. While the ORER bit is set to 1, transmission or reception is not resumed.

[3] To complete reception:
To complete reception, read receive data after clearing the RE bit to 0. When reading SSRDR without clearing the RE bit, reception is resumed.

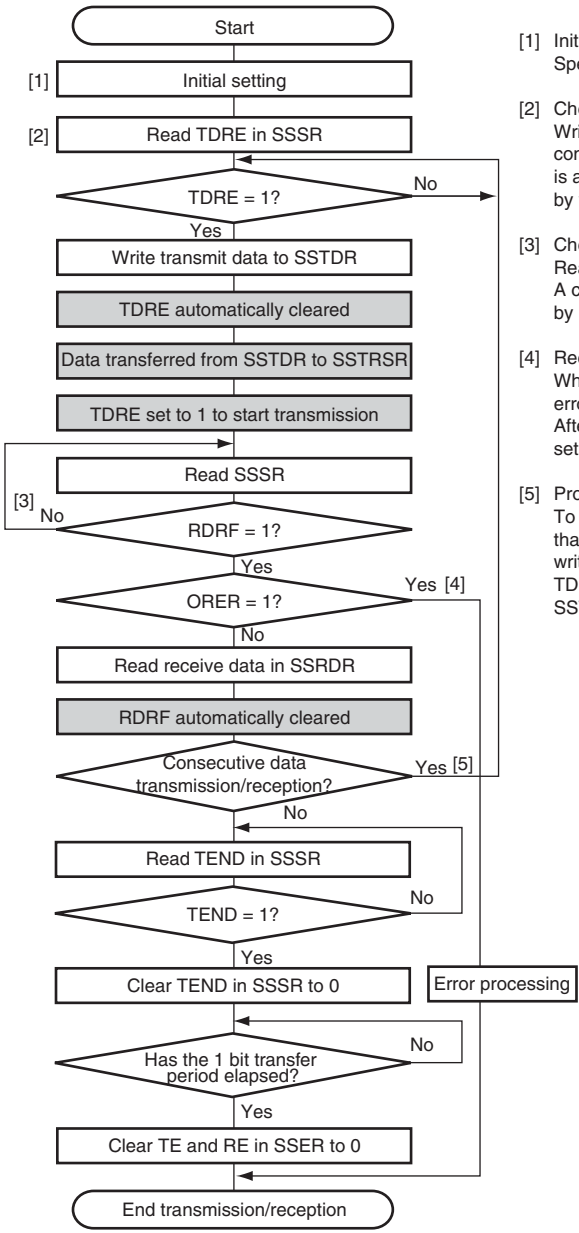
Note: Hatching boxes represent SSU internal operations.

Figure 14.16 Flowchart Example of Data Reception (Clock Synchronous Communication Mode)

(4) Data Transmission/Reception

Figure 14.17 shows a flowchart example of simultaneous transmission/reception. The data transmission/reception is performed combining the data transmission and data reception as mentioned above. The data transmission/reception is started by writing transmit data to SSTDR with TE = RE = 1.

Before switching transmission mode (TE = 1) or reception mode (RE = 1) to transmission/reception mode (TE = RE = 1), clear the TE and RE bits to 0. When starting the transfer, confirm that the TEND, RDRF, and ORER bits are cleared to 0 before setting the TE or RE bits to 1.



[1] Initial setting:
Specify the transmit/receive data format.

[2] Check the SSU state and write transmit data:
Write transmit data to SSTDR after reading and confirming that the TDRE bit in SSSR is 1. The TDRE bit is automatically cleared to 0 and transmission is started by writing data to SSTDR.

[3] Check the SSU state:
Read SSSR confirming that the RDRF bit is 1.
A change of the RDRF bit (from 0 to 1) can be notified by RXI interrupt.

[4] Receive error processing:
When a receive error occurs, execute the designated error processing after reading the ORER bit in SSSR. After that, clear the ORER bit to 0. While the ORER bit is set to 1, transmission or reception is not resumed.

[5] Procedure for consecutive data transmission/reception:
To continue serial data transmission/reception, confirm that the TDRE bit is 1 meaning that SSTDR is ready to be written to. After that, data can be written to SSTDR. The TDRE bit is automatically cleared to 0 by writing data to SSTDR.

Note: Hatching boxes represent SSU internal operations.

Figure 14.17 Flowchart Example of Simultaneous Transmission/Reception (Clock Synchronous Communication Mode)

14.5 Interrupt Requests

The SSU interrupt requests are an overrun error, a conflict error, a receive data register full, transmit data register empty, and a transmit end interrupts. Of these interrupt sources, a receive data register full, a transmit data register empty, and a transmit end interrupts can activate the DMAC for data transfer.

Since both an overrun error and a conflict error interrupts are allocated to the SSERI vector address, and both a transmit data register empty and a transmit end interrupts are allocated to the SSTXI vector address, the interrupt source should be decided by their flags. Table 14.7 lists the interrupt sources.

When an interrupt condition shown in table 14.7 is satisfied, an interrupt is requested. Clear the interrupt source by CPU or DMAC data transfer.

Table 14.7 Interrupt Sources

Channel	Abbreviation	Interrupt Source	Symbol	Interrupt Condition	DMAC Activation
0	SSERI0	Overrun error	OEI0	(RIE = 1) • (ORER = 1)	—
		Conflict error	CEI0	(CEIE = 1) • (CE = 1)	—
	SSRXI0	Receive data register full	RXI0	(RIE = 1) • (RDRF = 1)	Yes
	SSTXI0	Transmit data register empty	TXI0	(TIE = 1) • (TDRE = 1)	Yes
		Transmit end	TEI0	(TEIE = 1) • (TEND = 1)	Yes
1	SSERI1	Overrun error	OEI1	(RIE = 1) • (ORER = 1)	—
		Conflict error	CEI1	(CEIE = 1) • (CE = 1)	—
	SSRXI1	Receive data register full	RXI1	(RIE = 1) • (RDRF = 1)	Yes
	SSTXI1	Transmit data register empty	TXI1	(TIE = 1) • (TDRE = 1)	Yes
		Transmit end	TEI1	(TEIE = 1) • (TEND = 1)	Yes
2	SSERI2	Overrun error	OEI2	(RIE = 1) • (ORER = 1)	—
		Conflict error	CEI2	(CEIE = 1) • (CE = 1)	—
	SSRXI2	Receive data register full	RXI2	(RIE = 1) • (RDRF = 1)	Yes
	SSTXI2	Transmit data register empty	TXI2	(TIE = 1) • (TDRE = 1)	Yes
		Transmit end	TEI2	(TEIE = 1) • (TEND = 1)	Yes

14.6 Usage Note

14.6.1 Setting of Module Stop Mode

The SSU can be enabled/disabled by setting the module stop control register setting and is disabled by the initial value. Canceling module stop mode enables to access the SSU register. For details, see section19, Power-Down Modes.

Section 15 A/D Converter

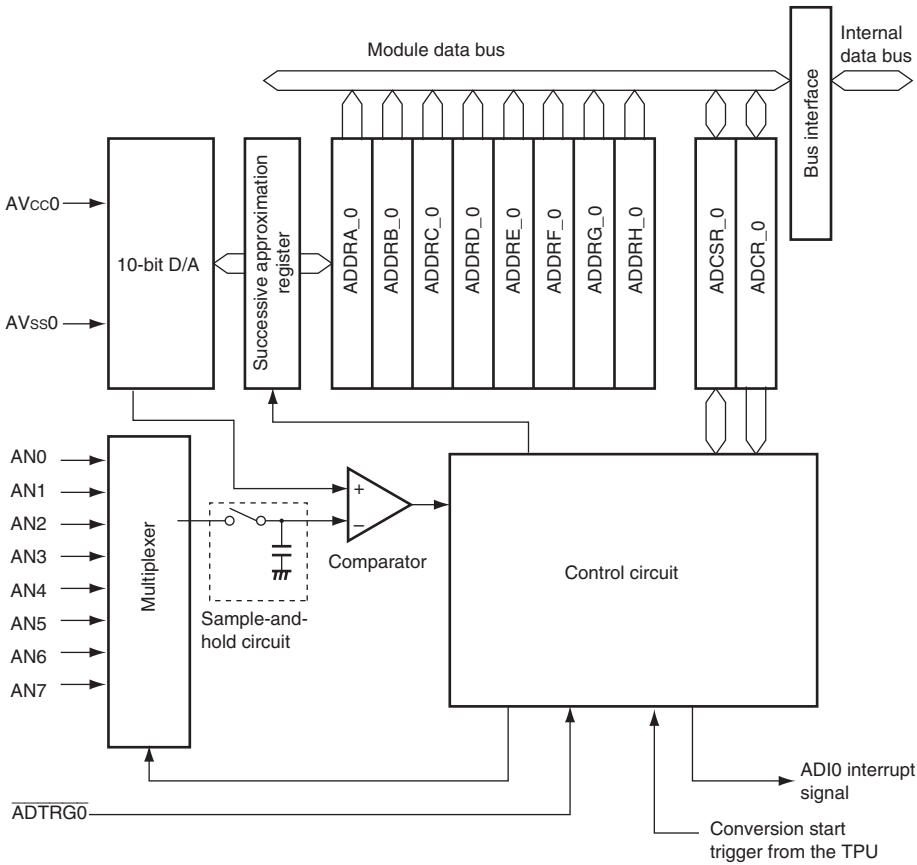
This LSI includes two units (unit 0 and unit 1) of successive approximation type 10-bit A/D converters that allow up to 16 analog input channels to be selected.

Figures 15.1 and 15.2 are block diagrams for unit 0 and unit 1, respectively.

This section describes unit 0, which has the same functions as the other unit.

15.1 Features

- 10-bit resolution
- 16 input channels (eight channels for unit 0 and eight channels for unit 1)
- Conversion time: 7.4 μs per channel (at 35-MHz operation)
- Two kinds of operating modes
  - Single mode: Single-channel A/D conversion
  - Scan mode: Continuous A/D conversion on 1 to 4 channels, or 1 to 8 channels
- 16 data registers (eight registers for unit 0 and eight registers for unit 1)
  A/D conversion results are held in a 16-bit data register for each channel
- Sample and hold function
- Three types of conversion start
  Conversion can be started by software, a conversion start trigger by the 16-bit timer pulse unit (TPU), or an external trigger signal.
- Interrupt source
  A/D conversion end interrupt (ADI) request can be generated.
- Module stop mode can be set



[Legend]

ADCR_0:	A/D control register_0	ADDRD_0:	A/D data register D_0
ADCSR_0:	A/D control/status register_0	ADDRE_0:	A/D data register E_0
ADDRA_0:	A/D data register A_0	ADDRF_0:	A/D data register F_0
ADDRB_0:	A/D data register B_0	ADDRG_0:	A/D data register G_0
ADDRC_0:	A/D data register C_0	ADDRH_0:	A/D data register H_0

Figure 15.1 Block Diagram of A/D Converter (Unit 0/AD_0)

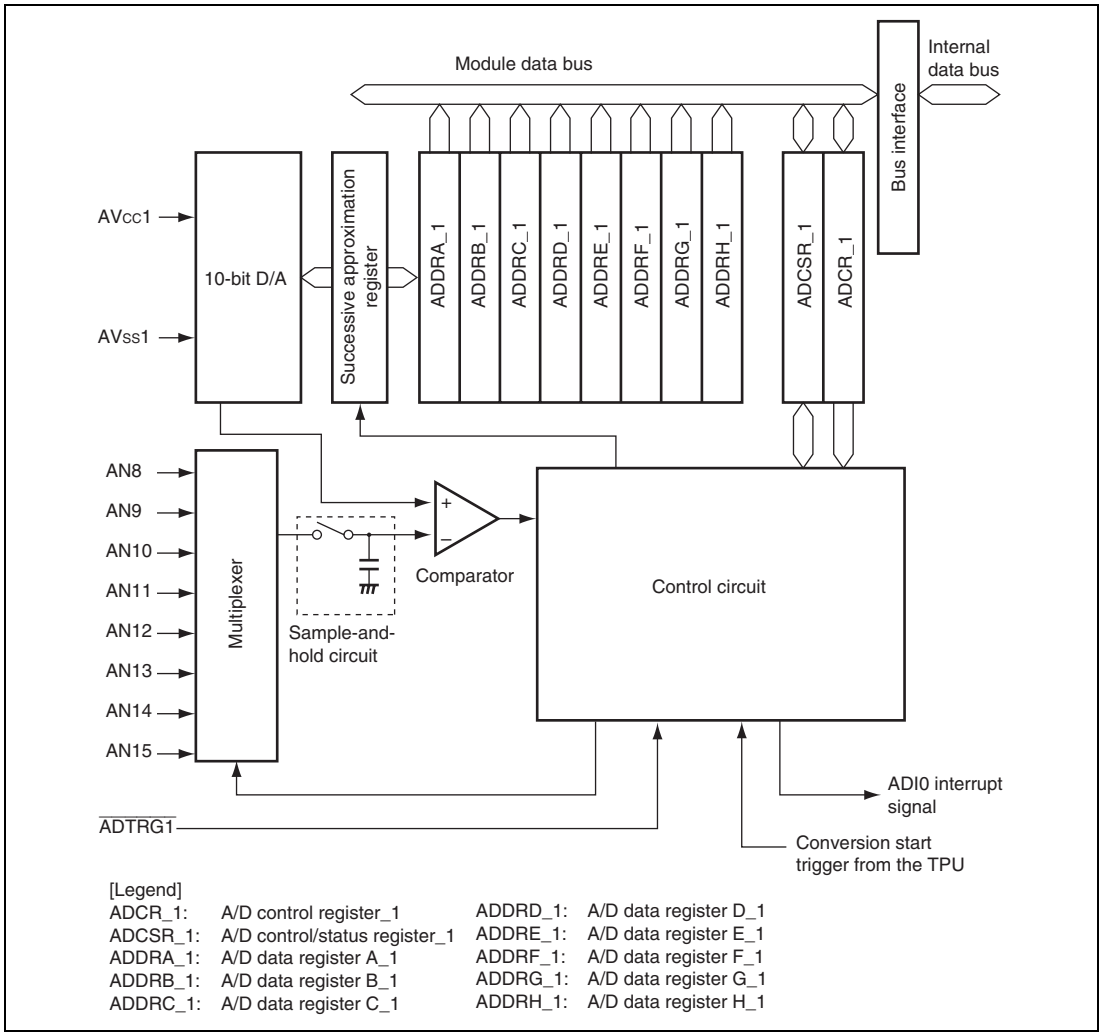


Figure 15.2 Block Diagram of A/D Converter (Unit 1/AD_1)

15.2 Input/Output Pins

Table 15.1 shows the pin configuration of the A/D converter.

Table 15.1 Pin Configuration

Unit	Abbr.	Pin Name	Symbol	I/O	Function
0	AD_0	Analog input pin 0	AN0	Input	Analog inputs
		Analog input pin 1	AN1	Input	
		Analog input pin 2	AN2	Input	
		Analog input pin 3	AN3	Input	
		Analog input pin 4	AN4	Input	
		Analog input pin 5	AN5	Input	
		Analog input pin 6	AN6	Input	
		Analog input pin 7	AN7	Input	
		A/D external trigger input pin 0	$\overline{\text{ADTRG0}}$	Input	External trigger input for starting A/D conversion
		Analog power supply pin 0	AV_{CC0}	Input	Analog block power supply
1	AD_1	Analog input pin 8	AN8	Input	Analog inputs
		Analog input pin 9	AN9	Input	
		Analog input pin 10	AN10	Input	
		Analog input pin 11	AN11	Input	
		Analog input pin 12	AN12	Input	
		Analog input pin 13	AN13	Input	
		Analog input pin 14	AN14	Input	
		Analog input pin 15	AN15	Input	
		A/D external trigger input pin 1	$\overline{\text{ADTRG1}}$	Input	External trigger input for starting A/D conversion
		Analog power supply pin 1	AV_{CC1}	Input	Analog block power supply
Common		Analog ground pin	AV_{SS}	Input	Analog block ground

15.3 Register Descriptions

The A/D converter has the following registers.

The registers for unit 0 (A/D_0) and unit 1 (A/D_1) have the same functions. In this descriptions, AN8 to AN15 correspond to AN0 to AN7.

- Unit 0 (A/D_0)
  - A/D data register A_0 (ADDRA_0)
  - A/D data register B_0 (ADDRB_0)
  - A/D data register C_0 (ADDRC_0)
  - A/D data register D_0 (ADDRD_0)
  - A/D data register E_0 (ADDRE_0)
  - A/D data register F_0 (ADDRF_0)
  - A/D data register G_0 (ADDRG_0)
  - A/D data register H_0 (ADDRH_0)
  - A/D control/status register_0 (ADCSR_0)
  - A/D control register_0 (ADCR_0)

- Unit 1 (A/D_1)
  - A/D data register A_1 (ADDRA_1)
  - A/D data register B_1 (ADDRB_1)
  - A/D data register C_1 (ADDRC_1)
  - A/D data register D_1 (ADDRD_1)
  - A/D data register E_1 (ADDRE_1)
  - A/D data register F_1 (ADDRF_1)
  - A/D data register G_1 (ADDRG_1)
  - A/D data register H_1 (ADDRH_1)
  - A/D control/status register_1 (ADCSR_1)
  - A/D control register_1 (ADCR_1)

15.3.1 A/D Data Registers A to H (ADDRA to ADDRH)

There are eight 16-bit read-only ADDR registers, ADDRA to ADDRH, used to store the results of A/D conversion. The ADDR registers, which store a conversion result for each channel, are shown in table 15.2.

The converted 10-bit data is stored in bits 15 to 6. The lower 6-bit data is always read as 0.

The data bus between the CPU and the A/D converter has a 16-bit width. The data can be read directly from the CPU. ADDR must not be accessed in 8-bit units and must be accessed in 16-bit units.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit Name											—	—	—	—	—	—
Initial Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

Table 15.2 Analog Input Channels and Corresponding ADDR Registers

Analog Input Channel	A/D Data Register Which Stores Conversion Result
AN0	ADDRA
AN1	ADDRB
AN2	ADDRC
AN3	ADDRD
AN4	ADDRE
AN5	ADDRF
AN6	ADDRG
AN7	ADDRH

15.3.2 A/D Control/Status Register (ADCSR)

ADCSR controls A/D conversion operations.

Bit	7	6	5	4	3	2	1	0
Bit Name	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0
Initial Value	0	0	0	0	0	0	0	0
R/W	R/(W)*	R/W	R/W	R	R/W	R/W	R/W	R/W

Note: * Only 0 can be written to this bit, to clear the flag.

Bit	Bit Name	Initial Value	R/W	Description
7	ADF	0	R/(W)*	A/D End Flag A status flag that indicates the end of A/D conversion. [Setting conditions] • When A/D conversion ends in single mode • When A/D conversion ends on all specified channels in scan mode [Clearing conditions] • When 0 is written after reading ADF = 1 (When the CPU is used to clear this flag by writing 0 while the corresponding interrupt is enabled, be sure to read the flag after writing 0 to it.) • When the DMAC is activated by an ADI interrupt and ADDR is read
6	ADIE	0	R/W	A/D Interrupt Enable When this bit is set to 1, ADI interrupts by ADF are enabled.
5	ADST	0	R/W	A/D Start Clearing this bit to 0 stops A/D conversion, and the A/D converter enters wait state. Setting this bit to 1 starts A/D conversion. In single mode, this bit is cleared to 0 automatically when A/D conversion on the specified channel ends. In scan mode, A/D conversion continues sequentially on the specified channels until this bit is cleared to 0 by software or a reset.

Bit	Bit Name	Initial Value	R/W	Description
4	—	0	R	Reserved This is a read-only bit and cannot be modified.
3	CH3	0	R/W	Channel Select 3 to 0
2	CH2	0	R/W	Selects analog input together with bits SCANE and SCANS in ADCR.
1	CH1	0	R/W	
0	CH0	0	R/W	• When SCANE = 0 and SCANS = X 0000: AN0 0001: AN1 0010: AN2 0011: AN3 0100: AN4 0101: AN5 0110: AN6 0111: AN7 1XXX: Setting prohibited • When SCANE = 1 and SCANS = 0 0000: AN0 0001: AN0 and AN1 0010: AN0 to AN2 0011: AN0 to AN3 0100: AN4 0101: AN4 and AN5 0110: AN4 to AN6 0111: AN4 to AN7 1XXX: Setting prohibited • When SCANE = 1 and SCANS = 1 0000: AN0 0001: AN0 and AN1 0010: AN0 to AN2 0011: AN0 to AN3 0100: AN0 to AN4 0101: AN0 to AN5 0110: AN0 to AN6 0111: AN0 to AN7 1XXX: Setting prohibited

[Legend]

X: Don't care

Note: * Only 0 can be written to this bit, to clear the flag.

15.3.3 A/D Control Register (ADCR)

ADCR enables A/D conversion to be started by an external trigger input.

Bit	7	6	5	4	3	2	1	0
Bit Name	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—
Initial Value	0	0	0	0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R

Bit	Bit Name	Initial Value	R/W	Description
7	TRGS1	0	R/W	Timer Trigger Select 1 and 0
6	TRGS0	0	R/W	These bits select enabling or disabling of the start of A/D conversion by a trigger signal. 00: A/D conversion start by external trigger is disabled 01: A/D conversion start by external trigger from TPU is enabled 10: Setting prohibited 11: A/D conversion start by the $\overline{\text{ADTRG}}$ pin is enabled*
5	SCANE	0	R/W	Scan Mode
4	SCANS	0	R/W	These bits select the A/D conversion operating mode. 0X: Single mode 10: Scan mode. A/D conversion is performed continuously for channels 1 to 4. 11: Scan mode. A/D conversion is performed continuously for channels 1 to 8.
3	CKS1	0	R/W	Clock Select 1 and 0
2	CKS0	0	R/W	These bits set the A/D conversion time. Set bits CKS1 and CKS0 only while A/D conversion is stopped (ADST = 0). 00: A/D conversion time = 530 states (max) 01: A/D conversion time = 266 states (max) 10: A/D conversion time = 134 states (max) 11: A/D conversion time = 68 states (max)
1, 0	—	All 0	R	Reserved These are read-only bits and cannot be modified.

[Legend]

X: Don't care

15.4 Operation

The A/D converter operates by successive approximation with 10-bit resolution. It has two operating modes: single mode and scan mode. When changing the operating mode or analog input channel, to prevent incorrect operation, first clear the ADST bit in ADCSR to 0 to halt A/D conversion. The ADST bit can be set to 1 at the same time as the operating mode or analog input channel is changed.

15.4.1 Single Mode

In single mode, A/D conversion is to be performed only once on the analog input of the specified single channel.

1. A/D conversion for the selected channel is started when the ADST bit in ADCSR is set to 1 by software or an external trigger input.
2. When A/D conversion is completed, the A/D conversion result is transferred to the corresponding A/D data register of the channel.
3. When A/D conversion is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.
4. The ADST bit remains set to 1 during A/D conversion, and is automatically cleared to 0 when A/D conversion ends. The A/D converter enters wait state. If the ADST bit is cleared to 0 during A/D conversion, A/D conversion stops and the A/D converter enters wait state.

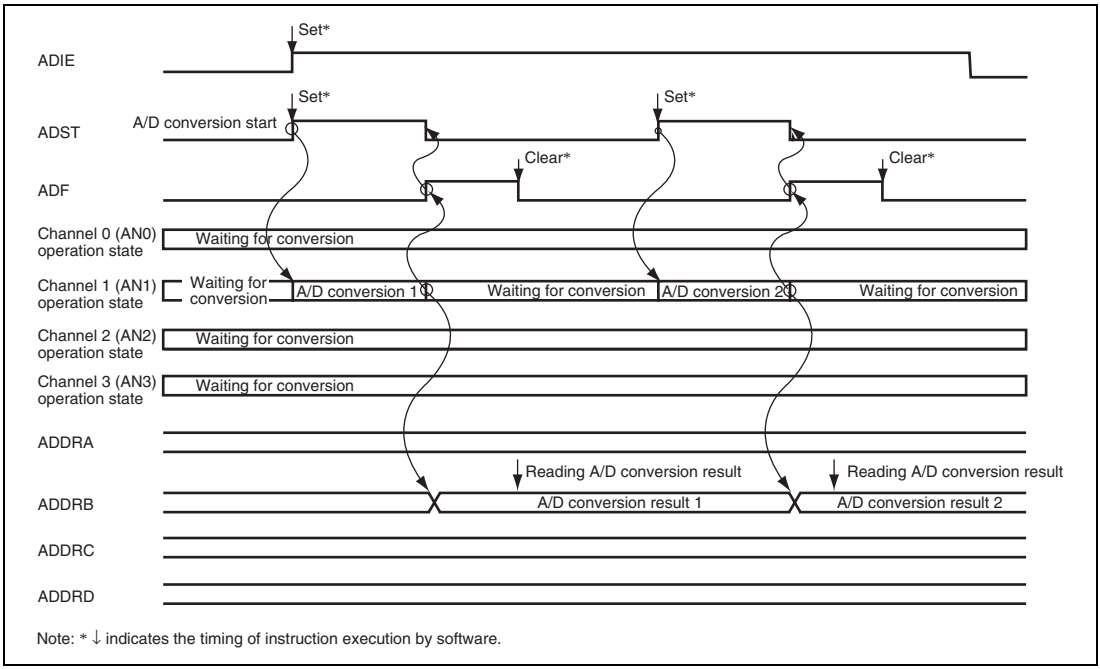


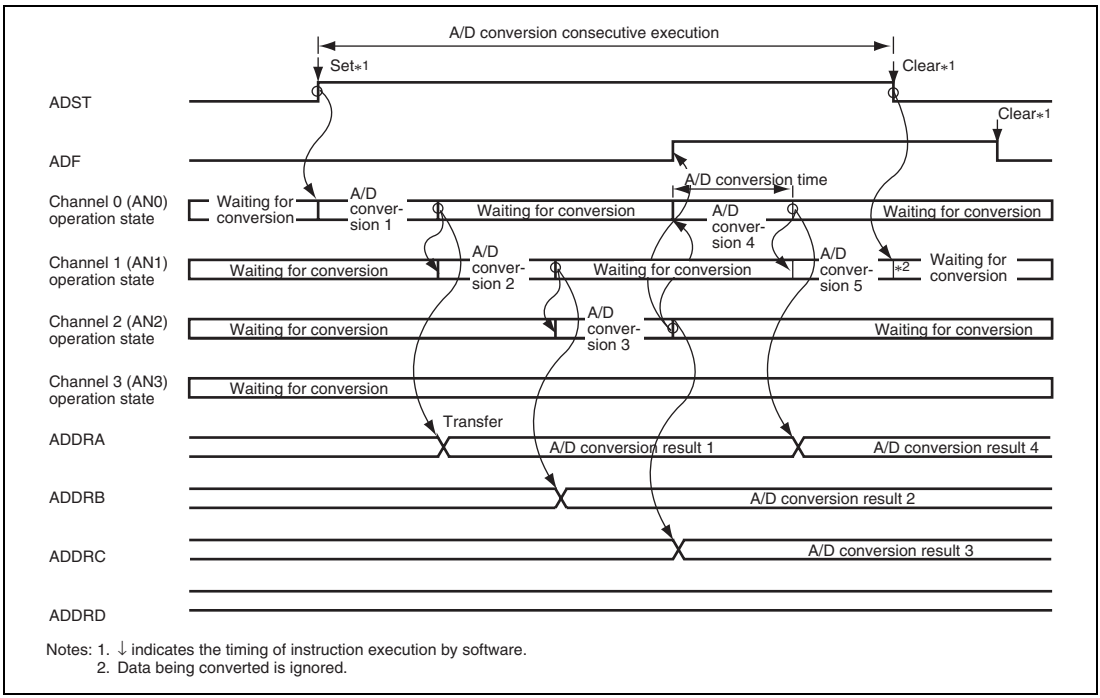
Figure 15.3 Example of A/D Converter Operation (Single Mode, Channel 1 Selected)

15.4.2 Scan Mode

In scan mode, A/D conversion is to be performed sequentially on the analog inputs of the specified channels up to four or eight channels.

1. When the ADST bit in ADCSR is set to 1 by software, TPU, or an external trigger input, A/D conversion starts on the first channel in the group. Consecutive A/D conversion on a maximum of four channels (SCANE and SCANS = B'10) or on a maximum of eight channels (SCANE and SCANS = B'11) can be selected. When consecutive A/D conversion is performed on four channels, A/D conversion starts on AN0 when CH3 and CH2 = B'00, on AN4 when CH3 and CH2 = B'01, on AN8 when CH3 and CH2 = B'10, and on AN12 when CH3 and CH2 = B'11. When consecutive A/D conversion is performed on eight channels, A/D conversion starts on AN0 when CH3 = B'0 and on AN8 when CH3 = B'1.
2. When A/D conversion for each channel is completed, the A/D conversion result is sequentially transferred to the corresponding ADDR of each channel.
3. When A/D conversion of all selected channels is completed, the ADF bit in ADCSR is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated. A/D conversion of the first channel in the group starts again.

4. The ADST bit is not cleared automatically, and steps [2] to [3] are repeated as long as the ADST bit remains set to 1. When the ADST bit is cleared to 0, A/D conversion stops and the A/D converter enters wait state. If the ADST bit is later set to 1, A/D conversion starts again from the first channel in the group.



Notes: 1. ↓ indicates the timing of instruction execution by software.
2. Data being converted is ignored.

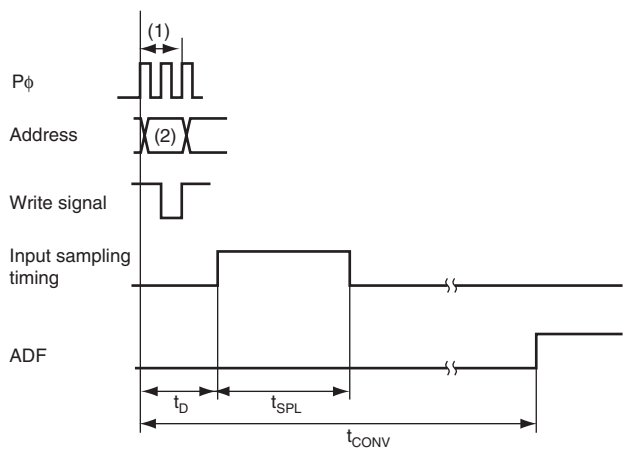
Figure 15.4 Example of A/D Conversion (Scan Mode, Three Channels (AN0 to AN2) Selected)

15.4.3 Input Sampling and A/D Conversion Time

The A/D converter has a built-in sample-and-hold circuit. The A/D converter samples the analog input when the A/D conversion start delay time (t_D) passes after the ADST bit in ADCSR is set to 1, then starts A/D conversion. Figure 15.5 shows the A/D conversion timing. Table 15.3 indicates the A/D conversion time.

As indicated in figure 15.5, the A/D conversion time (t_{CONV}) includes t_D and the input sampling time (t_{SPL}). The length of t_D varies depending on the timing of the write access to ADCSR. The total conversion time therefore varies within the ranges indicated in table 15.3.

In scan mode, the values given in table 15.3 apply to the first conversion time. The values given in table 15.4 apply to the second and subsequent conversions. In either case, bits CKS1 and CKS0 in ADCR should be set so that the conversion time is within the ranges indicated by the A/D conversion characteristics.



[Legend]
(1): ADCSR write cycle
(2): ADCSR address
t_D: A/D conversion start delay time
t_{SPL}: Input sampling time
t_{CONV}: A/D conversion time

Figure 15.5 A/D Conversion Timing

Table 15.3 A/D Conversion Characteristics (Single Mode)

		CKS1 = 0						CKS1 = 1					
		CKS0 = 0			CKS0 = 1			CKS0 = 0			CKS0 = 1		
Item	**Symbol**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**	**Min.**	**Typ.**	**Max.**
A/D conversion start delay time	t_D	18	—	33	10	—	17	6	—	9	4	—	5
Input sampling time	t_{SPL}	—	127	—	—	63	—	—	31	—	—	15	—
A/D conversion time	t_{CONV}	515	—	530	259	—	266	131	—	134	67	—	68

Note: Values in the table are the number of states.

Table 15.4 A/D Conversion Characteristics (Scan Mode)

CKS1	CKS0	Conversion Time (Number of States)
0	0	512 (Fixed)
	1	256 (Fixed)
1	0	128 (Fixed)
	1	64 (Fixed)

15.4.4 External Trigger Input Timing

A/D conversion can be externally triggered. When the TRGS1 and TRGS0 bits are set to B'11 in ADCR, an external trigger is input from the $\overline{\text{ADTRG}}$ pin. A/D conversion starts when the ADST bit in ADCSR is set to 1 on the falling edge of the $\overline{\text{ADTRG}}$ pin. Other operations, in both single and scan modes, are the same as when the ADST bit has been set to 1 by software. Figure 15.6 shows the timing.

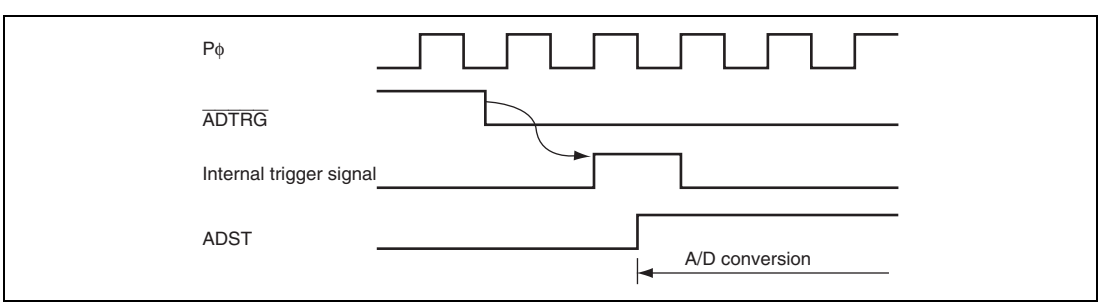


Figure 15.6 External Trigger Input Timing

15.5 Interrupt Source

The A/D converter generates an A/D conversion end interrupt (ADI) at the end of A/D conversion. Setting the ADIE bit to 1 when the ADF bit in ADCSR is set to 1 after A/D conversion is completed enables ADI interrupt requests. The DMA controller (DMAC) can be activated by an ADI interrupt. Having the converted data read by the DMAC in response to an ADI interrupt enables continuous conversion to be achieved without imposing a load on software.

Table 15.5 A/D Converter Interrupt Source

Unit	Abbr.	Interrupt Source	Interrupt Flag	DMAC Activation
0	ADI0	A/D_0 conversion end	ADF	Possible
1	ADI1	A/D_1 conversion end	ADF	Possible

15.6 A/D Conversion Accuracy Definitions

This LSI's A/D conversion accuracy definitions are given below.

- Resolution

  The number of A/D converter digital output codes.
- Quantization error

  The deviation inherent in the A/D converter, given by 1/2 LSB (see figure 15.7).
- Offset error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from the minimum voltage value B'0000000000 (H'000) to B'0000000001 (H'001) (see figure 15.8).
- Full-scale error

  The deviation of the analog input voltage value from the ideal A/D conversion characteristic when the digital output changes from B'1111111110 (H'3FE) to B'1111111111 (H'3FF) (see figure 15.8).
- Nonlinearity error

  The error with respect to the ideal A/D conversion characteristic between the zero voltage and the full-scale voltage. Does not include the offset error, full-scale error, or quantization error (see figure 15.8).
- Absolute accuracy

  The deviation between the digital value and the analog input value. Includes the offset error, full-scale error, quantization error, and nonlinearity error.

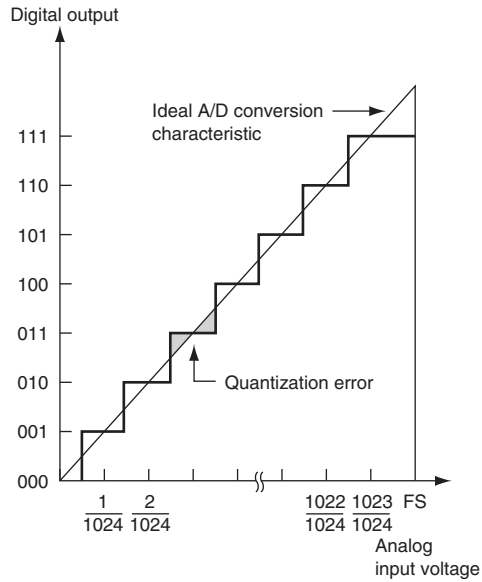


Figure 15.7 A/D Conversion Accuracy Definitions

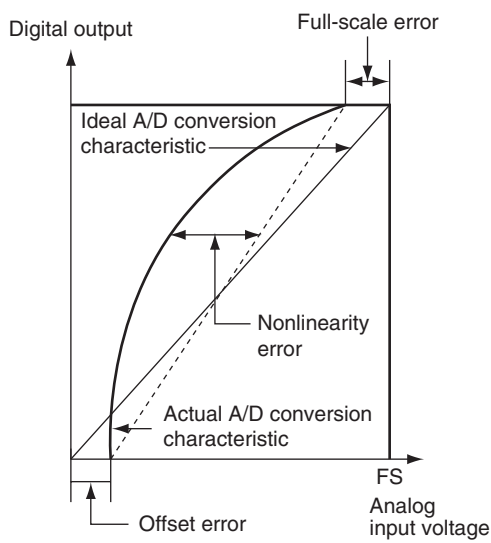


Figure 15.8 A/D Conversion Accuracy Definitions

15.7 Usage Notes

15.7.1 Module Stop Mode Setting

Operation of the A/D converter can be disabled or enabled using the module stop control register. The initial setting is for operation of the A/D converter to be halted. Register access is enabled by clearing module stop mode. For details, refer to section 19, Power-Down Modes.

15.7.2 Permissible Signal Source Impedance

This LSI's analog input is designed so that the conversion accuracy is guaranteed for an input signal for which the signal source impedance is 5 kΩ or less. This specification is provided to enable the A/D converter's sample-and-hold circuit input capacitance to be charged within the sampling time; if the sensor output impedance exceeds 5 kΩ, charging may be insufficient and it may not be possible to guarantee the A/D conversion accuracy. However, if a large capacitance is provided externally for conversion in single mode, the input load will essentially comprise only the internal input resistance of 10 kΩ, and the signal source impedance is ignored. However, since a low-pass filter effect is obtained in this case, it may not be possible to follow an analog signal with a large differential coefficient (e.g., 5 mV/μs or greater) (see figure 15.9). When converting a high-speed analog signal or conversion in scan mode, a low-impedance buffer should be inserted.

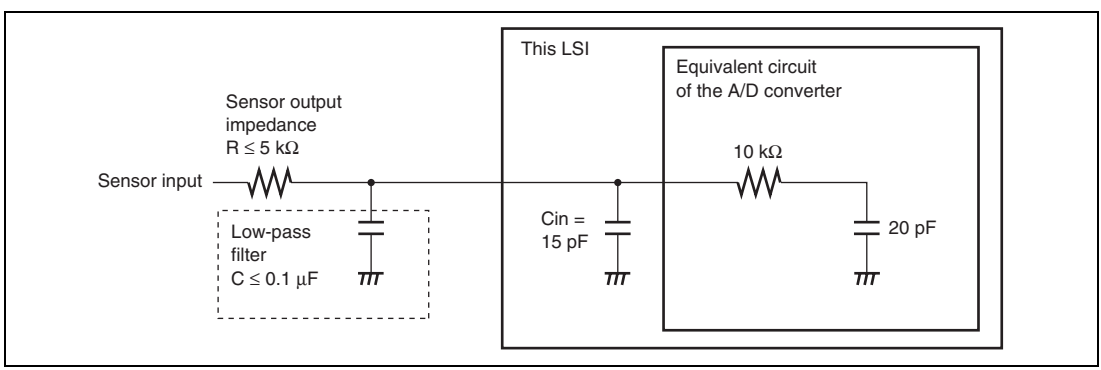


Figure 15.9 Example of Analog Input Circuit

15.7.3 Influences on Absolute Accuracy

Adding capacitance results in coupling with GND, and therefore noise in GND may adversely affect absolute accuracy. Be sure to make the connection to an electrically stable GND such as AVss.

Care is also required to insure that digital signals on the board do not interfere with filter circuits and filter circuits do not act as antennas.

15.7.4 Setting Range of Analog Power Supply and Other Pins

If the conditions shown below are not met, the reliability of the LSI may be adversely affected.

- Analog input voltage range

  The voltage applied to analog input pin ANn during A/D conversion should be in the range AVss ≤ V_{AN} ≤ AVcc0 and AVss ≤ V_{AN} ≤ AVcc1.
- Relation between AVcc0, AVcc1, AVss and Vcc, Vss

  As the relationship between AVcc0, AVcc1, AVss and Vcc, Vss, set AVcc0 = Vcc ± 0.3 V, AVcc1 = Vcc ± 0.3 V, and AVss = Vss. If the A/D converter is not used, set AVcc0 = Vcc, AVcc1 = Vcc, and AVss = Vss.

15.7.5 Notes on Board Design

In board design, digital circuitry and analog circuitry should be as mutually isolated as possible, and layout in which digital circuit signal lines and analog circuit signal lines cross or are in close proximity should be avoided as far as possible. Failure to do so may result in incorrect operation of the analog circuitry due to inductance, adversely affecting A/D conversion values.

Digital circuitry must be isolated from the analog input pins (AN0 to AN15) and analog power supply (AVcc0 and AVcc1) by the analog ground (AVss). Also, the analog ground (AVss) should be connected at one point to a stable ground (Vss) on the board.

15.7.6 Notes on Noise Countermeasures

A protection circuit connected to prevent damage due to an abnormal voltage such as an excessive surge at the analog input pins (AN0 to AN15) should be connected between AVcc0, AVcc1 and AVss as shown in figure 15.10. Also, the bypass capacitors connected to AVcc0 and AVcc1 and the filter capacitor connected to pins AN0 to AN15 must be connected to AVss.

If a filter capacitor is connected, the input currents at pins AN0 to AN15 are averaged, and so an error may arise. Also, when A/D conversion is performed frequently, as in scan mode, if the current charged and discharged by the capacitance of the sample-and-hold circuit in the A/D converter exceeds the current input via the input impedance (R_{in}), an error will arise in the analog input pin voltage. Careful consideration is therefore required when deciding the circuit constants.

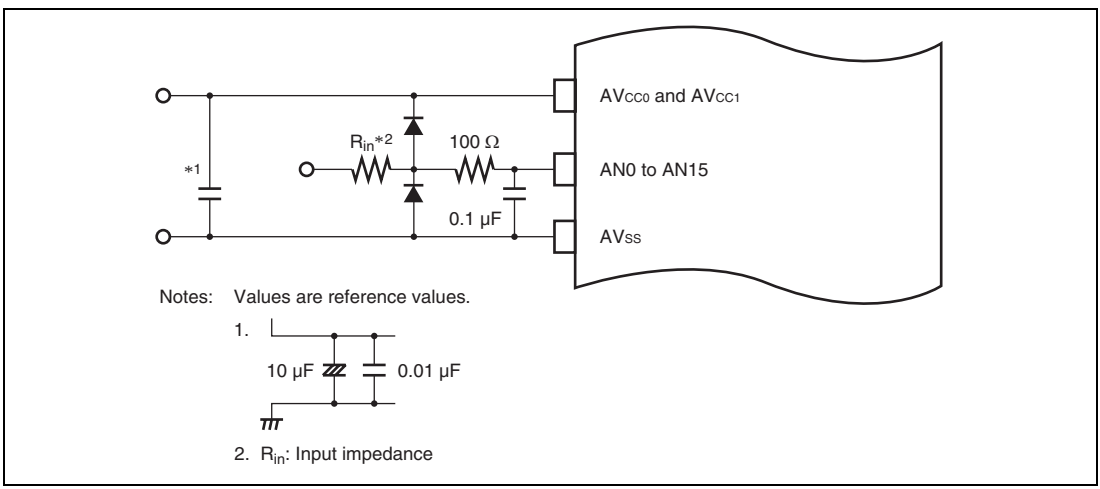
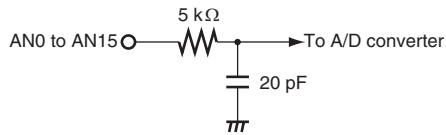


Figure 15.10 Example of Analog Input Protection Circuit

Table 15.6 Analog Pin Specifications

Item	Min	Max	Unit
Analog input capacitance	—	20	pF
Permissible signal source impedance	—	5	kΩ



Note: Values are reference values.

Figure 15.11 Analog Input Pin Equivalent Circuit

15.7.7 A/D Input Hold Function in Software Standby Mode

When this LSI enters software standby mode with A/D conversion enabled, the A/D conversion are retained, and the analog current is equal to as during A/D conversion. If the analog power supply current needs to be reduced in software standby mode, clear the ADST, TRGS1, and TRGS0 bits all to 0 to disable A/D conversion.

Section 16 RAM

This LSI has a 12-kbyte on-chip high-speed static RAM. The RAM is connected to the CPU by a 32-bit data bus, enabling 1-state read and 2-state write accesses by the CPU to all byte data, word data, and longword data.

The on-chip RAM can be enabled or disabled by means of the RAME bit in the system control register (SYSCR). For details on SYSCR, refer to section 3.2.2, System Control Register (SYSCR).

Product Classification		RAM Size	RAM Addresses
Flash memory version	H8SX/1527R	12 kbytes	H'FF9000 to H'FFBFFF
	H8SX/1525R	12 kbytes	H'FF9000 to H'FFBFFF

Section 17 Flash Memory (0.18-μm F-ZTAT Version)

The flash memory has the following features. Figure 17.1 is a block diagram of the flash memory.

17.1 Features

- Size

Product Classification		ROM Size	ROM Address
H8SX/1527R	R5F61527R	256 kbytes	H'000000 to H'03FFFF (modes 1 to 3)
H8SX/1525R	R5F61525R		

- Two memory MATs

  The start addresses of two memory spaces (memory MATs) are allocated to the same address. The mode setting in the initiation determines which memory MAT is initiated first. The memory MATs can be switched by using the bank-switching method after initiation.

  — User MAT initiated at a power-on reset in user mode: 256 kbytes

  — User boot MAT is initiated at a power-on reset in user boot mode: 10 kbytes

- Programming/erasing interface by the download of on-chip program

  This LSI has a programming/erasing program. After downloading this program to the on-chip RAM, programming/erasing can be performed by setting the parameters. The user branch is also supported.

  — User branch

  The program processing is performed in 128-byte units. It consists of the program pulse application, verify read, and several other steps. Erasing is performed in one divided-block units and consists of several steps. The user processing routine can be executed between the steps, the setting for which is called the user branch addition.

- Programming/erasing time

  Programming time: 3 ms (typ) for 128-byte simultaneous programming, 23.4 μs per byte

  Erasing time: 1000 ms (typ) per 1 block (64 kbytes)

- Number of programming

  The number of programming can be up to 100 times at the minimum. (1 to 100 times are guaranteed.)

- Three on-board programming modes

  Boot mode: Using the on-chip SCI_4, the user MAT and user boot MAT can be programmed/erased. In boot mode, the bit rate between the host and this LSI can be adjusted automatically.

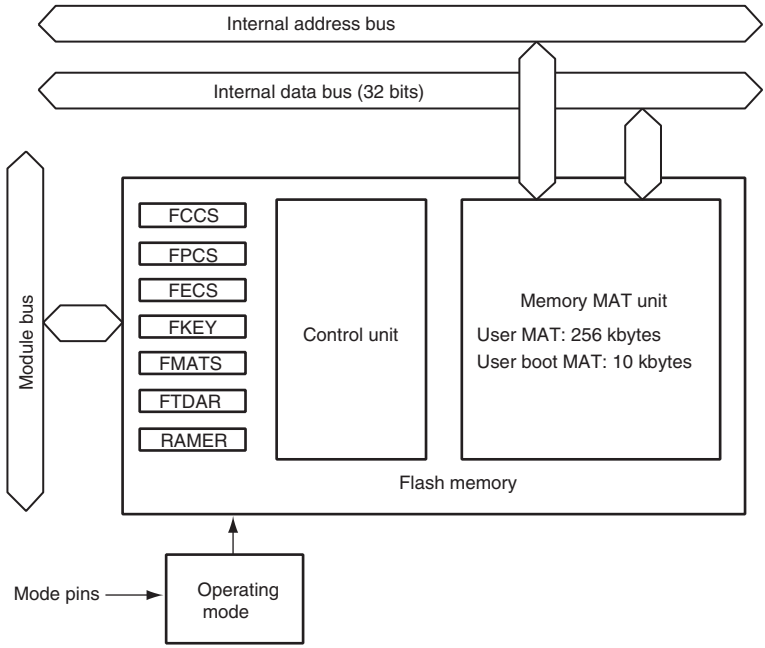
  User program mode: Using a desired interface, the user MAT can be programmed/erased.

  User boot mode: Using a desired interface, the user boot program can be made and the user MAT can be programmed/erased.
- Off-board programming mode

  Programmer mode: Using a PROM programmer, the user MAT and user boot MAT can be programmed/erased.
- Programming/erasing protection

  Protection against programming/erasing of the flash memory can be set by hardware protection, software protection, or error protection.
- Flash memory emulation function using the on-chip RAM

  Realtime emulation of the flash memory programming can be performed by overlaying parts of the flash memory (user MAT) area and the on-chip RAM.



[Legend]
FCCS: Flash code control/status register
FPCS: Flash program code select register
FECS: Flash erase code select register
FKEY: Flash key code register
FMATS: Flash MAT select register
FTDAR: Flash transfer destination address register
RAMER: RAM emulation register

Note: To read from or write to the registers, the FLSHE bit in SYSCR must be set to 1.

Figure 17.1 Block Diagram of Flash Memory

17.2 Mode Transition Diagram

When the mode pins are set in the reset state and reset start is performed, this LSI enters each operating mode as shown in figure 17.2. Although the flash memory can be read in user mode, it cannot be programmed or erased. The flash memory can be programmed or erased in boot mode, user program mode, user boot mode, and programmer mode. The differences between boot mode, user program mode, user boot mode, and programmer mode are shown in table 17.1.

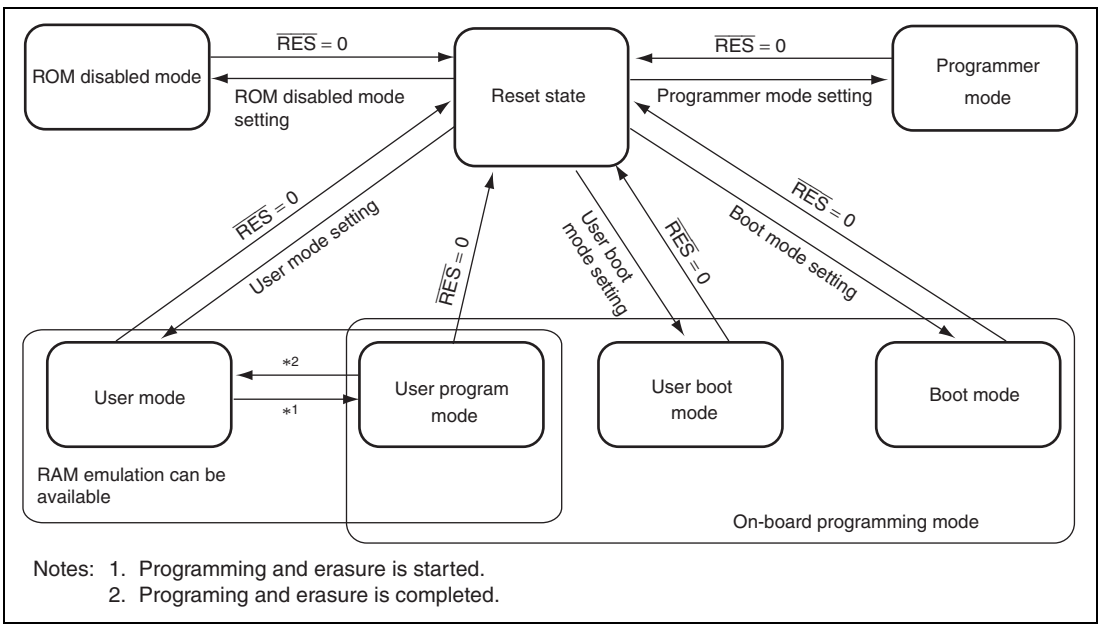


Figure 17.2 Mode Transition of Flash Memory

Table 17.1 Differences between Boot Mode, User Program Mode, User Boot Mode, and Programmer Mode

Item	Boot Mode	User Program Mode	User Boot Mode	Programmer Mode
Programming/ erasing environment	On-board programming	On-board programming	On-board programming	Off-board programming
Programming/ erasing enable MAT	• User MAT • User boot MAT	• User MAT	• User MAT	• User MAT • User boot MAT
Programming/ erasing control	Command	Programming/ erasing interface	Programming/ erasing interface	Command
All erasure	O (Automatic)	O	O	O (Automatic)
Block division erasure	O*[1]	O	O	×
Program data transfer	From host via SCI	From desired device via RAM	From desired device via RAM	Via programmer
User branch function	×	O	O	×
RAM emulation	×	O	O	×
Reset initiation MAT	Embedded program storage area	User MAT	User boot MAT*[2]	—
Transition to user mode	Changing mode and reset	Completing Programming/ erasure*[3]	Changing mode and reset	—

Notes: 1. All-erasure is performed. After that, the specified block can be erased.

2. First, the reset vector is fetched from the embedded program storage area. After the flash memory related registers are checked, the reset vector is fetched from the user boot MAT.

3. In this LSI, the user programming mode is defined as the period from the timing when a program concerning programming and erasure is started to the timing when the program is completed. For details on a program concerning programming and erasure, see section 17.8.2, User Program Mode.

17.3 Memory MAT Configuration

The memory MATs of flash memory in this LSI consists of the 256-kbyte user MAT and 10-kbyte user boot MAT. The start addresses of the user MAT and user boot MAT are allocated to the same address. Therefore, when the program execution or data access is performed between the two memory MATs, the memory MATs must be switched by the flash MAT select register (FMATS).

The user MAT or user boot MAT can be read in all modes. However, the user boot MAT can be programmed or erased only in boot mode and programmer mode.

The size of the user MAT is different from that of the user boot MAT. Addresses which exceed the size of the 10-kbyte user boot MAT should not be accessed. If an attempt is made, data is read as an undefined value.

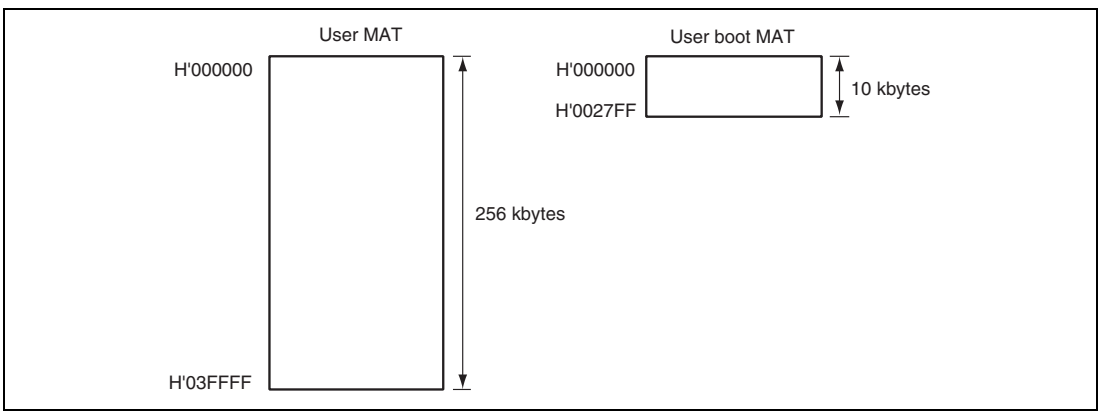


Figure 17.3 Memory MAT Configuration

17.4 Block Structure

Figure 17.4 shows the block structure of the 256-kbyte user MAT. The heavy-line frames indicate the erase blocks. The thin-line frames indicate the programming units and the values inside the frames stand for the addresses. The user MAT is divided into three 64-kbyte blocks, one 32-kbyte block, and eight 4-kbyte blocks. The user MAT can be erased in these divided block units. Programming is done in 128-byte units starting from where the lower address is H'00 or H'80. RAM emulation can be performed in the eight 4-kbyte blocks.

EB0 Erase unit: 4 kbytes	H'000000	H'000001	H'000002	←Programming unit: 128 bytes→	H'00007F
	H'000F80	H'000F81	H'000F82		H'000FFF
EB1 Erase unit: 4 kbytes	H'001000	H'001001	H'001002	←Programming unit: 128 bytes→	H'00107F
	H'001F80	H'001F81	H'001F82		H'001FFF
EB2 Erase unit: 4 kbytes	H'002000	H'002001	H'002002	←Programming unit: 128 bytes→	H'00207F
	H'002F80	H'002F81	H'002F82		H'002FFF
EB3 Erase unit: 4 kbytes	H'003000	H'003001	H'003002	←Programming unit: 128 bytes→	H'00307F
	H'003F80	H'003F81	H'003F82		H'003FFF
EB4 Erase unit: 4 kbytes	H'004000	H'004001	H'004002	←Programming unit: 128 bytes→	H'00407F
	H'004F80	H'004F81	H'004F82		H'004FFF
EB5 Erase unit: 4 kbytes	H'005000	H'005001	H'005002	←Programming unit: 128 bytes→	H'00507F
	H'005F80	H'005F81	H'005F82		H'005FFF
EB6 Erase unit: 4 kbytes	H'006000	H'006001	H'006002	←Programming unit: 128 bytes→	H'00607F
	H'006F80	H'006F81	H'006F82		H'006FFF
EB7 Erase unit: 4 kbytes	H'007000	H'007001	H'007002	←Programming unit: 128 bytes→	H'00707F
	H'007F80	H'007F81	H'007F82		H'007FFF
EB8 Erase unit: 32 kbytes	H'008000	H'008001	H'008002	←Programming unit: 128 bytes→	H'00807F
	H'00FF80	H'00FF81	H'00FF82		H'00FFFF
EB9 Erase unit: 64 kbytes	H'010000	H'010001	H'010002	←Programming unit: 128 bytes→	H'01007F
	H'01FF80	H'01FF81	H'01FF82		H'01FFFF
EB10 Erase unit: 64 kbytes	H'020000	H'020001	H'020002	←Programming unit: 128 bytes→	H'02007F
	H'02FF80	H'02FF81	H'02FF82		H'02FFFF
EB11 Erase unit: 64 kbytes	H'030000	H'030001	H'030002	←Programming unit: 128 bytes→	H'03007F
	H'03FF80	H'03FF81	H'03FF82		H'03FFFF

Figure 17.4 Block Structure of User MAT

17.5 Programming/Erasing Interface

Programming/erasing of the flash memory is done by downloading an on-chip programming/erasing program to the on-chip RAM and specifying the start address of the programming destination, the program data, and the erase block number using the programming/erasing interface registers and programming/erasing interface parameters.

The procedure program for user program mode and user boot mode is made by the user. Figure 17.5 shows the procedure for creating the procedure program. For details, see section 17.8.2, User Program Mode.

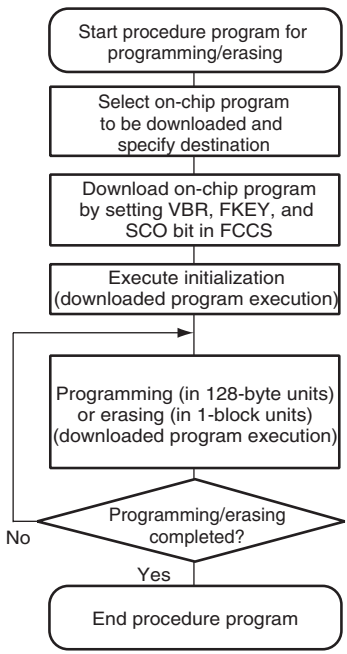


Figure 17.5 Procedure for Creating Procedure Program

(1) Selection of On-Chip Program to be Downloaded

For programming/erasing, the FLSHE bit in the system control register (SYSCR) must be set to 1 to select user program mode. This LSI has programming/erasing programs which can be downloaded to the on-chip RAM. The on-chip program to be downloaded is selected by the programming/erasing interface registers. The start address of the on-chip RAM where an on-chip program is downloaded is specified by the flash transfer destination address register (FTDAR).

(2) Download of On-Chip Program

The on-chip program is automatically downloaded by setting the flash key code register (FKEY) and the SCO bit in the flash code control/status register (FCCS) after initializing the vector base register (VBR). The memory MAT is replaced with the embedded program storage area during download. Since the memory MAT cannot be read during programming/erasing, the procedure program must be executed in a space other than the flash memory (for example, on-chip RAM). Since the download result is returned to the programming/erasing interface parameter, whether download is normally executed or not can be confirmed. The VBR contents can be changed after completion of download.

(3) Initialization of Programming/Erasing

The operating frequency and user branch are set before execution of programming/erasing. The user branch destination must be in an area other than the on-chip flash memory and the area where the on-chip program is downloaded. These settings are made by using the programming/erasing interface parameters.

(4) Execution of Programming/Erasing

For programming/erasing, the FLSHE bit in SYSCR must be set to 1 to make a transition to user program mode. The start address of the programming destination and the program data are specified in 128-byte units when programming. The block to be erased is specified with the erase block number in erase-block units when erasing. Specifications of the start address of the programming destination, program data, and erase block number are performed by the programming/erasing interface parameters, and the on-chip program is initiated. The on-chip program is executed by using the JSR or BSR instruction and executing the subroutine call of the specified address in the on-chip RAM. The execution result is returned to the programming/erasing interface parameter.

The area to be programmed must be erased in advance when programming flash memory. All interrupts are disabled during programming/erasing.

(5) When Programming/Erasing is Executed Consecutively

When processing does not end by 128-byte programming or 1-block erasure, consecutive programming/erasing can be realized by updating the start address of the programming destination and program data, or the erase block number. Since the downloaded on-chip program is left in the on-chip RAM even after programming/erasing completes, download and initialization are not required when the same processing is executed consecutively.

17.6 Input/Output Pins

The flash memory is controlled through the input/output pins shown in table 17.2.

Table 17.2 Pin Configuration

Pin Name	I/O	Function
$\overline{\text{RES}}$	Input	Reset
MD1 and MD0	Input	Set operating mode of this LSI
TxD4	Output	Serial transmit data output (used in boot mode)
RxD4	Input	Serial receive data input (used in boot mode)

17.7 Register Descriptions

The flash memory has the following registers. To access these registers, the FLSHE bit in the system control register (SYSCR) must be set to 1. For details on SYSCR, see section 3.2.2, System Control Register (SYSCR).

Programming/Erasing Interface Registers:

- Flash code control/status register (FCCS)
- Flash program code select register (FPCS)
- Flash erase code select register (FECS)
- Flash key code register (FKEY)
- Flash MAT select register (FMATS)
- Flash transfer destination address register (FTDAR)

Programming/Erasing Interface Parameters:

- Download pass and fail result parameter (DPFR)
- Flash pass and fail result parameter (FPFR)
- Flash program/erase frequency parameter (FPEFEQ)
- Flash user branch set parameter (FUBRA)
- Flash multipurpose address area parameter (FMPAR)
- Flash multipurpose data destination area parameter (FMPDR)
- Flash erase block select parameter (FEBS)

- RAM emulation register (RAMER)

There are several operating modes for accessing the flash memory. Respective operating modes, registers, and parameters are assigned to the user MAT and user boot MAT. The correspondence between operating modes and registers/parameters for use is shown in table 17.3.

Table 17.3 Registers/Parameters and Target Modes

Register/Parameter		Down-load	Initiali-zation	Program-ming	Erasure	Read	RAM Emulation
Programming/ erasing interface registers	FCCS	O	—	—	—	—	—
	FPCS	O	—	—	—	—	—
	FECS	O	—	—	—	—	—
	FKEY	O	—	O	O	—	—
	FMATS	—	—	O*[1]	O*[1]	O*[2]	—
	FTDAR	O	—	—	—	—	—
Programming/ erasing interface parameters	DPFR	O	—	—	—	—	—
	FPFR	—	O	O	O	—	—
	FPEFEQ	—	O	—	—	—	—
	FUBRA	—	O	—	—	—	—
	FMPAR	—	—	O	—	—	—
	FMPDR	—	—	O	—	—	—
	FEBS	—	—	—	O	—	—
RAM emulation	RAMER	—	—	—	—	—	O

Notes: 1. The setting is required when programming or erasing the user MAT in user boot mode.
2. The setting may be required according to the combination of initiation mode and read target memory MAT.

17.7.1 Programming/Erasing Interface Registers

The programming/erasing interface registers are 8-bit registers that can be accessed only in bytes. These registers are initialized by a power-on reset.

(1) Flash Code Control/Status Register (FCCS)

FCCS monitors errors during programming/erasing the flash memory and requests the on-chip program to be downloaded to the on-chip RAM.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	FLER	—	—	—	SCO
Initial Value	1	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	(R)/W

Bit	Bit Name	Initial Value	R/W	Description
7	—	1	R	Reserved
6	—	0	R	These are read-only bits and cannot be modified.
5	—	0	R	
4	FLER	0	R	Flash Memory Error Indicates that an error has occurred during programming or erasing the flash memory. When this bit is set to 1, the flash memory enters the error protection state. When this bit is set to 1, high voltage is applied to the internal flash memory. To reduce the damage to the flash memory, the reset must be released after the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs. 0: Flash memory operates normally (Error protection is invalid) [Clearing condition] • At a power-on reset 1: An error occurs during programming/erasing flash memory (Error protection is valid) [Setting conditions] • When an interrupt, such as NMI, occurs during programming/erasing. • When the flash memory is read during programming/erasing (including a vector read and an instruction fetch). • When the SLEEP instruction is executed during programming/erasing (including software standby mode). • When a bus master other than the CPU, such as the DMAC, obtains bus mastership during programming/erasing.
3 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.

Bit	Bit Name	Initial Value	R/W	Description
0	SCO	0	(R)/W*	Source Program Copy Operation Requests the on-chip programming/erasing program to be downloaded to the on-chip RAM. When this bit is set to 1, the on-chip program which is selected by FPCS or FECS is automatically downloaded in the on-chip RAM area specified by FTDAR. In order to set this bit to 1, the RAM emulation mode must be canceled, H'A5 must be written to FKEY, and this operation must be executed in the on-chip RAM. Dummy read of FCCS must be executed twice immediately after setting this bit to 1. All interrupts must be disabled during download. This bit is cleared to 0 when download is completed. During program download initiated with this bit, particular processing which accompanies bank-switching of the program storage area is executed. Before a download request, initialize the VBR contents to H'00000000. After download is completed, the VBR contents can be changed. 0: Download of the programming/erasing program is not requested. [Clearing condition] • When download is completed 1: Download of the programming/erasing program is requested. [Setting conditions] (When all of the following conditions are satisfied) • Not in RAM emulation mode (the RAMS bit in RAMER is cleared to 0) • H'A5 is written to FKEY • Setting of this bit is executed in the on-chip RAM

Note: * This is a write-only bit. This bit is always read as 0.

(2) Flash Program Code Select Register (FPCS)

FPCS selects the programming program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	PPVS
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	PPVS	0	R/W	Program Pulse Verify Selects the programming program to be downloaded. 0: Programming program is not selected. [Clearing condition] When transfer is completed. 1: Programming program is selected.

(3) Flash Erase Code Select Register (FECS)

FECS selects the erasing program to be downloaded.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	EPVB
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R	R	R	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 1	—	All 0	R	Reserved These are read-only bits and cannot be modified.
0	EPVB	0	R/W	Erase Pulse Verify Block Selects the erasing program to be downloaded. 0: Erasing program is not selected. [Clearing condition] When transfer is completed. 1: Erasing program is selected.

(4) Flash Key Code Register (FKEY)

FKEY is a register for software protection that enables to download the on-chip program and perform programming/erasing of the flash memory.

Bit	7	6	5	4	3	2	1	0
Bit Name	K7	K6	K5	K4	K3	K2	K1	K0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	K7	0	R/W	Key Code
6	K6	0	R/W	When H'A5 is written to FKEY, writing to the SCO bit in FCCS is enabled. When a value other than H'A5 is written, the SCO bit cannot be set to 1. Therefore, the on-chip program cannot be downloaded to the on-chip RAM.
5	K5	0	R/W	
4	K4	0	R/W	
3	K3	0	R/W	
2	K2	0	R/W	Only when H'5A is written can programming/erasing of the flash memory be executed. When a value other than H'5A is written, even if the programming/erasing program is executed, programming/erasing cannot be performed.
1	K1	0	R/W	
0	K0	0	R/W	
				H'A5: Writing to the SCO bit is enabled. (The SCO bit cannot be set to 1 when FKEY is a value other than H'A5.)
				H'5A: Programming/erasing of the flash memory is enabled. (When FKEY is a value other than H'5A, the software protection state is entered.)
				H'00: Initial value

(5) Flash MAT Select Register (FMATS)

FMATS selects the user MAT or user boot MAT. Writing to FMATS should be done when a program in the on-chip RAM is being executed.

Bit	7	6	5	4	3	2	1	0
Bit Name	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0
Initial Value	0/1*	0	0/1*	0	0/1*	0	0/1*	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: * This bit is set to 1 in user boot mode, otherwise cleared to 0.

Bit	Bit Name	Initial Value	R/W	Description
7	MS7	0/1*	R/W	MAT Select
6	MS6	0	R/W	The memory MATs can be switched by writing a value to FMATS.
5	MS5	0/1*	R/W	
4	MS4	0	R/W	When H'AA is written to FMATS, the user boot MAT is selected. When a value other than H'AA is written, the user MAT is selected. Switch the MATs following the memory MAT switching procedure in section 17.11, Switching between User MAT and User Boot MAT. The user boot MAT cannot be selected by FMATS in user programming mode. The user boot MAT can be selected in boot mode or programmer mode.
3	MS3	0/1*	R/W	
2	MS2	0	R/W	
1	MS1	0/1*	R/W	
0	MS0	0	R/W	
				H'AA: The user boot MAT is selected. (The user MAT is selected when FMATS is a value other than H'AA.) (Initial value when initiated in user boot mode.)
				H'00: The user MAT is selected. (Initial value when initiated in a mode except for user boot mode.)

Note: * This bit is set to 1 in user boot mode, otherwise cleared to 0.

(6) Flash Transfer Destination Address Register (FTDAR)

FTDAR specifies the start address of the on-chip RAM at which to download an on-chip program. FTDAR must be set before setting the SCO bit in FCCS to 1.

Bit	7	6	5	4	3	2	1	0
Bit Name	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
7	TDER	0	R/W	Transfer Destination Address Setting Error This bit is set to 1 when an error has occurred in setting the start address specified by bits TDA6 to TDA0. A start address error is determined by whether the value set in bits TDA6 to TDA0 is within the range of H'00 to H'02 when download is executed by setting the SCO bit in FCCS to 1. Make sure that this bit is cleared to 0 before setting the SCO bit to 1 and the value specified by bits TDA6 to TDA0 should be within the range of H'00 to H'02. 0: The value specified by bits TDA6 to TDA0 is within the range. 1: The value specified by bits TDA6 to TDA0 is between H'03 and H'FF and download has stopped.
6	TDA6	0	R/W	Transfer Destination Address
5	TDA5	0	R/W	Specifies the on-chip RAM start address of the download destination. A value between H'00 and H'02, and up to 4 kbytes can be specified as the start address of the on-chip RAM.
4	TDA4	0	R/W	
3	TDA3	0	R/W	
2	TDA2	0	R/W	H'00: H'FF9000 is specified as the start address.
1	TDA1	0	R/W	
0	TDA0	0	R/W	H'01: H'FFA000 is specified as the start address. H'02: H'FFB000 is specified as the start address. H'03 to H'7F: Setting prohibited. (Specifying a value from H'03 to H'7F sets the TDER bit to 1 and stops download of the on-chip program.)

17.7.2 Programming/Erasing Interface Parameters

The programming/erasing interface parameters specify the operating frequency, user branch destination address, storage place for program data, start address of programming destination, and erase block number, and exchanges the execution result. These parameters use the general registers of the CPU (ER0 and ER1) or the on-chip RAM area. The initial values of programming/erasing interface parameters are undefined at a power-on reset or a transition to software standby mode.

Since registers of the CPU except for R0 are saved in the stack area during download of an on-chip program, initialization, programming, or erasing, allocate the stack area before performing these operations (the maximum stack size is 128 bytes). The return value of the processing result is written in R0. The programming/erasing interface parameters are used in download control, initialization before programming or erasing, programming, and erasing. Table 17.4 shows the usable parameters and target modes. The meaning of the bits in the flash pass and fail result parameter (FPFR) varies in initialization, programming, and erasure.

Table 17.4 Parameters and Target Modes

Parameter	Download	Initialization	Programming	Erasure	R/W	Initial Value	Allocation
DPFR	O	—	—	—	R/W	Undefined	On-chip RAM*
FPFR	—	O	O	O	R/W	Undefined	R0L of CPU
FPEFEQ	—	O	—	—	R/W	Undefined	ER0 of CPU
FUBRA	—	O	—	—	R/W	Undefined	ER0 of CPU
FMPAR	—	—	O	—	R/W	Undefined	ER1 of CPU
FMPDR	—	—	O	—	R/W	Undefined	ER0 of CPU
FEBS	—	—	—	O	R/W	Undefined	ER0 of CPU

Note: * A single byte of the start address of the on-chip RAM specified by FTDAR

Download Control: The on-chip program is automatically downloaded by setting the SCO bit in FCCS to 1. The on-chip RAM area to download the on-chip program is the 4-kbyte area starting from the start address specified by FTDAR. Download is set by the programming/erasing interface registers, and the download pass and fail result parameter (DPFR) indicates the return value.

Initialization before Programming/Erasing: The on-chip program includes the initialization program. A pulse with the specified period must be applied when programming or erasing. The specified pulse width is made by the method in which wait loop is configured by the CPU instruction. Accordingly, the operating frequency of the CPU must be set. Since the user branch function is supported, the user branch destination address must be set. The initial program is set as a parameter of the programming/erasing program which has been downloaded to perform these settings.

Programming: When the flash memory is programmed, the start address of the programming destination on the user MAT and the program data must be passed to the programming program.

The start address of the programming destination on the user MAT must be stored in general register ER1. This parameter is called the flash multipurpose address area parameter (FMPAR).

The program data is always in 128-byte units. When the program data does not satisfy 128 bytes, 128-byte program data is prepared by filling the dummy code (H'FF). The boundary of the start address of the programming destination on the user MAT is aligned at an address where the lower eight bits (A7 to A0) are H'00 or H'80.

The program data for the user MAT must be prepared in consecutive areas. The program data must be in a consecutive space which can be accessed using the MOV.B instruction of the CPU and is not in the flash memory space.

The start address of the area that stores the data to be written in the user MAT must be set in general register ER0. This parameter is called the flash multipurpose data destination area parameter (FMPDR).

For details on the programming procedure, see section 17.8.2, User Program Mode.

Erasure: When the flash memory is erased, the erase block number on the user MAT must be passed to the erasing program which is downloaded.

The erase block number on the user MAT must be set in general register ER0. This parameter is called the flash erase block select parameter (FEBS).

One block is selected from the block numbers of 0 to 11 as the erase block number.

For details on the erasing procedure, see section 17.8.2, User Program Mode.

(1) Download Pass and Fail Result Parameter (DPFR: Single Byte of Start Address in On-Chip RAM Specified by FTDAR)

DPFR indicates the return value of the download result. The DPFR value is used to determine the download result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	SS	FK	SF

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused These bits return 0.
2	SS	—	R/W	Source Select Error Detect Only one type can be specified for the on-chip program which can be downloaded. When the program to be downloaded is not selected, more than two types of programs are selected, or a program which is not mapped is selected, an error occurs. 0: Download program selection is normal 1: Download program selection is abnormal
1	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'A5) and returns the result. 0: FKEY setting is normal (H'A5) 1: FKEY setting is abnormal (value other than H'A5)
0	SF	—	R/W	Success/Fail Returns the download result. Reads back the program downloaded to the on-chip RAM and determines whether it has been transferred to the on-chip RAM. 0: Download of the program has ended normally (no error) 1: Download of the program has ended abnormally (error occurs)

(2) Flash Pass and Fail Parameter (FPFR: General Register R0L of CPU)

FPFR indicates the return values of the initialization, programming, and erasure results. The meaning of the bits in FPFR varies depending on the processing.

(a) Initialization before programming/erasing

FPFR indicates the return value of the initialization result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	BR	FQ	SF

Bit	Bit Name	Initial Value	R/W	Description
7 to 3	—	—	—	Unused These bits return 0.
2	BR	—	R/W	User Branch Error Detect Returns the check result whether the specified user branch destination address is the area other than the storage area of the programming/erasing program which has been downloaded. 0: User branch address setting is normal 1: User branch address setting is abnormal
1	FQ	—	R/W	Frequency Error Detect Compares the specified CPU operating frequency with the operating frequencies supported by this LSI, and returns the result. 0: Setting of operating frequency is normal 1: Setting of operating frequency is abnormal
0	SF	—	R/W	Success/Fail Returns the initialization result. 0: Initialization has ended normally (no error) 1: Initialization has ended abnormally (error occurs)

(b) Programming

FPFR indicates the return value of the programming result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	—	WD	WA	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	Programming Mode Related Setting Error Detect Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 17.9.3, Error Protection. 0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Programming Execution Error Detect Writes 1 to this bit when the specified data could not be written because the user MAT was not erased. If this bit is set to 1, there is a high possibility that the user MAT has been written to partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when programming is performed. In this case, both the user MAT and user boot MAT have not been written to. Programming the user boot MAT should be performed in boot mode or programmer mode. 0: Programming has ended normally 1: Programming has ended abnormally (programming result is not guaranteed)
4	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'5A) before programming starts, and returns the result. 0: FKEY setting is normal (H'5A) 1: FKEY setting is abnormal (value other than H'5A)

Bit	Bit Name	Initial Value	R/W	Description
3	—	—	—	Unused Returns 0.
2	WD	—	R/W	Write Data Address Detect When an address not in the flash memory area is specified as the start address of the storage destination for the program data, an error occurs. 0: Setting of the start address of the storage destination for the program data is normal 1: Setting of the start address of the storage destination for the program data is abnormal
1	WA	—	R/W	Write Address Error Detect When the following items are specified as the start address of the programming destination, an error occurs. • An area other than flash memory • The specified address is not aligned with the 128-byte boundary (lower eight bits of the address are other than H'00 and H'80) 0: Setting of the start address of the programming destination is normal 1: Setting of the start address of the programming destination is abnormal
0	SF	—	R/W	Success/Fail Returns the programming result. 0: Programming has ended normally (no error) 1: Programming has ended abnormally (error occurs)

(c) Erasure

FPFR indicates the return value of the erasure result.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	MD	EE	FK	EB	—	—	SF

Bit	Bit Name	Initial Value	R/W	Description
7	—	—	—	Unused Returns 0.
6	MD	—	R/W	Erasure Mode Related Setting Error Detect Detects the error protection state and returns the result. When the error protection state is entered, this bit is set to 1. Whether the error protection state is entered or not can be confirmed with the FLER bit in FCCS. For conditions to enter the error protection state, see section 17.9.3, Error Protection. 0: Normal operation (FLER = 0) 1: Error protection state, and programming cannot be performed (FLER = 1)
5	EE	—	R/W	Erasure Execution Error Detect Returns 1 when the user MAT could not be erased or when the flash memory related register settings are partially changed. If this bit is set to 1, there is a high possibility that the user MAT has been erased partially. In this case, after removing the error factor, erase the user MAT. If FMATS is set to H'AA and the user boot MAT is selected, an error occurs when erasure is performed. In this case, both the user MAT and user boot MAT have not been erased. Erasing of the user boot MAT should be performed in boot mode or programmer mode. 0: Erasure has ended normally 1: Erasure has ended abnormally

Bit	Bit Name	Initial Value	R/W	Description
4	FK	—	R/W	Flash Key Register Error Detect Checks the FKEY value (H'5A) before erasure starts, and returns the result. 0: FKEY setting is normal (H'5A) 1: FKEY setting is abnormal (value other than H'5A)
3	EB	—	R/W	Erase Block Select Error Detect Checks whether the specified erase block number is in the block range of the user MAT, and returns the result. 0: Setting of erase block number is normal 1: Setting of erase block number is abnormal
2, 1	—	—	—	Unused These bits return 0.
0	SF	—	R/W	Success/Fail Indicates the erasure result. 0: Erasure has ended normally (no error) 1: Erasure has ended abnormally (error occurs)

(3) Flash Program/Erase Frequency Parameter (FPEFEQ: General Register ER0 of CPU)

FPEFEQ sets the operating frequency of the CPU. The CPU operating frequency available in this LSI ranges from 8 to 48 MHz.

Bit	31	30	29	28	27	26	25	24
Bit Name	—	—	—	—	—	—	—	—

Bit	23	22	21	20	19	18	17	16
Bit Name	—	—	—	—	—	—	—	—

Bit	15	14	13	12	11	10	9	8
Bit Name	F15	F14	F13	F12	F11	F10	F9	F8

Bit	7	6	5	4	3	2	1	0
Bit Name	F7	F6	F5	F4	F3	F2	F1	F0

Bit	Bit Name	Initial Value	R/W	Description
31 to 16	—	—	—	Unused These bits should be cleared to 0.
15 to 0	F15 to F0	—	R/W	Frequency Set These bits set the operating frequency of the CPU. When the PLL multiplication function is used, set the multiplied frequency. The setting value must be calculated as follows: 1. The operating frequency shown in MHz units must be rounded in a number of three decimal places and be shown in a number of two decimal places. 2. The value multiplied by 100 is converted to the binary digit and is written to FPEFEQ (general register ER0). For example, when the operating frequency of the CPU is 33.000 MHz, the value is as follows: 1. The number of three decimal places of 33.000 is rounded. 2. The formula of 33.00 × 100 = 3300 is converted to the binary digit and B'0000 1100 1110 0100 (H'0CE4) is set to ER0.

(4) Flash Multipurpose Address Area Parameter (FMPAR: General Register ER1 of CPU)

FMPAR stores the start address of the programming destination on the user MAT.

When an address in an area other than the flash memory is set, or the start address of the programming destination is not aligned with the 128-byte boundary, an error occurs. The error occurrence is indicated by the WA bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOA31	MOA30	MOA29	MOA28	MOA27	MOA26	MOA25	MOA24

Bit	23	22	21	20	19	18	17	16
Bit Name	MOA23	MOA22	MOA21	MOA20	MOA19	MOA18	MOA17	MOA16

Bit	15	14	13	12	11	10	9	8
Bit Name	MOA15	MOA14	MOA13	MOA12	MOA11	MOA10	MOA9	MOA8

Bit	7	6	5	4	3	2	1	0
Bit Name	MOA7	MOA6	MOA5	MOA4	MOA3	MOA2	MOA1	MOA0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOA31 to MOA0	—	R/W	These bits store the start address of the programming destination on the user MAT. Consecutive 128-byte programming is executed starting from the specified start address of the user MAT. Therefore, the specified start address of the programming destination becomes a 128-byte boundary, and MOA6 to MOA0 are always cleared to 0.

(5) Flash Multipurpose Data Destination Parameter (FMPDR: General Register ER0 of CPU)

FMPDR stores the start address in the area which stores the data to be programmed in the user MAT.

When the storage destination for the program data is in flash memory, an error occurs. The error occurrence is indicated by the WD bit in FPFR.

Bit	31	30	29	28	27	26	25	24
Bit Name	MOD31	MOD30	MOD29	MOD28	MOD27	MOD26	MOD25	MOD24

Bit	23	22	21	20	19	18	17	16
Bit Name	MOD23	MOD22	MOD21	MOD20	MOD19	MOD18	MOD17	MOD16

Bit	15	14	13	12	11	10	9	8
Bit Name	MOD15	MOD14	MOD13	MOD12	MOD11	MOD10	MOD9	MOD8

Bit	7	6	5	4	3	2	1	0
Bit Name	MOD7	MOD6	MOD5	MOD4	MOD3	MOD2	MOD1	MOD0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	MOD31 to MOD0	—	R/W	These bits store the start address of the area which stores the program data for the user MAT. Consecutive 128-byte data is programmed to the user MAT starting from the specified start address.

(6) Flash Erase Block Select Parameter (FEBS: General Register ER0 of CPU)

FEBS specifies the erase block number. Settable values for the erase block numbers range from 0 to 11 (H'00000000 to H'0000000B). A value of 0 corresponds to block EB0 and a value of 11 corresponds to block EB11. An error occurs when a value outside the range from 0 to 11 is set.

Bit	31	30	29	28	27	26	25	24
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	23	22	21	20	19	18	17	16
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	15	14	13	12	11	10	9	8
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name								
Initial Value	—	—	—	—	—	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

(7) Flash User Branch Address Set Parameter (FUBRA: General Register ER0 of CPU)

This parameter sets the user branch destination address. The user program which has been set can be executed in specified processing units when programming and erasing.

When the user branch is not required, address 0 (H'00000000) must be set.

The user branch destination must be an area other than the on-chip flash memory or an area other than the RAM area in which on-chip program has been transferred.

Note that the CPU must not branch to an area without the execution code and get out of control. The on-chip program download area and stack area must not be overwritten. If CPU runaway occurs or the download area or stack area is overwritten, the value of flash memory cannot be guaranteed.

The download of the on-chip program, initialization, initiation of the programming/erasing program must not be executed in the processing of the user branch destination. Programming or erasing cannot be guaranteed when returning from the user branch destination. The program data which has already been prepared must not be programmed.

Store general registers ER2 to ER7. General registers ER0 and ER1 are available without storing them.

Moreover, the programming/erasing interface registers must not be written to or RAM emulation mode must not be entered in the processing of the user branch destination.

After the processing of the user branch has ended, the programming/erasing program must be returned to by using the RTS instruction.

Bit	31	30	29	28	27	26	25	24
Bit Name	UA31	UA30	UA29	UA28	UA27	UA26	UA25	UA24

Bit	23	22	21	20	19	18	17	16
Bit Name	UA23	UA22	UA21	UA20	UA19	UA18	UA17	UA16

Bit	15	14	13	12	11	10	9	8
Bit Name	UA15	UA14	UA13	UA12	UA11	UA10	UA9	UA8

Bit	7	6	5	4	3	2	1	0
Bit Name	UA7	UA6	UA5	UA4	UA3	UA2	UA1	UA0

Bit	Bit Name	Initial Value	R/W	Description
31 to 0	UA31 to UA0	—	R/W	Sets the start address in the user branch destination.

17.7.3 RAM Emulation Register (RAMER)

RAMER specifies the user MAT area overlaid with part of the on-chip RAM (H'FFA000 to H'FFAFFF) when performing emulation of programming the user MAT. RAMER should be set in user mode or user program mode. To ensure dependable emulation, the memory MAT to be emulated must not be accessed immediately after changing the RAMER contents. When accessed at such a timing, correct operation is not guaranteed.

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	RAMS	RAM2	RAM1	RAM0
Initial Value	0	0	0	0	0	0	0	0
R/W	R	R	R	R	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
7 to 4	—	0	R	Reserved These are read-only bits and cannot be modified.
3	RAMS	0	R/W	RAM Select Selects the function which emulates the flash memory using the on-chip RAM. 0: Disables RAM emulation function 1: Enables RAM emulation function (all blocks of the user MAT are protected against programming and erasing)
2	RAM2	0	R/W	Flash Memory Area Select These bits select the user MAT area overlaid with the on-chip RAM when RAMS = 1. The following areas correspond to the 4-kbyte erase blocks. 000: H'000000 to H'000FFF (EB0) 001: H'001000 to H'001FFF (EB1) 010: H'002000 to H'002FFF (EB2) 011: H'003000 to H'003FFF (EB3) 100: H'004000 to H'004FFF (EB4) 101: H'005000 to H'005FFF (EB5) 110: H'006000 to H'006FFF (EB6) 111: H'007000 to H'007FFF (EB7)
1	RAM1	0	R/W	
0	RAM0	0	R/W	

17.8 On-Board Programming Mode

When the mode pins (MD0, MD1, and MD2) are set to on-board programming mode and the reset start is executed, a transition is made to on-board programming mode in which the on-chip flash memory can be programmed/erased. On-board programming mode has three operating modes: boot mode, user boot mode, and user program mode.

Table 17.5 shows the pin setting for each operating mode. For details on the state transition of each operating mode for flash memory, see figure 17.2.

Table 17.5 On-Board Programming Mode Setting

Mode Setting	MD1	MD0
User boot mode	0	1
Boot mode	1	0
User program mode	1	1

17.8.1 Boot Mode

Boot mode executes programming/erasing of the user MAT or user boot MAT by means of the control command and program data transmitted from the externally connected host via the on-chip SCI_4.

In boot mode, the tool for transmitting the control command and program data, and the program data must be prepared in the host. The serial communication mode is set to asynchronous mode. The system configuration in boot mode is shown in figure 17.6. Interrupts are ignored in boot mode. Configure the user system so that interrupts do not occur.

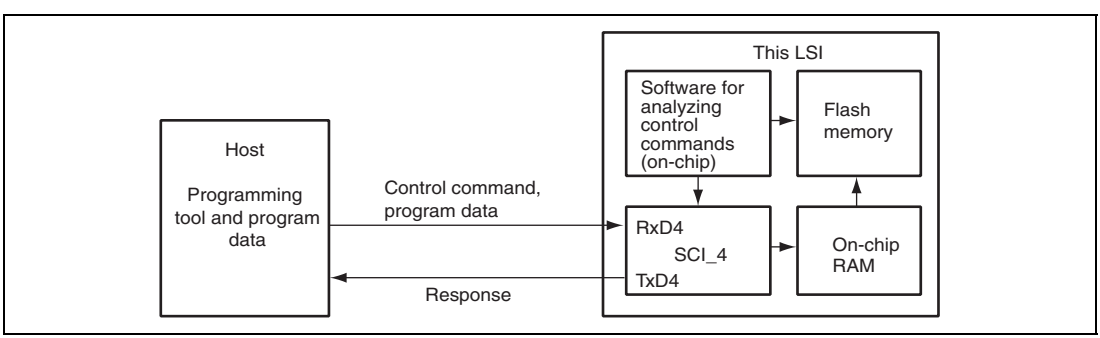


Figure 17.6 System Configuration in Boot Mode

(1) Serial Interface Setting by Host

The SCI_4 is set to asynchronous mode, and the serial transmit/receive format is set to 8-bit data, one stop bit, and no parity.

When a transition to boot mode is made, the boot program embedded in this LSI is initiated.

When the boot program is initiated, this LSI measures the low period of asynchronous serial communication data (H'00) transmitted consecutively by the host, calculates the bit rate, and adjusts the bit rate of the SCI_4 to match that of the host.

When bit rate adjustment is completed, this LSI transmits 1 byte of H'00 to the host as the bit adjustment end sign. When the host receives this bit adjustment end sign normally, it transmits 1 byte of H'55 to this LSI. When reception is not executed normally, initiate boot mode again. The bit rate may not be adjusted within the allowable range depending on the combination of the bit rate of the host and the system clock frequency of this LSI. Therefore, the transfer bit rate of the host and the system clock frequency of this LSI must be as shown in table 17.6.

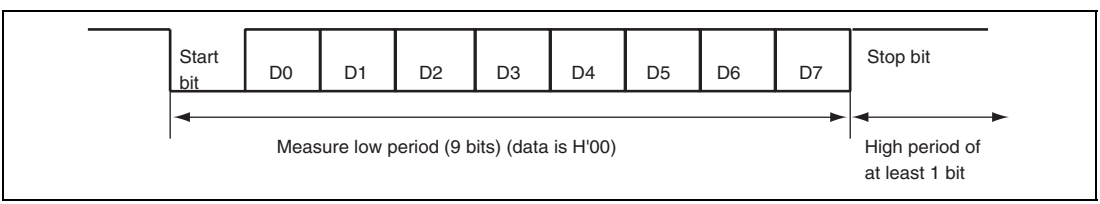


Figure 17.7 Automatic-Bit-Rate Adjustment Operation

Table 17.6 System Clock Frequency for Automatic-Bit-Rate Adjustment

Bit Rate of Host	System Clock Frequency of This LSI	External Clock Frequency
9,600 bps	8 to 18 MHz	4 to 9 MHz
19,200 bps	16 to 18 MHz	8 to 9 MHz

(2) State Transition Diagram

The state transition after boot mode is initiated is shown in figure 17.8.

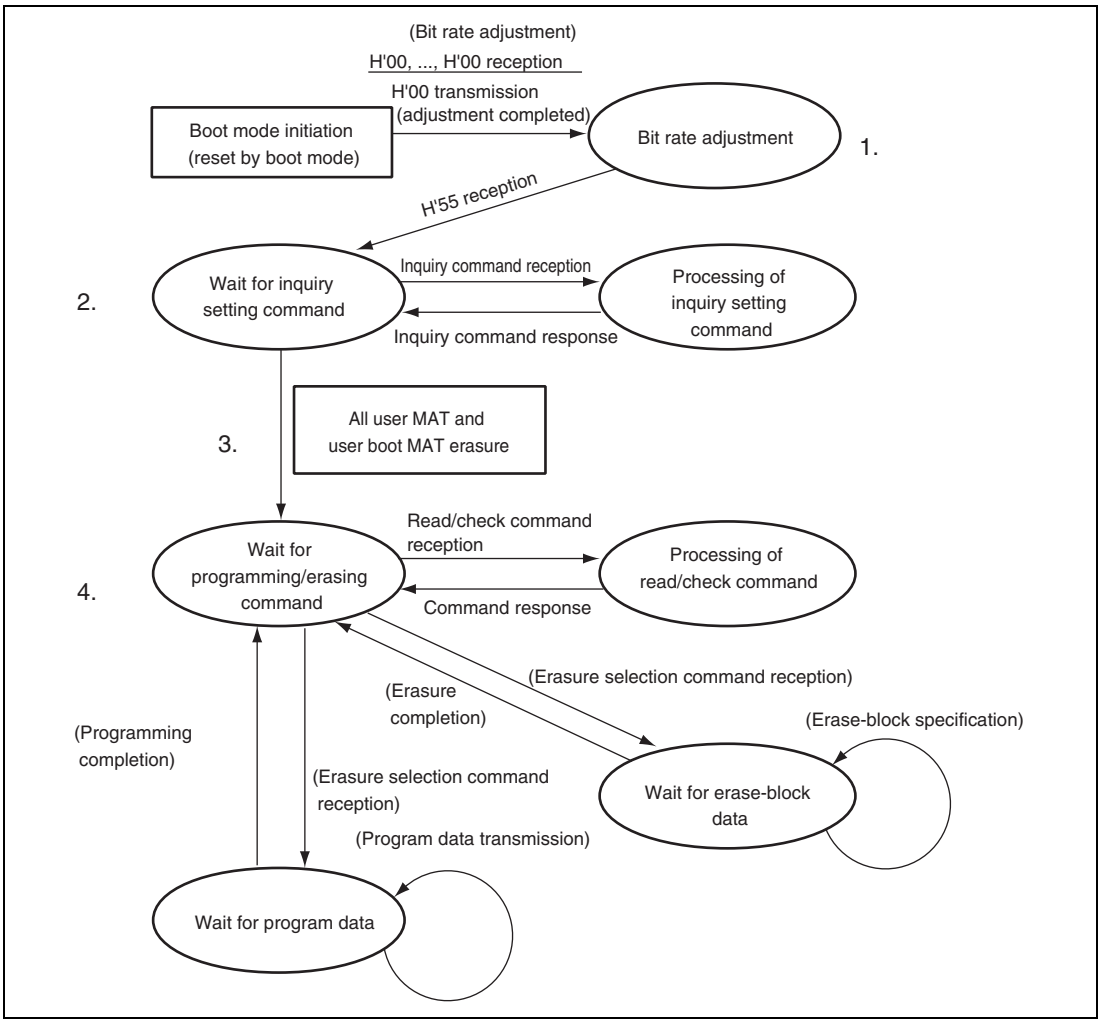


Figure 17.8 Boot Mode State Transition Diagram

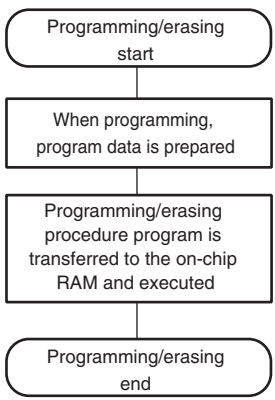
1. After boot mode is initiated, the bit rate of the SCI_4 is adjusted with that of the host.
2. Inquiry information about the size, configuration, start address, and support status of the user MAT is transmitted to the host.
3. After inquiries have finished, all user MAT and user boot MAT are automatically erased.
4. When the program preparation notice is received, the state of waiting for program data is entered. The start address of the programming destination and program data must be transmitted after the programming command is transmitted. When programming is finished, the start address of the programming destination must be set to H'FFFFFFFF and transmitted. Then the state of waiting for program data is returned to the state of waiting for programming/erasing command. When the erasure preparation notice is received, the state of waiting for erase block data is entered. The erase block number must be transmitted after the erasing command is transmitted. When the erasure is finished, the erase block number must be set to H'FF and transmitted. Then the state of waiting for erase block data is returned to the state of waiting for programming/erasing command. Erasure must be executed when the specified block is programmed without a reset start after programming is executed in boot mode. When programming can be executed by only one operation, all blocks are erased before entering the state of waiting for programming/erasing command or another command. Thus, in this case, the erasing operation is not required. The commands other than the programming/erasing command perform sum check, blank check (erasure check), and memory read of the user MAT/user boot MAT and acquisition of current status information.

Memory read of the user MAT/user boot MAT can only read the data programmed after all user MAT/user boot MAT has automatically been erased. No other data can be read.

17.8.2 User Program Mode

Programming/erasing of the user MAT is executed by downloading an on-chip program. The user boot MAT cannot be programmed/erased in user program mode. The programming/erasing flow is shown in figure 17.9.

Since high voltage is applied to the internal flash memory during programming/erasing, a transition to the reset state or hardware standby mode must not be made during programming/erasing. A transition to the reset state or hardware standby mode during programming/erasing may damage the flash memory. If a reset is input, the reset must be released after the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.



1. Exit RAM emulation mode beforehand. Download is not allowed in emulation mode.
2. When the program data is adjusted in emulation mode, select the download destination specified by FTDAR carefully. Make sure that the download area does not overlap the emulation area.
3. Programming/erasing is executed only in the on-chip RAM.
4. After programming/erasing is finished, protect the flash memory by the hardware protection.

Figure 17.9 Programming/Erasing Flow

(1) On-Chip RAM Address Map when Programming/Erasing is Executed

Parts of the procedure program that is made by the user, like download request, programming/erasing procedure, and decision of the result, must be executed in the on-chip RAM. Since the on-chip program to be downloaded is embedded in the on-chip RAM, make sure the on-chip program and procedure program do not overlap. Figure 17.10 shows the area of the on-chip program to be downloaded.

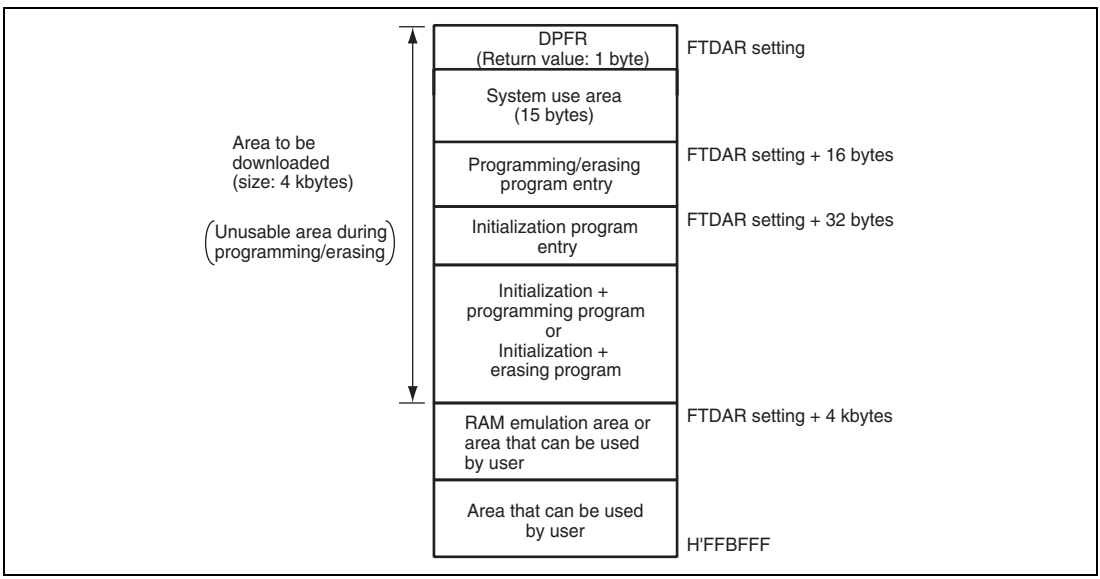


Figure 17.10 RAM Map when Programming/Erasing is Executed

(2) Programming Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and programming are shown in figure 17.11.

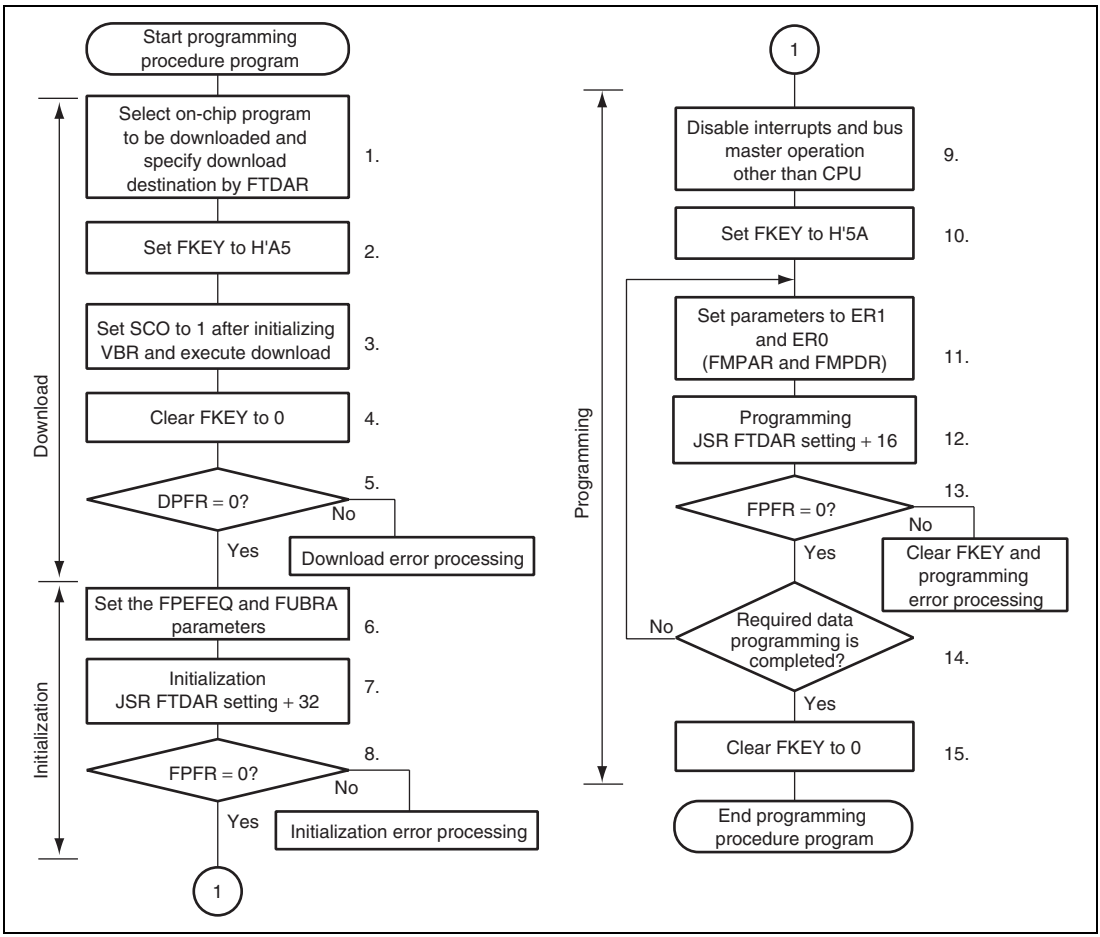


Figure 17.11 Programming Procedure in User Program Mode

The procedure program must be executed in an area other than the flash memory to be programmed. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM and user MAT) is shown in section 17.8.4, On-Chip Program and Storable Area for Program Data. The following description assumes that the area to be programmed on the user MAT is erased and that program data is prepared in the consecutive area.

The program data for one programming operation is always 128 bytes. When the program data exceeds 128 bytes, the start address of the programming destination and program data parameters are updated in 128-byte units and programming is repeated. When the program data is less than 128 bytes, invalid data is filled to prepare 128-byte program data. If the invalid data to be added is H'FF, the program processing time can be shortened.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.
2. Write H'A5 in FKEY. If H'A5 is not written to FKEY, the SCO bit in FCCS cannot be set to 1 to request download of the on-chip program.
3. After initializing VBR to H'00000000, set the SCO bit to 1 to execute download. To set the SCO bit to 1, all of the following conditions must be satisfied.
   - RAM emulation mode has been canceled.
   - H'A5 is written to FKEY.
   - Setting the SCO bit is executed in the on-chip RAM.

   When the SCO bit is set to 1, download is started automatically. Since the SCO bit is cleared to 0 when the procedure program is resumed, the SCO bit cannot be confirmed to be 1 in the procedure program. The download result can be confirmed by the return value of the DPFR parameter. To prevent incorrect decision, before setting the SCO bit to 1, set one byte of the on-chip RAM start address specified by FTDAR, which becomes the DPFR parameter, to a value other than the return value (e.g. H'FF). Since particular processing that is accompanied by bank switching as described below is performed when download is executed, initialize the VBR contents to H'00000000. Dummy read of FCCS must be performed twice immediately after the SCO bit is set to 1.
   - The user-MAT space is switched to the on-chip program storage area.
   - After the program to be downloaded and the on-chip RAM start address specified by FTDAR are checked, they are transferred to the on-chip RAM.
   - FPCS, FECS, and the SCO bit in FCCS are cleared to 0.
   - The return value is set in the DPFR parameter.
   - After the on-chip program storage area is returned to the user-MAT space, the procedure program is resumed. After that, VBR can be set again.
   - The values of general registers of the CPU are held.
   - During download, no interrupts can be accepted. However, since the interrupt requests are held, when the procedure program is resumed, the interrupts are requested.

— To hold a level-detection interrupt request, the interrupt must continue to be input until the download is completed.

— Allocate a stack area of 128 bytes at the maximum in the on-chip RAM before setting the SCO bit to 1.

— If access to the flash memory is requested by the DMAC during download, the operation cannot be guaranteed. Make sure that an access request by the DMAC is not generated.

4. FKEY is cleared to H'00 for protection.
5. The download result must be confirmed by the value of the DPFR parameter. Check the value of the DPFR parameter (one byte of start address of the download destination specified by FTDAR). If the value of the DPFR parameter is H'00, download has been performed normally. If the value is not H'00, the source that caused download to fail can be investigated by the description below.

   — If the value of the DPFR parameter is the same as that before downloading, the setting of the start address of the download destination in FTDAR may be abnormal. In this case, confirm the setting of the TDER bit in FTDAR.

   — If the value of the DPFR parameter is different from that before downloading, check the SS bit or FK bit in the DPFR parameter to confirm the download program selection and FKEY setting, respectively.
6. The operating frequency and the user branch destination are set to the FPEFEQ and FUBRA parameters, respectively, for initialization.

   — The operating frequency of the CPU is set to the FPEFEQ parameter. The operating frequency settable to the FPEFEQ parameter ranges from 8 to 48 MHz. When a value out of this frequency range is set, an error is returned to the FPFR parameter of the initialization program and initialization is not performed. For details on the frequency setting, see the description in (3) Flash Program/Erase Frequency Parameter (FPEFEQ) of 17.7.2 Programming/Erasing Interface Parameters.

   — The start address in the user branch destination is set to the FUBRA parameter. When the user branch processing is not required, H'00000000 must be set to FUBRA. When the user branch is executed, the branch destination is executed in flash memory other than the one that is to be programmed. The area of the on-chip program that is downloaded cannot be set. The program processing must be returned from the user branch processing by the RTS instruction. See the description in (7) Flash User Branch Address Set Parameter (FUBRA) of 17.7.2 Programming/Erasing Interface Parameters.

7. Initialization is executed. The initialization program is downloaded together with the programming program to the on-chip RAM. The entry point of the initialization program is at the address which is 32 bytes after #DLTOP (when the start address of the download destination specified by FTDAR is #DLTOP). Call the subroutine to execute initialization by using the following steps.

```
MOV.L #DLTOP+32,ER2    ; Set entry address to ER2
JSR   @ER2             ; Call initialization routine
NOP
```

   — The general registers other than ER0 and ER1 are held in the initialization program.
   — R0L is a return value of the FPFR parameter.
   — Since the stack area is used in the initialization program, a stack area of 128 bytes at the maximum must be allocated in RAM.
   — Interrupts can be accepted during execution of the initialization program. Make sure the program storage area and stack area in the on-chip RAM and register values are not overwritten.
8. The return value in the initialization program, the FPFR parameter is determined.
9. All interrupts and the use of a bus master other than the CPU are disabled during programming/erasing. The specified voltage is applied for the specified time when programming or erasing. If interrupts occur or the bus mastership is moved to other than the CPU during programming/erasing, causing a voltage exceeding the specifications to be applied, the flash memory may be damaged. Therefore, interrupts are disabled by setting bit 7 (I bit) in the condition code register (CCR) to B'1 in interrupt control mode 0 and by setting bits 2 to 0 (I2 to I0 bits) in the extend register (EXR) to B'111 in interrupt control mode 2. Accordingly, interrupts other than NMI are held and not executed. Configure the user system so that NMI interrupts do not occur. The interrupts that are held must be executed after all programming completes. When the bus mastership is moved to other than the CPU, such as to the DMAC, the error protection state is entered. Therefore, make sure the DMAC does not acquire the bus.
10. FKEY must be set to H'5A and the user MAT must be prepared for programming.

11. The parameters required for programming are set. The start address of the programming destination on the user MAT (FMPAR parameter) is set in general register ER1. The start address of the program data storage area (FMPDR parameter) is set in general register ER0.
    - Example of FMPAR parameter setting: When an address other than one in the user MAT area is specified for the start address of the programming destination, even if the programming program is executed, programming is not executed and an error is returned to the FPFR parameter. Since the program data for one programming operation is 128 bytes, the lower eight bits of the address must be H'00 or H'80 to be aligned with the 128-byte boundary.
    - Example of FMPDR parameter setting: When the storage destination for the program data is flash memory, even if the programming routine is executed, programming is not executed and an error is returned to the FPFR parameter. In this case, the program data must be transferred to the on-chip RAM and then programming must be executed.
12. Programming is executed. The entry point of the programming program is at the address which is 16 bytes after #DLTOP (when start address of the download destination specified by FTDAR is #DLTOP). Call the subroutine to execute programming by using the following steps.

```
MOV.L    #DLTOP+16,ER2   ; Set entry address to ER2
JSR      @ER2            ; Call programming routine
NOP
```

    - The general registers other than ER0 and ER1 are held in the programming program.
    - R0L is a return value of the FPFR parameter.
    - Since the stack area is used in the programming program, a stack area of 128 bytes at the maximum must be allocated in RAM.
13. The return value in the programming program, the FPFR parameter is determined.
14. Determine whether programming of the necessary data has finished. If more than 128 bytes of data are to be programmed, update the FMPAR and FMPDR parameters in 128-byte units, and repeat steps 11 to 14. Increment the programming destination address by 128 bytes and update the programming data pointer correctly. If an address which has already been programmed is written to again, not only will a programming error occur, but also flash memory will be damaged.
15. After programming finishes, clear FKEY and specify software protection. If this LSI is restarted by a reset immediately after programming has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.

(3) Erasing Procedure in User Program Mode

The procedures for download of the on-chip program, initialization, and erasing are shown in figure 17.12.

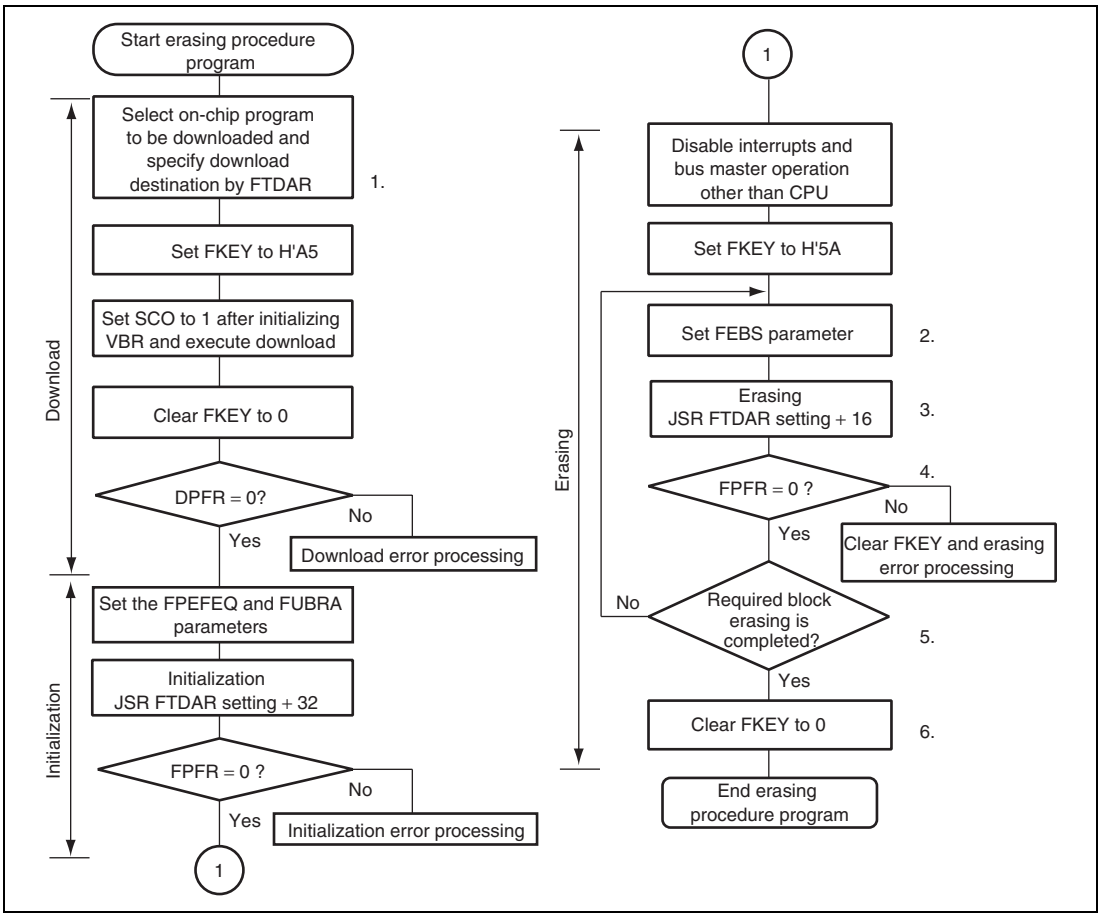


Figure 17.12 Erasing Procedure in User Program Mode

The procedure program must be executed in an area other than the user MAT to be erased. Setting the SCO bit in FCCS to 1 to request download must be executed in the on-chip RAM. The area that can be executed in the steps of the procedure program (on-chip RAM and user MAT) is shown in section 17.8.4, On-Chip Program and Storable Area for Program Data. For the downloaded on-chip program area, see figure 17.10.

One erasure processing erases one block. For details on block divisions, refer to figure 17.4. To erase two or more blocks, update the erase block number and repeat the erasing processing for each block.

1. Select the on-chip program to be downloaded and the download destination. When the PPVS bit in FPCS is set to 1, the programming program is selected. Several programming/erasing programs cannot be selected at one time. If several programs are selected, a download error is returned to the SS bit in the DPFR parameter. The on-chip RAM start address of the download destination is specified by FTDAR.

   For the procedures to be carried out after setting FKEY, see section 17.8.2 (2), Programming Procedure in User Program Mode.
2. Set the FEBS parameter necessary for erasure. Set the erase block number (FEBS parameter) of the user MAT in general register ER0. If a value other than an erase block number of the user MAT is set, no block is erased even though the erasing program is executed, and an error is returned to the FPFR parameter.
3. Erasure is executed. Similar to as in programming, the entry point of the erasing program is at the address which is 16 bytes after #DLTOP (when the start address of the download destination specified by FTDAR is #DLTOP). Call the subroutine to execute erasure by using the following steps.

```
MOV.L #DLTOP+16, ER2      ; Set entry address to ER2
JSR   @ER2                ; Call erasing routine
NOP
```

   - The general registers other than ER0 and ER1 are held in the erasing program.
   - R0L is a return value of the FPFR parameter.
   - Since the stack area is used in the erasing program, a stack area of 128 bytes at the maximum must be allocated in RAM.
4. The return value in the erasing program, the FPFR parameter is determined.
5. Determine whether erasure of the necessary blocks has finished. If more than one block is to be erased, update the FEBS parameter and repeat steps 2 to 5.
6. After erasure completes, clear FKEY and specify software protection. If this LSI is restarted by a power-on reset immediately after erasure has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs.

(4) Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

By changing the on-chip RAM start address of the download destination in FTDAR, the erasing program and programming program can be downloaded to separate on-chip RAM areas.

Figure 17.13 shows a repeating procedure of erasing, programming, and RAM emulation.

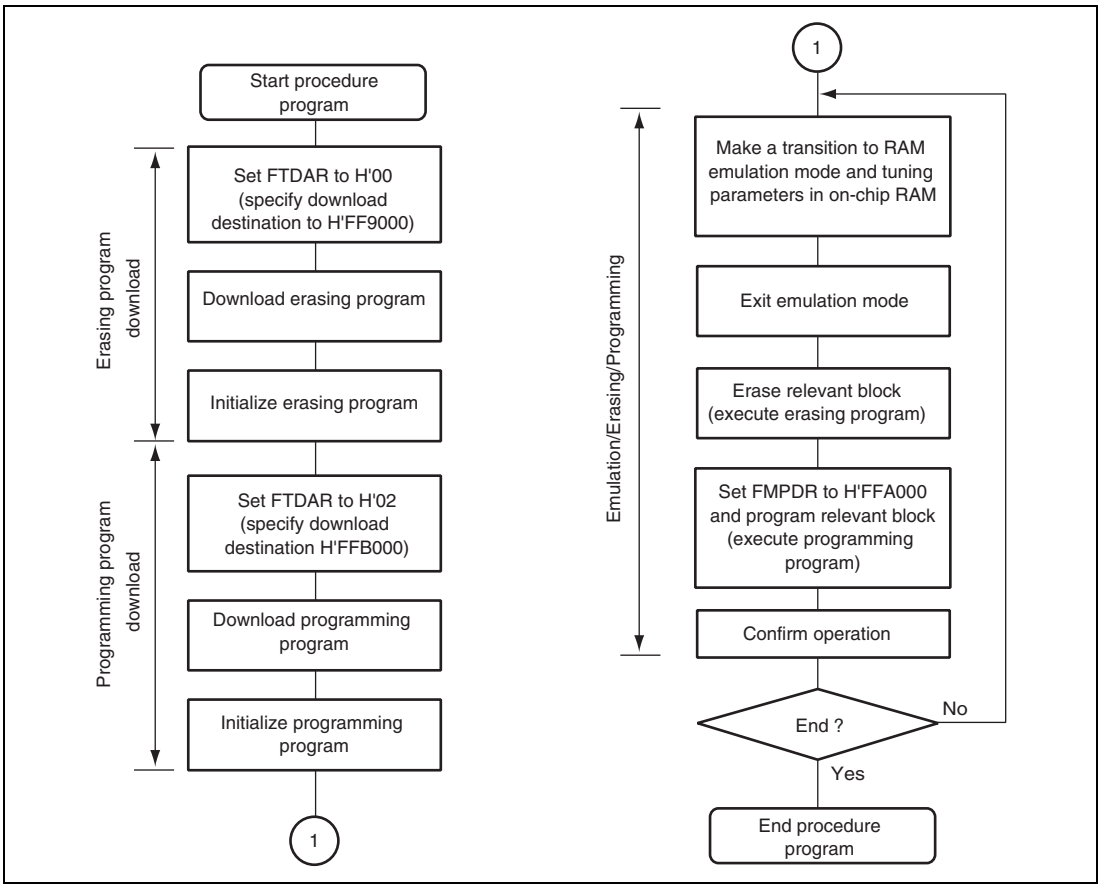


Figure 17.13 Repeating Procedure of Erasing, Programming, and RAM Emulation in User Program Mode

In figure 17.13, since RAM emulation is performed, the erasing/programming program is downloaded to avoid the 4-kbyte on-chip RAM area (H'FFA000 to H'FFAFFF). Download and initialization are performed only once at the beginning. Note the following when executing the procedure program.

- Be careful not to overwrite data in the on-chip RAM with overlay settings. In addition to the programming program area, erasing program area, and RAM emulation area, areas for the procedure programs, work area, and stack area are reserved in the on-chip RAM. Do not make settings that will overwrite data in these areas.
- Be sure to initialize both the programming program and erasing program. When the FPEFEQ and FUBRA parameters are initialized, also initialize both the erasing program and programming program. Initialization must be executed for both entry addresses: 32 bytes after #DLTOP (when the start address of download destination for erasing program is #DLTOP), and 32 bytes after #DLTOP (when the start address of download destination for programming program is #DLTOP).

17.8.3 User Boot Mode

Branching to a programming/erasing program prepared by the user enables user boot mode which is a user-arbitrary boot mode to be used.

Only the user MAT can be programmed/erased in user boot mode. Programming/erasing of the user boot MAT is only enabled in boot mode or programmer mode.

(1) Initiation in User Boot Mode

When the reset start is executed with the mode pins set to user boot mode, the built-in check routine runs and checks the user MAT and user boot MAT states. While the check routine is running, NMI and all other interrupts cannot be accepted. Next, processing starts from the execution start address of the reset vector in the user boot MAT. At this point, the user boot MAT is selected (FMATS = H'AA) as the execution memory MAT.

(2) User MAT Programming in User Boot Mode

Figure 17.14 shows the procedure for programming the user MAT in user boot mode.

The difference between the programming procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 17.14. For programming the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after programming completes.

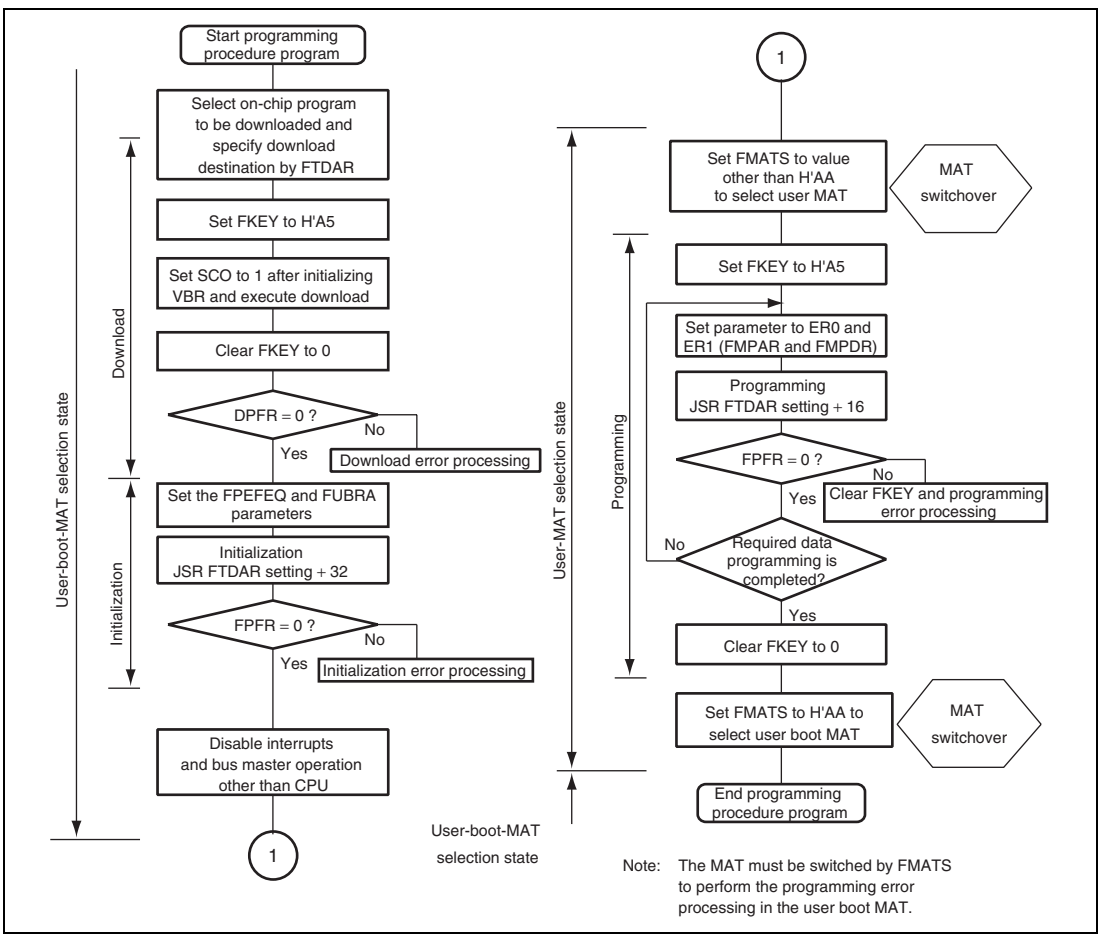


Figure 17.14 Procedure for Programming User MAT in User Boot Mode

In user boot mode, though the user boot MAT can be seen in the flash memory space, the user MAT is hidden in the background. Therefore, the user MAT and user boot MAT are switched while the user MAT is being programmed. Because the user boot MAT is hidden while the user MAT is being programmed, the procedure program must be executed in an area other than flash memory. After programming completes, switch the memory MATs again to return to the first state.

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 17.11, Switching between User MAT and User Boot MAT.

Except for memory MAT switching, the programming procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 17.8.4, On-Chip Program and Storable Area for Program Data.

(3) User MAT Erasing in User Boot Mode

Figure 17.15 shows the procedure for erasing the user MAT in user boot mode.

The difference between the erasing procedures in user program mode and user boot mode is the memory MAT switching as shown in figure 17.15. For erasing the user MAT in user boot mode, additional processing made by setting FMATS is required: switching from the user boot MAT to the user MAT, and switching back to the user boot MAT after erasing completes.

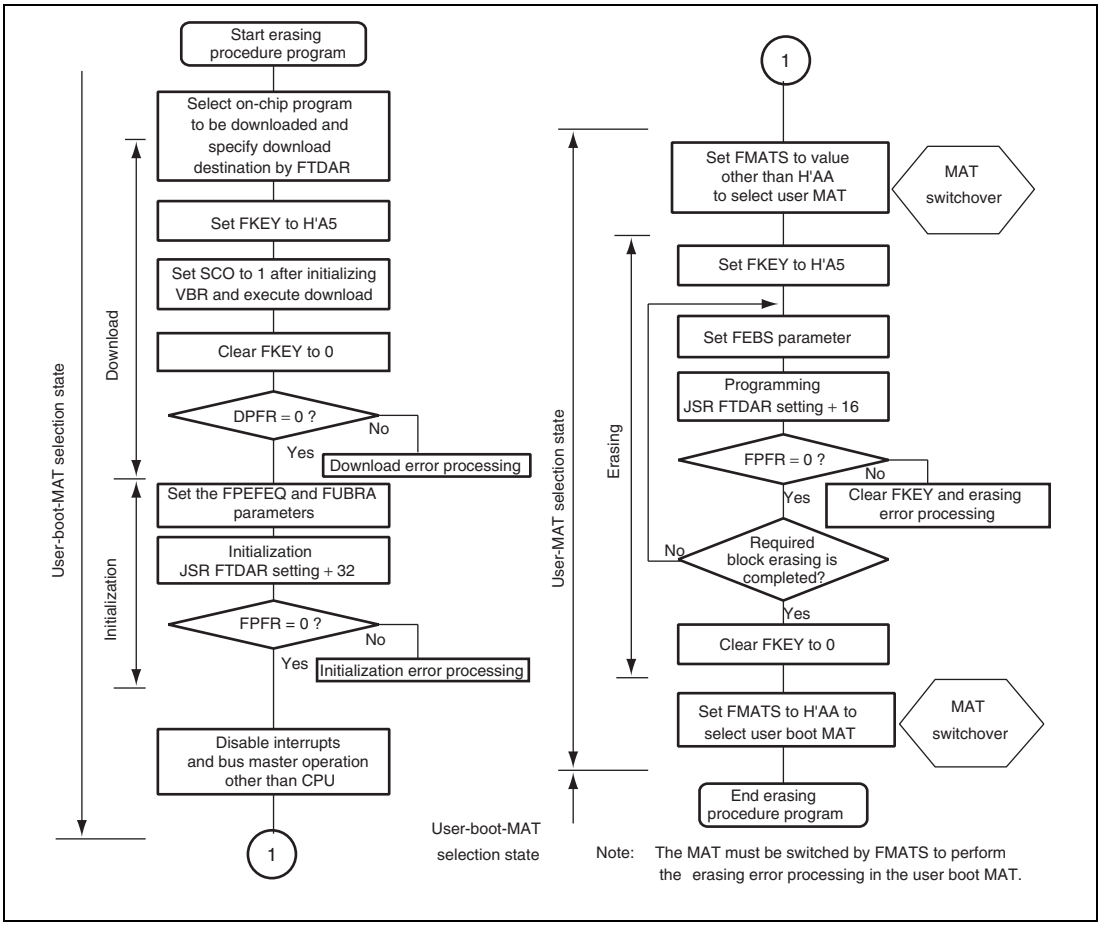


Figure 17.15 Procedure for Erasing User MAT in User Boot Mode

Memory MAT switching is enabled by setting FMATS. However note that access to a memory MAT is not allowed until memory MAT switching is completed. During memory MAT switching, the LSI is in an unstable state, e.g. if an interrupt occurs, from which memory MAT the interrupt vector is read is undetermined. Perform memory MAT switching in accordance with the description in section 17.11, Switching between User MAT and User Boot MAT.

Except for memory MAT switching, the erasing procedure is the same as that in user program mode.

The area that can be executed in the steps of the procedure program (on-chip RAM, user MAT, and external space) is shown in section 17.8.4, On-Chip Program and Storable Area for Program Data.

17.8.4 On-Chip Program and Storable Area for Program Data

In the descriptions in this manual, the on-chip programs and program data storage areas are assumed to be in the on-chip RAM. However, they can be executed from part of the flash memory which is not to be programmed or erased as long as the following conditions are satisfied.

- The on-chip program is downloaded to and executed in the on-chip RAM specified by FTDAR. Therefore, this on-chip RAM area is not available for use.
- Since the on-chip program uses a stack area, allocate 128 bytes at the maximum as a stack area.
- Download requested by setting the SCO bit in FCCS to 1 should be executed from the on-chip RAM because it will require switching of the memory MATs.
- In an operating mode in which the external address space is not accessible, such as single-chip mode, the required procedure programs, NMI handling vector table, and NMI handling routine should be transferred to the on-chip RAM before programming/erasing starts (download result is determined).
- The flash memory is not accessible during programming/erasing. Programming/erasing is executed by the program downloaded to the on-chip RAM. Therefore, the procedure program that initiates operation, the user program at the user branch destination for programming/erasing, the NMI handling vector table, and the NMI handling routine should be stored in the on-chip RAM other than the flash memory.
- After programming/erasing starts, access to the flash memory should be inhibited until FKEY is cleared. The reset input state (period of $\overline{RES}$ = 0) must be set to at least 100 μs when the operating mode is changed and the reset start executed on completion of programming/erasing. Transitions to the reset state are inhibited during programming/erasing. When the reset signal is input, a reset input state (period of $\overline{RES}$ = 0) of at least 100 μs is needed before the reset signal is released.
- Switching of the memory MATs by FMATS should be needed when programming/erasing of the user MAT is operated in user boot mode. The program which switches the memory MATs should be executed from the on-chip RAM. For details, see section 17.11, Switching between User MAT and User Boot MAT. Make sure you know which memory MAT is currently selected when switching them.
- When the program data storage area is within the flash memory area, an error will occur even when the data stored is normal program data. Therefore, the data should be transferred to the on-chip RAM to place the address that the FMPDR parameter indicates in an area other than the flash memory.

In consideration of these conditions, the areas in which the program data can be stored and executed are determined by the combination of the processing contents, operating mode, and bank structure of the memory MATs, as shown in tables 17.7 to 17.11.

Table 17.7 Executable Memory MAT

	Operating Mode	
Processing Contents	**User Program Mode**	**User Boot Mode***
Programming	See table 17.8	See table 17.10
Erasing	See table 17.9	See table 17.11

Note: * Programming/Erasing is possible to the user MAT.

Table 17.8 Usable Area for Programming in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	**On-Chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage MAT**
Storage area for program data	O	×*	—	—
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting programming parameter	O	×	O	
Execution of programming	O	×	O	
Decision of programming result	O	×	O	
Operation for programming error	O	×	O	
Operation for clearing FKEY	O	×	O	

Note: * Transferring the program data to the on-chip RAM beforehand enables this area to be used.

Table 17.9 Usable Area for Erasure in User Program Mode

	Storable/Executable Area		Selected MAT	
Item	**On-Chip RAM**	**User MAT**	**User MAT**	**Embedded Program Storage MAT**
Operation for selecting on-chip program to be downloaded	O	O	O	
Operation for writing H'A5 to FKEY	O	O	O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×		O
Operation for clearing FKEY	O	O	O	
Decision of download result	O	O	O	
Operation for download error	O	O	O	
Operation for setting initialization parameter	O	O	O	
Execution of initialization	O	×	O	
Decision of initialization result	O	O	O	
Operation for initialization error	O	O	O	
NMI handling routine	O	×	O	
Operation for disabling interrupts	O	O	O	
Operation for writing H'5A to FKEY	O	O	O	
Operation for setting erasure parameter	O	×	O	
Execution of erasure	O	×	O	
Decision of erasure result	O	×	O	
Operation for erasure error	O	×	O	
Operation for clearing FKEY	O	×	O	

Table 17.10 Usable Area for Programming in User Boot Mode

Item	Storable/Executable Area		Selected MAT		
	On-Chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage MAT
Storage area for program data	O	×*1	—	—	—
Operation for selecting on-chip program to be downloaded	O	O		O	
Operation for writing H'A5 to FKEY	O	O		O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×			O
Operation for clearing FKEY	O	O		O	
Decision of download result	O	O		O	
Operation for download error	O	O		O	
Operation for setting initialization parameter	O	O		O	
Execution of initialization	O	×		O	
Decision of initialization result	O	O		O	
Operation for initialization error	O	O		O	
NMI handling routine	O	×		O	
Operation for disabling interrupts	O	O		O	
Switching memory MATs by FMATS	O	×	O		
Operation for writing H'5A to FKEY	O	×	O		
Operation for setting programming parameter	O	×	O		
Execution of programming	O	×	O		
Decision of programming result	O	×	O		
Operation for programming error	O	×*2	O		
Operation for clearing FKEY	O	×	O		
Switching memory MATs by FMATS	O	×		O	

Notes: 1. Transferring the program data to the on-chip RAM beforehand enables this area to be used.
2. Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

Table 17.11 Usable Area for Erasure in User Boot Mode

Item	Storable/Executable Area			Selected MAT	
	On-Chip RAM	User Boot MAT	User MAT	User Boot MAT	Embedded Program Storage MAT
Operation for selecting on-chip program to be downloaded	O	O		O	
Operation for writing H'A5 to FKEY	O	O		O	
Execution of writing 1 to SCO bit in FCCS (download)	O	×			O
Operation for clearing FKEY	O	O		O	
Decision of download result	O	O		O	
Operation for download error	O	O		O	
Operation for setting initialization parameter	O	O		O	
Execution of initialization	O	×		O	
Decision of initialization result	O	O		O	
Operation for initialization error	O	O		O	
NMI handling routine	O	×		O	
Operation for disabling interrupts	O	O		O	
Switching memory MATs by FMATS	O	×	O		
Operation for writing H'5A to FKEY	O	×	O		
Operation for setting erasure parameter	O	×	O		
Execution of erasure	O	×	O		
Decision of erasure result	O	×	O		
Operation for erasure error	O	×*	O		
Operation for clearing FKEY	O	×	O		
Switching memory MATs by FMATS	O	×	O		

Note: Switching memory MATs by FMATS by a program in the on-chip RAM enables this area to be used.

17.9 Protection

There are three types of protection against the flash memory programming/erasing: hardware protection, software protection, and error protection.

17.9.1 Hardware Protection

Programming and erasure of the flash memory is forcibly disabled or suspended by hardware protection. In this state, download of an on-chip program and initialization are possible. However, programming or erasure of the user MAT cannot be performed even if the programming/erasing program is initiated, and the error in programming/erasing is indicated by the FPFR parameter.

Table 17.12 Hardware Protection

		Function to be Protected	
Item	**Description**	**Download**	**Programming/ Erasing**
Reset protection	• The programming/erasing interface registers are initialized in the reset state (including a reset by the WDT) and the programming/erasing protection state is entered. • The reset state will not be entered by a reset using the $\overline{RES}$ pin unless the $\overline{RES}$ pin is held low until oscillation has settled after a power is initially supplied. In the case of a reset during operation, hold the $\overline{RES}$ pin low for the $\overline{RES}$ pulse width given in the AC characteristics. If a reset is input during programming or erasure, data in the flash memory is not guaranteed. In this case, execute erasure and then execute programming again.	O	O

17.9.2 Software Protection

The software protection protects the flash memory against programming/erasing by disabling download of the programming/erasing program, using the key code, and by the RAMER setting.

Table 17.13 Software Protection

Item	Description	Function to be Protected	
		Download	Programming/ Erasing
Protection by SCO bit	• The programming/erasing protection state is entered when the SCO bit in FCCS is cleared to 0 to disable download of the programming/erasing programs.	O	O
Protection by FKEY	• The programming/erasing protection state is entered because download and programming/erasing are disabled unless the required key code is written in FKEY.	O	O
Emulation protection	• The programming/erasing protection state is entered when the RAMS bit in the RAM emulation register (RAMER) is set to 1.	O	O

17.9.3 Error Protection

Error protection is a mechanism for aborting programming or erasure when a CPU runaway occurs or operations not according to the programming/erasing procedures are detected during programming/erasing of the flash memory. Aborting programming or erasure in such cases prevents damage to the flash memory due to excessive programming or erasing.

If an error occurs during programming/erasing of the flash memory, the FLER bit in FCCS is set to 1 and the error protection state is entered.

- When an interrupt request, such as NMI, occurs during programming/erasing.
- When the flash memory is read from during programming/erasing (including a vector read or an instruction fetch).
- When a SLEEP instruction is executed (including software-standby mode) during programming/erasing.
- When a bus master other than the CPU, such as the DMAC, obtains bus mastership during programming/erasing.

Error protection is canceled by a reset. Note that the reset should be released after the reset input period of at least 100 μs has passed. Since high voltages are applied during programming/erasing of the flash memory, some voltage may remain after the error protection state has been entered. For this reason, it is necessary to reduce the risk of damaging the flash memory by extending the reset input period so that the charge is released.

The state-transition diagram in figure 17.16 shows transitions to and from the error protection state.

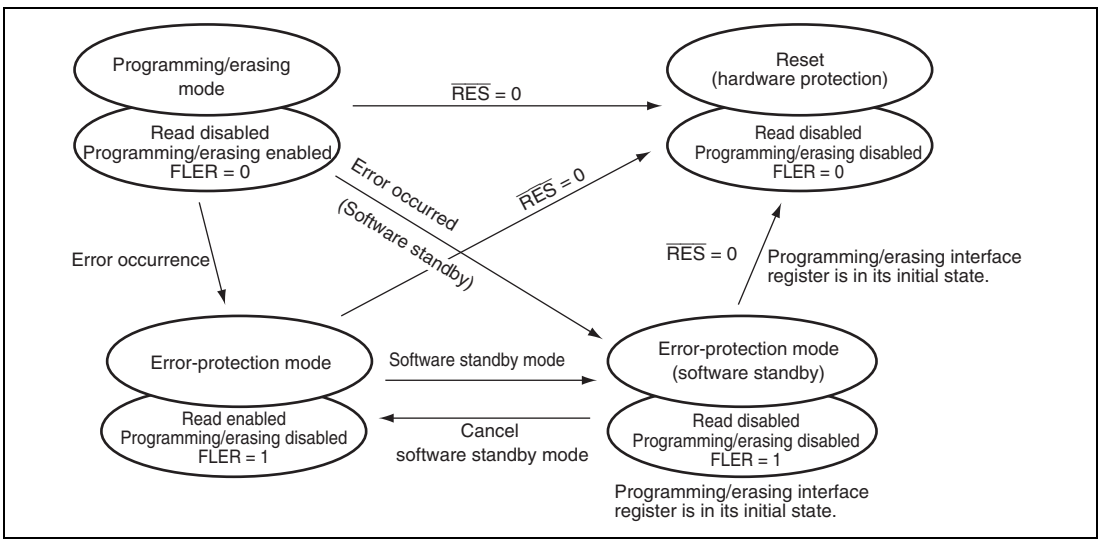


Figure 17.16 Transitions to Error Protection State

17.10 Flash Memory Emulation Using RAM

For realtime emulation of the data written to the flash memory using the on-chip RAM, the on-chip RAM area can be overlaid with several flash memory blocks (user MAT) using the RAM emulation register (RAMER).

The overlaid area can be accessed from both the user MAT area specified by RAMER and the overlaid RAM area. The emulation can be performed in user mode and user program mode.

Figure 17.17 shows an example of emulating realtime programming of the user MAT.

Note: When performing the flash memory emulation using the RAM, set the RAME bit in the system control register (SYSCR).

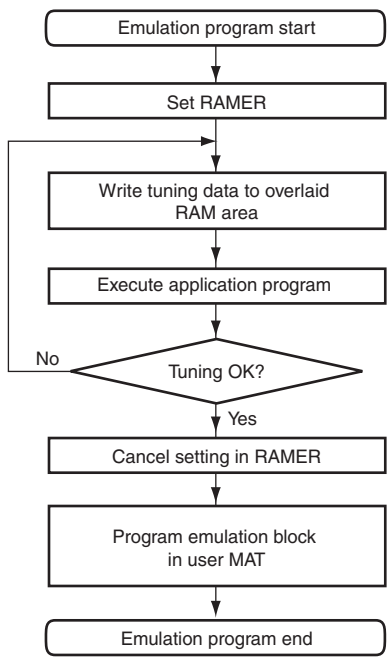


Figure 17.17 RAM Emulation Flow

Figure 17.18 shows an example of overlaying flash memory block area EB0.

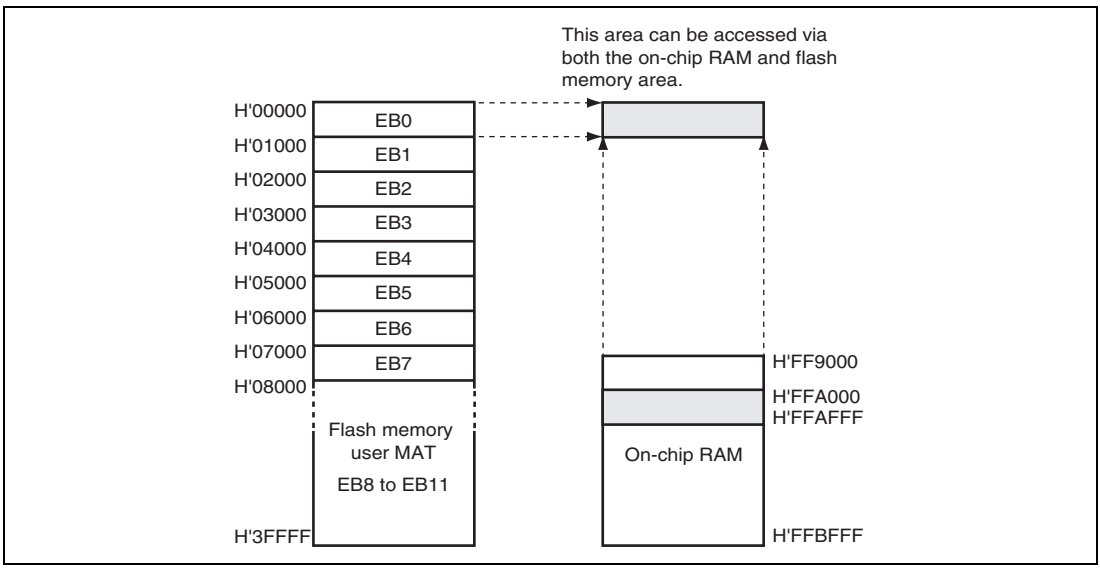


Figure 17.18 Address Map of Overlaid RAM Area

The flash memory area that can be emulated is the one area selected by bits RAM2 to RAM0 in RAMER from among the eight blocks, EB0 to EB7, of the user MAT.

To overlay a part of the on-chip RAM with block EB0 for realtime emulation, set the RAMS bit in RAMER to 1 and bits RAM2 to RAM0 to B'000.

For programming/erasing the user MAT, the procedure programs including a download program of the on-chip program must be executed. At this time, the download area should be specified so that the overlaid RAM area is not overwritten by downloading the on-chip program. Since the area in which the tuned data is stored is overlaid with the download area when FTDAR = H'01, the tuned data must be saved in an unused area beforehand.

Figure 17.19 shows an example of the procedure to program the tuned data in block EB0 of the user MAT.

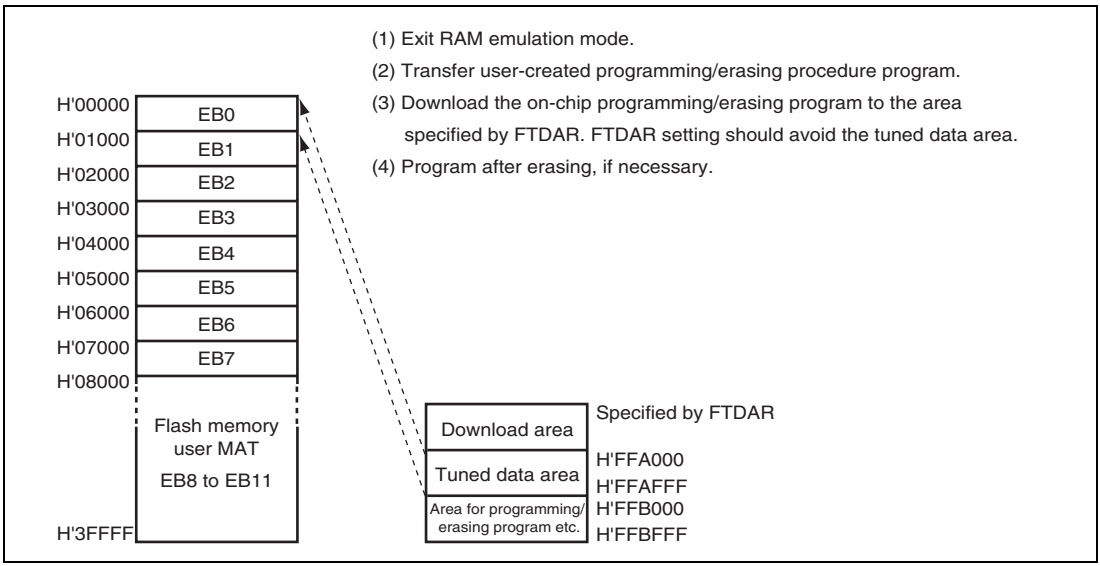


Figure 17.19 Programming Tuned Data

1. After tuning program data is completed, clear the RAMS bit in RAMER to 0 to cancel the overlaid RAM.
2. Transfer the user-created procedure program to the on-chip RAM.
3. Start the procedure program and download the on-chip program to the on-chip RAM. The start address of the download destination should be specified by FTDAR so that the tuned data area does not overlay the download area.
4. When block EB0 of the user MAT has not been erased, the programming program must be downloaded after block EB0 is erased. Specify the tuned data saved in the FMPAR and FMPDR parameters and then execute programming.

Note: Setting the RAMS bit to 1 makes all the blocks of the user MAT enter the programming/erasing protection state (emulation protection state) regardless of the setting of the RAM2 to RAM0 bits. Under this condition, the on-chip program cannot be downloaded. When data is to be actually programmed and erased, clear the RAMS bit to 0.

17.11 Switching between User MAT and User Boot MAT

It is possible to switch between the user MAT and user boot MAT. However, the following procedure is required because the start addresses of these MATs are allocated to the same address.

Switching to the user boot MAT disables programming and erasing. Programming of the user boot MAT should take place in boot mode or programmer mode.

1. Memory MAT switching by FMATS should always be executed from the on-chip RAM.
2. When accessing the memory MAT immediately after switching the memory MATs by FMATS from the on-chip RAM, similarly execute the NOP instruction in the on-chip RAM for eight times (this prevents access to the flash memory during memory MAT switching).
3. If an interrupt request has occurred during memory MAT switching, there is no guarantee of which memory MAT is accessed. Always mask the maskable interrupts before switching memory MATs. In addition, configure the system so that NMI interrupts do not occur during memory MAT switching.
4. After the memory MATs have been switched, take care because the interrupt vector table will also have been switched. If interrupt processing is to be the same before and after memory MAT switching, transfer the interrupt processing routines to the on-chip RAM and specify VBR to place the interrupt vector table in the on-chip RAM.
5. Memory sizes of the user MAT and user boot MAT are different. When accessing the user boot MAT, do not access addresses which exceed the 10-kbyte memory space. If an access is equal to or greater than 10 kbytes, the read values are undefined.

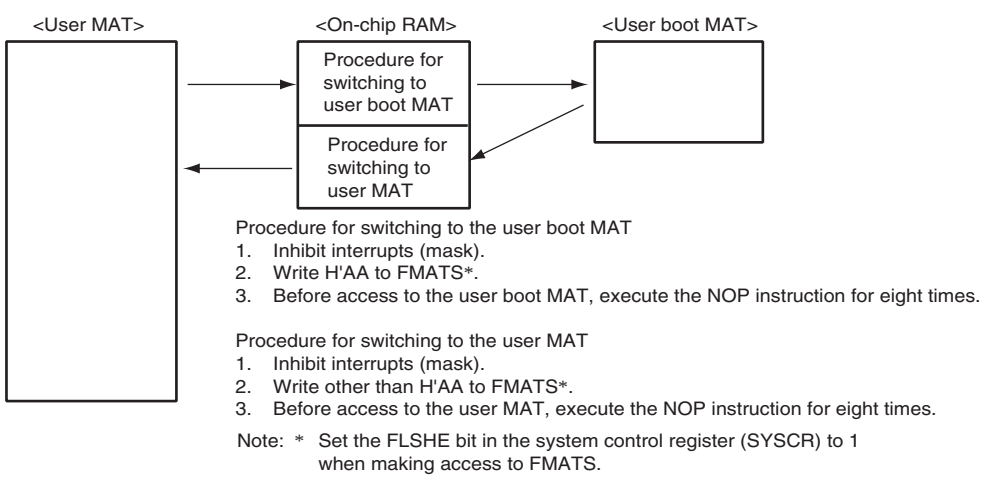


Figure 17.20 Switching between User MAT and User Boot MAT

17.12 Programmer Mode

Along with its on-board programming mode, this LSI also has a programmer mode as a further mode for the writing and erasing of programs and data. In programmer mode, a general-purpose PROM programmer that supports the device types shown in table 17.14 can be used to write programs to the on-chip ROM without any limitation.

Table 17.14 Device Types Supported in Programmer Mode

Target Memory MAT	Size	Device Type
User MAT	256 kbytes	FZTAT256V5A
User boot MAT	10 kbytes	FZTATUSBTV5A

17.13 Standard Serial Communication Interface Specifications for Boot Mode

The boot program initiated in boot mode performs serial communication using the host and on-chip SCI_4. The serial communication interface specifications are shown below.

The boot program has three states.

1. Bit-rate-adjustment state

   In this state, the boot program adjusts the bit rate to achieve serial communication with the host. Initiating boot mode enables starting of the boot program and entry to the bit-rate-adjustment state. The program receives the command from the host to adjust the bit rate. After adjusting the bit rate, the program enters the inquiry/selection state.
2. Inquiry/selection state

   In this state, the boot program responds to inquiry commands from the host. The device name, clock mode, and bit rate are selected. After selection of these settings, the program is made to enter the programming/erasing state by the command for a transition to the programming/erasing state. The program transfers the libraries required for erasure to the on-chip RAM and erases the user MATs and user boot MATs before the transition.
3. Programming/erasing state

   Programming and erasure by the boot program take place in this state. The boot program is made to transfer the programming/erasing programs to the on-chip RAM by commands from the host. Sum checks and blank checks are executed by sending these commands from the host.

These boot program states are shown in figure 17.21.

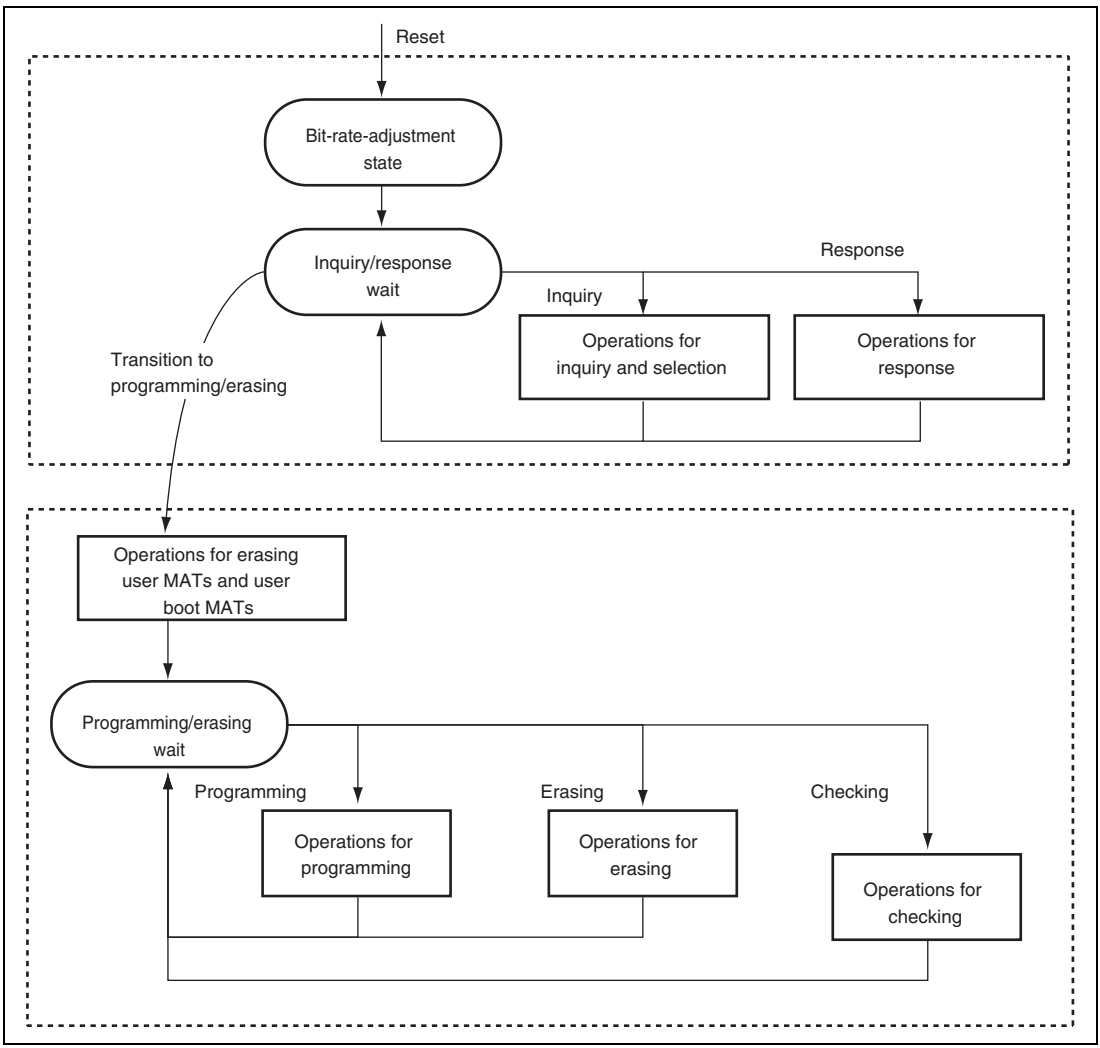


Figure 17.21 Boot Program States

(1) Bit-Rate-Adjustment State

The bit rate is calculated by measuring the period of transfer of a low-level byte (H'00) from the host. The bit rate can be changed by the command for a new bit rate selection. After the bit rate has been adjusted, the boot program enters the inquiry and selection state. The bit-rate-adjustment sequence is shown in figure 17.22.

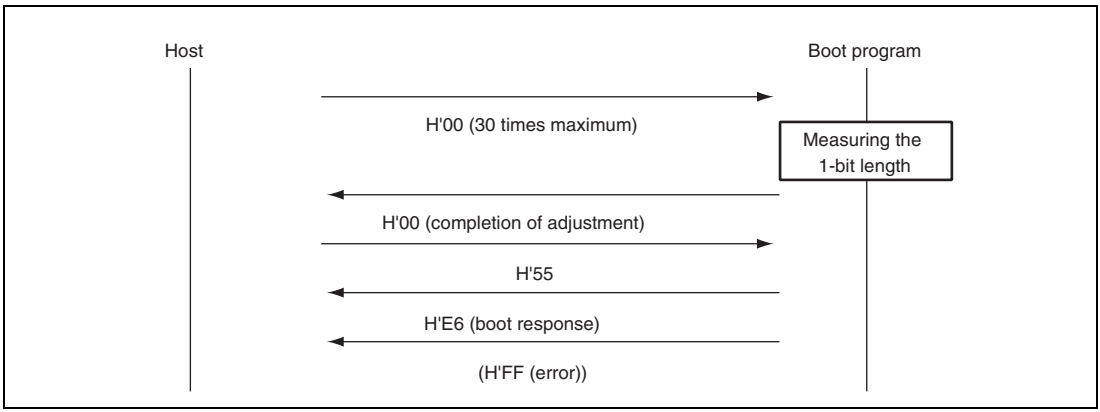


Figure 17.22 Bit-Rate-Adjustment Sequence

(2) Communications Protocol

After adjustment of the bit rate, the protocol for serial communications between the host and the boot program is as shown below.

1. One-byte commands and one-byte responses

   These one-byte commands and one-byte responses consist of the inquiries and the ACK for successful completion.
2. n-byte commands or n-byte responses

   These commands and responses are comprised of n bytes of data. These are selections and responses to inquiries.

   The program data size is not included under this heading because it is determined in another command.
3. Error response

   The error response is a response to inquiries. It consists of an error response and an error code and comes two bytes.
4. Programming of 128 bytes

   The size is not specified in commands. The size of n is indicated in response to the programming unit inquiry.

5. Memory read response

   This response consists of four bytes of data.

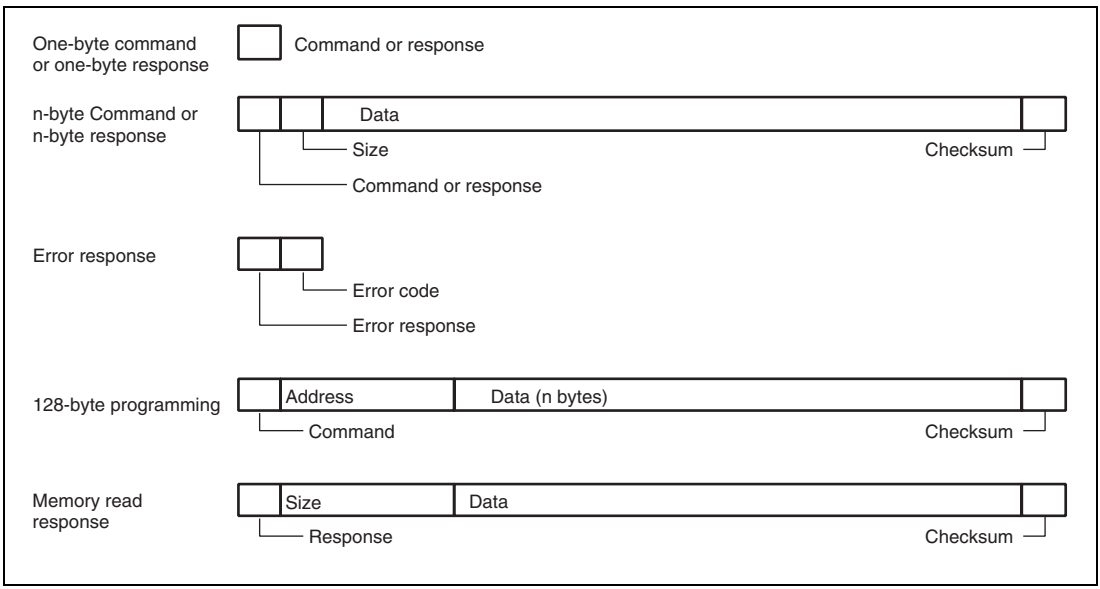


Figure 17.23 Communication Protocol Format

- Command (one byte): Commands including inquiries, selection, programming, erasing, and checking
- Response (one byte): Response to an inquiry
- Size (one byte): The amount of data for transmission excluding the command, amount of data, and checksum
- Checksum (one byte): The checksum is calculated so that the total of all values from the command byte to the SUM byte becomes H'00.
- Data (n bytes): Detailed data of a command or response
- Error response (one byte): Error response to a command
- Error code (one byte): Type of the error
- Address (four bytes): Address for programming
- Data (n bytes): Data to be programmed (the size is indicated in the response to the programming unit inquiry.)
- Size (four bytes): Four-byte response to a memory read

(3) Inquiry and Selection States

The boot program returns information from the flash memory in response to the host's inquiry commands and sets the device code, clock mode, and bit rate in response to the host's selection command.

Table 17.15 lists the inquiry and selection commands.

Table 17.15 Inquiry and Selection Commands

Command	Command Name	Description
H'20	Supported device inquiry	Inquiry regarding device codes
H'10	Device selection	Selection of device code
H'21	Clock mode inquiry	Inquiry regarding numbers of clock modes and values of each mode
H'11	Clock mode selection	Indication of the selected clock mode
H'22	Multiplication ratio inquiry	Inquiry regarding the number of frequency-multiplied clock types, the number of multiplication ratios, and the values of each multiple
H'23	Operating clock frequency inquiry	Inquiry regarding the maximum and minimum values of the main clock and peripheral clocks
H'24	User boot MAT information inquiry	Inquiry regarding the number of user boot MATs and the start and last addresses of each MAT
H'25	User MAT information inquiry	Inquiry regarding the a number of user MATs and the start and last addresses of each MAT
H'26	Block for erasing information Inquiry	Inquiry regarding the number of blocks and the start and last addresses of each block
H'27	Programming unit inquiry	Inquiry regarding the unit of program data
H'3F	New bit rate selection	Selection of new bit rate
H'40	Transition to programming/erasing state	Erasing of user MAT and user boot MAT, and entry to programming/erasing state
H'4F	Boot program status inquiry	Inquiry into the operated status of the boot program

The selection commands, which are device selection (H'10), clock mode selection (H'11), and new bit rate selection (H'3F), should be sent from the host in that order. When two or more selection commands are sent at once, the last command will be valid.

All of these commands, except for the boot program status inquiry command (H'4F), will be valid until the boot program receives the programming/erasing transition (H'40). The host can choose the needed commands and make inquiries while the above commands are being transmitted. H'4F is valid even after the boot program has received H'40.

(a) Supported Device Inquiry

The boot program will return the device codes of supported devices and the product code in response to the supported device inquiry.

Command

H'20

- Command, H'20, (one byte): Inquiry regarding supported devices

Response

H'30	Size	Number of devices	
Number of characters	Device code		Product name
...			
SUM			

- Response, H'30, (one byte): Response to the supported device inquiry
- Size (one byte): Number of bytes to be transmitted, excluding the command, size, and checksum, that is, the amount of data contributes by the number of devices, characters, device codes and product names
- Number of devices (one byte): The number of device types supported by the boot program
- Number of characters (one byte): The number of characters in the device codes and boot program's name
- Device code (four bytes): ASCII code of the supporting product
- Product name (n bytes): Type name of the boot program in ASCII-coded characters
- SUM (one byte): Checksum

  The checksum is calculated so that the total number of all values from the command byte to the SUM byte becomes H'00.

(b) Device Selection

The boot program will set the supported device to the specified device code. The program will return the selected device code in response to the inquiry after this setting has been made.

Command

H'10	Size	Device code	SUM

- Command, H'10, (one byte): Device selection
- Size (one byte): Amount of device-code data
  This is fixed at 4.
- Device code (four bytes): Device code (ASCII code) returned in response to the supported device inquiry
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to the device selection command
  ACK will be returned when the device code matches.

Error response

H'90	ERROR

- Error response, H'90, (one byte): Error response to the device selection command

  ERROR : (one byte): Error code
  H'11: Sum check error
  H'21: Device code error, that is, the device code does not match

(c) Clock Mode Inquiry

The boot program will return the supported clock modes in response to the clock mode inquiry.

Command

H'21

- Command, H'21, (one byte): Inquiry regarding clock mode

Response

H'31	Size	Number of modes	Mode	···	SUM

- Response, H'31, (one byte): Response to the clock-mode inquiry
- Size (one byte): Amount of data that represents the number of modes and modes
- Number of clock modes (one byte): The number of supported clock modes
  H'00 indicates no clock mode or the device allows to read the clock mode.
- Mode (one byte): Values of the supported clock modes (i.e. H'01 means clock mode 1.)
- SUM (one byte): Checksum

(d) Clock Mode Selection

The boot program will set the specified clock mode. The program will return the selected clock-mode information after this setting has been made.

The clock-mode selection command should be sent after the device-selection commands.

Command	H'11	Size	Mode	SUM

- Command, H'11, (one byte): Selection of clock mode
- Size (one byte): Amount of data that represents the modes
- Mode (one byte): A clock mode returned in reply to the supported clock mode inquiry.
- SUM (one byte): Checksum

Response	H'06

- Response, H'06, (one byte): Response to the clock mode selection command
  ACK will be returned when the clock mode matches.

Error Response	H'91	ERROR

- Error response, H'91, (one byte): Error response to the clock mode selection command
- ERROR : (one byte): Error code
  H'11: Checksum error
  H'22: Clock mode error, that is, the clock mode does not match.

Even if the clock mode numbers are H'00 and H'01 by a clock mode inquiry, the clock mode must be selected using these respective values.

(e) Multiplication Ratio Inquiry

The boot program will return the supported multiplication and division ratios.

Command

H'22

- Command, H'22, (one byte): Inquiry regarding multiplication ratio

Response

H'32	Size	Number of types					
Number of multiplication ratios	Multiplica-tion ratio	···					
···							
SUM							

- Response, H'32, (one byte): Response to the multiplication ratio inquiry
- Size (one byte): The amount of data that represents the number of clock sources and multiplication ratios and the multiplication ratios
- Number of types (one byte): The number of supported multiplied clock types (e.g. when there are two multiplied clock types, which are the main and peripheral clocks, the number of types will be H'02.)
- Number of multiplication ratios (one byte): The number of multiplication ratios for each type (e.g. the number of multiplication ratios to which the main clock can be set and the peripheral clock can be set.)
- Multiplication ratio (one byte)
- Multiplication ratio: The value of the multiplication ratio (e.g. when the clock-frequency multiplier is four, the value of multiplication ratio will be H'04.)

  Division ratio: The inverse of the division ratio, i.e. a negative number (e.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)

  The number of multiplication ratios returned is the same as the number of multiplication ratios and as many groups of data are returned as there are types.
- SUM (one byte): Checksum

(f) Operating Clock Frequency Inquiry

The boot program will return the number of operating clock frequencies, and the maximum and minimum values.

Command

H'23

- Command, H'23, (one byte): Inquiry regarding operating clock frequencies

Response

H'33	Size	Number of operating clock frequencies
Minimum value of operating clock frequency		Maximum value of operating clock frequency
...		
SUM		

- Response, H'33, (one byte): Response to operating clock frequency inquiry
- Size (one byte): The number of bytes that represents the minimum values, maximum values, and the number of frequencies.
- Number of operating clock frequencies (one byte): The number of supported operating clock frequency types
  (e.g. when there are two operating clock frequency types, which are the main and peripheral clocks, the number of types will be H'02.)
- Minimum value of operating clock frequency (two bytes): The minimum value of the multiplied or divided clock frequency.

  The minimum and maximum values of the operating clock frequency represent the values in MHz, valid to the hundredths place of MHz, and multiplied by 100. (e.g. when the value is 17.00 MHz, it will be 2000, which is H'07D0.)
- Maximum value (two bytes): Maximum value among the multiplied or divided clock frequencies.

  There are as many pairs of minimum and maximum values as there are operating clock frequencies.
- SUM (one byte): Checksum

(g) User Boot MAT Information Inquiry

The boot program will return the number of user boot MATs and their addresses.

Command

H'24

- Command, H'24, (one byte): Inquiry regarding user boot MAT information

Response

H'34	Size	Number of areas	
Area-start address			Area-last address
...			
SUM			

- Response, H'34, (one byte): Response to user boot MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start addresses, and area-last address
- Number of Areas (one byte): The number of consecutive user boot MAT areas
  When user boot MAT areas are consecutive, the number of areas returned is H'01.
- Area-start address (four byte): Start address of the area
- Area-last address (four byte): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(h) User MAT Information Inquiry

The boot program will return the number of user MATs and their addresses.

Command

H'25

- Command, H'25, (one byte): Inquiry regarding user MAT information

Response

H'35	Size	Number of areas	
Start address area			Last address area
...			
SUM			

- Response, H'35, (one byte): Response to the user MAT information inquiry
- Size (one byte): The number of bytes that represents the number of areas, area-start address and area-last address
- Number of areas (one byte): The number of consecutive user MAT areas
  When the user MAT areas are consecutive, the number of areas is H'01.
- Area-start address (four bytes): Start address of the area

- Area-last address (four bytes): Last address of the area
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(i) Erased Block Information Inquiry

The boot program will return the number of erased blocks and their addresses.

Command

H'26

- Command, H'26, (two bytes): Inquiry regarding erased block information

Response

H'36	Size	Number of blocks

Block start address	Block last address
...	

SUM

- Response, H'36, (one byte): Response to the number of erased blocks and addresses
- Size (three bytes): The number of bytes that represents the number of blocks, block-start addresses, and block-last addresses.
- Number of blocks (one byte): The number of erased blocks
- Block start address (four bytes): Start address of a block
- Block last Address (four bytes): Last address of a block
  There are as many groups of data representing the start and last addresses as there are areas.
- SUM (one byte): Checksum

(j) Programming Unit Inquiry

The boot program will return the programming unit used to program data.

Command

H'27

- Command, H'27, (one byte): Inquiry regarding programming unit

Response

H'37	Size	Programming unit	SUM

- Response, H'37, (one byte): Response to programming unit inquiry
- Size (one byte): The number of bytes that indicate the programming unit, which is fixed to 2
- Programming unit (two bytes): A unit for programming
  This is the unit for reception of programming.
- SUM (one byte): Checksum

(k) New Bit-Rate Selection

The boot program will set a new bit rate and return the new bit rate.

This selection should be sent after sending the clock mode selection command.

Command

H'3F	Size	Bit rate	Input frequency
Number of multiplication ratios	Multiplication ratio 1	Multiplication ratio 2	
SUM			

- Command, H'3F, (one byte): Selection of new bit rate
- Size (one byte): The number of bytes that represents the bit rate, input frequency, number of multiplication ratios, and multiplication ratio
- Bit rate (two bytes): New bit rate
  One hundredth of the value (e.g. when the value is 19200 bps, it will be 192, which is H'00C0.)
- Input frequency (two bytes): Frequency of the clock input to the boot program
  This is valid to the hundredths place and represents the value in MHz multiplied by 100. (E.g. when the value is 20.00 MHz, it will be 2000, which is H'07D0.)
- Number of multiplication ratios (one byte): The number of multiplication ratios to which the device can be set.
- Multiplication ratio 1 (one byte): The value of multiplication or division ratios for the main operating frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  Division ratio: The inverse of the division ratio, as a negative number (e.g. when the clock frequency is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- Multiplication ratio 2 (one byte): The value of multiplication or division ratios for the peripheral frequency
  Multiplication ratio (one byte): The value of the multiplication ratio (e.g. when the clock frequency is multiplied by four, the multiplication ratio will be H'04.)
  (Division ratio: The inverse of the division ratio, as a negative number (E.g. when the clock is divided by two, the value of division ratio will be H'FE. H'FE = D'-2)
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to selection of a new bit rate
  When it is possible to set the bit rate, the response will be ACK.

Error Response	H'BF	ERROR

- Error response, H'BF, (one byte): Error response to selection of new bit rate
- ERROR: (one byte): Error code

H'11:	Sum checking error
H'24:	Bit-rate selection error The rate is not available.
H'25:	Error in input frequency This input frequency is not within the specified range.
H'26:	Multiplication-ratio error The ratio does not match an available ratio.
H'27:	Operating frequency error The frequency is not within the specified range.

(4) Receive Data Check

The methods for checking of receive data are listed below.

1. Input frequency

   The received value of the input frequency is checked to ensure that it is within the range of minimum to maximum frequencies which matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.
2. Multiplication ratio

   The received value of the multiplication ratio or division ratio is checked to ensure that it matches the clock modes of the specified device. When the value is out of this range, an input-frequency error is generated.
3. Operating frequency error

   Operating frequency is calculated from the received value of the input frequency and the multiplication or division ratio. The input frequency is input to the LSI and the LSI is operated at the operating frequency. The expression is given below.

   Operating frequency = Input frequency × Multiplication ratio, or
   Operating frequency = Input frequency ÷ Division ratio

   The calculated operating frequency should be checked to ensure that it is within the range of minimum to maximum frequencies which are available with the clock modes of the specified device. When it is out of this range, an operating frequency error is generated.

4. Bit rate

   To facilitate error checking, the value (n) of clock select (CKS) in the serial mode register (SMR), and the value (N) in the bit rate register (BRR), which are found from the peripheral operating clock frequency (ϕ) and bit rate (B), are used to calculate the error rate to ensure that it is less than 4%. If the error is more than 4%, a bit rate error is generated. The error is calculated using the following expression:

$$\text{Error (\%)} = \{[\frac{\phi \times 10^6}{(N + 1) \times B \times 64 \times 2^{(2\times n - 1)}}] - 1\} \times 100$$

When the new bit rate is selectable, the rate will be set in the register after sending ACK in response. The host will send an ACK with the new bit rate for confirmation and the boot program will response with that rate.

Confirmation | H'06 |

- Confirmation, H'06, (one byte): Confirmation of a new bit rate

Response | H'06 |

- Response, H'06, (one byte): Response to confirmation of a new bit rate

The sequence of new bit-rate selection is shown in figure 17.24.

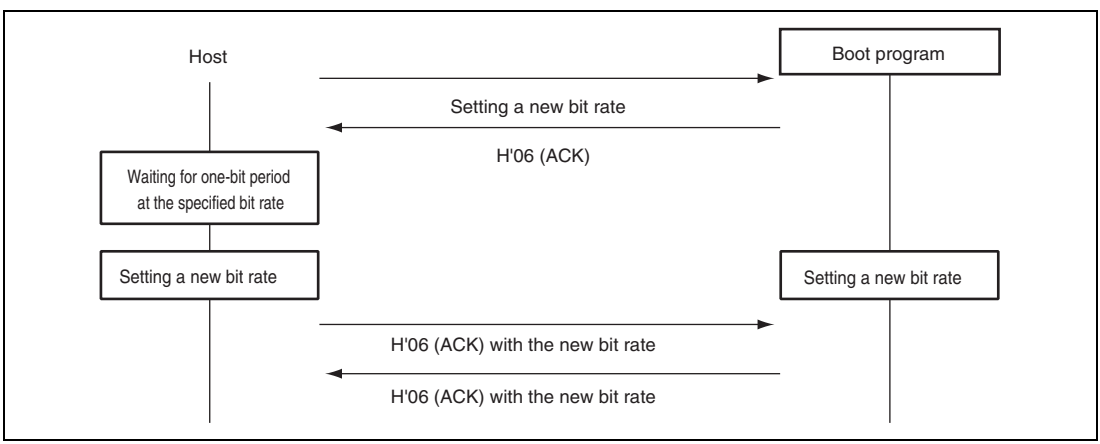


Figure 17.24 New Bit-Rate Selection Sequence

(5) Transition to Programming/Erasing State

The boot program will transfer the erasing program, and erase the user MATs and user boot MATs in that order. On completion of this erasure, ACK will be returned and will enter the programming/erasing state.

The host should select the device code, clock mode, and new bit rate with device selection, clock-mode selection, and new bit-rate selection commands, and then send the command for the transition to programming/erasing state. These procedures should be carried out before sending of the programming selection command or program data.

Command

H'40

- Command, H'40, (one byte): Transition to programming/erasing state

Response

H'06

- Response, H'06, (one byte): Response to transition to programming/erasing state
  The boot program will send ACK when the user MAT and user boot MAT have been erased by the transferred erasing program.

Error Response

H'C0	H'51

- Error response, H'C0, (one byte): Error response for user boot MAT blank check
- Error code, H'51, (one byte): Erasing error
  An error occurred and erasure was not completed.

(6) Command Error

A command error will occur when a command is undefined, the order of commands is incorrect, or a command is unacceptable. Issuing a clock-mode selection command before a device selection or an inquiry command after the transition to programming/erasing state command, are examples.

Error Response

H'80	H'xx

- Error response, H'80, (one byte): Command error
- Command, H'xx, (one byte): Received command

(7) Command Order

The order for commands in the inquiry selection state is shown below.

1. A supported device inquiry (H'20) should be made to inquire about the supported devices.
2. The device should be selected from among those described by the returned information and set with a device-selection (H'10) command.
3. A clock-mode inquiry (H'21) should be made to inquire about the supported clock modes.
4. The clock mode should be selected from among those described by the returned information and set.
5. After selection of the device and clock mode, inquiries for other required information should be made, such as the multiplication-ratio inquiry (H'22) or operating frequency inquiry (H'23), which are needed for a new bit-rate selection.
6. A new bit rate should be selected with the new bit-rate selection (H'3F) command, according to the returned information on multiplication ratios and operating frequencies.
7. After selection of the device and clock mode, the information of the user boot MAT and user MAT should be made to inquire about the user boot MATs information inquiry (H'24), user MATs information inquiry (H'25), erased block information inquiry (H'26), and programming unit inquiry (H'27).
8. After making inquiries and selecting a new bit rate, issue the transition to programming/erasing state command (H'40). The boot program will then enter the programming/erasing state.

(8) Programming/Erasing State

A programming selection command makes the boot program select the programming method, a 128-byte programming command makes it program the memory with data, and an erasing selection command and block erasing command make it erase the block. Table 17.16 lists the programming/erasing commands.

Table 17.16 Programming/Erasing Commands

Command	Command Name	Description
H'42	User boot MAT programming selection	Transfers the user boot MAT programming program
H'43	User MAT programming selection	Transfers the user MAT programming program
H'50	128-byte programming	Programs 128 bytes of data
H'48	Erasing selection	Transfers the erasing program
H'58	Block erasing	Erases a block of data
H'52	Memory read	Reads the contents of memory
H'4A	User boot MAT sum check	Checks the checksum of the user boot MAT
H'4B	User MAT sum check	Checks the checksum of the user MAT
H'4C	User boot MAT blank check	Checks the blank data of the user boot MAT
H'4D	User MAT blank check	Checks the blank data of the user MAT
H'4F	Boot program status inquiry	Inquires into the boot program's status

- Programming

  Programming is executed by the programming selection and 128-byte programming commands.

  Firstly, the host should send the programming selection command and select the programming method and programming MATs. There are two programming selection commands, and selection is according to the area and method for programming.

  1. User boot MAT programming selection
  2. User MAT programming selection

  After issuing the programming selection command, the host should send the 128-byte programming command. The 128-byte programming command that follows the selection command represents the data programmed according to the method specified by the selection command. When more than 128-byte data is programmed, 128-byte commands should repeatedly be executed. Sending a 128-byte programming command with H'FFFFFFFF as the address will stop the programming. On completion of programming, the boot program will wait for selection of programming or erasing.

  Where the sequence of programming operations that is executed includes programming with another method or of another MAT, the procedure must be repeated from the programming selection command.

  The sequence for the programming selection and 128-byte programming commands is shown in figure 17.25.

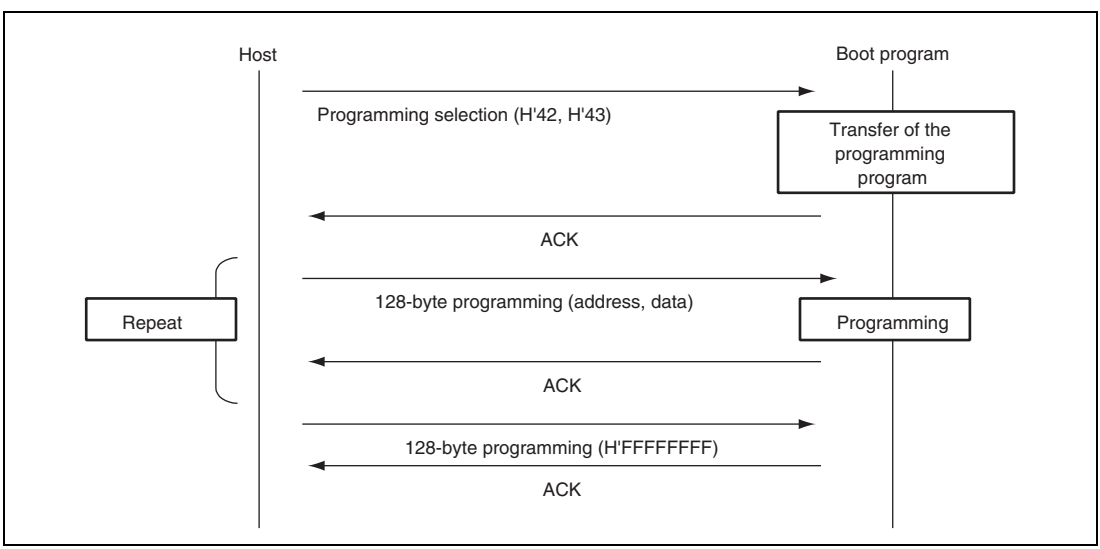


Figure 17.25 Programming Sequence

- Erasure

  Erasure is executed by the erasure selection and block erasure commands.

  Firstly, erasure is selected by the erasure selection command and the boot program then erases the specified block. The command should be repeatedly executed if two or more blocks are to be erased. Sending a block erasure command from the host with the block number H'FF will stop the erasure operating. On completion of erasing, the boot program will wait for selection of programming or erasing.

  The sequence for the erasure selection and block erasure commands is shown in figure 17.26.

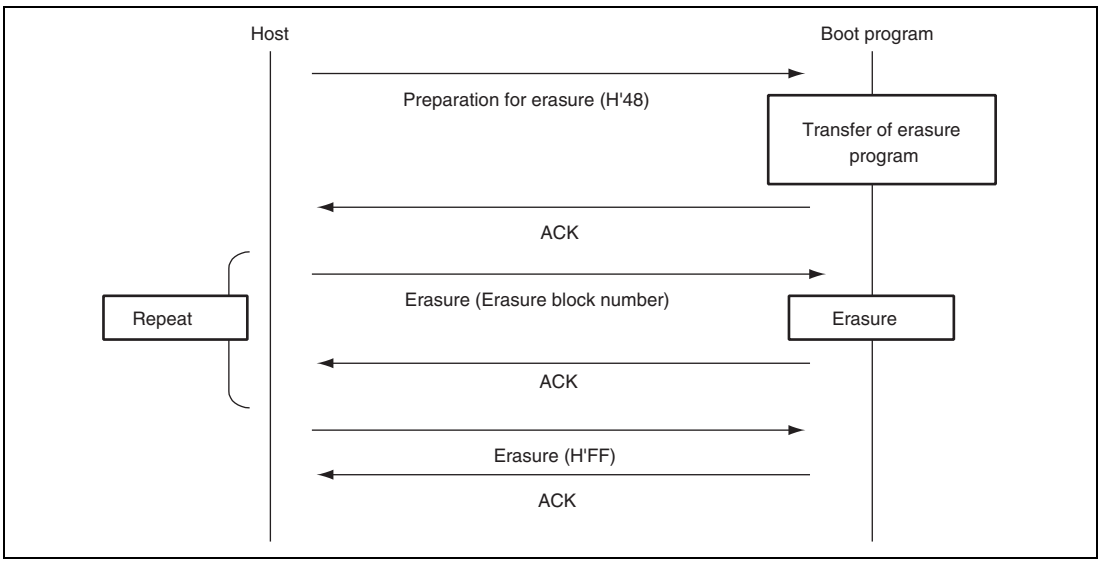


Figure 17.26 Erasure Sequence

(a) User Boot MAT Programming Selection

The boot program will transfer a programming program. The data is programmed to the user boot MATs by the transferred programming program.

Command

H'42

- Command, H'42, (one byte): User boot-program programming selection

Response

H'06

- Response, H'06, (one byte): Response to user boot-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response

H'C2	ERROR

- Error response : H'C2 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code
  H'54: Selection processing error (transfer error occurs and processing is not completed)

(b) User MAT Programming Selection

The boot program will transfer a program for user MAT programming selection. The data is programmed to the user MATs by the transferred program for programming.

Command

H'43

- Command, H'43, (one byte): User-program programming selection

Response

H'06

- Response, H'06, (one byte): Response to user-program programming selection
  When the programming program has been transferred, the boot program will return ACK.

Error Response

H'C3	ERROR

- Error response : H'C3 (1 byte): Error response to user boot MAT programming selection
- ERROR : (1 byte): Error code
  H'54: Selection processing error (transfer error occurs and processing is not completed)

(c) 128-Byte Programming

The boot program will use the programming program transferred by the programming selection to program the user boot MATs or user MATs in response to 128-byte programming.

Command

H'50	Address						
Data	···						
···							
SUM							

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): Start address for programming
  Multiple of the size specified in response to the programming unit inquiry
  (i.e. H'00, H'01, H'00, H'00 : H'00010000)
- Program data (128 bytes): Data to be programmed
  The size is specified in the response to the programming unit inquiry.
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response

H'D0	ERROR

- Error response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

  H'11: Checksum Error

  H'2A: Address error
  The address is not in the MAT.

  H'53: Programming error
  A programming error has occurred and programming cannot be continued.

The specified address should match the unit for programming of data. For example, when the programming is in 128-byte units, the lower eight bits of the address should be H'00 or H'80. When there are less than 128 bytes of data to be programmed, the host should fill the rest with H'FF.

Sending the 128-byte programming command with the address of H'FFFFFFFF will stop the programming operation. The boot program will interpret this as the end of the programming and wait for selection of programming or erasing.

Command

H'50	Address	SUM

- Command, H'50, (one byte): 128-byte programming
- Programming Address (four bytes): End code is H'FF, H'FF, H'FF, H'FF.
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to 128-byte programming
  On completion of programming, the boot program will return ACK.

Error Response

H'D0	ERROR

- Error Response, H'D0, (one byte): Error response for 128-byte programming
- ERROR: (one byte): Error code

  H'11: Checksum error

  H'53: Programming error

  An error has occurred in programming and programming cannot be continued.

(d) Erasure Selection

The boot program will transfer the erasure program. User MAT data is erased by the transferred erasure program.

Command

H'48

- Command, H'48, (one byte): Erasure selection

Response

H'06

- Response, H'06, (one byte): Response for erasure selection
  After the erasure program has been transferred, the boot program will return ACK.

Error Response

H'C8	ERROR

- Error Response, H'C8, (one byte): Error response to erasure selection
- ERROR: (one byte): Error code

  H'54: Selection processing error (transfer error occurs and processing is not completed)

(e) Block Erasure

The boot program will erase the contents of the specified block.

Command

H'58	Size	Block number	SUM

- Command, H'58, (one byte): Erasure
- Size (one byte): The number of bytes that represents the erase block number
  This is fixed to 1.
- Block number (one byte): Number of the block to be erased
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to Erasure
  After erasure has been completed, the boot program will return ACK.

Error Response

H'D8	ERROR

- Error Response, H'D8, (one byte): Response to Erasure
- ERROR (one byte): Error code

H'11:	Sum check error
H'29:	Block number error Block number is incorrect.
H'51:	Erasure error An error has occurred during erasure.

On receiving block number H'FF, the boot program will stop erasure and wait for a selection command.

Command

H'58	Size	Block number	SUM

- Command, H'58, (one byte): Erasure
- Size, (one byte): The number of bytes that represents the block number
  This is fixed to 1.
- Block number (one byte): H'FF
  Stop code for erasure
- SUM (one byte): Checksum

Response

H'06

- Response, H'06, (one byte): Response to end of erasure (ACK)
  When erasure is to be performed after the block number H'FF has been sent, the procedure should be executed from the erasure selection command.

(f) Memory Read

The boot program will return the data in the specified address.

Command

H'52	Size	Area	Read address
Read size			SUM

- Command: H'52 (1 byte): Memory read
- Size (1 byte): Amount of data that represents the area, read address, and read size (fixed at 9)
- Area (1 byte)

  H'00: User boot MAT
  H'01: User MAT
  An address error occurs when the area setting is incorrect.
- Read address (4 bytes): Start address to be read from
- Read size (4 bytes): Size of data to be read
- SUM (1 byte): Checksum

Response

H'52	Read size						
Data	···						
SUM							

- Response: H'52 (1 byte): Response to memory read
- Read size (4 bytes): Size of data to be read
- Data (n bytes): Data for the read size from the read address
- SUM (1 byte): Checksum

Error Response

H'D2	ERROR

- Error response: H'D2 (1 byte): Error response to memory read
- ERROR: (1 byte): Error code

  H'11: Sum check error
  H'2A: Address error
  The read address is not in the MAT.
  H'2B: Size error
  The read size exceeds the MAT.

(g) User-Boot Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user-boot program, as a four-byte value.

Command	H'4A

- Command, H'4A, (one byte): Sum check for user-boot program

Response	H'5A	Size	Checksum of user boot program	SUM

- Response, H'5A, (one byte): Response to the sum check of user-boot program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user boot MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(h) User-Program Sum Check

The boot program will return the byte-by-byte total of the contents of the bytes of the user program.

Command	H'4B

- Command, H'4B, (one byte): Sum check for user program

Response	H'5B	Size	Checksum of user program	SUM

- Response, H'5B, (one byte): Response to the sum check of the user program
- Size (one byte): The number of bytes that represents the checksum
  This is fixed to 4.
- Checksum of user boot program (four bytes): Checksum of user MATs
  The total of the data is obtained in byte units.
- SUM (one byte): Sum check for data being transmitted

(i) User Boot MAT Blank Check

The boot program will check whether or not all user boot MATs are blank and return the result.

Command	H'4C

- Command, H'4C, (one byte): Blank check for user boot MAT

Response

H'06

- Response, H'06, (one byte): Response to the blank check of user boot MAT
  If all user MATs are blank (H'FF), the boot program will return ACK.

Error Response

H'CC	H'52

- Error Response, H'CC, (one byte): Response to blank check for user boot MAT
- Error Code, H'52, (one byte): Erasure has not been completed.

(j) User MAT Blank Check

The boot program will check whether or not all user MATs are blank and return the result.

Command

H'4D

- Command, H'4D, (one byte): Blank check for user MATs

Response

H'06

- Response, H'06, (one byte): Response to the blank check for user MATs
  If the contents of all user MATs are blank (H'FF), the boot program will return ACK.

Error Response

H'CD	H'52

- Error Response, H'CD, (one byte): Error response to the blank check of user MATs.
- Error code, H'52, (one byte): Erasure has not been completed.

(k) Boot Program State Inquiry

The boot program will return indications of its present state and error condition. This inquiry can be made in the inquiry/selection state or the programming/erasing state.

Command

H'4F

- Command, H'4F, (one byte): Inquiry regarding boot program's state

Response

H'5F	Size	Status	ERROR	SUM

- Response, H'5F, (one byte): Response to boot program state inquiry
- Size (one byte): The number of bytes. This is fixed to 2.
- Status (one byte): State of the boot program
- ERROR (one byte): Error status
  ERROR = 0 indicates normal operation.
  ERROR = 1 indicates error has occurred.
- SUM (one byte): Sum check

Table 17.17 Status Code

Code	Description
H'11	Device selection wait
H'12	Clock mode selection wait
H'13	Bit rate selection wait
H'1F	Programming/erasing state transition wait (bit rate selection is completed)
H'31	Programming state for erasure
H'3F	Programming/erasing selection wait (erasure is completed)
H'4F	Program data receive wait
H'5F	Erase block specification wait (erasure is completed)

Table 17.18 Error Code

Code	Description
H'00	No error
H'11	Sum check error
H'12	Program size error
H'21	Device code mismatch error
H'22	Clock mode mismatch error
H'24	Bit rate selection error
H'25	Input frequency error
H'26	Multiplication ratio error
H'27	Operating frequency error
H'29	Block number error
H'2A	Address error
H'2B	Data length error
H'51	Erasure error
H'52	Erasure incomplete error
H'53	Programming error
H'54	Selection processing error
H'80	Command error
H'FF	Bit-rate-adjustment confirmation error

17.14 Usage Notes

1. The initial state of the product at its shipment is in the erased state. For the product whose revision of erasing is undefined, we recommend to execute automatic erasure for checking the initial state (erased state) and compensating.
2. For the PROM programmer suitable for programmer mode in this LSI and its program version, refer to the instruction manual of the socket adapter.
3. If the socket, socket adapter, or product index does not match the specifications, too much current flows and the product may be damaged.
4. Use a PROM programmer that supports the device with 256-kbyte on-chip flash memory and 5.0-V programming voltage. Do not select HN28F101 and 3.3-V programming voltage with the programmer parameters. Use only the specified socket adapter.
5. Do not remove the chip from the PROM programmer nor input a reset signal during programming/erasing in which a high voltage is applied to the flash memory. Doing so may damage the flash memory permanently. If a reset is input accidentally, the reset must be released after the reset input period of at least 100 μs.
6. The flash memory is not accessible until FKEY is cleared after programming/erasing starts. If the operating mode is changed and this LSI is restarted by a reset immediately after programming/erasing has finished, secure the reset input period (period of $\overline{RES}$ = 0) of at least 100 μs. Transition to the reset state during programming/erasing is inhibited. If a reset is input accidentally, the reset must be released after the reset input period of at least 100 μs.
7. At powering on or off the Vcc power supply, fix the $\overline{RES}$ pin to low and set the flash memory to hardware protection state. This power on/off timing must also be satisfied at a power-off and power-on caused by a power failure and other factors.
8. In on-board programming mode or programmer mode, programming of the 128-byte programming-unit block must be performed only once. Perform programming in the state where the programming-unit block is fully erased.
9. When the chip is to be reprogrammed with the programmer after execution of programming or erasure in on-board programming mode, it is recommended that automatic programming is performed after execution of automatic erasure.
10. To program the flash memory, the program data and program must be placed at higher addresses than those of the external interrupt vector table, and H'FF must be written to all the system reserved areas in the exception handling vector table.

11. The programming program that includes the initialization routine and the erasing program that includes the initialization routine are each 4 kbytes or less. Accordingly, when the CPU clock frequency is 48 MHz, the download for each program takes approximately 35 μs at the maximum.
12. A programming/erasing program for the flash memory used in a conventional F-ZTAT H8, H8S microcomputer which does not support download of the on-chip program by setting the SCO bit in FCCS to 1 cannot run in this LSI. Be sure to download the on-chip program to execute programming/erasing of the flash memory in this F-ZTAT H8SX microcomputer.
13 Unlike a conventional F-ZTAT H8 or H8S microcomputers, measures against a program crash are not taken by WDT during programming/erasing. When needed, measures should be taken by user. A periodic interrupt generated by the WDT can be used as the measures, as an example. The interrupt generation cycle should take into consideration time to download a programming/erasing program and time to program/erase the flash memory.
14 When downloading the programming/erasing program, do not clear the SCO bit in FCCS to 0 after immediately setting it to 1. Otherwise, download cannot be performed normally. Immediately after executing the instruction to set the SCO bit to 1, dummy read of the FCCS must be executed twice.
15. The contents of some general registers are not saved in a programming/erasing program. When needed, save general registers in the procedure program.
16. The intervals for executing the user branch processing differ in programming and erasing. The processing phase also differs. Table 17.19 lists the maximum and minimum intervals for initiating the user branch processing when the CPU clock frequency is 40/48 MHz.

Table 17.19 Initiation Intervals of User Branch Processing

Processing Name	Maximum Interval	Minimum Interval
Programming	Approximately 1 ms	Approximately 19 μs
Erasing	Approximately 5 ms	Approximately 19 μs

17. While an instruction in on-chip RAM is being executed, the DMAC can write to the SCO bit in FCCS that is used for a download request or FMATS that is used for MAT switching. Make sure that these registers are not accidentally written to, otherwise an on-chip program may be downloaded and damage RAM or a MAT switchover may occur and the CPU get out of control.

18. States in which NMI Interrupt is Ignored

In the following modes or period, the NMI interrupt requests are ignored; they are not executed and the interrupt sources are not retained.

— Boot mode
— Programmer mode
— Checking the flash memory-related registers immediately after user boot mode is initiated (Approximately a period of A (s) if operation is done at an input clock frequency of B (Hz) after the reset signal is released)

$$A = \frac{1}{B \times 2} \times 1200$$

The following is an example when an input clock frequency is 5 MHz.

$$\frac{1}{5 \times 10^6 \times 2} \times 1200 = 120 \times 10^{-6} = 120\ \mu s$$

19. Notes on NMI Interrupt During Programming/Erasing

Do not execute interrupt processing during programming/erasing. Doing so causes the error protection state to be entered, aborts programming/erasing, and prevents flash memory from being successfully programmed/erased.

When executing an NMI interrupt during programming/erasing for error processing, the following should be noted.

— The NMI interrupt is an only interrupt that can be used during programming/erasing.
— When flash memory is being programmed or erased, both the user MAT and user boot MAT cannot be accessed. Prepare the interrupt vector table and interrupt processing routine in on-chip RAM by setting VBR. Make sure the flash memory being programmed or erased is not accessed by the interrupt processing routine. If flash memory is read, the read values are not guaranteed.
— Since generation of NMI requests (or other interrupt requests) during programming/erasing causes the error protection state to be entered, programming or erasing is aborted. For details on the error protection, see section 17.9.3, Error Protection.

20. Notes on Returning From the Error Protection State

Allow a reset period of 100 μs or longer before releasing the reset signal to return from the error protection state. A transition to the error protection state aborts programming or erasing, and programming or erasing is not performed properly. Note the following when returning from the error protection state to each mode.

— Boot mode

The boot program programmed in the LSI is initiated when the reset signal is released. All of the user MAT and user boot MAT are automatically erased before programming.

— User boot mode

The embedded check routine that is programmed in the LSI runs when the reset signal is released. Then, processing starts from the execution start address of the reset vector in the user boot MAT. First, erase all of the user MAT. Since data in the flash memory may be damaged, do not start with partial erasure or programming.

— User program mode

The program programmed in the user MAT is initiated when the reset signal is released. However, the program may not be initiated properly from the user MAT after releasing the reset signal. Erase all of the user MAT in boot mode etc. before programming. Then, set to user program mode before releasing the reset signal again.

Section 18 Clock Pulse Generator

This LSI has an on-chip clock pulse generator (CPG) that generates the system clock (Iϕ), peripheral module clock (Pϕ), and external clock (Bϕ).

The clock pulse generator consists of an oscillator, PLL (Phase Locked Loop) circuit, and divider. Figure 18.1 shows a block diagram of the clock pulse generator.

Clock frequencies can be changed by the PLL circuit and divider in the CPG. Changing the system clock control register (SCKCR) setting by software can change the clock frequencies.

This LSI supports three types of clocks: a system clock provided to the CPU and bus masters, a peripheral module clock provided to the peripheral modules, and an external clock provided to the external bus. These clocks can be specified independently. Note, however, that the frequencies of the peripheral clock and external clock are lower than that of the system clock.

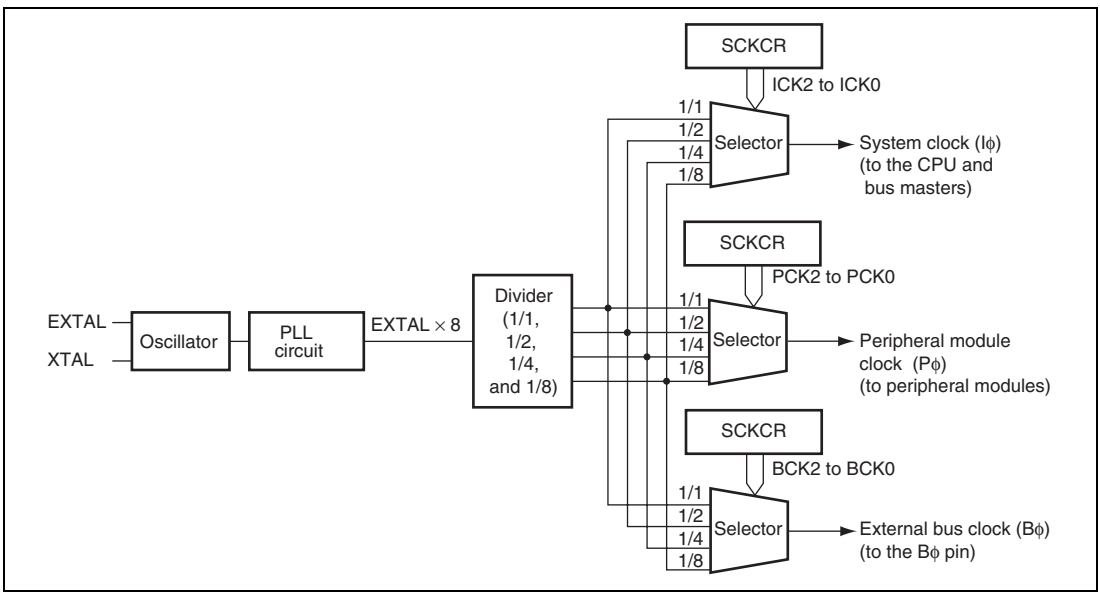


Figure 18.1 Block Diagram of Clock Pulse Generator

18.1 Register Description

The clock pulse generator has the following register.

- System clock control register (SCKCR)

18.1.1 System Clock Control Register (SCKCR)

SCKCR controls Bϕ clock output and frequencies of the system, peripheral module, and external clocks, and selects the Bϕ clock to be output.

Bit	15	14	13	12	11	10	9	8
Bit Name	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0
Initial Value	0	0	0	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0
Initial Value	0	0	1	0	0	0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	Bit Name	Initial Value	R/W	Description
15	PSTOP1	0	R/W	Bϕ Clock Output Enable Controls Bϕ output on PA7. • Normal operation 0: Bϕ output 1: Fixed high • Software standby mode X: Fixed high • Hardware standby mode X: Hi-Z
14	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
13	POSEL1	0	R/W	Bϕ Output Select 1 Controls the Bϕ output on PA7. 0: External clock (Bϕ) 1: Setting prohibited
12, 11	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
10	ICK2	0	R/W	System Clock (Iϕ) Select
9	ICK1	1	R/W	These bits select the frequency of the system clock provided to the CPU and DMAC. The ratio to the input clock is as follows:
8	ICK0	0	R/W	000: ×8 001: ×4 010: ×2 011: ×1 1xx: Setting prohibited The frequency of the peripheral module clock changes to the same frequency as the system clock if the frequency of the system clock is lower than that of the peripheral module clock.
7	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.

Bit	Bit Name	Initial Value	R/W	Description
6	PCK2	0	R/W	Peripheral Module Clock (Pϕ) Select
5	PCK1	1	R/W	These bits select the frequency of the peripheral module clock. The ratio to the input clock is as follows:
4	PCK0	0	R/W	000: ×8 001: ×4 010: ×2 011: ×1 1XX: Setting prohibited The frequency of the peripheral module clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the peripheral module clock higher than that of the system clock, the clocks will have the same frequency in reality.
3	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
2	BCK2	0	R/W	External clock (Bϕ) Select
1	BCK1	1	R/W	These bits select the frequency of the external clock. The ratio to the input clock is as follows:
0	BCK0	0	R/W	000: ×8 001: ×4 010: ×2 011: ×1 1xx: Setting prohibited The frequency of the external clock should be lower than that of the system clock. Though these bits can be set so as to make the frequency of the external clock higher than that of the system clock, the clocks will have the same frequency in reality.

[Legend]

X: Don't care

18.2 Oscillator

Clock pulses can be supplied by connecting a crystal resonator, or by input of an external clock.

18.2.1 Connecting Crystal Resonator

A crystal resonator can be connected as shown in the example in figure 18.2. Select the damping resistance R_d according to table 18.1. An AT-cut parallel-resonance type should be used.

When the clock is provided by connecting a crystal resonator, a crystal resonator having a frequency of 4 to 9 MHz should be connected.

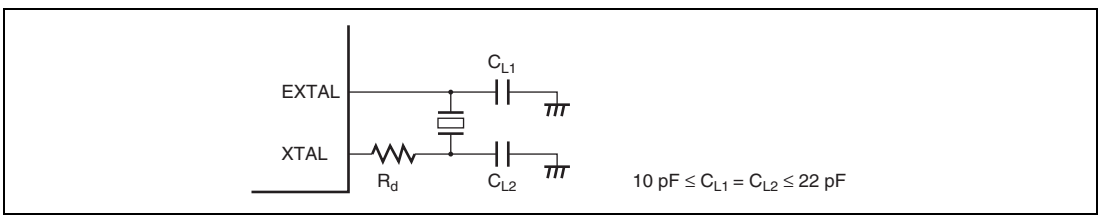


Figure 18.2 Connection of Crystal Resonator (Example)

Table 18.1 Damping Resistance Value

Frequency (MHz)	4	6	8	9
R_d (Ω)	500	300	200	100

Figure 18.3 shows an equivalent circuit of the crystal resonator. Use a crystal resonator that has the characteristics shown in table 18.2.

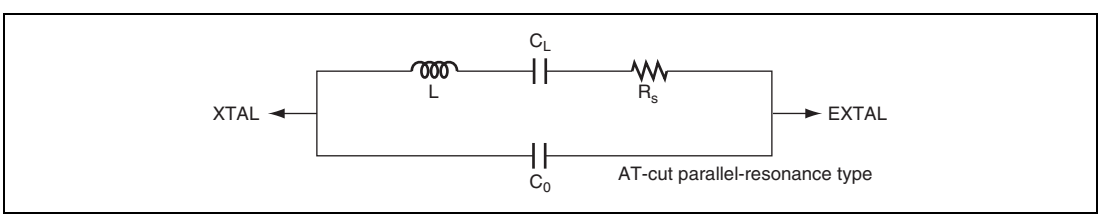


Figure 18.3 Crystal Resonator Equivalent Circuit

Table 18.2 Crystal Resonator Characteristics

Frequency (MHz)	4	6	8	9
R_S Max. (Ω)	120	100	80	80
C_0 Max. (pF)	7	7	7	7

18.2.2 External Clock Input

An external clock signal can be input as shown in the examples in figure 18.4. If the XTAL pin is left open, make sure that parasitic capacitance is no more than 10 pF. When the counter clock is input to the XTAL pin, make sure that the external clock is held high in standby mode.

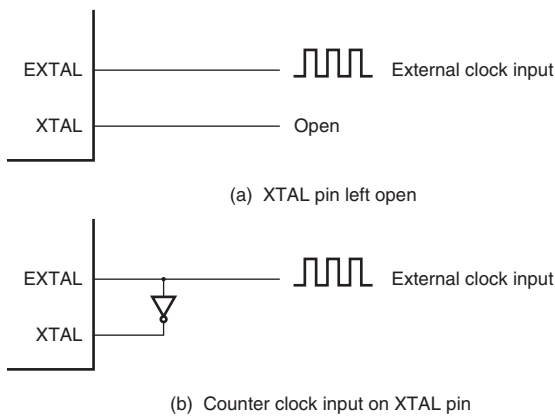


Figure 18.4 External Clock Input (Examples)

For the input conditions of the external clock, refer to table 21.4, Clock Timing, in section 21.3.1, Clock Timing. The input external clock should be from 4 to 9 MHz.

18.3 PLL Circuit

The PLL circuit has the function of multiplying the frequency of the clock from the oscillator by a factor of 8. The frequency multiplication factor is fixed.

18.4 Frequency Divider

The frequency divider divides the PLL clock to generate a 1/2, 1/4, or 1/8 clock. After bits ICK2 to ICK0 and PCK 2 to PCK0 are modified, this LSI operates at the modified frequency.

18.5 Usage Notes

18.5.1 Notes on Clock Pulse Generator

1. The following points should be noted since the frequency of ϕ (Iϕ: system clock and Pϕ: peripheral module clock) supplied to each module changes according to the setting of SCKCR.

   Select a clock division ratio that is within the operation guaranteed range of clock cycle time t_{cyc} shown in the AC timing of electrical characteristics as follows.

- When the SSU is in use

  Ensure the following settings:

  Iϕ (min.) = 8 MHz, Pϕ (min.) = 8 MHz, Iϕ (max.) = 48 MHz, and Pϕ (max.) = 24 MHz.

  The following settings are not permitted:
  Iϕ < 8 MHz, Iϕ > 48 MHz, Pϕ < 8 MHz, and Pϕ > 24 MHz.

- When the SSU is not in use

  Ensure the following settings:

  Iϕ (min.) = 8 MHz, Pϕ (min.) = 8 MHz, Iϕ (max.) = 48 MHz, and Pϕ (max.) = 35 MHz.

  The following settings are not permitted:
  Iϕ < 8 MHz, Iϕ > 48 MHz, Pϕ < 8 MHz, and Pϕ > 35 MHz.

2. All the on-chip peripheral modules (except for the DMAC) operate on the Pϕ. Therefore, note that the time processing of modules such as a timer and SCI differs before and after changing the clock division ratio.

   In addition, wait time for clearing software standby mode differs by changing the clock division ratio. For details, see section 19.7.3, Setting Oscillation Settling Time after Clearing Software Standby Mode.

3. The relationship between the system clock and peripheral module clock is Iϕ ≥ Pϕ. In addition, the system clock setting has priority. Accordingly, Pϕ may have the frequency set by bits ICK2 to ICK0 regardless of the settings of bits PCK2 to PCK0.

4. Figure 18.5 shows the clock modification timing. After a value is written to SCKCR, this LSI waits for the current bus cycle to complete. After the current bus cycle completes, each clock frequency will be modified within one cycle (worst case) of the external clock.

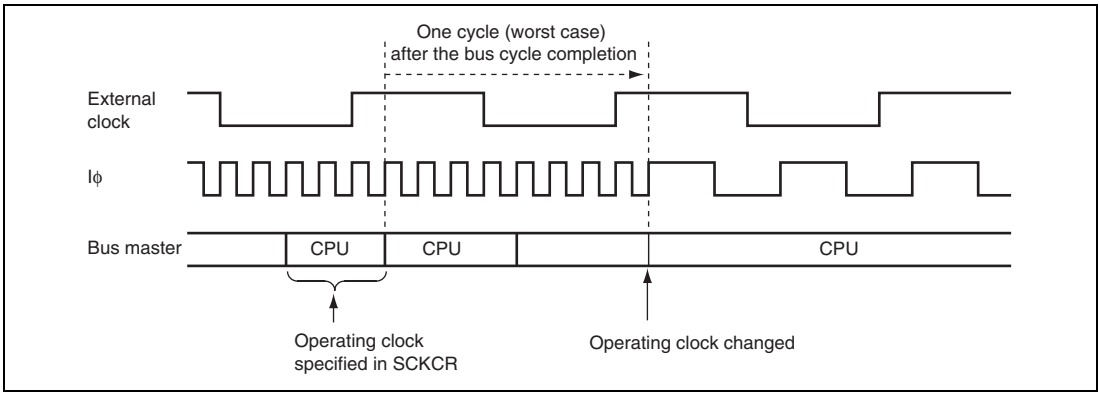


Figure 18.5 Clock Modification Timing

18.5.2 Notes on Resonator

Since various characteristics related to the resonator are closely linked to the user's board design, thorough evaluation is necessary on the user's part, using the resonator connection examples shown in this section as a reference. As the parameters for the resonator will depend on the floating capacitance of the resonator and the mounting circuit, the parameters should be determined in consultation with the resonator manufacturer. The design must ensure that a voltage exceeding the maximum rating is not applied to the resonator pin.

18.5.3 Notes on Board Design

When using the crystal resonator, place the crystal resonator and its load capacitors as close as possible to the XTAL and EXTAL pins. Other signal lines should be routed away from the oscillation circuit as shown in figure 18.6 to prevent induction from interfering with correct oscillation.

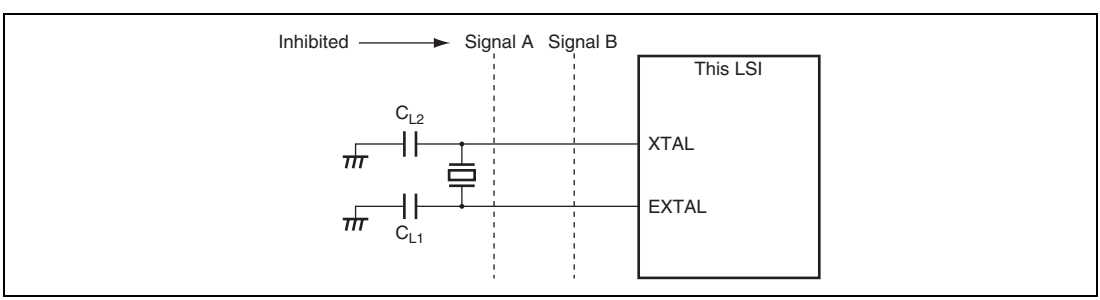
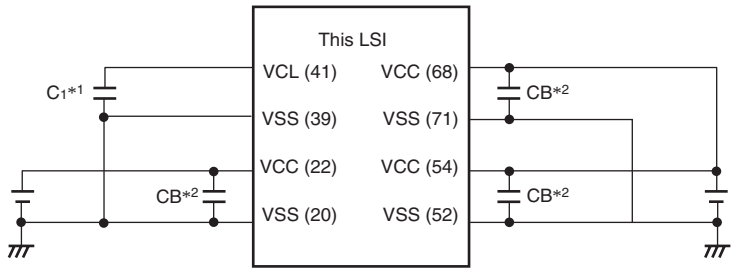


Figure 18.6 Note on Board Design for Oscillation Circuit

Figure 18.7 shows a connection example of bypass capacitor. Please be sure to insert bypass capacitor (CB) close to the Vcc and Vss pins and its capacitance meets the characteristics of the user system board.



Note: Numbers in parenthesis are pin numbers.
1. A 0.1-μF capacitor should be used here.
2. CB is a laminated ceramic capacitor.

Figure 18.7 Connection Example of Bypass Capacitor

18.5.4 Notes on Input Clock Frequency

The frequency of the input clock is multiplied in the PLL circuit by a factor of 8. To reduce noises, a lower frequency ranging of 4 to 9 MHz is recommended.

Section 19 Power-Down Modes

This LSI has power consumption reduction functions, such as multi-clock function, module stop function, and transition function to power-d own mode.

19.1 Features

- Multi-clock function

  The frequency division ratio is settable independently for the system clock, peripheral module clock, and external bus clock.
- Module stop function

  The functions for each peripheral modules can be stopped to make a transition to a power-down mode.
- Transition function to power-down mode

  Transition to a power-down mode is possible to stop the CPU, all the on-chip peripheral modules, and oscillator.
- Three power-down modes

  Sleep mode

  All-module-clock-stop mode

  Software standby mode

Table 19.1 shows conditions for making a transition to a power-down mode, states of the CPU and peripheral modules, and clearing method for each mode. After the reset state, since this LSI operates in normal program execution state, the modules, other than the DMAC are stopped.

Table 19.1 Operating States

Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode
Transition condition	Control register + instruction	Control register + instruction	Control register + instruction
Cancellation method	Interrupt	Interrupt*[2]	External interrupt
Oscillator	Functions	Functions	Halted
CPU	Halted (retained)	Halted (retained)	Halted (retained)
Watchdog timer	Functions	Functions	Halted (retained)
Other peripheral modules	Functions	Halted*[3]	Halted*[1]
I/O port	Functions	Retained	Retained

Notes: "Halted (retained)" in the table means that the internal register values are retained and internal operations are suspended.

1. SCI, RCAN-ET, and SSU enter the reset state, and other peripheral modules retain their states.
2. External interrupt and some internal interrupts (watchdog timer)
3. RCAN-ET and SSU enter the reset state, and other peripheral modules retain their states.

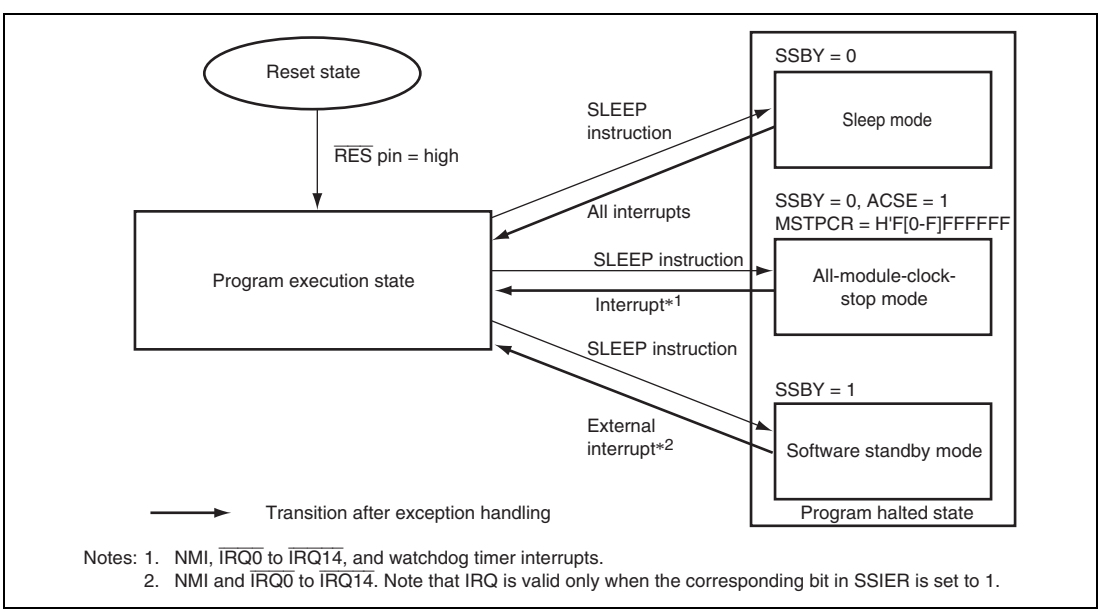


Figure 19.1 Mode Transitions

19.2 Register Descriptions

The registers related to the power-down modes are shown below. For details on the system clock control register (SCKCR), refer to section 18.1.1, System Clock Control Register (SCKCR).

- Standby control register (SBYCR)
- Module stop control register A (MSTPCRA)
- Module stop control register B (MSTPCRB)
- Module stop control register C (MSTPCRC)

19.2.1 Standby Control Register (SBYCR)

SBYCR controls software standby mode.

Bit	15	14	13	12	11	10	9	8
Bit Name	SSBY	—	—	STS4	STS3	STS2	STS1	STS0
Initial Value	0	1	0	0	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	—	—	—	—	—	—	—	—
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Description
15	SSBY	0	R/W	Software Standby Specifies the transition mode after executing the SLEEP instruction 0: Shifts to sleep mode after the SLEEP instruction is executed 1: Shifts to software standby mode after the SLEEP instruction is executed This bit does not change when clearing the software standby mode by using external interrupts and shifting to normal operation. For clearing, write 0 to this bit. When the WDT is used as the watchdog timer, the setting of this bit is disabled. In this case, a transition is always made to sleep mode or all-module-clock-stop mode after the SLEEP instruction is executed.

Bit	Bit Name	Initial Value	R/W	Description
14	—	1	R/W	Reserved This bit is always read as 1. The write value should always be 1.
13	—	0	R/W	Reserved This bit is always read as 0. The write value should always be 0.
12	STS4	0	R/W	Standby Timer Select 4 to 0
11	STS3	1	R/W	These bits select the time the MCU waits for the clock to settle when software standby mode is cleared by an external interrupt. With a crystal resonator, refer to table 19.2 and make a selection according to the operating frequency so that the standby time is at least equal to the oscillation settling time. With an external clock, a PLL circuit settling time is necessary. Refer to table 19.2 to set the standby time. While oscillation is being settled, the timer is counted on the Pϕ clock frequency. Careful consideration is required in multi-clock mode. 00000: Reserved 00001: Reserved 00010: Reserved 00011: Reserved 00100: Reserved 00101: Standby time = 64 states 00110: Standby time = 512 states 00111: Standby time = 1024 states 01000: Standby time = 2048 states 01001: Standby time = 4096 states 01010: Standby time = 16384 states 01011: Standby time = 32768 states 01100: Standby time = 65536 states 01101: Standby time = 131072 states 01110: Standby time = 262144 states 01111: Standby time = 524288 states 10000: Reserved 10001: Reserved 1001X: Reserved 101XX: Reserved 11XXX: Reserved
10	STS2	1	R/W	
9	STS1	1	R/W	
8	STS0	1	R/W	

Bit	Bit Name	Initial Value	R/W	Description
7 to 0	—	All 0	R/W	Reserved These bits are always read as 0. The write value should always be 0.

Note: The flash memory settling time must be reserved.

19.2.2 Module Stop Control Registers A and B (MSTPCRA and MSTPCRB)

MSTPCRA and MSTPCRB control module stop mode. Setting a bit to 1 makes the corresponding module enter module stop mode, while clearing the bit to 0 clears module stop mode.

- MSTPCRA

Bit	15	14	13	12	11	10	9	8
Bit Name	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8
Initial Value	0	0	0	0	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MSTPCRB

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

- MSTPCRA

Bit	Bit Name	Initial Value	R/W	Module
15	ACSE	0	R/W	All-Module-Clock-Stop Mode Enable Enables/disables all-module-clock-stop mode for reducing current consumption by stopping the bus controller and I/O ports operations when the CPU executes the SLEEP instruction after module stop mode has been set for all the on-chip peripheral modules controlled by MSTPCR. 0: All-module-clock-stop mode disabled 1: All-module-clock-stop mode enabled
14	MSTPA14	0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
13	MSTPA13	0	R/W	DMA controller (DMAC)
12	MSTPA12	0	R/W	Reserved These bits are always read as 0. The write value should always be 0.
11	MSTPA11	1	R/W	Reserved These bits are always read as 1. The write value should always be 1.
10	MSTPA10	1	R/W	
9	MSTPA9	1	R/W	
8	MSTPA8	1	R/W	
7	MSTPA7	1	R/W	
6	MSTPA6	1	R/W	
5	MSTPA5	1	R/W	
4	MSTPA4	1	R/W	A/D converter (unit 1)
3	MSTPA3	1	R/W	A/D converter (unit 0)
2	MSTPA2	1	R/W	Reserved These bits are always read as 1. The write value should always be 1.
1	MSTPA1	1	R/W	16-bit timer pulse unit (TPU channels 11 to 6)
0	MSTPA0	1	R/W	16-bit timer pulse unit (TPU channels 5 to 0)*

Note: * Supported only by the H8SX/1527R.

- MSTPCRB

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPB15	1	R/W	Programmable pulse generator (PPG)*
14	MSTPB14	1	R/W	Reserved
13	MSTPB13	1	R/W	These bits are always read as 1. The write value should always be 1.
12	MSTPB12	1	R/W	Serial communication interface_4 (SCI_4)
11	MSTPB11	1	R/W	Serial communication interface_3 (SCI_3)
10	MSTPB10	1	R/W	Reserved
9	MSTPB9	1	R/W	These bits are always read as 1. The write value should always be 1.
8	MSTPB	1	R/W	
7	MSTPB7	1	R/W	
6	MSTPB6	1	R/W	
5	MSTPB5	1	R/W	
4	MSTPB4	1	R/W	
3	MSTPB3	1	R/W	
2	MSTPB2	1	R/W	
1	MSTPB1	1	R/W	
0	MSTPB0	1	R/W	

Note: * Supported only by the H8SX/1527R.

19.2.3 Module Stop Control Register C (MSTPCRC)

When bits MSTPC1 and MSTPC0 are set to 1, the corresponding on-chip RAM stops. Do not set the corresponding MSTPC1 and MSTPC0 bits to 1 while accessing the on-chip RAM.

Bit	15	14	13	12	11	10	9	8
Bit Name	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8
Initial Value	1	1	1	1	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit	7	6	5	4	3	2	1	0
Bit Name	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0
Initial Value	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

Bit	Bit Name	Initial Value	R/W	Module
15	MSTPC15	1	R/W	Reserved
14	MSTPC14	1	R/W	These bits are always read as 1. The write value should always be 1.
13	MSTPC13	1	R/W	
12	MSTPC12	1	R/W	
11	MSTPC11	1	R/W	Controller area network (RCAN-ET)
10	MSTPC10	1	R/W	Synchronous serial communication unit 2 (SSU_2)
9	MSTPC9	1	R/W	Synchronous serial communication unit 1 (SSU_1)
8	MSTPC8	1	R/W	Synchronous serial communication unit 0 (SSU_0)
7	MSTPC7	0	R/W	Reserved
6	MSTPC6	0	R/W	These bits are always read as 0. The write value should always be 0.
5	MSTPC5	0	R/W	
4	MSTPC4	0	R/W	
3	MSTPC3	0	R/W	
2	MSTPC2	0	R/W	
1	MSTPC1	0	R/W	On-chip RAM (H'FFF9000 to H'FFFBFFF)
0	MSTPC0	0	R/W	The write value to MSTPC1 and MSTPC0 should always be the same.

19.3 Multi-Clock Function

When bits ICK2 to ICK0, PCK2 to PCK0, and BCK2 to BCK0 in SCKCR are set, a transition is made to multi-clock mode at the end of the bus cycle. In multi-clock mode, the CPU and bus masters operate on the operating clock specified by bits ICK2 to ICK0. The peripheral modules operate on the operating clock specified by bits PCK2 to PCK0. The external bus operates on the operating clock specified by bits BCK2 to BCK0.

Even if the frequencies specified by bits PCK2 to PCK0 and BCK2 to BCK0 are higher than the frequency specified by bits ICK2 to ICK0, the specified values are not reflected in the peripheral module and external bus clocks. The peripheral module and external bus clocks are restricted to the operating clock specified by bits ICK2 to ICK0.

Multi-clock mode is cleared by clearing all of bits ICK2 to ICK0, PCK2 to PCK0, and BCK2 to BCK0 to 0. A transition is made to normal mode at the end of the bus cycle, and multi-clock mode is cleared.

If a SLEEP instruction is executed while the SSBY bit in SBYCR is cleared to 0, this LSI enters sleep mode. When sleep mode is cleared by an interrupt, multi-clock mode is restored.

If a SLEEP instruction is executed while the SSBY bit in SBYCR is set to 1, this LSI enters software standby mode. When software standby mode is cleared by an external interrupt, multi-clock mode is restored.

When the $\overline{\text{RES}}$ pin is driven low, the reset state is entered and multi-clock mode is cleared. The same applies to a reset caused by watchdog timer overflow.

19.4 Module Stop Mode

Module stop mode can be set for individual on-chip peripheral modules.

When the corresponding MSTP bit in MSTPCRA, MSTPCRB, or MSTPCRC is set to 1, module operation stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

When the corresponding MSTP bit is cleared to 0, module stop mode is cleared and the module starts operating at the end of the bus cycle. In module stop mode, the internal states of modules other than the SCI, RCAN-ET and SSU are retained.

After the reset state is cleared, all modules other than the DMAC and on-chip RAM are in module stop mode.

The registers of the module for which module stop mode is selected cannot be read from or written to.

19.5 Sleep Mode

19.5.1 Transition to Sleep Mode

When the SLEEP instruction is executed when the SSBY bit in SBYCR is 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other peripheral functions do not stop.

19.5.2 Clearing Sleep Mode

Sleep mode is exited by any interrupt, signals on the $\overline{\text{RES}}$ pin, and a reset caused by a watchdog timer overflow.

1. Clearing by interrupt

   When an interrupt occurs, sleep mode is exited and interrupt exception processing starts. Sleep mode is not exited if the interrupt is disabled, or interrupts other than NMI are masked by the CPU.
2. Clearing by $\overline{\text{RES}}$ pin

   Setting the $\overline{\text{RES}}$ pin level low selects the reset state. After the stipulated reset input duration, driving the $\overline{\text{RES}}$ pin high makes the CPU start the reset exception processing.
3. Clearing by reset caused by watchdog timer overflow

   Sleep mode is exited by an internal reset caused by a watchdog timer overflow.

19.6 All-Module-Clock-Stop Mode

When the ACSE bit in MSTPCRA is set to 1 and all modules controlled by MSTPCR are stopped (MSTPCRA, MSTPCRB = H'FFFFFFFF, MSTPCRC = H'FF00), executing a SLEEP instruction with the SSBY bit in SBYCR cleared to 0 will cause all modules (except for the watchdog timer), the bus controller, and the I/O ports to stop operating, and to make a transition to all-module-clock-stop mode at the end of the bus cycle.

All-module-clock-stop mode is cleared by an external interrupt (NMI or $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ14}}$ pins), $\overline{\text{RES}}$ pin input, or an internal interrupt (watchdog timer), and the CPU returns to the normal program execution state via the exception handling state. All-module-clock-stop mode is not cleared if interrupts are disabled or interrupts other than NMI are masked on the CPU side.

19.7 Software Standby Mode

19.7.1 Transition to Software Standby Mode

If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, software standby mode is entered. In this mode, the CPU, on-chip peripheral functions, and oscillator all stop. However, the contents of the CPU's internal registers, on-chip RAM data, and the states of on-chip peripheral functions other than the SCI, RCAN-ET, and SSU, and the states of the I/O ports, are retained. In this mode the oscillator stops, allowing power consumption to be significantly reduced.

If the WDT is used as a watchdog timer, it is impossible to make a transition to software standby mode. The WDT should be stopped before the SLEEP instruction execution.

19.7.2 Clearing Software Standby Mode

Software standby mode is cleared by an external interrupt (NMI pin, or pins $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ14}}$*), or by means of the $\overline{\text{RES}}$ pin.

1. Clearing by interrupt

   When an NMI or IRQ0 to IRQ14* interrupt request signal is input, clock oscillation starts, and after the elapse of the time set in bits STS4 to STS0 in SBYCR, stable clocks are supplied to the entire LSI, software standby mode is cleared, and interrupt exception handling is started.

   When clearing software standby mode with an IRQ0 to IRQ14* interrupt, set the corresponding enable bit to 1 and ensure that no interrupt with a higher priority than interrupts IRQ0 to IRQ11* is generated. Software standby mode cannot be cleared if the interrupt has been masked on the CPU side or has been designated as a DMAC activation source.

   Note: * By setting the SSIn bit in SSIER to 1, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ14}}$ can be used as a software standby mode clearing source.

2. Clearing by $\overline{\text{RES}}$ pin

   When the $\overline{\text{RES}}$ pin is driven low, clock oscillation is started. At the same time as clock oscillation starts, clocks are supplied to the entire LSI. Note that the $\overline{\text{RES}}$ pin must be held low until clock oscillation settles. When the $\overline{\text{RES}}$ pin goes high, the CPU begins reset exception handling.

19.7.3 Setting Oscillation Settling Time after Clearing Software Standby Mode

Bits STS4 to STS0 in SBYCR should be set as described below.

1. Using a crystal resonator
   Set bits STS4 to STS0 so that the standby time is at least equal to the oscillation settling time. Table 19.2 shows the standby times for operating frequencies and settings of bits STS4 to STS0.
2. Using an external clock
   A PLL circuit settling time is necessary. Refer to table 19.2 to set the standby time.

Table 19.2 Oscillation Settling Time Settings

STS4	STS3	STS2	STS1	STS0	Standby Time	Pϕ* [MHz] 35	25	20	Unit
0	0	0	0	0	Reserved	—	—	—	μs
				1	Reserved	—	—	—	
			1	0	Reserved	—	—	—	
				1	Reserved	—	—	—	
		1	0	0	Reserved	—	—	—	
				1	64	1.8	2.6	3.2	
			1	0	512	14.6	20.5	25.6	
				1	1024	29.3	41.0	51.2	
	1	0	0	0	2048	58.5	81.9	102.4	
				1	4096	0.12	0.16	0.20	ms
			1	0	16384	0.47	0.66	0.82	
				1	32768	0.94	1.31	1.64	
		1	0	0	65536	1.87	2.62	3.28	
				1	131072	3.74	5.24	6.55	
			1	0	262144	7.49	10.49	13.11	
				1	524288	14.98	20.97	26.21	
1	0	0	0	0	Reserved	—	—	—	

: Recommended time setting when using a crystal resonator.
: Recommended time setting when using an external clock.

Note: * Pϕ is the output from the peripheral module frequency divider.

STS4	STS3	STS2	STS1	STS0	Standby Time	Pϕ* [MHz] 13	10	8	Unit
0	0	0	0	0	Reserved	—	—	—	μs
				1	Reserved	—	—	—	
			1	0	Reserved	—	—	—	
				1	Reserved	—	—	—	
		1	0	0	Reserved	—	—	—	
				1	64	4.9	6.4	8.0	
			1	0	512	39.4	51.2	64.0	
				1	1024	78.8	102.4	128.0	
	1	0	0	0	2048	157.5	204.8	256.0	
				1	4096	0.32	0.41	0.51	ms
			1	0	16384	1.26	1.64	2.05	
				1	32765	2.52	3.28	4.10	
		1	0	0	65536	5.04	6.55	8.19	
				1	131072	10.08	13.11	16.38	
			1	0	262144	20.16	26.21	32.77	
				1	524288	40.33	52.43	65.54	
1	0	0	0	0	Reserved	—	—	—	

: Recommended time setting when using a crystal resonator.

: Recommended time setting when using an external clock.

Note: * ϕ is the output from the peripheral module frequency divider.

19.7.4 Software Standby Mode Application Example

Figure 19.2 shows an example in which a transition is made to software standby mode at the falling edge on the NMI pin, and software standby mode is cleared at the rising edge on the NMI pin.

In this example, an NMI interrupt is accepted with the NMIEG bit in INTCR cleared to 0 (falling edge specification), then the NMIEG bit is set to 1 (rising edge specification), the SSBY bit is set to 1, and a SLEEP instruction is executed, causing a transition to software standby mode.

Software standby mode is then cleared at the rising edge on the NMI pin.

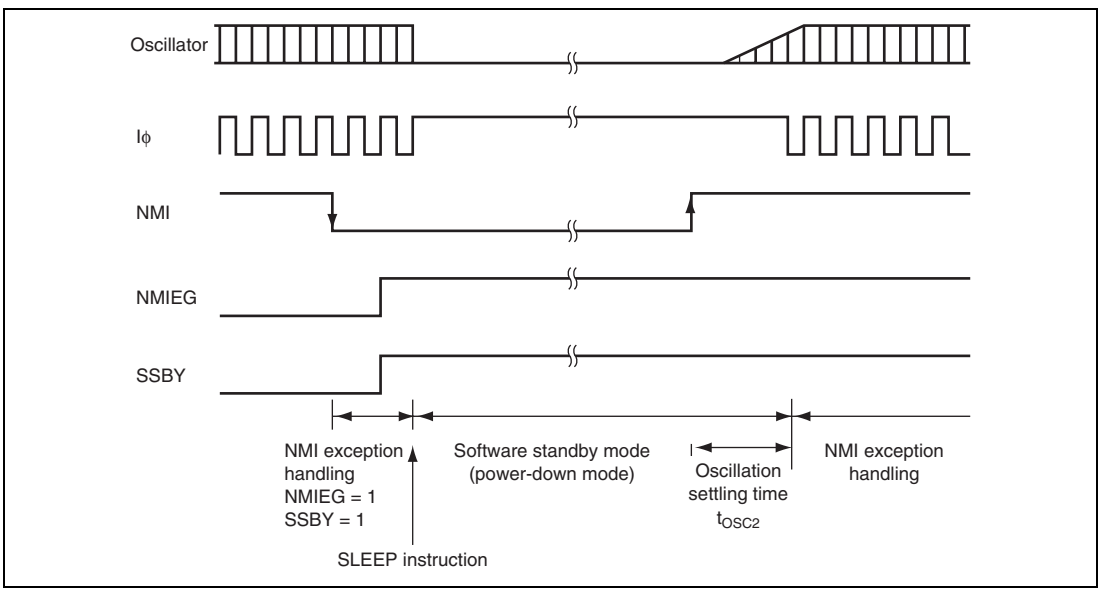


Figure 19.2 Software Standby Mode Application Example

19.8 Bϕ Clock Output Control

Output of the Bϕ clock can be controlled by bits PSTOP1 and POSEL1 in SCKCR, and DDR for the corresponding PA7 pin.

Clearing both bits PSTOP1 and POSEL1 to 0 enables the Bϕ clock output on the PA7 pin. When bit PSTOP1 is set to 1, the Bϕ clock output stops at the end of the bus cycle, and the Bϕ clock output goes high. When DDR for the PA7 pin is cleared to 0, the Bϕ clock output is disabled and the pin becomes an input port.

Disabling Bϕ output can reduce electromagnetic interference (EMI). Take it into consideration for design of the user system board.

Tables 19.3 shows the states of the Bϕ pin in each processing state.

Table 19.3 Bϕ Pin (PA7) State in Each Processing State

Register Setting Value			Normal Operating State	Sleep Mode	All-Module-Clock-Stop Mode	Software Standby Mode	
DDR	PSTOP1	POSEL1				OPE = 0	OPE = 1
0	X	X	Hi-Z	Hi-Z	Hi-Z	Hi-Z	Hi-Z
1	0	0	Bϕ output	Bϕ output	Bϕ output	High	High
1	0	1	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited
1	1	X	High	High	High	High	High

19.9 Usage Notes

19.9.1 I/O Port Status

In software standby mode, the I/O port states are retained. Therefore, there is no reduction in current consumption for the output current when a high-level signal is output.

19.9.2 Current Consumption during Oscillation Settling Standby Period

Current consumption increases during the oscillation settling standby period.

19.9.3 DMAC Module Stop

Depending on the operating state of the DMAC, bit MSTPA13 may not be set to 1. Setting of the DMAC module stop mode should be carried out only when the DMAC is not activated.

For details, refer to section 7, DMA Controller (DMAC).

19.9.4 On-Chip Peripheral Module Interrupts

Relevant interrupt operations cannot be performed in module stop mode. Consequently, if module stop mode is entered when an interrupt has been requested, it will not be possible to clear the CPU interrupt source or the DMAC activation source. Interrupts should therefore be disabled before entering module stop mode.

19.9.5 Writing to MSTPCRA, MSTPCRB, and MSTPCRC

MSTPCRA, MSTPCRB, and MSTPCRC should only be written to by the CPU.

Section 20 List of Registers

The register list gives information on the on-chip I/O register addresses, how the register bits are configured, and the register states in each operating mode. The information is given as shown below.

1. Register addresses (address order)

- Registers are listed from the lower allocation addresses.
- Registers are classified according to functional modules.
- Undefined and reserved addresses cannot be accessed. Do not access these addresses; otherwise, the operation when accessing these bits and subsequent operations cannot be guaranteed.

2. Register bits

- Bit configurations of the registers are listed in the same order as the register addresses.
- Reserved bits are indicated by — in the bit name column.
- Space in the bit name field indicates that the entire register is allocated to either the counter or data.
- For the registers of 16 or 32 bits, the MSB is listed first.

  Byte configuration description order is subject to big endian.

3. Register states in each operating mode

- Register states are listed in the same order as the register addresses.
- For the initialized state of each bit, refer to the register description in the corresponding section.
- The register states shown here are for the basic operating modes. If there is a specific reset for an on-chip peripheral module, refer to the section on that on-chip peripheral module.

20.1 Register Addresses (Address Order)

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Master control register		MCR	16	H'FEA00	RCAN-ET	16	4Pϕ/4Pϕ
General status register		GSR	16	H'FEA02	RCAN-ET	16	4Pϕ/4Pϕ
Bit configuration register 1		BCR1	16	H'FEA04	RCAN-ET	16	4Pϕ/4Pϕ
Bit configuration register 0		BCR0	16	H'FEA06	RCAN-ET	16	4Pϕ/4Pϕ
Interrupt request register		IRR	16	H'FEA08	RCAN-ET	16	4Pϕ/4Pϕ
Interrupt mask register		IMR	16	H'FEA0A	RCAN-ET	16	4Pϕ/4Pϕ
Transmit error counter/ Receive error counter		TEC/ REC	16	H'FEA0C	RCAN-ET	16	4Pϕ/4Pϕ
Transmit wait register 1		TXPR1	16	H'FEA20	RCAN-ET	16	4Pϕ/4Pϕ
Transmit wait register 0		TXPR0	16	H'FEA22	RCAN-ET	16	4Pϕ/4Pϕ
Transmit cancel register 0		TXCR0	16	H'FEA2A	RCAN-ET	16	4Pϕ/4Pϕ
Transmit acknowledge register 0		TXACK0	16	H'FEA32	RCAN-ET	16	4Pϕ/4Pϕ
Abort acknowledge register 0		ABACK0	16	H'FEA3A	RCAN-ET	16	4Pϕ/4Pϕ
Data frame receive pending register 0		RXPR0	16	H'FEA42	RCAN-ET	16	4Pϕ/4Pϕ
Remote frame receive pending register 0		RFPR0	16	H'FEA4A	RCAN-ET	16	4Pϕ/4Pϕ
Mailbox interrupt mask register 0		MBIMR0	16	H'FEA52	RCAN-ET	16	4Pϕ/4Pϕ
Unread message status register 0		UMSR0	16	H'FEA5A	RCAN-ET	16	4Pϕ/4Pϕ
MB[0].	CONTROL0H	—	16	H'FEB00	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEB02	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEB04	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEB06	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEB08	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEB09	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEB0A	RCAN-ET	8, 16	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEB0B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEB0C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEB0D	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEB0E	RCAN-ET	8, 16	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEB0F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEB10	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEB11	RCAN-ET	16	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[1].	CONTROL0H	—	16	H'FEB20	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEB22	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEB24	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEB26	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEB28	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEB29	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEB2A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEB2B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEB2C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEB2D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEB2E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEB2F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEB30	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEB31	RCAN-ET	8	4Pϕ/4Pϕ
MB[2].	CONTROL0H	—	16	H'FEB40	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEB42	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEB44	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEB46	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEB48	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEB49	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEB4A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEB4B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEB4C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEB4D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEB4E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEB4F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEB50	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEB51	RCAN-ET	8	4Pϕ/4Pϕ
MB[3].	CONTROL0H	—	16	H'FEB60	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEB62	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEB64	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEB66	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEB68	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEB69	RCAN-ET	8	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[3].	MSG_DATA[2]	—	8	H'FEB6A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEB6B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEB6C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEB6D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEB6E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEB6F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEB70	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEB71	RCAN-ET	8	4Pϕ/4Pϕ
MB[4].	CONTROL0H	—	16	H'FEB80	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEB82	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEB84	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEB86	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEB88	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEB89	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEB8A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEB8B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEB8C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEB8D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEB8E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEB8F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEB90	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEB91	RCAN-ET	8	4Pϕ/4Pϕ
MB[5].	CONTROL0H	—	16	H'FEBA0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEBA2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEBA4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEBA6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEBA8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEBA9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEBAA	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEBAB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEBAC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEBAD	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEBAE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEBAF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEBB0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEBB1	RCAN-ET	8	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[6].	CONTROL0H	—	16	H'FEBC0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEBC2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEBC4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEBC6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEBC8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEBC9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEBCA	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEBCB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEBCC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEBCD	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEBCE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEBCF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEBD0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEBD1	RCAN-ET	8	4Pϕ/4Pϕ
MB[7].	CONTROL0H	—	16	H'FEBE0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEBE2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEBE4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEBE6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEBE8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEBE9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEBEA	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEBEB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEBEC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEBED	RCAN-ET	8	4Pϕ/4Pϕ
	.MSG_DATA[6]	—	8	H'FEBEE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEBEF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEBF0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEBF1	RCAN-ET	8	4Pϕ/4Pϕ
MB[8].	CONTROL0H	—	16	H'FEC00	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEC02	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEC04	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEC06	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEC08	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEC09	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEC0A	RCAN-ET	16, 32	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[8].	MSG_DATA[3]	—	8	H'FEC0B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEC0C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEC0D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEC0E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEC0F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEC10	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEC11	RCAN-ET	8	4Pϕ/4Pϕ
MB[9].	CONTROL0H	—	16	H'FEC20	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEC22	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEC24	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEC26	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEC28	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEC29	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEC2A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEC2B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEC2C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEC2D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEC2E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEC2F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEC30	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEC31	RCAN-ET	8	4Pϕ/4Pϕ
MB[10].	CONTROL0H	—	16	H'FEC40	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEC42	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEC44	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEC46	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEC48	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEC49	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEC4A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEC4B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEC4C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEC4D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEC4E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEC4F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEC50	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEC51	RCAN-ET	8	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[11].	CONTROL0H	—	16	H'FEC60	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEC62	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEC64	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEC66	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEC68	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEC69	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEC6A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEC6B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEC6C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEC6D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEC6E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEC6F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEC70	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEC71	RCAN-ET	8	4Pϕ/4Pϕ
MB[12].	CONTROL0H	—	16	H'FEC80	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FEC82	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FEC84	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FEC86	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FEC88	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FEC89	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FEC8A	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FEC8B	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FEC8C	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FEC8D	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FEC8E	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FEC8F	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FEC90	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FEC91	RCAN-ET	8	4Pϕ/4Pϕ
MB[13].	CONTROL0H	—	16	H'FECA0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FECA2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FECA4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FECA6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FECA8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FECA9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FECAA	RCAN-ET	16, 32	4Pϕ/4Pϕ

Register Name		Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
MB[13].	MSG_DATA[3]	—	8	H'FECAB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FECAC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FECAD	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FECAE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FECAF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FECB0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FECB1	RCAN-ET	8	4Pϕ/4Pϕ
MB[14].	CONTROL0H	—	16	H'FECC0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FECC2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FECC4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FECC6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FECC8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FECC9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FECCA	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FECCB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FECCC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FECCD	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FECCE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FECCF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FECD0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FECD1	RCAN-ET	8	4Pϕ/4Pϕ
MB[15].	CONTROL0H	—	16	H'FECE0	RCAN-ET	16, 32	4Pϕ/4Pϕ
	CONTROL0L	—	16	H'FECE2	RCAN-ET	16	4Pϕ/4Pϕ
	LAFMH	—	16	H'FECE4	RCAN-ET	16, 32	4Pϕ/4Pϕ
	LAFML	—	16	H'FECE6	RCAN-ET	16	4Pϕ/4Pϕ
	MSG_DATA[0]	—	8	H'FECE8	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[1]	—	8	H'FECE9	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[2]	—	8	H'FECEA	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[3]	—	8	H'FECEB	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[4]	—	8	H'FECEC	RCAN-ET	8, 16, 32	4Pϕ/4Pϕ
	MSG_DATA[5]	—	8	H'FECED	RCAN-ET	8	4Pϕ/4Pϕ
	MSG_DATA[6]	—	8	H'FECEE	RCAN-ET	16, 32	4Pϕ/4Pϕ
	MSG_DATA[7]	—	8	H'FECEF	RCAN-ET	8	4Pϕ/4Pϕ
	CONTROL1H	—	8	H'FECF0	RCAN-ET	8, 16	4Pϕ/4Pϕ
	CONTROL1L	—	8	H'FECF1	RCAN-ET	8	4Pϕ/4Pϕ
RCAN monitor register		RCANMON	8	H'FED00	RCAN-ET	16	4Pϕ/4Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
SS control register H_0	SSCRH_0	8	H'FF200	SSU_0	16	3Pϕ/3Pϕ
SS control register L_0	SSCRL_0	8	H'FF201	SSU_0	16	3Pϕ/3Pϕ
SS mode register_0	SSMR_0	8	H'FF202	SSU_0	16	3Pϕ/3Pϕ
SS enable register_0	SSER_0	8	H'FF203	SSU_0	16	3Pϕ/3Pϕ
SS status register_0	SSSR_0	8	H'FF204	SSU_0	16	3Pϕ/3Pϕ
SS control register 2_0	SSCR2_0	8	H'FF205	SSU_0	16	3Pϕ/3Pϕ
SS transmit data register 0_0	SSTDR0_0	8	H'FF206	SSU_0	16	3Pϕ/3Pϕ
SS transmit data register 1_0	SSTDR1_0	8	H'FF207	SSU_0	16	3Pϕ/3Pϕ
SS transmit data register 2_0	SSTDR2_0	8	H'FF208	SSU_0	16	3Pϕ/3Pϕ
SS transmit data register 3_0	SSTDR3_0	8	H'FF209	SSU_0	16	3Pϕ/3Pϕ
SS receive data register 0_0	SSRDR0_0	8	H'FF20A	SSU_0	16	3Pϕ/3Pϕ
SS receive data register 1_0	SSRDR1_0	8	H'FF20B	SSU_0	16	3Pϕ/3Pϕ
SS receive data register 2_0	SSRDR2_0	8	H'FF20C	SSU_0	16	3Pϕ/3Pϕ
SS receive data register 3_0	SSRDR3_0	8	H'FF20D	SSU_0	16	3Pϕ/3Pϕ
SS control register H_1	SSCRH_1	8	H'FF210	SSU_1	16	3Pϕ/3Pϕ
SS control register L_1	SSCRL_1	8	H'FF211	SSU_1	16	3Pϕ/3Pϕ
SS mode register_1	SSMR_1	8	H'FF212	SSU_1	16	3Pϕ/3Pϕ
SS enable register_1	SSER_1	8	H'FF213	SSU_1	16	3Pϕ/3Pϕ
SS status register_1	SSSR_1	8	H'FF214	SSU_1	16	3Pϕ/3Pϕ
SS control register 2_1	SSCR2_1	8	H'FF215	SSU_1	16	3Pϕ/3Pϕ
SS transmit data register 0_1	SSTDR0_1	8	H'FF216	SSU_1	16	3Pϕ/3Pϕ
SS transmit data register 1_1	SSTDR1_1	8	H'FF217	SSU_1	16	3Pϕ/3Pϕ
SS transmit data register 2_1	SSTDR2_1	8	H'FF218	SSU_1	16	3Pϕ/3Pϕ
SS transmit data register 3_1	SSTDR3_1	8	H'FF219	SSU_1	16	3Pϕ/3Pϕ
SS receive data register 0_1	SSRDR0_1	8	H'FF21A	SSU_1	16	3Pϕ/3Pϕ
SS receive data register 1_1	SSRDR1_1	8	H'FF21B	SSU_1	16	3Pϕ/3Pϕ
SS receive data register 2_1	SSRDR2_1	8	H'FF21C	SSU_1	16	3Pϕ/3Pϕ
SS receive data register 3_1	SSRDR3_1	8	H'FF21D	SSU_1	16	3Pϕ/3Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
SS control register H_2	SSCRH_2	8	H'FF220	SSU_2	16	3Pϕ/3Pϕ
SS control register L_2	SSCRL_2	8	H'FF221	SSU_2	16	3Pϕ/3Pϕ
SS mode register_2	SSMR_2	8	H'FF222	SSU_2	16	3Pϕ/3Pϕ
SS enable register_2	SSER_2	8	H'FF223	SSU_2	16	3Pϕ/3Pϕ
SS status register_2	SSSR_2	8	H'FF224	SSU_2	16	3Pϕ/3Pϕ
SS control register2_2	SSCR2_2	8	H'FF225	SSU_2	16	3Pϕ/3Pϕ
SS transmit data register r0_2	SSTDR0_2	8	H'FF226	SSU_2	16	3Pϕ/3Pϕ
SS transmit data register 1_2	SSTDR1_2	8	H'FF227	SSU_2	16	3Pϕ/3Pϕ
SS transmit data register 2_2	SSTDR2_2	8	H'FF228	SSU_2	16	3Pϕ/3Pϕ
SS transmit data register 3_2	SSTDR3_2	8	H'FF229	SSU_2	16	3Pϕ/3Pϕ
SS receive data register 0_2	SSRDR0_2	8	H'FF22A	SSU_2	16	3Pϕ/3Pϕ
SS receive data register 1_2	SSRDR1_2	8	H'FF22B	SSU_2	16	3Pϕ/3Pϕ
SS receive data register r2_2	SSRDR2_2	8	H'FF22C	SSU_2	16	3Pϕ/3Pϕ
SS receive data register 3_2	SSRDR3_2	8	H'FF22D	SSU_2	16	3Pϕ/3Pϕ
Port H realtime input data register	PHRTIDR	8	H'FF240	PORT	16	3Pϕ/3Pϕ
A/D data register A_1	ADDRA_1	16	H'FFA90	A/D_1	16	2Pϕ/2Pϕ
A/D data register B_1	ADDRB_1	16	H'FFA92	A/D_1	16	2Pϕ/2Pϕ
A/D data register C_1	ADDRC_1	16	H'FFA94	A/D_1	16	2Pϕ/2Pϕ
A/D data register D_1	ADDRD_1	16	H'FFA96	A/D_1	16	2Pϕ/2Pϕ
A/D data register E_1	ADDRE_1	16	H'FFA98	A/D_1	16	2Pϕ/2Pϕ
A/D data register F_1	ADDRF_1	16	H'FFA9A	A/D_1	16	2Pϕ/2Pϕ
A/D data register G_1	ADDRG_1	16	H'FFA9C	A/D_1	16	2Pϕ/2Pϕ
A/D data register H_1	ADDRH_1	16	H'FFA9E	A/D_1	16	2Pϕ/2Pϕ
A/D control/status register_1	ADCSR_1	8	H'FFAA0	A/D_1	16	2Pϕ/2Pϕ
A/D control register_1	ADCR_1	8	H'FFAA1	A/D_1	16	2Pϕ/2Pϕ
Timer start register	TSTRB	8	H'FFB00	TPU	16	2Pϕ/2Pϕ
Timer synchronous register	TSYRB	8	H'FFB01	TPU	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Timer control register_6	TCR_6	8	H'FFB10	TPU_6	16	2Pϕ/2Pϕ
Timer mode register_6	TMDR_6	8	H'FFB11	TPU_6	16	2Pϕ/2Pϕ
Timer I/O control register H_6	TIORH_6	8	H'FFB12	TPU_6	16	2Pϕ/2Pϕ
Timer I/O control register L_6	TIORL_6	8	H'FFB13	TPU_6	16	2Pϕ/2Pϕ
Timer interrupt enable register_6	TIER_6	8	H'FFB14	TPU_6	16	2Pϕ/2Pϕ
Timer status register_6	TSR_6	8	H'FFB15	TPU_6	16	2Pϕ/2Pϕ
Timer counter_6	TCNT_6	16	H'FFB16	TPU_6	16	2Pϕ/2Pϕ
Timer general register A_6	TGRA_6	16	H'FFB18	TPU_6	16	2Pϕ/2Pϕ
Timer general register B_6	TGRB_6	16	H'FFB1A	TPU_6	16	2Pϕ/2Pϕ
Timer general register C_6	TGRC_6	16	H'FFB1C	TPU_6	16	2Pϕ/2Pϕ
Timer general register D_6	TGRD_6	16	H'FFB1E	TPU_6	16	2Pϕ/2Pϕ
Timer control register_7	TCR_7	8	H'FFB20	TPU_7	16	2Pϕ/2Pϕ
Timer mode register_7	TMDR_7	8	H'FFB21	TPU_7	16	2Pϕ/2Pϕ
Timer I/O control register_7	TIOR_7	8	H'FFB22	TPU_7	16	2Pϕ/2Pϕ
Timer interrupt enable register_7	TIER_7	8	H'FFB24	TPU_7	16	2Pϕ/2Pϕ
Timer status register_7	TSR_7	8	H'FFB25	TPU_7	16	2Pϕ/2Pϕ
Timer counter_7	TCNT_7	16	H'FFB26	TPU_7	16	2Pϕ/2Pϕ
Timer general register A_7	TGRA_7	16	H'FFB28	TPU_7	16	2Pϕ/2Pϕ
Timer general register B_7	TGRB_7	16	H'FFB2A	TPU_7	16	2Pϕ/2Pϕ
Timer control register_8	TCR_8	8	H'FFB30	TPU_8	16	2Pϕ/2Pϕ
Timer mode register_8	TMDR_8	8	H'FFB31	TPU_8	16	2Pϕ/2Pϕ
Timer I/O control register_8	TIOR_8	8	H'FFB32	TPU_8	16	2Pϕ/2Pϕ
Timer interrupt enable register_8	TIER_8	8	H'FFB34	TPU_8	16	2Pϕ/2Pϕ
Timer status register_8	TSR_8	8	H'FFB35	TPU_8	16	2Pϕ/2Pϕ
Timer counter_8	TCNT_8	16	H'FFB36	TPU_8	16	2Pϕ/2Pϕ
Timer general register A_8	TGRA_8	16	H'FFB38	TPU_8	16	2Pϕ/2Pϕ
Timer general register B_8	TGRB_8	16	H'FFB3A	TPU_8	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Timer control register_9	TCR_9	8	H'FFB40	TPU_9	16	2Pϕ/2Pϕ
Timer mode register_9	TMDR_9	8	H'FFB41	TPU_9	16	2Pϕ/2Pϕ
Timer I/O control register H_9	TIORH_9	8	H'FFB42	TPU_9	16	2Pϕ/2Pϕ
Timer I/O control register L_9	TIORL_9	8	H'FFB43	TPU_9	16	2Pϕ/2Pϕ
Timer interrupt enable register_9	TIER_9	8	H'FFB44	TPU_9	16	2Pϕ/2Pϕ
Timer status register_9	TSR_9	8	H'FFB45	TPU_9	16	2Pϕ/2Pϕ
Timer counter_9	TCNT_9	16	H'FFB46	TPU_9	16	2Pϕ/2Pϕ
Timer general register A_9	TGRA_9	16	H'FFB48	TPU_9	16	2Pϕ/2Pϕ
Timer general register B_9	TGRB_9	16	H'FFB4A	TPU_9	16	2Pϕ/2Pϕ
Timer general register C_9	TGRC_9	16	H'FFB4C	TPU_9	16	2Pϕ/2Pϕ
Timer general register D_9	TGRD_9	16	H'FFB4E	TPU_9	16	2Pϕ/2Pϕ
Timer control register_10	TCR_10	8	H'FFB50	TPU_10	16	2Pϕ/2Pϕ
Timer mode register_10	TMDR_10	8	H'FFB51	TPU_10	16	2Pϕ/2Pϕ
Timer I/O control register_10	TIOR_10	8	H'FFB52	TPU_10	16	2Pϕ/2Pϕ
Timer interrupt enable register_10	TIER_10	8	H'FFB54	TPU_10	16	2Pϕ/2Pϕ
Timer status register_10	TSR_10	8	H'FFB55	TPU_10	16	2Pϕ/2Pϕ
Timer counter_10	TCNT_10	16	H'FFB56	TPU_10	16	2Pϕ/2Pϕ
Timer general register A_10	TGRA_10	16	H'FFB58	TPU_10	16	2Pϕ/2Pϕ
Timer general register B_10	TGRB_10	16	H'FFB5A	TPU_10	16	2Pϕ/2Pϕ
Timer control register_11	TCR_11	8	H'FFB60	TPU_11	16	2Pϕ/2Pϕ
Timer mode register_11	TMDR_11	8	H'FFB61	TPU_11	16	2Pϕ/2Pϕ
Timer I/O control register_11	TIOR_11	8	H'FFB62	TPU_11	16	2Pϕ/2Pϕ
Timer interrupt enable register_11	TIER_11	8	H'FFB64	TPU_11	16	2Pϕ/2Pϕ
Timer status register_11	TSR_11	8	H'FFB65	TPU_11	16	2Pϕ/2Pϕ
Timer counter_11	TCNT_11	16	H'FFB66	TPU_11	16	2Pϕ/2Pϕ
Timer general register A_11	TGRA_11	16	H'FFB68	TPU_11	16	2Pϕ/2Pϕ
Timer general register B_11	TGRB_11	16	H'FFB6A	TPU_11	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Port 1 data direction register	P1DDR	8	H'FFB80	I/O port	8	2Pϕ/2Pϕ
Port 2 data direction register	P2DDR	8	H'FFB81	I/O port	8	2Pϕ/2Pϕ
Port 3 data direction register	P3DDR	8	H'FFB82	I/O port	8	2Pϕ/2Pϕ
Port 6 data direction register	P6DDR	8	H'FFB85	I/O port	8	2Pϕ/2Pϕ
Port A data direction register	PADDR	8	H'FFB89	I/O port	8	2Pϕ/2Pϕ
Port D data direction register	PDDDR	8	H'FFB8C	I/O port	8	2Pϕ/2Pϕ
Port 1 input buffer control register	P1ICR	8	H'FFB90	I/O port	8	2Pϕ/2Pϕ
Port 2 input buffer control register	P2ICR	8	H'FFB91	I/O port	8	2Pϕ/2Pϕ
Port 3 input buffer control register	P3ICR	8	H'FFB92	I/O port	8	2Pϕ/2Pϕ
Port 4 input buffer control register	P4ICR	8	H'FFB93	I/O port	8	2Pϕ/2Pϕ
Port 5 input buffer control register	P5ICR	8	H'FFB94	I/O port	8	2Pϕ/2Pϕ
Port 6 input buffer control register	P6ICR	8	H'FFB95	I/O port	8	2Pϕ/2Pϕ
Port A input buffer control register	PAICR	8	H'FFB99	I/O port	8	2Pϕ/2Pϕ
Port D input buffer control register	PDICR	8	H'FFB9C	I/O port	8	2Pϕ/2Pϕ
Port H register	PORTH	8	H'FFBA0	I/O port	8	2Pϕ/2Pϕ
Port J register	PORTJ	8	H'FFBA2	I/O port	8	2Pϕ/2Pϕ
Port K register	PORTK	8	H'FFBA3	I/O port	8	2Pϕ/2Pϕ
Port H data register	PHDR	8	H'FFBA4	I/O port	8	2Pϕ/2Pϕ
Port J data register	PJDR	8	H'FFBA6	I/O port	8	2Pϕ/2Pϕ
Port K data register	PKDR	8	H'FFBA7	I/O port	8	2Pϕ/2Pϕ
Port H data direction register	PHDDR	8	H'FFBA8	I/O port	8	2Pϕ/2Pϕ
Port J data direction register	PJDDR	8	H'FFBAA	I/O port	8	2Pϕ/2Pϕ
Port K data direction register	PKDDR	8	H'FFBAB	I/O port	8	2Pϕ/2Pϕ
Port H input buffer control register	PHICR	8	H'FFBAC	I/O port	8	2Pϕ/2Pϕ
Port J input buffer control register	PJICR	8	H'FFBAE	I/O port	8	2Pϕ/2Pϕ
Port K input buffer control register	PKICR	8	H'FFBAF	I/O port	8	2Pϕ/2Pϕ
Port D pull-up MOS control register	PDPCR	8	H'FFBB4	I/O port	8	2Pϕ/2Pϕ
Port H pull-up MOS control register	PHPCR	8	H'FFBB8	I/O port	8	2Pϕ/2Pϕ
Port J pull-up MOS control register	PJPCR	8	H'FFBBA	I/O port	8	2Pϕ/2Pϕ
Port K pull-up MOS control register	PKPCR	8	H'FFBBB	I/O port	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Port 2 open drain control register	P2ODR	8	H'FFBBC	I/O port	8	2Pϕ/2Pϕ
Port function control register 9	PFCR9	8	H'FFBC9	I/O port	8	2Pϕ/3Pϕ
Port function control register A	PFCRA	8	H'FFBCA	I/O port	8	2Pϕ/3Pϕ
Port function control register B	PFCRB	8	H'FFBCB	I/O port	8	2Pϕ/3Pϕ
Software standby release IRQ enable register	SSIER	16	H'FFBCE	INTC	8	2Pϕ/3Pϕ
DMA source address register_0	DSAR_0	32	H'FFC00	DMAC_0	16	2Iϕ/2Iϕ
DMA destination address register_0	DDAR_0	32	H'FFC04	DMAC_0	16	2Iϕ/2Iϕ
DMA offset register_0	DOFR_0	32	H'FFC08	DMAC_0	16	2Iϕ/2Iϕ
DMA transfer count register_0	DTCR_0	32	H'FFC0C	DMAC_0	16	2Iϕ/2Iϕ
DMA block size register_0	DBSR_0	32	H'FFC10	DMAC_0	16	2Iϕ/2Iϕ
DMA mode control register_0	DMDR_0	32	H'FFC14	DMAC_0	16	2Iϕ/2Iϕ
DMA address control register_0	DACR_0	32	H'FFC18	DMAC_0	16	2Iϕ/2Iϕ
DMA source address register_1	DSAR_1	32	H'FFC20	DMAC_1	16	2Iϕ/2Iϕ
DMA destination address register_1	DDAR_1	32	H'FFC24	DMAC_1	16	2Iϕ/2Iϕ
DMA offset register_1	DOFR_1	32	H'FFC28	DMAC_1	16	2Iϕ/2Iϕ
DMA transfer count register_1	DTCR_1	32	H'FFC2C	DMAC_1	16	2Iϕ/2Iϕ
DMA block size register_1	DBSR_1	32	H'FFC30	DMAC_1	16	2Iϕ/2Iϕ
DMA mode control register_1	DMDR_1	32	H'FFC34	DMAC_1	16	2Iϕ/2Iϕ
DMA address control register_1	DACR_1	32	H'FFC38	DMAC_1	16	2Iϕ/2Iϕ
DMA source address register_2	DSAR_2	32	H'FFC40	DMAC_2	16	2Iϕ/2Iϕ
DMA destination address register_2	DDAR_2	32	H'FFC44	DMAC_2	16	2Iϕ/2Iϕ
DMA offset register_2	DOFR_2	32	H'FFC48	DMAC_2	16	2Iϕ/2Iϕ
DMA transfer count register_2	DTCR_2	32	H'FFC4C	DMAC_2	16	2Iϕ/2Iϕ
DMA block size register_2	DBSR_2	32	H'FFC50	DMAC_2	16	2Iϕ/2Iϕ
DMA mode control register_2	DMDR_2	32	H'FFC54	DMAC_2	16	2Iϕ/2Iϕ
DMA address control register_2	DACR_2	32	H'FFC58	DMAC_2	16	2Iϕ/2Iϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
DMA source address register_3	DSAR_3	32	H'FFC60	DMAC_3	16	2Iϕ/2Iϕ
DMA destination address register_3	DDAR_3	32	H'FFC64	DMAC_3	16	2Iϕ/2Iϕ
DMA offset register_3	DOFR_3	32	H'FFC68	DMAC_3	16	2Iϕ/2Iϕ
DMA transfer count register_3	DTCR_3	32	H'FFC6C	DMAC_3	16	2Iϕ/2Iϕ
DMA block size register_3	DBSR_3	32	H'FFC70	DMAC_3	16	2Iϕ/2Iϕ
DMA mode control register_3	DMDR_3	32	H'FFC74	DMAC_3	16	2Iϕ/2Iϕ
DMA address control register_3	DACR_3	32	H'FFC78	DMAC_3	16	2Iϕ/2Iϕ
DMA module request select register_0	DMRSR_0	8	H'FFD20	DMAC_0	16	2Iϕ/2Iϕ
DMA module request select register_1	DMRSR_1	8	H'FFD21	DMAC_1	16	2Iϕ/2Iϕ
DMA module request select register_2	DMRSR_2	8	H'FFD22	DMAC_2	16	2Iϕ/2Iϕ
DMA module request select register_3	DMRSR_3	8	H'FFD23	DMAC_3	16	2Iϕ/2Iϕ
Interrupt priority register A	IPRA	16	H'FFD40	INTC	16	2Iϕ/3Iϕ
Interrupt priority register B	IPRB	16	H'FFD42	INTC	16	2Iϕ/3Iϕ
Interrupt priority register C	IPRC	16	H'FFD44	INTC	16	2Iϕ/3Iϕ
Interrupt priority register D	IPRD	16	H'FFD46	INTC	16	2Iϕ/3Iϕ
Interrupt priority register E	IPRE	16	H'FFD48	INTC	16	2Iϕ/3Iϕ
Interrupt priority register F	IPRF	16	H'FFD4A	INTC	16	2Iϕ/3Iϕ
Interrupt priority register G	IPRG	16	H'FFD4C	INTC	16	2Iϕ/3Iϕ
Interrupt priority register I	IPRI	16	H'FFD50	INTC	16	2Iϕ/3Iϕ
Interrupt priority register K	IPRK	16	H'FFD54	INTC	16	2Iϕ/3Iϕ
Interrupt priority register L	IPRL	16	H'FFD56	INTC	16	2Iϕ/3Iϕ
Interrupt priority register M	IPRM	16	H'FFD58	INTC	16	2Iϕ/3Iϕ
Interrupt priority register N	IPRN	16	H'FFD5A	INTC	16	2Iϕ/3Iϕ
Interrupt priority register O	IPRO	16	H'FFD5C	INTC	16	2Iϕ/3Iϕ
Interrupt priority register Q	IPRQ	16	H'FFD60	INTC	16	2Iϕ/3Iϕ
Interrupt priority register R	IPRR	16	H'FFD62	INTC	16	2Iϕ/3Iϕ
IRQ sense control register H	ISCRH	16	H'FFD68	INTC	16	2Iϕ/3Iϕ
IRQ sense control register L	ISCRL	16	H'FFD6A	INTC	16	2Iϕ/3Iϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Bus control register 2	BCR2	8	H'FFD94	BSC	16	2Iϕ/3Iϕ
RAM emulation register	RAMER	8	H'FFD9E	BSC	16	2Iϕ/3Iϕ
Mode control register	MDCR	16	H'FFDC0	SYSTEM	16	2Iϕ/3Iϕ
System control register	SYSCR	16	H'FFDC2	SYSTEM	16	2Iϕ/3Iϕ
System clock control register	SCKCR	16	H'FFDC4	SYSTEM	16	2Iϕ/3Iϕ
Standby control register	SBYCR	16	H'FFDC6	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register A	MSTPCRA	16	H'FFDC8	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register B	MSTPCRB	16	H'FFDCA	SYSTEM	16	2Iϕ/3Iϕ
Module stop control register C	MSTPCRC	16	H'FFDCC	SYSTEM	16	2Iϕ/3Iϕ
Serial mode register_3	SMR_3	8	H'FFE88	SCI_3	8	2Pϕ/2Pϕ
Bit rate register_3	BRR_3	8	H'FFE89	SCI_3	8	2Pϕ/2Pϕ
Serial control register_3	SCR_3	8	H'FFE8A	SCI_3	8	2Pϕ/2Pϕ
Transmit data register_3	TDR_3	8	H'FFE8B	SCI_3	8	2Pϕ/2Pϕ
Serial status register_3	SSR_3	8	H'FFE8C	SCI_3	8	2Pϕ/2Pϕ
Receive data register_3	RDR_3	8	H'FFE8D	SCI_3	8	2Pϕ/2Pϕ
Smart card mode register_3	SCMR_3	8	H'FFE8E	SCI_3	8	2Pϕ/2Pϕ
Serial mode register_4	SMR_4	8	H'FFE90	SCI_4	8	2Pϕ/2Pϕ
Bit rate register_4	BRR_4	8	H'FFE91	SCI_4	8	2Pϕ/2Pϕ
Serial control register_4	SCR_4	8	H'FFE92	SCI_4	8	2Pϕ/2Pϕ
Transmit data register_4	TDR_4	8	H'FFE93	SCI_4	8	2Pϕ/2Pϕ
Serial status register_4	SSR_4	8	H'FFE94	SCI_4	8	2Pϕ/2Pϕ
Receive data register_4	RDR_4	8	H'FFE95	SCI_4	8	2Pϕ/2Pϕ
Smart card mode register_4	SCMR_4	8	H'FFE96	SCI_4	8	2Pϕ/2Pϕ
Flash code control/status register	FCCS	8	H'FFEA8	FLASH	8	2Pϕ/2Pϕ
Flash program code select register	FPCS	8	H'FFEA9	FLASH	8	2Pϕ/2Pϕ
Flash erase code select register	FECS	8	H'FFEAA	FLASH	8	2Pϕ/2Pϕ
Flash key code register	FKEY	8	H'FFEAC	FLASH	8	2Pϕ/2Pϕ
Flash MAT select register	FMATS	8	H'FFEAD	FLASH	8	2Pϕ/2Pϕ
Flash transfer destination address register	FTDAR	8	H'FFEAE	FLASH	8	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Timer control register_4	TCR_4	8	H'FFEE0	TPU_4*2	16	2Pϕ/2Pϕ
Timer mode register_4	TMDR_4	8	H'FFEE1	TPU_4*2	16	2Pϕ/2Pϕ
Timer I/O control register_4	TIOR_4	8	H'FFEE2	TPU_4*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_4	TIER_4	8	H'FFEE4	TPU_4*2	16	2Pϕ/2Pϕ
Timer status register_4	TSR_4	8	H'FFEE5	TPU_4*2	16	2Pϕ/2Pϕ
Timer counter_4	TCNT_4	16	H'FFEE6	TPU_4*2	16	2Pϕ/2Pϕ
Timer general register A_4	TGRA_4	16	H'FFEE8	TPU_4*2	16	2Pϕ/2Pϕ
Timer general register B_4	TGRB_4	16	H'FFEEA	TPU_4*2	16	2Pϕ/2Pϕ
Timer control register_5	TCR_5	8	H'FFEF0	TPU_5*2	16	2Pϕ/2Pϕ
Timer mode register_5	TMDR_5	8	H'FFEF1	TPU_5*2	16	2Pϕ/2Pϕ
Timer I/O control register_5	TIOR_5	8	H'FFEF2	TPU_5*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_5	TIER_5	8	H'FFEF4	TPU_5*2	16	2Pϕ/2Pϕ
Timer status register_5	TSR_5	8	H'FFEF5	TPU_5*2	16	2Pϕ/2Pϕ
Timer counter_5	TCNT_5	16	H'FFEF6	TPU_5*2	16	2Pϕ/2Pϕ
Timer general register A_5	TGRA_5	16	H'FFEF8	TPU_5*2	16	2Pϕ/2Pϕ
Timer general register B_5	TGRB_5	16	H'FFEFA	TPU_5*2	16	2Pϕ/2Pϕ
Interrupt control register	INTCR	8	H'FFF32	INTC	16	2Iϕ/3Iϕ
CPU priority control register	CPUPCR	8	H'FFF33	INTC	16	2Iϕ/3Iϕ
IRQ enable register	IER	16	H'FFF34	INTC	16	2Iϕ/3Iϕ
IRQ status register	ISR	16	H'FFF36	INTC	16	2Iϕ/3Iϕ
Port 1 register	PORT1	8	H'FFF40	I/O port	8	2Pϕ/-
Port 2 register	PORT2	8	H'FFF41	I/O port	8	2Pϕ/-
Port 3 register	PORT3	8	H'FFF42	I/O port	8	2Pϕ/-
Port 4 register	PORT4	8	H'FFF43	I/O port	8	2Pϕ/-
Port 5 register	PORT5	8	H'FFF44	I/O port	8	2Pϕ/-
Port 6 register	PORT6	8	H'FFF45	I/O port	8	2Pϕ/-
Port A register	PORTA	8	H'FFF49	I/O port	8	2Pϕ/-
Port D register	PORTD	8	H'FFF4C	I/O port	8	2Pϕ/-

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Port 1 data register	P1DR	8	H'FFF50	I/O port	8	2Pϕ/2Pϕ
Port 2 data register	P2DR	8	H'FFF51	I/O port	8	2Pϕ/2Pϕ
Port 3 data register	P3DR	8	H'FFF52	I/O port	8	2Pϕ/2Pϕ
Port 6 data register	P6DR	8	H'FFF55	I/O port	8	2Pϕ/2Pϕ
Port A data register	PADR	8	H'FFF59	I/O port	8	2Pϕ/2Pϕ
Port D data register	PDDR	8	H'FFF5C	I/O port	8	2Pϕ/2Pϕ
PPG output control register	PCR	8	H'FFF76	PPG*[2]	8	2Pϕ/2Pϕ
PPG output mode register	PMR	8	H'FFF77	PPG*[2]	8	2Pϕ/2Pϕ
Next data enable register H	NDERH	8	H'FFF78	PPG*[2]	8	2Pϕ/2Pϕ
Next data enable register L	NDERL	8	H'FFF79	PPG*[2]	8	2Pϕ/2Pϕ
Output data register H	PODRH	8	H'FFF7A	PPG*[2]	8	2Pϕ/2Pϕ
Output data register L	PODRL	8	H'FFF7B	PPG*[2]	8	2Pϕ/2Pϕ
Next data register H	NDRH	8	H'FFF7C	PPG*[2]	8	2Pϕ/2Pϕ
Next data register L	NDRL	8	H'FFF7D	PPG*[2]	8	2Pϕ/2Pϕ
Next data register H	NDRH	8	H'FFF7E	PPG*[2]	8	2Pϕ/2Pϕ
Next data register L	NDRL	8	H'FFF7F	PPG*[2]	8	2Pϕ/2Pϕ
A/D data register A_0	ADDRA_0	16	H'FFF90	A/D_0	16	2Pϕ/2Pϕ
A/D data register B_0	ADDRB_0	16	H'FFF92	A/D_0	16	2Pϕ/2Pϕ
A/D data register C_0	ADDRC_0	16	H'FFF94	A/D_0	16	2Pϕ/2Pϕ
A/D data register D_0	ADDRD_0	16	H'FFF96	A/D_0	16	2Pϕ/2Pϕ
A/D data register E_0	ADDRE_0	16	H'FFF98	A/D_0	16	2Pϕ/2Pϕ
A/D data register F_0	ADDRF_0	16	H'FFF9A	A/D_0	16	2Pϕ/2Pϕ
A/D data register G_0	ADDRG_0	16	H'FFF9C	A/D_0	16	2Pϕ/2Pϕ
A/D data register H_0	ADDRH_0	16	H'FFF9E	A/D_0	16	2Pϕ/2Pϕ
A/D control/status register_0	ADCSR_0	8	H'FFFA0	A/D_0	16	2Pϕ/2Pϕ
A/D control register_0	ADCR_0	8	H'FFFA1	A/D_0	16	2Pϕ/2Pϕ
Timer control/status register	TCSR	8	H'FFFA4	WDT		2Pϕ/3Pϕ
Timer counter	TCNT	8	H'FFFA5	WDT		2Pϕ/3Pϕ
Reset control/status register	RSTCSR	8	H'FFFA7	WDT		2Pϕ/3Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Timer start register	TSTR	8	H'FFFBC	TPU*2	16	2Pϕ/2Pϕ
Timer synchronous register	TSYR	8	H'FFFBD	TPU*2	16	2Pϕ/2Pϕ
Timer control register_0	TCR_0	8	H'FFFC0	TPU_0*2	16	2Pϕ/2Pϕ
Timer mode register_0	TMDR_0	8	H'FFFC1	TPU_0*2	16	2Pϕ/2Pϕ
Timer I/O control register H_0	TIORH_0	8	H'FFFC2	TPU_0*2	16	2Pϕ/2Pϕ
Timer I/O control register L_0	TIORL_0	8	H'FFFC3	TPU_0*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_0	TIER_0	8	H'FFFC4	TPU_0*2	16	2Pϕ/2Pϕ
Timer status register_0	TSR_0	8	H'FFFC5	TPU_0*2	16	2Pϕ/2Pϕ
Timer counter_0	TCNT_0	16	H'FFFC6	TPU_0*2	16	2Pϕ/2Pϕ
Timer general register A_0	TGRA_0	16	H'FFFC8	TPU_0*2	16	2Pϕ/2Pϕ
Timer general register B_0	TGRB_0	16	H'FFFCA	TPU_0*2	16	2Pϕ/2Pϕ
Timer general register C_0	TGRC_0	16	H'FFFCC	TPU_0*2	16	2Pϕ/2Pϕ
Timer general register D_0	TGRD_0	16	H'FFFCE	TPU_0*2	16	2Pϕ/2Pϕ
Timer control register_1	TCR_1	8	H'FFFD0	TPU_1*2	16	2Pϕ/2Pϕ
Timer mode register_1	TMDR_1	8	H'FFFD1	TPU_1*2	16	2Pϕ/2Pϕ
Timer I/O control register_1	TIOR_1	8	H'FFFD2	TPU_1*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_1	TIER_1	8	H'FFFD4	TPU_1*2	16	2Pϕ/2Pϕ
Timer status register_1	TSR_1	8	H'FFFD5	TPU_1*2	16	2Pϕ/2Pϕ
Timer counter_1	TCNT_1	16	H'FFFD6	TPU_1*2	16	2Pϕ/2Pϕ
Timer general register A_1	TGRA_1	16	H'FFFD8	TPU_1*2	16	2Pϕ/2Pϕ
Timer general register B_1	TGRB_1	16	H'FFFDA	TPU_1*2	16	2Pϕ/2Pϕ
Timer control register_2	TCR_2	8	H'FFFE0	TPU_2*2	16	2Pϕ/2Pϕ
Timer mode register_2	TMDR_2	8	H'FFFE1	TPU_2*2	16	2Pϕ/2Pϕ
Timer I/O control register_2	TIOR_2	8	H'FFFE2	TPU_2*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_2	TIER_2	8	H'FFFE4	TPU_2*2	16	2Pϕ/2Pϕ
Timer status register_2	TSR_2	8	H'FFFE5	TPU_2*2	16	2Pϕ/2Pϕ
Timer counter_2	TCNT_2	16	H'FFFE6	TPU_2*2	16	2Pϕ/2Pϕ
Timer general register A_2	TGRA_2	16	H'FFFE8	TPU_2*2	16	2Pϕ/2Pϕ
Timer general register B_2	TGRB_2	16	H'FFFEA	TPU_2*2	16	2Pϕ/2Pϕ

Register Name	Abbreviation	Number of Bits	Address	Module	Data Width	Access Cycles
Timer control register_3	TCR_3	8	H'FFFF0	TPU_3*2	16	2Pϕ/2Pϕ
Timer mode register_3	TMDR_3	8	H'FFFF1	TPU_3*2	16	2Pϕ/2Pϕ
Timer I/O control register H_3	TIORH_3	8	H'FFFF2	TPU_3*2	16	2Pϕ/2Pϕ
Timer I/O control register L_3	TIORL_3	8	H'FFFF3	TPU_3*2	16	2Pϕ/2Pϕ
Timer interrupt enable register_3	TIER_3	8	H'FFFF4	TPU_3*2	16	2Pϕ/2Pϕ
Timer status register_3	TSR_3	8	H'FFFF5	TPU_3*2	16	2Pϕ/2Pϕ
Timer counter_3	TCNT_3	16	H'FFFF6	TPU_3*2	16	2Pϕ/2Pϕ
Timer general register A_3	TGRA_3	16	H'FFFF8	TPU_3*2	16	2Pϕ/2Pϕ
Timer general register B_3	TGRB_3	16	H'FFFFA	TPU_3*2	16	2Pϕ/2Pϕ
Timer general register C_3	TGRC_3	16	H'FFFFC	TPU_3*2	16	2Pϕ/2Pϕ
Timer general register D_3	TGRD_3	16	H'FFFFE	TPU_3*2	16	2Pϕ/2Pϕ

Notes: 1. The lower 20 bits are indicated.
2. Supported only by the H8SX/1527R.

20.2 Register Bits

Register addresses and bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, and 16-bit and 32-bit registers are shown as 2 or 4 lines, respectively.

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MCR	MCR15	MCR14	—	—	—	TST2	TST1	TST0	RCAN-ET
	MCR7	MCR6	MCR5	—	—	MCR2	MCR1	MCR0	
GSR	—	—	—	—	—	—	—	—	
	—	—	GSR5	GSR4	GSR3	GSR2	GSR1	GSR0	
BCR1	TSG13	TSG12	TSG11	TSG10	—	TSG22	TSG21	TSG20	
	—	—	SJW1	SJW0	—	—	—	BSP	
BCR0	—	—	—	—	—	—	—	—	
	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	
IRR	—	—	IRR13	IRR12	—	—	IRR9	IRR8	
	IRR7	IRR6	IRR5	IRR4	IRR3	IRR2	IRR1	IRR0	
IMR	IMR15	IMR14	IMR13	IMR12	IMR11	IMR10	IMR9	IMR8	
	IMR7	IMR6	IMR5	IMR4	IMR3	IMR2	IMR1	IMR0	
TEC	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0	
REC	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0	
TXPR1	TXPR115	TXPR114	TXPR113	TXPR112	TXPR111	TXPR110	TXPR19	TXPR18	
	TXPR17	TXPR16	TXPR15	TXPR14	TXPR13	TXPR12	TXPR11	TXPR10	
TXPR0	TXPR015	TXPR014	TXPR013	TXPR012	TXPR011	TXPR010	TXPR09	TXPR08	
	TXPR07	TXPR06	TXPR05	TXPR04	TXPR03	TXPR02	TXPR01	—	
TXCR0	TXCR015	TXCR014	TXCR013	TXCR012	TXCR011	TXCR010	TXCR09	TXCR08	
	TXCR07	TXCR06	TXCR05	TXCR04	TXCR03	TXCR02	TXCR01	—	
TXACK0	TXACK015	TXACK014	TXACK013	TXACK012	TXACK011	TXACK010	TXACK09	TXACK08	
	TXACK07	TXACK06	TXACK05	TXACK04	TXACK03	TXACK02	TXACK01	—	
ABACK0	ABACK015	ABACK014	ABACK013	ABACK012	ABACK011	ABACK010	ABACK09	ABACK08	
	ABACK07	ABACK06	ABACK05	ABACK04	ABACK03	ABACK02	ABACK01	—	
RXPR0	RXPR015	RXPR014	RXPR013	RXPR012	RXPR011	RXPR010	RXPR09	RXPR08	
	RXPR07	RXPR06	RXPR05	RXPR04	RXPR03	RXPR02	RXPR01	RXPR00	
RFPR0	RFPR015	RFPR014	RFPR013	RFPR012	RFPR011	RFPR010	RFPR09	RFPR08	
	RFPR07	RFPR06	RFPR05	RFPR04	RFPR03	RFPR02	RFPR01	RFPR00	
MBIMR0	MBIMR015	MBIMR014	MBIMR013	MBIMR012	MBIMR011	MBIMR010	MBIMR09	MBIMR08	
	MBIMR07	MBIMR06	MBIMR05	MBIMR04	MBIMR03	MBIMR02	MBIMR01	MBIMR00	
UMSR0	UMSR015	UMSR014	UMSR013	UMSR012	UMSR011	UMSR010	UMSR09	UMSR08	
	UMSR07	UMSR06	UMSR05	UMSR04	UMSR03	UMSR02	UMSR01	UMSR00	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MB[0]. CONTROL0H	IDE	RTR	—	STDID10	STDID9	STDID8	STDID7	STDID6	RCAN-ET (MCR15 = 0)
	STDID5	STDID4	STDID3	STDID2	STDID1	STDID0	EXTID17	EXTID16	
MB[0]. CONTROL0H	—	STDID10	STDID9	STDID8	STDID7	STDID6	STDID5	STDID4	RCAN-ET (MCR15 = 1)
	STDID3	STDID2	STDID1	STDID0	RTR	IDE	EXTID17	EXTID16	
MB[0]. CONTROL0L	EXTID15	EXTID14	EXTID13	EXTID12	EXTID11	EXTID10	EXTID9	EXTID8	RCAN-ET
	EXTID7	EXTID6	EXTID5	EXTID4	EXTID3	EXTID2	EXTID1	EXTID0	
MB[0]. LAFMH	IED_LAFM	—	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	RCAN-ET (MCR15 = 0)
	STDID_ LAFM5	STDID_ LAFM4	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	EXTID_LAF M17	EXTID_LAF M16	
MB[0]. LAFMH	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	STDID_ LAFM5	STDID_ LAFM4	RCAN-ET (MCR15 = 0)
	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	—	IED_LAFM	EXTID_LAF M17	EXTID_LAF M16	
MB[0]. LAFML	EXTID_ LAFM15	EXTID_ LAFM14	EXTID_ LAFM13	EXTID_ LAFM12	EXTID_ LAFM11	EXTID_ LAFM10	EXTID_ LAFM9	EXTID_ LAFM8	RCAN-ET
	EXTID_ LAFM7	EXTID_ LAFM6	EXTID_ LAFM5	EXTID_ LAFM4	EXTID_ LAFM3	EXTID_ LAFM2	EXTID_ LAFM1	EXTID_ LAFM0	
MB[0]. MSG_DATA[0]	MSG_DATA0								
MB[0]. MSG_DATA[1]	MSG_DATA1								
MB[0]. MSG_DATA[2]	MSG_DATA2								
MB[0]. MSG_DATA[3]	MSG_DATA3								
MB[0]. MSG_DATA[4]	MSG_DATA4								
MB[0]. MSG_DATA[5]	MSG_DATA5								
MB[0]. MSG_DATA[6]	MSG_DATA6								
MB[0]. MSG_DATA[7]	MSG_DATA7								
MB[0]. CONTROL1H	—	—	NMC	—	—	MBC2	MBC1	MBC0	
MB[0]. CONTROL1L	—	—	—	—	DLC3	DLC2	DLC1	DLC0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MB[1]. CONTROL0H	IDE	RTR	—	STDID10	STDID9	STDID8	STDID7	STDID6	RCAN-ET (MCR15 = 0)
	STDID5	STDID4	STDID3	STDID2	STDID1	STDID0	EXTID17	EXTID16	
MB[1]. CONTROL0H	—	STDID10	STDID9	STDID8	STDID7	STDID6	STDID5	STDID4	RCAN-ET (MCR15 = 1)
	STDID3	STDID2	STDID1	STDID0	RTR	IDE	EXTID17	EXTID16	
MB[1]. CONTROL0L	EXTID15	EXTID14	EXTID13	EXTID12	EXTID11	EXTID10	EXTID9	EXTID8	RCAN-ET
	EXTID7	EXTID6	EXTID5	EXTID4	EXTID3	EXTID2	EXTID1	EXTID0	
MB[1]. LAFMH	IED_LAFM	—	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	RCAN-ET (MCR15 = 0)
	STDID_ LAFM5	STDID_ LAFM4	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	EXTID_LAF M17	EXTID_LAF M16	
MB[1]. LAFMH	—	STDID_ LAFM10	STDID_ LAFM9	STDID_ LAFM8	STDID_ LAFM7	STDID_ LAFM6	STDID_ LAFM5	STDID_ LAFM4	RCAN-ET (MCR15 = 0)
	STDID_ LAFM3	STDID_ LAFM2	STDID_ LAFM1	STDID_ LAFM0	—	IED_LAFM	EXTID_LAF M17	EXTID_LAF M16	
MB[1]. LAFML	EXTID_ LAFM15	EXTID_ LAFM14	EXTID_ LAFM13	EXTID_ LAFM12	EXTID_ LAFM11	EXTID_ LAFM10	EXTID_ LAFM9	EXTID_ LAFM8	RCAN-ET
	EXTID_ LAFM7	EXTID_ LAFM6	EXTID_ LAFM5	EXTID_ LAFM4	EXTID_ LAFM3	EXTID_ LAFM2	EXTID_ LAFM1	EXTID_ LAFM0	
MB[1]. MSG_DATA[0]	MSG_DATA0								
MB[1]. MSG_DATA[1]	MSG_DATA1								
MB[1]. MSG_DATA[2]	MSG_DATA2								
MB[1]. MSG_DATA[3]	MSG_DATA3								
MB[1]. MSG_DATA[4]	MSG_DATA4								
MB[1]. MSG_DATA[5]	MSG_DATA5								
MB[1]. MSG_DATA[6]	MSG_DATA6								
MB[1]. MSG_DATA[7]	MSG_DATA7								
MB[1]. CONTROL1H	—	—	NMC	ATX	DART	MBC2	MBC1	MBC0	
MB[1]. CONTROL1L	—	—	—	—	DLC3	DLC2	DLC1	DLC0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
MB[2]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration								
MB[3]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration								
↓	(Repeated)								
MB[13]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration								
MB[14]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration								
MB[15]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration								
RCANMON	—	CTxSTP	RCANE	—	—	—	CTxD	CRxD	
SSCRH_0	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0	SSU_0
SSCRL_0	—	SSUMS	SRES	—	—	—	DATS1	DATS0	
SSMR_0	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0	
SSER_0	TE	RE	—	—	TEIE	TIE	RIE	CEIE	
SSSR_0	—	ORER	—	—	TEND	TDRE	RDRF	CE	
SSCR2_0	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—	
SSTDR0_0									
SSTDR1_0									
SSTDR2_0									
SSTDR3_0									
SSRDR0_0									
SSRDR1_0									
SSRDR2_0									
SSRDR3_0									
SSCRH_1	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0	SSU_1
SSCRL_1	—	SSUMS	SRES	—	—	—	DATS1	DATS0	
SSMR_1	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0	
SSER_1	TE	RE	—	—	TEIE	TIE	RIE	CEIE	
SSSR_1	—	ORER	—	—	TEND	TDRE	RDRF	CE	
SSCR2_1	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SSTDR0_1									SSU_1
SSTDR1_1									
SSTDR2_1									
SSTDR3_1									
SSRDR0_1									
SSRDR1_1									
SSRDR2_1									
SSRDR3_1									
SSCRH_2	MSS	BIDE	—	SOL	SOLP	SCKS	CSS1	CSS0	SSU_2
SSCRL_2	—	SSUMS	SRES	—	—	—	DATS1	DATS0	
SSMR_2	MLS	CPOS	CPHS	—	—	CKS2	CKS1	CKS0	
SSER_2	TE	RE	—	—	TEIE	TIE	RIE	CEIE	
SSSR_2	—	ORER	—	—	TEND	TDRE	RDRF	CE	
SSCR2_2	SDOS	SSCKOS	SCSOS	TENDSTS	SCSATS	SSODTS	—	—	
SSTDR0_2									
SSTDR1_2									
SSTDR2_2									
SSTDR3_2									
SSRDR0_2									
SSRDR1_2									
SSRDR2_2									
SSRDR3_2									
PHRTIDR	PHRTIDR7	PHRTIDR6	PHRTIDR5	PHRTIDR4	PHRTIDR3	PHRTIDR2	PHRTIDR1	PHRTIDR0	I/O port
ADDRA_1									A/D_1
			—	—	—	—	—	—	
ADDRB_1									
			—	—	—	—	—	—	
ADDRC_1									
			—	—	—	—	—	—	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
ADDRD_1									A/D_1
			—	—	—	—	—	—	
ADDRE_1									
			—	—	—	—	—	—	
ADDRF_1									
			—	—	—	—	—	—	
ADDRG_1									
			—	—	—	—	—	—	
ADDRH_1									
			—	—	—	—	—	—	
ADCSR_1	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR_1	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—	
TSTRB	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU
TSYRB	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0	
TCR_6	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_6
TMDR_6	—	—	BFB	BFA	MD3	MD2	MD1	MD0	
TIORH_6	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_6	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_6	—	—	TCIEU	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_6	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_6									
TGRA_6									
TGRB_6									
TGRC_6									
TGRD_6									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_7	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_7
TMDR_7	—	—	—	—	—	MD2	MD1	MD0	
TIOR_7	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_7	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_7	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_7									
TGRA_7									
TGRB_7									
TCR_8	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_8
TMDR_8	—	—	—	—	—	MD2	MD1	MD0	
TIOR_8	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_8	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_8	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_8									
TGRA_8									
TGRB_8									
TCR_9	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_9
TMDR_9	—	—	BFB	BFA	—	MD2	MD1	MD0	
TIORH_9	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_9	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_9	—	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_9	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCNT_9									TPU_9
TGRA_9									
TGRB_9									
TGRC_9									
TGRD_9									
TCR_10	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_10
TMDR_10	—	—	—	—	—	MD2	MD1	MD0	
TIOR_10	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_10	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_10	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_10									
TGRA_10									
TGRB_10									
TCR_11	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_11
TMDR_11	—	—	—	—	—	MD2	MD1	MD0	
TIOR_11	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_11	—	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_11	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_11									
TGRA_11									
TGRB_11									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR	I/O port
P2DDR	—	—	—	—	P23DDR	P22DDR	P21DDR	P20DDR	
P3DDR	P37DDR	P36DDR	P35DDR	P34DDR	P33DDR	P32DDR	P31DDR	P30DDR	
P6DDR	—	P66DDR	P65DDR	P64DDR	P63DDR	P62DDR	P61DDR	P60DDR	
PADDR	PA7DDR	PA6DDR	PA5DDR	PA4DDR	PA3DDR	PA2DDR	PA1DDR	—	
PDDDR	PD7DDR	PD6DDR	PD5DDR	PD4DDR	PD3DDR	PD2DDR	PD1DDR	PD0DDR	
P1ICR	P17ICR	P16ICR	P15ICR	P14ICR	P13ICR	P12ICR	P11ICR	P10ICR	
P2ICR	—	—	—	—	P23ICR	P22ICR	P21ICR	P20ICR	
P3ICR	P37ICR	P36ICR	P35ICR	P34ICR	P33ICR	P32ICR	P31ICR	P30ICR	
P4ICR	P47ICR	P46ICR	P45ICR	P44ICR	P43ICR	P42ICR	P41ICR	P40ICR	
P5ICR	P57ICR	P56ICR	P55ICR	P54ICR	P53ICR	P52ICR	P51ICR	P50ICR	
P6ICR	—	P66ICR	P65ICR	P64ICR	P63ICR	P62ICR	P61ICR	P60ICR	
PAICR	PA7ICR	PA6ICR	PA5ICR	PA4ICR	PA3ICR	PA2ICR	PA1ICR	—	
PDICR	PD7ICR	PD6ICR	PD5ICR	PD4ICR	PD3ICR	PD2ICR	PD1ICR	PD0ICR	
PORTH	PH7	PH6	PH5	PH4	PH3	PH2	PH1	PH0	
PORTJ	PJ7	PJ6	PJ5	PJ4	PJ3	PJ2	PJ1	PJ0	
PORTK	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0	
PHDR	PH7DR	PH6DR	PH5DR	PH4DR	PH3DR	PH2DR	PH1DR	PH0DR	
PJDR	PJ7DR	PJ6DR	PJ5DR	PJ4DR	PJ3DR	PJ2DR	PJ1DR	PJ0DR	
PKDR	PK7DR	PK6DR	PK5DR	PK4DR	PK3DR	PK2DR	PK1DR	PK0DR	
PHDDR	PH7DDR	PH6DDR	PH5DDR	PH4DDR	PH3DDR	PH2DDR	PH1DDR	PH0DDR	
PJDDR	PJ7DDR	PJ6DDR	PJ5DDR	PJ4DDR	PJ3DDR	PJ2DDR	PJ1DDR	PJ0DDR	
PKDDR	PK7DDR	PK6DDR	PK5DDR	PK4DDR	PK3DDR	PK2DDR	PK1DDR	PK0DDR	
PHICR	PH7ICR	PH6ICR	PH5ICR	PH4ICR	PH3ICR	PH2ICR	PH1ICR	PH0ICR	
PJICR	PJ7ICR	PJ6ICR	PJ5ICR	PJ4ICR	PJ3ICR	PJ2ICR	PJ1ICR	PJ0ICR	
PKICR	PK7ICR	PK6ICR	PK5ICR	PK4ICR	PK3ICR	PK2ICR	PK1ICR	PK0ICR	
PDPCR	PD7PCR	PD6PCR	PD5PCR	PD4PCR	PD3PCR	PD2PCR	PD1PCR	PD0PCR	
PHPCR	PH7PCR	PH6PCR	PH5PCR	PH4PCR	PH3PCR	PH2PCR	PH1PCR	PH0PCR	
PJPCR	PJ7PCR	PJ6PCR	PJ5PCR	PJ4PCR	PJ3PCR	PJ2PCR	PJ1PCR	PJ0PCR	
PKPCR	PK7PCR	PK6PCR	PK5PCR	PK4PCR	PK3PCR	PK2PCR	PK1PCR	PK0PCR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
P2ODR	—	—	—	—	P23ODR	P22ODR	P21ODR	P20ODR	I/O port
PFCR9	—	—	TPUMS3A	TPUMS3B	TPUMS2	TPUMS1	TPUMS0A	TPUMS0B	
PFCRA	TPUMS11	TPUMS10	TPUMS9A	TPUMS9B	TPUMS8	TPUMS7	TPUMS6A	TPUMS6B	
PFCRB	—	ITS14	ITS13	ITS12	ITS11	ITS10	ITS9	ITS8	
SSIER	—	SSI14	SSI13	SSI12	SSI11	SSI10	SSI9	SSI8	INTC
	SSI7	SSI6	SSI5	SSI4	SSI3	SSI2	SSI1	SSI0	
DSAR_0									DMAC_0
DDAR_0									
DOFR_0									
DTCR_0									
DBSR_0	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_0	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	ERRF	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DACR_0	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	DMAC_0
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_1									DMAC_1
DDAR_1									
DOFR_1									
DTCR_1									
DBSR_1	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_1	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DACR_1	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	DMAC_1
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_2									DMAC_2
DDAR_2									
DOFR_2									
DTCR_2									
DBSR_2	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_2	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DACR_2	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	DMAC_2
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DSAR_3									DMAC_3
DDAR_3									
DOFR_3									
DTCR_3									
DBSR_3	BKSZH31	BKSZH30	BKSZH29	BKSZH28	BKSZH27	BKSZH26	BKSZH25	BKSZH24	
	BKSZH23	BKSZH22	BKSZH21	BKSZH20	BKSZH19	BKSZH18	BKSZH17	BKSZH16	
	BKSZ15	BKSZ14	BKSZ13	BKSZ12	BKSZ11	BKSZ10	BKSZ9	BKSZ8	
	BKSZ7	BKSZ6	BKSZ5	BKSZ4	BKSZ3	BKSZ2	BKSZ1	BKSZ0	
DMDR_3	DTE	DACKE	TENDE	—	DREQS	NRD	—	—	
	ACT	—	—	—	—	—	ESIF	DTIF	
	DTSZ1	DTSZ0	MDS1	MDS0	TSEIE	—	ESIE	DTIE	
	DTF1	DTF0	DTA	—	—	DMAP2	DMAP1	DMAP0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
DACR_3	AMS	DIRS	—	—	—	RPTIE	ARS1	ARS0	DMAC_3
	—	—	SAT1	SAT0	—	—	DAT1	DAT0	
	SARIE	—	—	SARA4	SARA3	SARA2	SARA1	SARA0	
	DARIE	—	—	DARA4	DARA3	DARA2	DARA1	DARA0	
DMRSR_0									DMAC_0
DMRSR_1									DMAC_1
DMRSR_2									DMAC_2
DMRSR_3									DMAC_3
IPRA	—	IPRA14	IPRA13	IPRA12	—	IPRA10	IPRA9	IPRA8	INTC
	—	IPRA6	IPRA5	IPRA4	—	IPRA2	IPRA1	IPRA0	
IPRB	—	IPRB14	IPRB13	IPRB12	—	IPRB10	IPRB9	IPRB8	
	—	IPRB6	IPRB5	IPRB4	—	IPRB2	IPRB1	IPRB0	
IPRC	—	IPRC14	IPRC13	IPRC12	—	IPRC10	IPRC9	IPRC8	INTC
	—	IPRC6	IPRC5	IPRC4	—	IPRC2	IPRC1	IPRC0	
IPRD	—	IPRD14	IPRD13	IPRD12	—	IPRD10	IPRD9	IPRD8	
	—	IPRD6	IPRD5	IPRD4	—	—	—	—	
IPRE	—	—	—	—	—	IPRE10	IPRE9	IPRE8	
	—	—	—	—	—	—	—	—	
IPRF	—	—	—	—	—	IPRF10	IPRF9	IPRF8	
	—	IPRF6	IPRF5	IPRF4	—	IPRF2	IPRF1	IPRF0	
IPRG	—	IPRG14	IPRG13	IPRG12	—	IPRG10	IPRG9	IPRG8	
	—	IPRG6	IPRG5	IPRG4	—	IPRG2	IPRG1	IPRG0	
IPRI	—	IPRI14	IPRI13	IPRI12	—	IPRI10	IPRI9	IPRI8	
	—	IPRI6	IPRI5	IPRI4	—	IPRI2	IPRI1	IPRI0	
IPRK	—	IPRK14	IPRK13	IPRK12	—	—	—	—	
	—	—	—	—	—	—	—	—	
IPRL	—	—	—	—	—	IPRL10	IPRL9	IPRL8	
	—	IPRL6	IPRL5	IPRL4	—	IPRL2	IPRL1	IPRL0	
IPRM	—	IPRM14	IPRM13	IPRM12	—	IPRM10	IPRM9	IPRM8	
	—	IPRM6	IPRM5	IPRM4	—	IPRM2	IPRM1	IPRM0	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
IPRN	—	IPRN14	IPRN13	IPRN12	—	IPRN10	IPRN9	IPRN8	INTC
	—	IPRN6	IPRN5	IPRN4	—	IPRN2	IPRN1	IPRN0	
IPRO	—	IPRO14	IPRO13	IPRO12	—	IPRO10	IPRO9	IPRO8	
	—	IPRO6	IPRO5	IPRO4	—	—	—	—	
IPRQ	—	—	—	—	—	—	—	—	
	—	—	—	—	—	IPRQ2	IPRQ1	IPRQ0	
IPRR	—	IPRR14	IPRR13	IPRR12	—	IPRR10	IPRR9	IPRR8	
	—	IPRR6	IPRR5	IPRR4	—	IPRR2	IPRR1	IPRR0	
ISCRH	—	—	IRQ14SR	IRQ14SF	IRQ13SR	IRQ13SF	IRQ12SR	IRQ12SF	
	IRQ11SR	IRQ11SF	IRQ10SR	IRQ10SF	IRQ9SR	IRQ9SF	IRQ8SR	IRQ8SF	
ISCRL	IRQ7SR	IRQ7SF	IRQ6SR	IRQ6SF	IRQ5SR	IRQ5SF	IRQ4SR	IRQ4SF	
	IRQ3SR	IRQ3SF	IRQ2SR	IRQ2SF	IRQ1SR	IRQ1SF	IRQ0SR	IRQ0SF	
BCR2	—	—	—	IBCCS	—	—	—	PWDBE	BSC
RAMER	—	—	—	—	RAMS	RAM2	RAM1	RAM0	
MDCR	—	—	—	—	MDS3	MDS2	MDS1	MDS0	SYSTEM
	—	—	—	—	—	—	—	—	
SYSCR	—	—	MACS	—	—	—	—	RAME	
	FLSHE	—	—	—	—	—	—	—	
SCKCR	PSTOP1	—	POSEL1	—	—	ICK2	ICK1	ICK0	
	—	PCK2	PCK1	PCK0	—	BCK2	BCK1	BCK0	
SBYCR	SSBY	OPE	—	STS4	STS3	STS2	STS1	STS0	
	—	—	—	—	—	—	—	—	
MSTPCRA	ACSE	MSTPA14	MSTPA13	MSTPA12	MSTPA11	MSTPA10	MSTPA9	MSTPA8	
	MSTPA7	MSTPA6	MSTPA5	MSTPA4	MSTPA3	MSTPA2	MSTPA1	MSTPA0	
MSTPCRB	MSTPB15	MSTPB14	MSTPB13	MSTPB12	MSTPB11	MSTPB10	MSTPB9	MSTPB8	
	MSTPB7	MSTPB6	MSTPB5	MSTPB4	MSTPB3	MSTPB2	MSTPB1	MSTPB0	
MSTPCRC	MSTPC15	MSTPC14	MSTPC13	MSTPC12	MSTPC11	MSTPC10	MSTPC9	MSTPC8	
	MSTPC7	MSTPC6	MSTPC5	MSTPC4	MSTPC3	MSTPC2	MSTPC1	MSTPC0	
SMR_3*[1]	C/$\overline{A}$ (GM)	CHR (BLK)	PE	O/$\overline{E}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_3
BRR_3									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
SCR_3*[1]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	SCI_3
TDR_3									
SSR_3*[1]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_3									
SCMR_3	—	—	—	—	SDIR	SINV	—	SMIF	
SMR_4*[1]	C/$\overline{A}$ (GM)	CHR (BLK)	PE	O/$\overline{E}$	STOP (BCP1)	MP (BCP0)	CKS1	CKS0	SCI_4
BRR_4									
SCR_4*[1]	TIE	RIE	TE	RE	MPIE	TEIE	CKE1	CKE0	
TDR_4									
SSR_4*[1]	TDRE	RDRF	ORER	FER (ERS)	PER	TEND	MPB	MPBT	
RDR_4									
SCMR_4	—	—	—	—	SDIR	SINV	—	SMIF	
FCCS	—	—	—	FLER	—	—	—	SCO	FLASH
FPCS	—	—	—	—	—	—	—	PPVS	
FECS	—	—	—	—	—	—	—	EPVB	
FKEY	K7	K6	K5	K4	K3	K2	K1	K0	
FMATS	MS7	MS6	MS5	MS4	MS3	MS2	MS1	MS0	
FTDAR	TDER	TDA6	TDA5	TDA4	TDA3	TDA2	TDA1	TDA0	
TCR_4	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_4*[2]
TMDR_4	—	—	—	—	—	MD2	MD1	MD0	
TIOR_4	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_4	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_4	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_4									
TGRA_4									
TGRB_4									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_5	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_5*[2]
TMDR_5	—	—	—	—	—	MD2	MD1	MD0	
TIOR_5	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_5	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_5	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_5									
TGRA_5									
TGRB_5									
INTCR	—	—	INTM1	INTM0	NMIEG	—	—	—	INTC
CPUPCR	CPUPCE	—	—	—	IPSETE	CPUP2	CPUP1	CPUP0	
IER	—	IRQ14E	IRQ13E	IRQ12E	IRQ11E	IRQ10E	IRQ9E	IRQ8E	
	IRQ7E	IRQ6E	IRQ5E	IRQ4E	IRQ3E	IRQ2E	IRQ1E	IRQ0E	
ISR	—	IRQ14F	IRQ13F	IRQ12F	IRQ11F	IRQ10F	IRQ9F	IRQ8F	
	IRQ7F	IRQ6F	IRQ5F	IRQ4F	IRQ3F	IRQ2F	IRQ1F	IRQ0F	
PORT1	P17	P16	P15	P14	P13	P12	P11	P10	I/O port
PORT2	—	—	—	—	P23	P22	P21	P20	
PORT3	P37	P36	P35	P34	P33	P32	P31	P30	
PORT4	P47	P46	P45	P44	P43	P42	P41	P40	
PORT5	P57	P56	P55	P54	P53	P52	P51	P50	
PORT6	—	P66	P65	P64	P63	P62	P61	P60	
PORTA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	—	
PORTD	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0	
P1DR	P17DR	P16DR	P15DR	P14DR	P13DR	P12DR	P11DR	P10DR	
P2DR	—	—	—	—	P23DR	P22DR	P21DR	P20DR	
P3DR	P37DR	P36DR	P35DR	P34DR	P33DR	P32DR	P31DR	P30DR	
P6DR	—	P66DR	P65DR	P64DR	P63DR	P62DR	P61DR	P60DR	
PADR	PA7DR	PA6DR	PA5DR	PA4DR	PA3DR	PA2DR	PA1DR	—	
PDDR	PD7DR	PD6DR	PD5DR	PD4DR	PD3DR	PD2DR	PD1DR	PD0DR	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
PCR	G3CMS1	G3CMS0	G2CMS1	G2CMS0	—	—	—	—	PPG*[2]
PMR	G3INV	G2INV	—	—	G3NOV	G2NOV	—	—	
NDERH	NDER15	NDER14	NDER13	NDER12	NDER11	NDER10	NDER9	NDER8	
NDERL	NDER7	NDER6	NDER5	NDER4	NDER3	NDER2	NDER1	NDER0	
PODRH	POD15	POD14	POD13	POD12	POD11	POD10	POD9	POD8	
PODRL	POD7	POD6	POD5	POD4	POD3	POD2	POD1	POD0	
NDRH	NDR15	NDR14	NDR13	NDR12	NDR11	NDR10	NDR9	NDR8	
NDRL	NDR7	NDR6	NDR5	NDR4	NDR3	NDR2	NDR1	NDR0	
ADDRA_0									A/D_0
			—	—	—	—	—	—	
ADDRB_0									
			—	—	—	—	—	—	
ADDRC_0									
			—	—	—	—	—	—	
ADDRD_0									
			—	—	—	—	—	—	
ADDRE_0									
			—	—	—	—	—	—	
ADDRF_0									
			—	—	—	—	—	—	
ADDRG_0									
			—	—	—	—	—	—	
ADDRH_0									
			—	—	—	—	—	—	
ADCSR_0	ADF	ADIE	ADST	—	CH3	CH2	CH1	CH0	
ADCR_0	TRGS1	TRGS0	SCANE	SCANS	CKS1	CKS0	—	—	
TCSR	OVF	WT/IT	TME	—	—	CKS2	CKS1	CKS0	WDT
TCNT									
RSTCSR	WOVF	RSTE	—	—	—	—	—	—	

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TSTR	—	—	CST5	CST4	CST3	CST2	CST1	CST0	TPU*2
TSYR	—	—	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0	
TCR_0	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_0*2
TMDR_0	—	—	BFB	BFA	—	MD2	MD1	MD0	
TIORH_0	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_0	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_0	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_0	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_0									
TGRA_0									
TGRB_0									
TGRC_0									
TGRD_0									
TCR_1	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_1*2
TMDR_1	—	—	—	—	—	MD2	MD1	MD0	
TIOR_1	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_1	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_1	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_1									
TGRA_1									
TGRB_1									

Register Abbreviation	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0	Module
TCR_2	—	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_2*[2]
TMDR_2	—	—	—	—	—	MD2	MD1	MD0	
TIOR_2	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIER_2	TTGE	—	TCIEU	TCIEV	—	—	TGIEB	TGIEA	
TSR_2	TCFD	—	TCFU	TCFV	—	—	TGFB	TGFA	
TCNT_2									
TGRA_2									
TGRB_2									
TCR_3	CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC1	TPSC0	TPU_3*[2]
TMDR_3	—	—	BFB	BFA	—	MD2	MD1	MD0	
TIORH_3	IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0	
TIORL_3	IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0	
TIER_3	TTGE	—	—	TCIEV	TGIED	TGIEC	TGIEB	TGIEA	
TSR_3	—	—	—	TCFV	TGFD	TGFC	TGFB	TGFA	
TCNT_3									
TGRA_3									
TGRB_3									
TGRC_3									
TGRD_3									

Notes: 1. Parts of the bit functions differ in normal mode and the smart card interface mode.
2. Supported only by the H8SX/1527R.

20.3 Register States in Each Operating Mode

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Module
MCR	Initialized	—	Initialized	Initialized	Initialized	RCAN-ET
GSR	Initialized	—	Initialized	Initialized	Initialized	
BCR1	Initialized	—	Initialized	Initialized	Initialized	
BCR0	Initialized	—	Initialized	Initialized	Initialized	
IRR	Initialized	—	Initialized	Initialized	Initialized	
IMR	Initialized	—	Initialized	Initialized	Initialized	
TEC	Initialized	—	Initialized	Initialized	Initialized	
REC	Initialized	—	Initialized	Initialized	Initialized	
TXPR1	Initialized	—	Initialized	Initialized	Initialized	
TXPR0	Initialized	—	Initialized	Initialized	Initialized	
TXCR0	Initialized	—	Initialized	Initialized	Initialized	
TXACK0	Initialized	—	Initialized	Initialized	Initialized	
ABACK0	Initialized	—	Initialized	Initialized	Initialized	
RXPR0	Initialized	—	Initialized	Initialized	Initialized	
RFPR0	Initialized	—	Initialized	Initialized	Initialized	
MBIMR0	Initialized	—	Initialized	Initialized	Initialized	
UMSR0	Initialized	—	Initialized	Initialized	Initialized	
MB[0]. CONTROL0H	—	—	—	—	—	
MB[0]. CONTROL0L	—	—	—	—	—	
MB[0]. LAFMH	—	—	—	—	—	
MB[0]. LAFML	—	—	—	—	—	
MB[0]. MSG_DATA[0]	—	—	—	—	—	
MB[0]. MSG_DATA[1]	—	—	—	—	—	
MB[0]. MSG_DATA[2]	—	—	—	—	—	
MB[0]. MSG_DATA[3]	—	—	—	—	—	
MB[0]. MSG_DATA[4]	—	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
MB[0]. MSG_DATA[5]	—	—	—	—	—	RCAN-ET
MB[0]. MSG_DATA[6]	—	—	—	—	—	
MB[0]. MSG_DATA[7]	—	—	—	—	—	
MB[0]. CONTROL1H	Initialized	—	Initialized	Initialized	Initialized	
MB[0]. CONTROL1L	Initialized	—	Initialized	Initialized	Initialized	
MB[1]. CONTROL0H	—	—	—	—	—	
MB[1]. CONTROL0L	—	—	—	—	—	
MB[1]. LAFMH	—	—	—	—	—	
MB[1]. LAFML	—	—	—	—	—	
MB[1]. MSG_DATA[0]	—	—	—	—	—	
MB[1]. MSG_DATA[1]	—	—	—	—	—	
MB[1]. MSG_DATA[2]	—	—	—	—	—	
MB[1]. MSG_DATA[3]	—	—	—	—	—	
MB[1]. MSG_DATA[4]	—	—	—	—	—	
MB[1]. MSG_DATA[5]	—	—	—	—	—	
MB[1]. MSG_DATA[6]	—	—	—	—	—	
MB[1]. MSG_DATA[7]	—	—	—	—	—	
MB[1]. CONTROL1H	Initialized	—	Initialized	Initialized	Initialized	
MB[1]. CONTROL1L	Initialized	—	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
MB[2]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration					RCAN-ET
MB[3]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration					
↓	(Repeated)					
MB[13]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration					
MB[14]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration					
MB[15]	Same as MB[1].CONTROL0H to MB[1].CONTROL1L bit configuration					
RCANMON	Initialized	—	Initialized	Initialized	Initialized	
SSCRH_0	Initialized	—	Initialized	Initialized	Initialized	SSU_0
SSCRL_0	Initialized	—	Initialized	Initialized	Initialized	
SSMR_0	Initialized	—	Initialized	Initialized	Initialized	
SSER_0	Initialized	—	Initialized	Initialized	Initialized	
SSSR_0	Initialized	—	Initialized	Initialized	Initialized	
SSCR2_0	Initialized	—	Initialized	Initialized	Initialized	
SSTDR0_0	Initialized	—	Initialized	Initialized	Initialized	
SSTDR1_0	Initialized	—	Initialized	Initialized	Initialized	
SSTDR2_0	Initialized	—	Initialized	Initialized	Initialized	
SSTDR3_0	Initialized	—	Initialized	Initialized	Initialized	
SSRDR0_0	Initialized	—	Initialized	Initialized	Initialized	
SSRDR1_0	Initialized	—	Initialized	Initialized	Initialized	
SSRDR2_0	Initialized	—	Initialized	Initialized	Initialized	
SSRDR3_0	Initialized	—	Initialized	Initialized	Initialized	
SSCRH_1	Initialized	—	Initialized	Initialized	Initialized	SSU_1
SSCRL_1	Initialized	—	Initialized	Initialized	Initialized	
SSMR_1	Initialized	—	Initialized	Initialized	Initialized	
SSER_1	Initialized	—	Initialized	Initialized	Initialized	
SSSR_1	Initialized	—	Initialized	Initialized	Initialized	
SSCR2_1	Initialized	—	Initialized	Initialized	Initialized	
SSTDR0_1	Initialized	—	Initialized	Initialized	Initialized	
SSTDR1_1	Initialized	—	Initialized	Initialized	Initialized	
SSTDR2_1	Initialized	—	Initialized	Initialized	Initialized	
SSTDR3_1	Initialized	—	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
SSRDR0_1	Initialized	—	Initialized	Initialized	Initialized	SSU_1
SSRDR1_1	Initialized	—	Initialized	Initialized	Initialized	
SSRDR2_1	Initialized	—	Initialized	Initialized	Initialized	
SSRDR3_1	Initialized	—	Initialized	Initialized	Initialized	
SSCRH_2	Initialized	—	Initialized	Initialized	Initialized	SSU_2
SSCRL_2	Initialized	—	Initialized	Initialized	Initialized	
SSMR_2	Initialized	—	Initialized	Initialized	Initialized	
SSER_2	Initialized	—	Initialized	Initialized	Initialized	
SSSR_2	Initialized	—	Initialized	Initialized	Initialized	
SSCR2_2	Initialized	—	Initialized	Initialized	Initialized	
SSTDR0_2	Initialized	—	Initialized	Initialized	Initialized	
SSTDR1_2	Initialized	—	Initialized	Initialized	Initialized	
SSTDR2_2	Initialized	—	Initialized	Initialized	Initialized	
SSTDR3_2	Initialized	—	Initialized	Initialized	Initialized	
SSRDR0_2	Initialized	—	Initialized	Initialized	Initialized	
SSRDR1_2	Initialized	—	Initialized	Initialized	Initialized	
SSRDR2_2	Initialized	—	Initialized	Initialized	Initialized	
SSRDR3_2	Initialized	—	Initialized	Initialized	Initialized	
PHRTIDR	Initialized	—	—	—	—	I/O port
ADDRA_1	Initialized	—	—	—	—	A/D_1
ADDRB_1	Initialized	—	—	—	—	
ADDRC_1	Initialized	—	—	—	—	
ADDRD_1	Initialized	—	—	—	—	
ADDRE_1	Initialized	—	—	—	—	
ADDRF_1	Initialized	—	—	—	—	
ADDRG_1	Initialized	—	—	—	—	
ADDRH_1	Initialized	—	—	—	—	
ADCSR_1	Initialized	—	—	—	—	
ADCR_1	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
TSTRB	Initialized	—	—	—	—	TPU
TSYRB	Initialized	—	—	—	—	
TCR_6	Initialized	—	—	—	—	TPU_6
TMDR_6	Initialized	—	—	—	—	
TIORH_6	Initialized	—	—	—	—	
TIORL_6	Initialized	—	—	—	—	
TIER_6	Initialized	—	—	—	—	
TSR_6	Initialized	—	—	—	—	
TCNT_6	Initialized	—	—	—	—	
TGRA_6	Initialized	—	—	—	—	
TGRB_6	Initialized	—	—	—	—	
TGRC_6	Initialized	—	—	—	—	
TGRD_6	Initialized	—	—	—	—	
TCR_7	Initialized	—	—	—	—	TPU_7
TMDR_7	Initialized	—	—	—	—	
TIOR_7	Initialized	—	—	—	—	
TIER_7	Initialized	—	—	—	—	
TSR_7	Initialized	—	—	—	—	
TCNT_7	Initialized	—	—	—	—	
TGRA_7	Initialized	—	—	—	—	
TGRB_7	Initialized	—	—	—	—	
TCR_8	Initialized	—	—	—	—	TPU_8
TMDR_8	Initialized	—	—	—	—	
TIOR_8	Initialized	—	—	—	—	
TIER_8	Initialized	—	—	—	—	
TSR_8	Initialized	—	—	—	—	
TCNT_8	Initialized	—	—	—	—	
TGRA_8	Initialized	—	—	—	—	
TGRB_8	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
TCR_9	Initialized	—	—	—	—	TPU_9
TMDR_9	Initialized	—	—	—	—	
TIORH_9	Initialized	—	—	—	—	
TIORL_9	Initialized	—	—	—	—	
TIER_9	Initialized	—	—	—	—	
TSR_9	Initialized	—	—	—	—	
TCNT_9	Initialized	—	—	—	—	
TGRA_9	Initialized	—	—	—	—	
TGRB_9	Initialized	—	—	—	—	
TGRC_9	Initialized	—	—	—	—	
TGRD_9	Initialized	—	—	—	—	
TCR_10	Initialized	—	—	—	—	TPU_10
TMDR_10	Initialized	—	—	—	—	
TIOR_10	Initialized	—	—	—	—	
TIER_10	Initialized	—	—	—	—	
TSR_10	Initialized	—	—	—	—	
TCNT_10	Initialized	—	—	—	—	
TGRA_10	Initialized	—	—	—	—	
TGRB_10	Initialized	—	—	—	—	
TCR_11	Initialized	—	—	—	—	TPU_11
TMDR_11	Initialized	—	—	—	—	
TIOR_11	Initialized	—	—	—	—	
TIER_11	Initialized	—	—	—	—	
TSR_11	Initialized	—	—	—	—	
TCNT_11	Initialized	—	—	—	—	
TGRA_11	Initialized	—	—	—	—	
TGRB_11	Initialized	—	—	—	—	
P1DDR	Initialized	—	—	—	—	I/O port
P2DDR	Initialized	—	—	—	—	
P3DDR	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
P6DDR	Initialized	—	—	—	—	I/O port
PADDR	Initialized	—	—	—	—	
PDDDR	Initialized	—	—	—	—	
P1ICR	Initialized	—	—	—	—	
P2ICR	Initialized	—	—	—	—	
P3ICR	Initialized	—	—	—	—	
P4ICR	Initialized	—	—	—	—	
P5ICR	Initialized	—	—	—	—	
P6ICR	Initialized	—	—	—	—	
PAICR	Initialized	—	—	—	—	
PDICR	Initialized	—	—	—	—	
PORTH	—	—	—	—	—	
PORTJ	—	—	—	—	—	
PORTK	—	—	—	—	—	
PHDR	Initialized	—	—	—	—	
PJDR	Initialized	—	—	—	—	
PKDR	Initialized	—	—	—	—	
PHDDR	Initialized	—	—	—	—	
PJDDR	Initialized	—	—	—	—	
PKDDR	Initialized	—	—	—	—	
PHICR	Initialized	—	—	—	—	
PJICR	Initialized	—	—	—	—	
PKICR	Initialized	—	—	—	—	
PDPCR	Initialized	—	—	—	—	
PHPCR	Initialized	—	—	—	—	
PJPCR	Initialized	—	—	—	—	
PKPCR	Initialized	—	—	—	—	
P2ODR	Initialized	—	—	—	—	
PFCR9	Initialized	—	—	—	—	
PFCRA	Initialized	—	—	—	—	
PFCRB	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
SSIER	Initialized	—	—	—	—	INTC
DSAR_0	Initialized	—	—	—	—	DMAC_0
DDAR_0	Initialized	—	—	—	—	
DOFR_0	Initialized	—	—	—	—	
DTCR_0	Initialized	—	—	—	—	
DBSR_0	Initialized	—	—	—	—	
DMDR_0	Initialized	—	—	—	—	
DACR_0	Initialized	—	—	—	—	
DSAR_1	Initialized	—	—	—	—	DMAC_1
DDAR_1	Initialized	—	—	—	—	
DOFR_1	Initialized	—	—	—	—	
DTCR_1	Initialized	—	—	—	—	
DBSR_1	Initialized	—	—	—	—	
DMDR_1	Initialized	—	—	—	—	
DACR_1	Initialized	—	—	—	—	
DSAR_2	Initialized	—	—	—	—	DMAC_2
DDAR_2	Initialized	—	—	—	—	
DOFR_2	Initialized	—	—	—	—	
DTCR_2	Initialized	—	—	—	—	
DBSR_2	Initialized	—	—	—	—	
DMDR_2	Initialized	—	—	—	—	
DACR_2	Initialized	—	—	—	—	
DSAR_3	Initialized	—	—	—	—	DMAC_3
DDAR_3	Initialized	—	—	—	—	
DOFR_3	Initialized	—	—	—	—	
DTCR_3	Initialized	—	—	—	—	
DBSR_3	Initialized	—	—	—	—	
DMDR_3	Initialized	—	—	—	—	
DACR_3	Initialized	—	—	—	—	
DMRSR_0	Initialized	—	—	—	—	DMAC_0
DMRSR_1	Initialized	—	—	—	—	DMAC_1
DMRSR_2	Initialized	—	—	—	—	DMAC_2
DMRSR_3	Initialized	—	—	—	—	DMAC_3

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
IPRA	Initialized	—	—	—	—	INTC
IPRB	Initialized	—	—	—	—	
IPRC	Initialized	—	—	—	—	
IPRD	Initialized	—	—	—	—	
IPRE	Initialized	—	—	—	—	
IPRF	Initialized	—	—	—	—	
IPRG	Initialized	—	—	—	—	
IPRI	Initialized	—	—	—	—	
IPRK	Initialized	—	—	—	—	
IPRL	Initialized	—	—	—	—	
IPRM	Initialized	—	—	—	—	
IPRN	Initialized	—	—	—	—	
IPRO	Initialized	—	—	—	—	
IPRQ	Initialized	—	—	—	—	
IPRR	Initialized	—	—	—	—	
ISCRH	Initialized	—	—	—	—	
ISCRL	Initialized	—	—	—	—	
BCR2	Initialized	—	—	—	—	BSC
RAMER	Initialized	—	—	—	—	
MDCR	Initialized	—	—	—	—	SYSTEM
SYSCR	Initialized	—	—	—	—	
SCKCR	Initialized	—	—	—	—	
SBYCR	Initialized	—	—	—	—	
MSTPCRA	Initialized	—	—	—	—	
MSTPCRB	Initialized	—	—	—	—	
MSTPCRC	Initialized	—	—	—	—	
SMR_3	Initialized	—	—	—	—	SCI_3
BRR_3	Initialized	—	—	—	—	
SCR_3	Initialized	—	—	—	—	
TDR_3	Initialized	—	Initialized	Initialized	Initialized	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
SSR_3	Initialized	—	Initialized	Initialized	Initialized	SCI_3
RDR_3	Initialized	—	Initialized	Initialized	Initialized	
SCMR_3	Initialized	—	—	—	—	
SMR_4	Initialized	—	—	—	—	SCI_4
BRR_4	Initialized	—	—	—	—	
SCR_4	Initialized	—	—	—	—	
TDR_4	Initialized	—	Initialized	Initialized	Initialized	
SSR_4	Initialized	—	Initialized	Initialized	Initialized	
RDR_4	Initialized	—	Initialized	Initialized	Initialized	
SCMR_4	Initialized	—	—	—	—	
FCCS	Initialized	—	—	—	—	FLASH
FPCS	Initialized	—	—	—	—	
FECS	Initialized	—	—	—	—	
FKEY	Initialized	—	—	—	—	
FMATS	Initialized	—	—	—	—	
FTDAR	Initialized	—	—	—	—	
TCR_4	Initialized	—	—	—	—	TPU_4*
TMDR_4	Initialized	—	—	—	—	
TIOR_4	Initialized	—	—	—	—	
TIER_4	Initialized	—	—	—	—	
TSR_4	Initialized	—	—	—	—	
TCNT_4	Initialized	—	—	—	—	
TGRA_4	Initialized	—	—	—	—	
TGRB_4	Initialized	—	—	—	—	
TCR_5	Initialized	—	—	—	—	TPU_5*
TMDR_5	Initialized	—	—	—	—	
TIOR_5	Initialized	—	—	—	—	
TIER_5	Initialized	—	—	—	—	
TSR_5	Initialized	—	—	—	—	
TCNT_5	Initialized	—	—	—	—	
TGRA_5	Initialized	—	—	—	—	
TGRB_5	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
INTCR	Initialized	—	—	—	—	INTC
CPUPCR	Initialized	—	—	—	—	
IER	Initialized	—	—	—	—	
ISR	Initialized	—	—	—	—	
PORT1	—	—	—	—	—	I/O port
PORT2	—	—	—	—	—	
PORT3	—	—	—	—	—	
PORT4	—	—	—	—	—	
PORT5	—	—	—	—	—	
PORT6	—	—	—	—	—	
PORTA	—	—	—	—	—	
PORTD	—	—	—	—	—	
P1DR	Initialized	—	—	—	—	
P2DR	Initialized	—	—	—	—	
P3DR	Initialized	—	—	—	—	
P6DR	Initialized	—	—	—	—	
PADR	Initialized	—	—	—	—	
PDDR	Initialized	—	—	—	—	
PCR	Initialized	—	—	—	—	PPG*
PMR	Initialized	—	—	—	—	
NDERH	Initialized	—	—	—	—	
NDERL	Initialized	—	—	—	—	
PODRH	Initialized	—	—	—	—	
PODRL	Initialized	—	—	—	—	
NDRH	Initialized	—	—	—	—	
NDRL	Initialized	—	—	—	—	
ADDRA_0	Initialized	—	—	—	—	A/D_0
ADDRB_0	Initialized	—	—	—	—	
ADDRC_0	Initialized	—	—	—	—	
ADDRD_0	Initialized	—	—	—	—	
ADDRE_0	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
ADDRF_0	Initialized	—	—	—	—	A/D_0
ADDRG_0	Initialized	—	—	—	—	
ADDRH_0	Initialized	—	—	—	—	
ADCSR_0	Initialized	—	—	—	—	
ADCR_0	Initialized	—	—	—	—	
TCSR	Initialized	—	—	—	—	WDT
TCNT	Initialized	—	—	—	—	
RSTCSR	Initialized	—	—	—	—	
TSTR	Initialized	—	—	—	—	TPU*
TSYR	Initialized	—	—	—	—	
TCR_0	Initialized	—	—	—	—	TPU_0*
TMDR_0	Initialized	—	—	—	—	
TIORH_0	Initialized	—	—	—	—	
TIORL_0	Initialized	—	—	—	—	
TIER_0	Initialized	—	—	—	—	
TSR_0	Initialized	—	—	—	—	
TCNT_0	Initialized	—	—	—	—	
TGRA_0	Initialized	—	—	—	—	
TGRB_0	Initialized	—	—	—	—	
TGRC_0	Initialized	—	—	—	—	
TGRD_0	Initialized	—	—	—	—	
TCR_1	Initialized	—	—	—	—	TPU_1*
TMDR_1	Initialized	—	—	—	—	
TIOR_1	Initialized	—	—	—	—	
TIER_1	Initialized	—	—	—	—	
TSR_1	Initialized	—	—	—	—	
TCNT_1	Initialized	—	—	—	—	
TGRA_1	Initialized	—	—	—	—	
TGRB_1	Initialized	—	—	—	—	

Register Abbreviation	Reset	Sleep	Module Stop	All-Module-Clock-Stop	Software Standby	Reset
TCR_2	Initialized	—	—	—	—	TPU_2*
TMDR_2	Initialized	—	—	—	—	
TIOR_2	Initialized	—	—	—	—	
TIER_2	Initialized	—	—	—	—	
TSR_2	Initialized	—	—	—	—	
TCNT_2	Initialized	—	—	—	—	
TGRA_2	Initialized	—	—	—	—	
TGRB_2	Initialized	—	—	—	—	
TCR_3	Initialized	—	—	—	—	TPU_3*
TMDR_3	Initialized	—	—	—	—	
TIORH_3	Initialized	—	—	—	—	
TIORL_3	Initialized	—	—	—	—	
TIER_3	Initialized	—	—	—	—	
TSR_3	Initialized	—	—	—	—	
TCNT_3	Initialized	—	—	—	—	
TGRA_3	Initialized	—	—	—	—	
TGRB_3	Initialized	—	—	—	—	
TGRC_3	Initialized	—	—	—	—	
TGRD_3	Initialized	—	—	—	—	

Note: * Supported only by the H8SX/1527R.

Section 21 Electrical Characteristics

21.1 Absolute Maximum Ratings

Table 21.1 Absolute Maximum Ratings

Item	Symbol	Value	Unit
Power supply voltage	V_{CC}	–0.3 to +7.0	V
Input voltage (except ports 4 and 5)	V_{in}	–0.3 to V_{CC} + 0.3	V
Input voltage (port 4)	V_{in}	–0.3 to AV_{CC1} + 0.3	V
Input voltage (port 5)	V_{in}	–0.3 to AV_{CC0} + 0.3	V
Analog power supply voltage	AV_{CC0}	–0.3 to +7.0	V
	AV_{CC1}	–0.3 to +7.0	V
Analog input voltage (port 4)	V_{AN}	–0.3 to AV_{CC1} + 0.3	V
Analog input voltage (port 5)	V_{AN}	–0.3 to AV_{CC0} + 0.3	V
Operating temperature	T_{opr}	Wide-range specifications: –40 to +85*	°C
Storage temperature	T_{stg}	–55 to +125	°C

Caution: Permanent damage to the LSI may result if absolute maximum ratings are exceeded.

Note: * The operating temperature when programming/erasing the flash memory ranges from 0°C to +85°C for wide-range specification products.

21.2 DC Characteristics

Table 21.2 DC Characteristics (1)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V, V_{SS} = AV_{SS} = 0 V*[1], T_a = –40°C to +85°C (wide-range specifications)

	Item	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Schmitt trigger input voltage	$\overline{IRQ}$ input pin, TPU input pin, ports 2, 3, J, K	VT^-	$V_{CC} \times 0.2$	—	—	V	
		VT^+	—	—	$V_{CC} \times 0.7$		
		$VT^+ - VT^-$	$V_{CC} \times 0.05$	—	—		
Input high voltage (except Schmitt trigger input pin)	MD, $\overline{RES}$, NMI	V_{IH}	$V_{CC} - 0.7$	—	$V_{CC} + 0.3$	V	
	EXTAL		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Other input pins		$V_{CC} \times 0.7$	—	$V_{CC} + 0.3$		
	Port 4		$AV_{CC1} \times 0.7$	—	$AV_{CC1} + 0.3$		
	Port 5		$AV_{CC0} \times 0.7$	—	$AV_{CC0} + 0.3$		
Input low voltage (except Schmitt trigger input pin)	$\overline{RES}$, MD, NMI	V_{IL}	–0.3	—	$V_{CC} \times 0.1$	V	
	EXTAL		–0.3	—	$V_{CC} \times 0.2$		
	Other pins		–0.3	—	$V_{CC} \times 0.2$		
	Port 4		–0.3	—	$AV_{CC1} \times 0.2$		
	Port 5		–0.3	—	$AV_{CC0} \times 0.2$		
Output high voltage	All output pins	V_{OH}	$V_{CC} - 0.5$	—	—	V	I_{OH} = –200 μA
			$V_{CC} - 1.0$	—	—		I_{OH} = –1 mA
Output low voltage	All output pins	V_{OL}	—	—	0.4	V	I_{OL} = 1.6 mA
Input leakage current	$\overline{RES}$, NMI, MD	$\|I_{in}\|$	—	—	1.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
	Port 4		—	—	1.0		V_{in} = 0.5 to AV_{CC1} – 0.5 V
	Port 5		—	—	1.0		V_{in} = 0.5 to AV_{CC0} – 0.5 V

Table 21.2 DC Characteristics (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V, V_{SS} = AV_{SS} = 0 V*[1], T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Tri-state leakage current (off state)	Ports 1 to 3, 6, A, D, H, J, K	$\lvert I_{TSI} \rvert$	—	—	1.0	μA	V_{in} = 0.5 to V_{CC} – 0.5 V
Input pull-up MOS current	Ports D, H, J, K	$-I_p$	50	—	300	μA	V_{in} = 0 V
Input capacitance	All input pins	C_{in}	—	—	15	pF	V_{in} = 0 V f = 1 MHz T_a = 25°C
Current consumption*[2]	Normal operation	I_{CC}*[4]	—	75	87	mA	f = 48 MHz
	Sleep mode		—	58	68		
	Standby mode*[3]		—	50	300		$T_a \leq$ 50°C
			—	—	1		50°C < T_a
	All-module-clock-stop mode*[5]		—	36	45		
Analog power supply current	During A/D conversion	AI_{CC0}	—	3.5	5		AV_{CC0} = 5.0 V
	Standby for A/D conversion		—	10	100	μA	
	During A/D conversion	AI_{CC1}	—	3.5	5	mA	AV_{CC1} = 5.0 V
	Standby for A/D conversion		—	10	100	μA	
RAM standby voltage		V_{RAM}	3.0	—	—	V	

Notes:
1. When the A/D converter is not used, the AV_{CC0}, AV_{CC1}, and AV_{SS} pins should not be open. Connect the AV_{CC0} and AV_{CC1} pins to V_{CC}, and the AV_{SS} pin to V_{SS}.
2. Current consumption values are for V_{IH} = AV_{CC0} (port 5), AV_{CC1} (port 4), V_{CC} (others) and V_{IL} = 0 V with all output pins unloaded and all input pull-up MOSs in the off state.
3. The values are for $V_{RAM} \leq V_{CC} < 4.5$ V, V_{IH}min. = V_{CC} – 0.1 V, and V_{IL}max. = 0.1 V.
4. I_{CC} depends on V_{CC} and f as follows:
   I_{CC}max = 12 (mA) + 0.28 (mA/(MHz × V)) × V_{CC} × f (normal operation)
   I_{CC}max = 12 (mA) + 0.21 (mA/(MHz × V)) × V_{CC} × f (sleep mode)
5. The values are for reference.

Table 21.3 Permissible Output Currents

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V, V_{SS} = AV_{SS} = 0 V*, T_a = –40°C to +85°C (wide-range specifications)

Item		Symbol	Min.	Typ.	Max.	Unit
Permissible output low current (per pin)	All output pins	I_{OL}	—	—	10	mA
Permissible output low current (total)	Total of all output pins	ΣI_{OL}	—	—	100	mA
Permissible output high current (per pin)	All output pins	$-I_{OH}$	—	—	2.0	mA
Permissible output high current (total)	Total of all output pins	$\Sigma -I_{OH}$	—	—	30	mA

Caution: To protect the LSI's reliability, do not exceed the output current values in table 21.11.

Note: * When the A/D converter is not used, the AV_{CC0}, AV_{CC1}, and AV_{SS} pins should not be open. Connect the AV_{CC0} and AV_{CC1} pins to V_{CC}, and the AV_{SS} pin to V_{SS}.

21.3 AC Characteristics

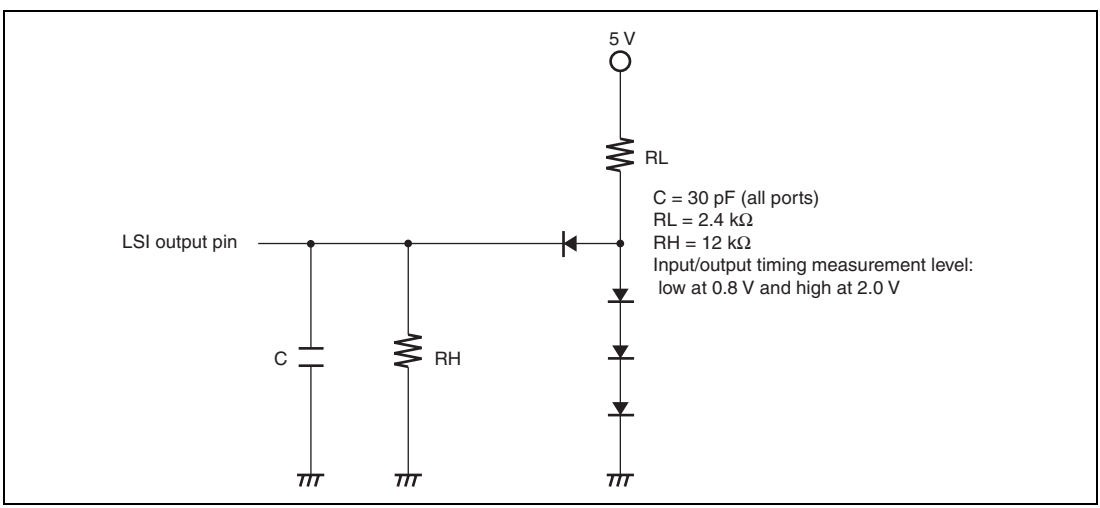


Figure 21.1 Output Load Circuit

21.3.1 Clock Timing

Table 21.4 Clock Timing

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Iϕ = 8 to 48 MHz, Pϕ = 8 to 35 MHz,
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit.	Test Conditions
Clock cycle time	t_{cyc}	20.8	125	ns	Figure 21.2
Clock high pulse width	t_{CH}	3	—	ns	
Clock low pulse width	t_{CL}	3	—	ns	
Clock rising time	t_{Cr}	—	5	ns	
Clock falling time	t_{Cf}	—	5	ns	
Oscillation settling time after reset (crystal)	t_{OSC1}	20	—	ms	Figure 21.4
Oscillation settling time after leaving software standby mode (crystal)	t_{OSC2}	10	—	ms	Figure 21.3
External clock output delay settling time	t_{DEXT}	2	—	ms	Figure 21.4
External clock input low pulse width	T_{EXL}	45	—	ns	Figure 21.5
External clock input high pulse width	T_{EXH}	45	—	ns	External clock input frequency = 4 to 9 MHz
External clock rising time	T_{EXr}	—	5	ns	
External clock falling time	T_{EXf}	—	5	ns	

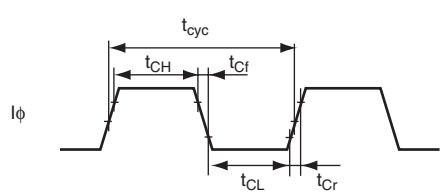


Figure 21.2 System Bus Clock Timing

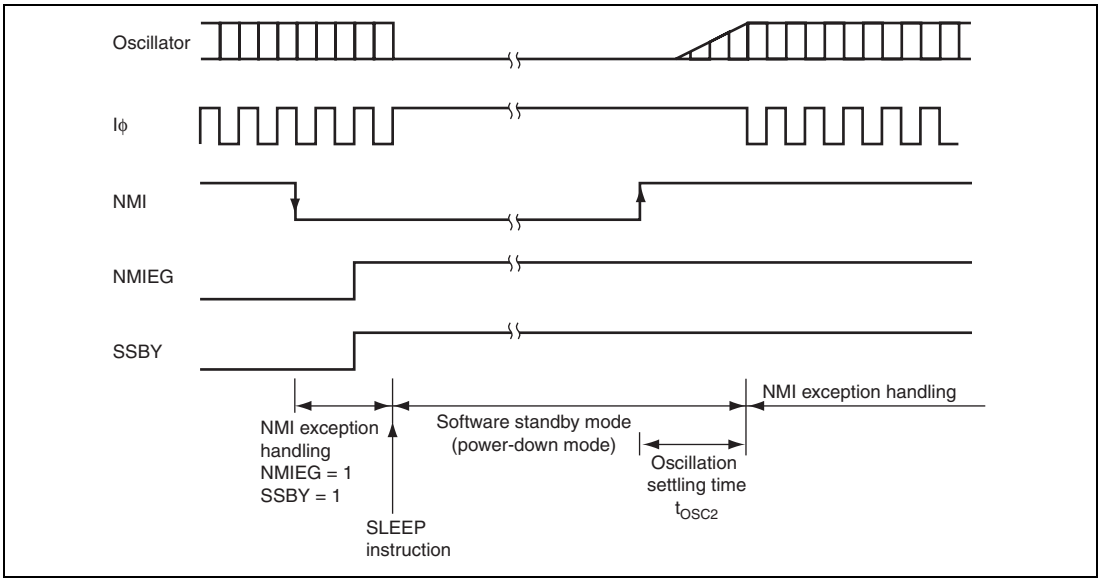


Figure 21.3 Oscillation Settling Timing after Software Standby Mode

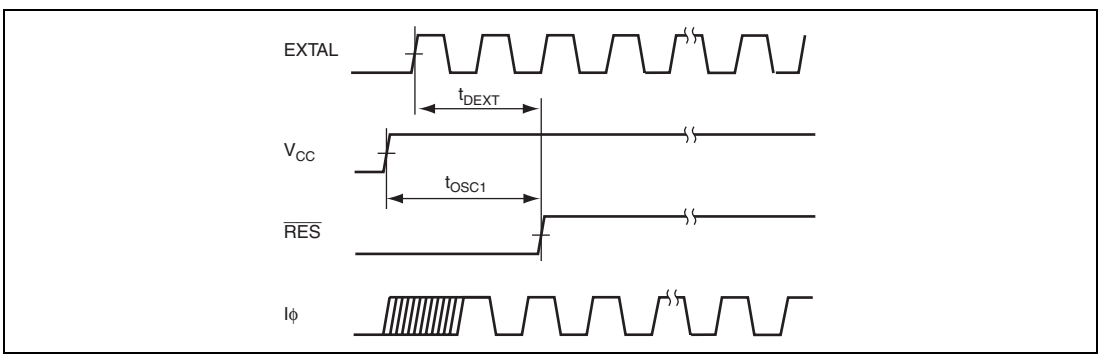


Figure 21.4 Oscillation Settling Timing

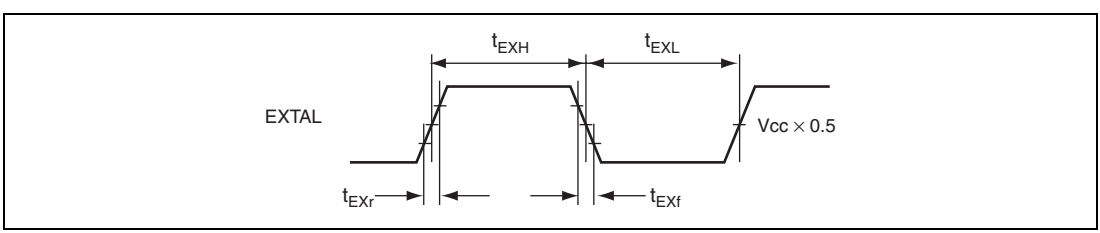


Figure 21.5 External Input Clock Timing

21.3.2 Control Signal Timing

Table 21.5 Control Signal Timing

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Iϕ = 8 to 48 MHz,
T_a = –40°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Max.	Unit	Test Conditions
$\overline{\text{RES}}$ setup time	t_{RESS}	1000	—	ns	Figure 21.6
$\overline{\text{RES}}$ pulse width	t_{RESW}	50	—	t_{cyc}	
NMI setup time	t_{NMIS}	150	—	ns	Figure 21.7
NMI hold time	t_{NMIH}	10	—	ns	
NMI pulse width (after leaving software standby mode)	t_{NMIW}	200	—	ns	
$\overline{\text{IRQ}}$ setup time	t_{IRQS}	150	—	ns	
$\overline{\text{IRQ}}$ hold time	t_{IRQH}	10	—	ns	
$\overline{\text{IRQ}}$ pulse width (after leaving software standby mode)	t_{IRQW}	200	—	ns	

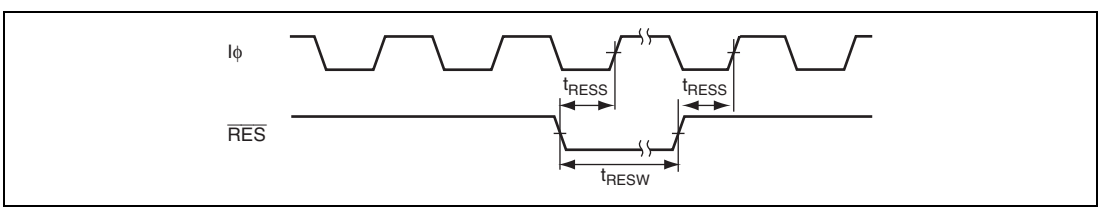
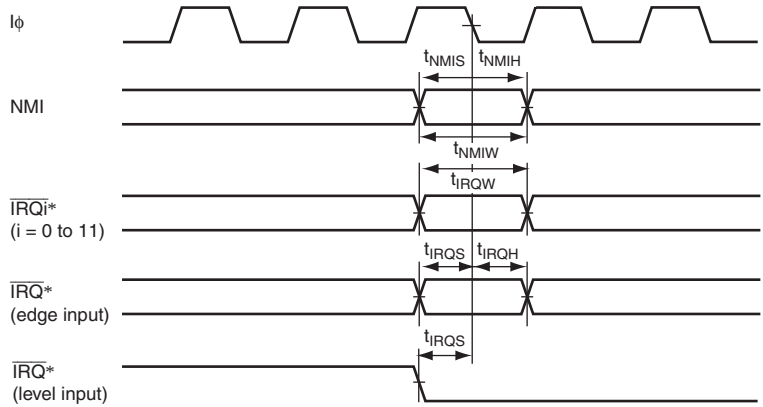


Figure 21.6 Reset Input Timing



Note: * SSIER must be set to cancel software standby mode.

Figure 21.7 Interrupt Input Timing

21.3.3 Timing of On-Chip Peripheral Modules

Table 21.6 Timing of On-Chip Peripheral Modules (1)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 to 35 MHz,
T_a = –40°C to +85°C (wide-range specifications)

Item			Symbol	Min.	Max.	Unit	Test Conditions
I/O ports	Output data delay time		t_{PWD}	—	40	ns	Figure 21.8
	Input data setup time		t_{PRS}	25	—	ns	
	Input data hold time		t_{PRH}	25	—	ns	
	Realtime input port data hold time		t_{RTIPH}	4	—	t_{cyc}	Figure 21.9
TPU	Timer output delay time		t_{TOCD}	—	40	ns	Figure 21.10
	Timer input setup time		t_{TICS}	25	—	ns	
	Timer clock input setup time		t_{TCKS}	25	—	ns	Figure 21.11
	Timer clock pulse width	Single-edge setting	t_{TCKWH}	1.5	—	t_{cyc}	
		Both-edge setting	t_{TCKWL}	2.5	—	t_{cyc}	

Item			Symbol	Min.	Max.	Unit	Test Conditions
PPG	Pulse output delay time		t_{POD}	—	40	ns	Figure 21.12
SCI	Input clock cycle	Asynchronous	t_{Scyc}	4	—	t_{cyc}	Figure 21.13
		Clocked synchronous		6	—		
	Input clock pulse width		t_{SCKW}	0.4	0.6	t_{Scyc}	
	Input clock rise time		t_{SCKr}	—	20	ns	Figure 21.13
	Input clock fall time		t_{SCKf}	—	20	ns	Measurement voltages: VCC × 0.3 V to VCC × 0.7 V
	Output clock cycle	Asynchronous	t_{Scyc}	30	—	t_{cyc}	Figure 21.13
		Clocked synchronous		4	—		
	Output clock pulse width		t_{SCKW}	0.4	0.6	t_{Scyc}	
	Output clock rise time		t_{SCKr}	—	20	ns	Figure 21.13
	Output clock fall time		t_{SCKf}	—	20	ns	Measurement voltages: VCC × 0.3 V to VCC × 0.7 V
	Transmit data delay time		t_{TXD}	—	40	ns	Figure 21.14
	Receive data setup time (clocked synchronous)		t_{RXS}	40	—	ns	
	Receive data hold time (clocked synchronous)		t_{RXH}	40	—	ns	
A/D converter	Trigger input setup time		t_{TRGS}	30	—	ns	Figure 21.15

Table 21.6 Timing of On-Chip Peripheral Modules (2)

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 to 24 MHz,
T_a = –40°C to +85°C (wide-range specifications)

Item			Symbol	Min.	Max.	Unit	Test Conditions
RCAN-ET*	Transmit data delay time		t_{CTXD}	—	100	ns	Figure 21.16
	Receive data setup time		t_{CRXS}	100	—	ns	
	Receive data hold time		t_{CRXH}	100	—	ns	
SSU	Clock cycle time	Master	t_{SUcyc}	4	256	t_{cyc}	Figure 21.17
		Slave		4	256		Figure 21.18
	Clock high pulse width	Master	t_{HI}	80	—	ns	Figure 21.19
		Slave		80	—		Figure 21.20
	Clock low pulse width	Master	t_{LO}	80	—	ns	
		Slave		80	—		
	Clock rising time		t_{RISE}	—	20	ns	
	Clock falling time		t_{FALL}	—	20	ns	
	Data input setup time	Master	t_{SU}	25	—	ns	
		Slave		30	—		
	Data input hold time	Master	t_{H}	10	—	ns	
		Slave		10	—		
	$\overline{SCS}$ setup time	Master	t_{LEAD}	2.5	—	t_{cyc}	
		Slave		2.5	—		
	$\overline{SCS}$ hold time	Master	t_{LAG}	2.5	—	t_{cyc}	
		Slave		2.5	—		
	Data output delay time	Master	t_{OD}	—	40	ns	
		Slave		—	40		
	Data output hold time	Master	t_{OH}	30	—	ns	
		Slave		30	—		
	Consecutive transmit delay time	Master	t_{TD}	2.5	—	t_{cyc}	
		Slave		2.5	—		

Item		Symbol	Min.	Max.	Unit	Test Conditions
SSU	Slave access time	t_{SA}	—	1	t_{cyc}	Figure 21.19
	Slave out release time	t_{REL}	—	1	t_{cyc}	Figure 21.20

Note: * Although the RCAN-ET input signals are asynchronous signals, they are received as the signals in synchronization with every other rising edge of the Pϕ clock (see Figure 21.16). The RCAN-ET output signals are also asynchronous signals, however, change their levels based on every other rising edge of the Pϕ clock.

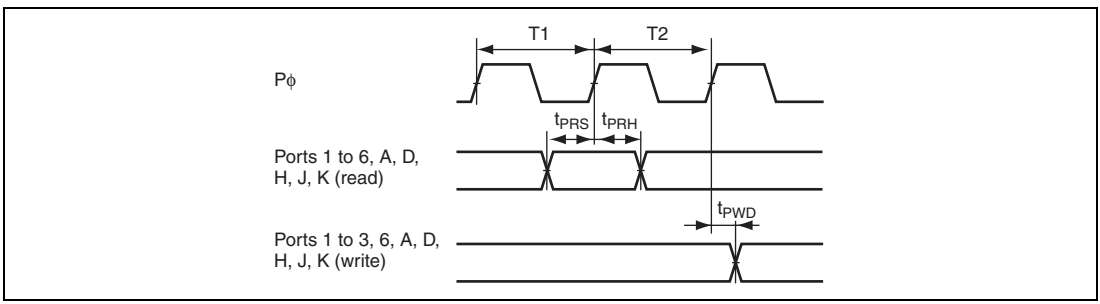


Figure 21.8 I/O Port Input/Output Timing

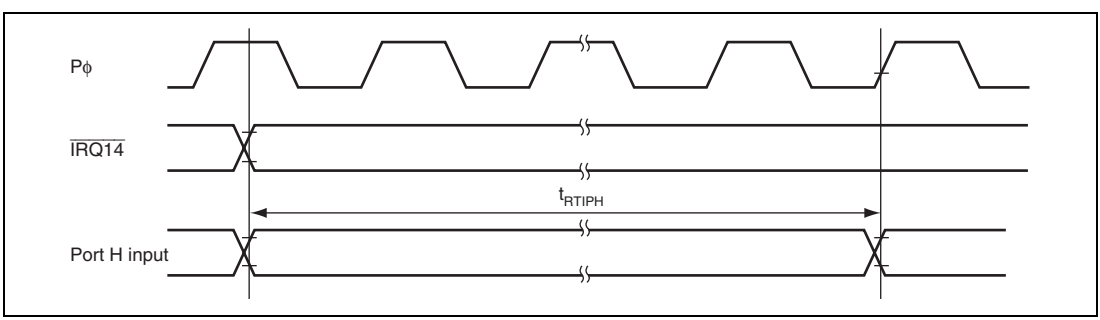
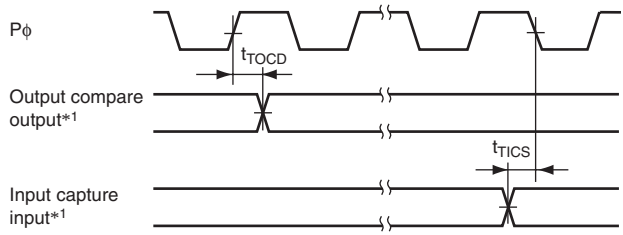
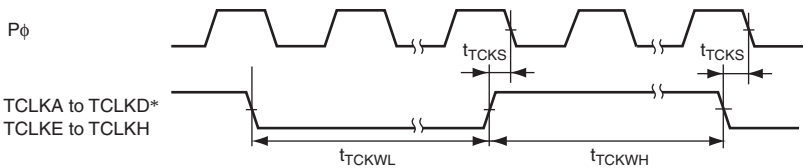


Figure 21.9 Data Input Timing for Realtime Input Port



Notes: 1. TIOCA0 to TIOCA3*2, TIOCB0 to TIOCB3*2, TIOCC0*2, TIOCC3*2, TIOCD0*2, TIOCD3*2, TIOCA6 to TIOCA11, TIOCB6 to TIOCB11, TIOCC6, TIOCC9, TIOCD6, and TIOCD9
2. Supported only by the H8SX/1527R.

Figure 21.10 TPU Input/Output Timing



Note: * Supported only by the H8SX/1527R.

Figure 21.11 TPU Clock Input Timing

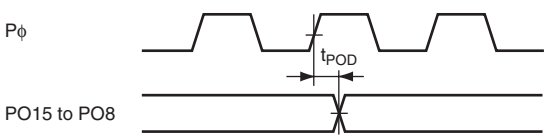


Figure 21.12 PPG Output Timing

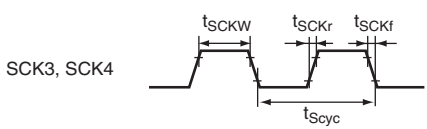


Figure 21.13 SCK Clock Input/Output Timing

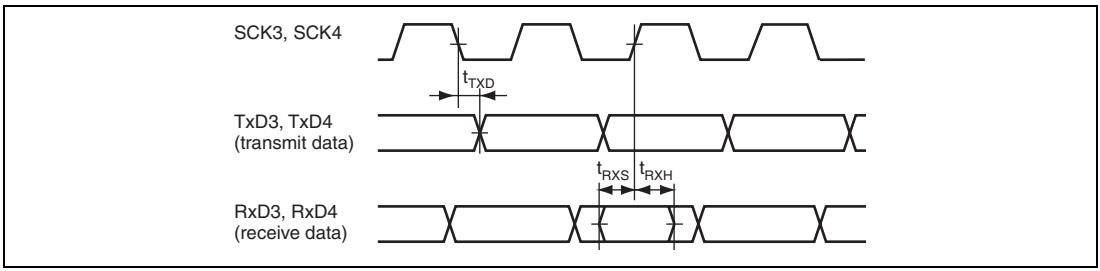


Figure 21.14 SCI Input/Output Timing: Clocked Synchronous Mode

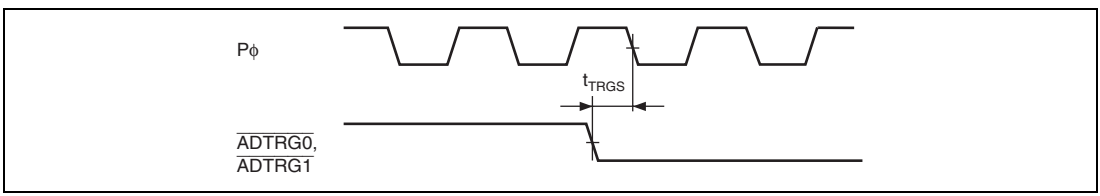


Figure 21.15 A/D Converter External Trigger Input Timing

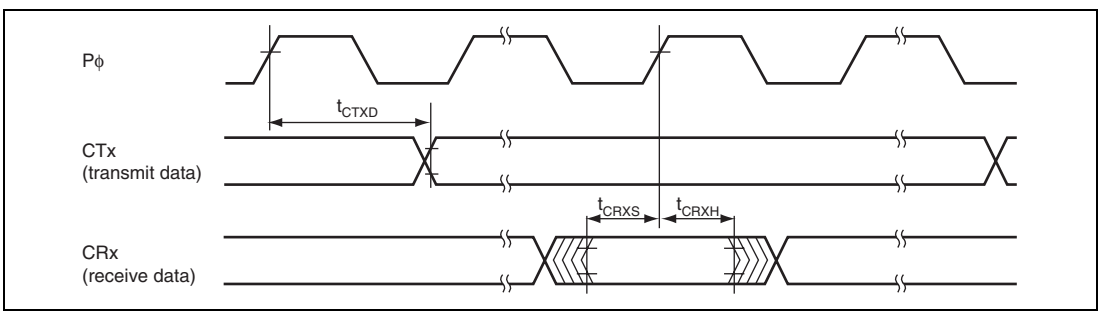


Figure 21.16 RCAN-ET Input/Output Timing

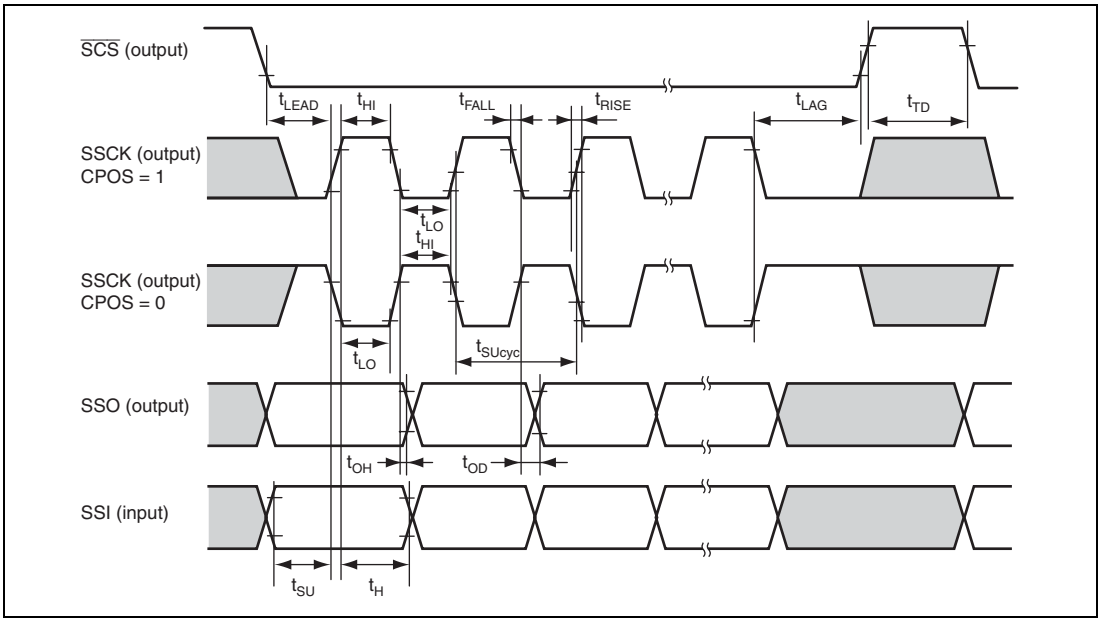


Figure 21.17 SSU Timing (Master, CPHS = 1)

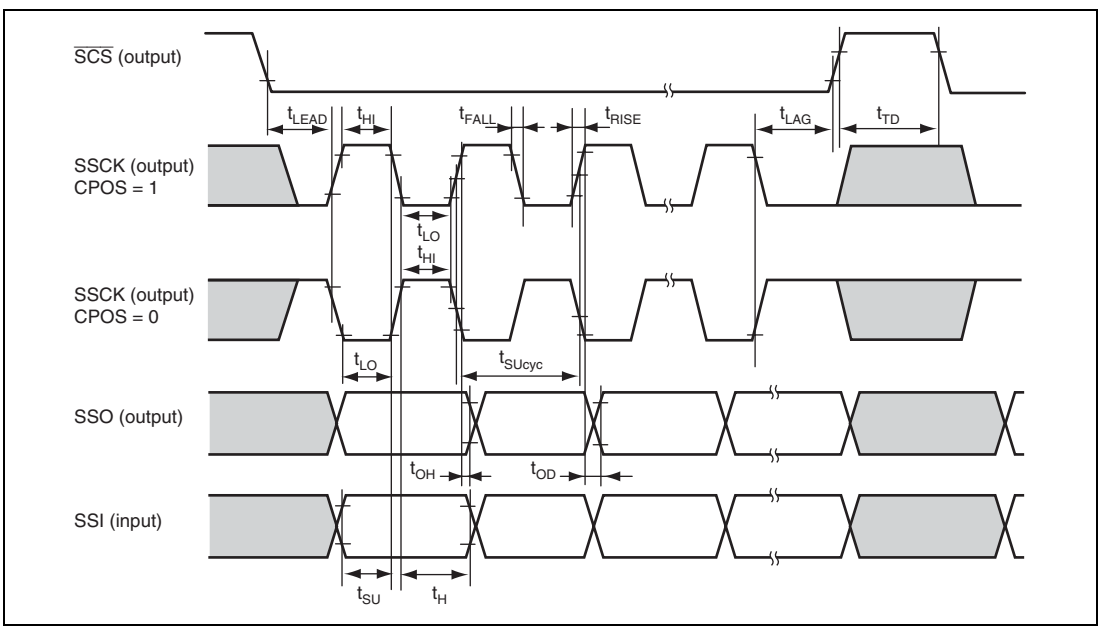


Figure 21.18 SSU Timing (Master, CPHS = 0)

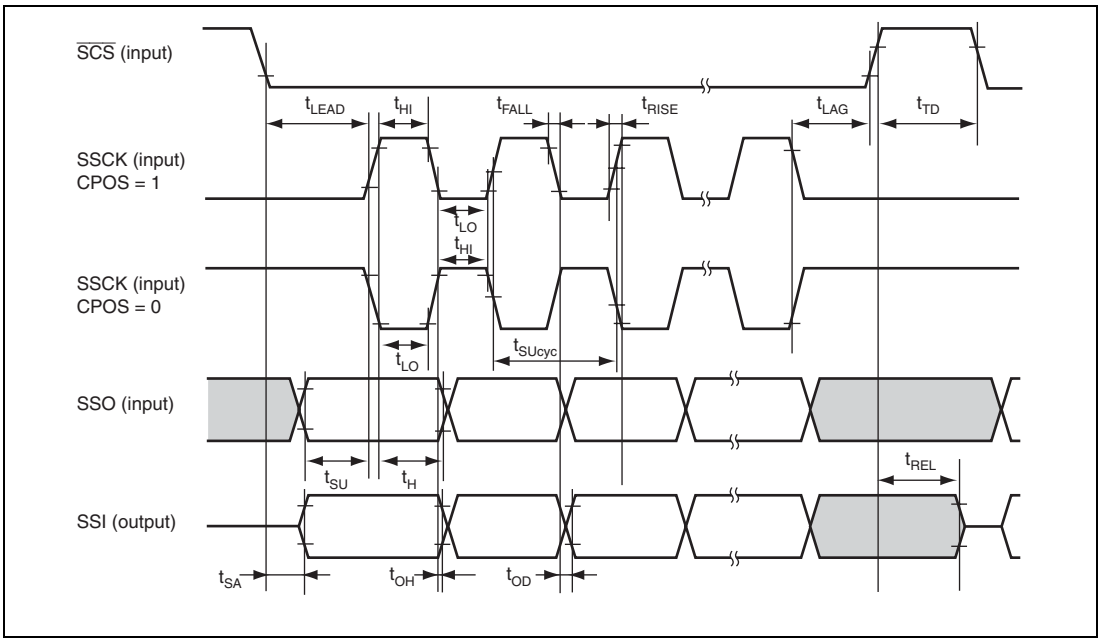


Figure 21.19 SSU Timing (Slave, CPHS = 1)

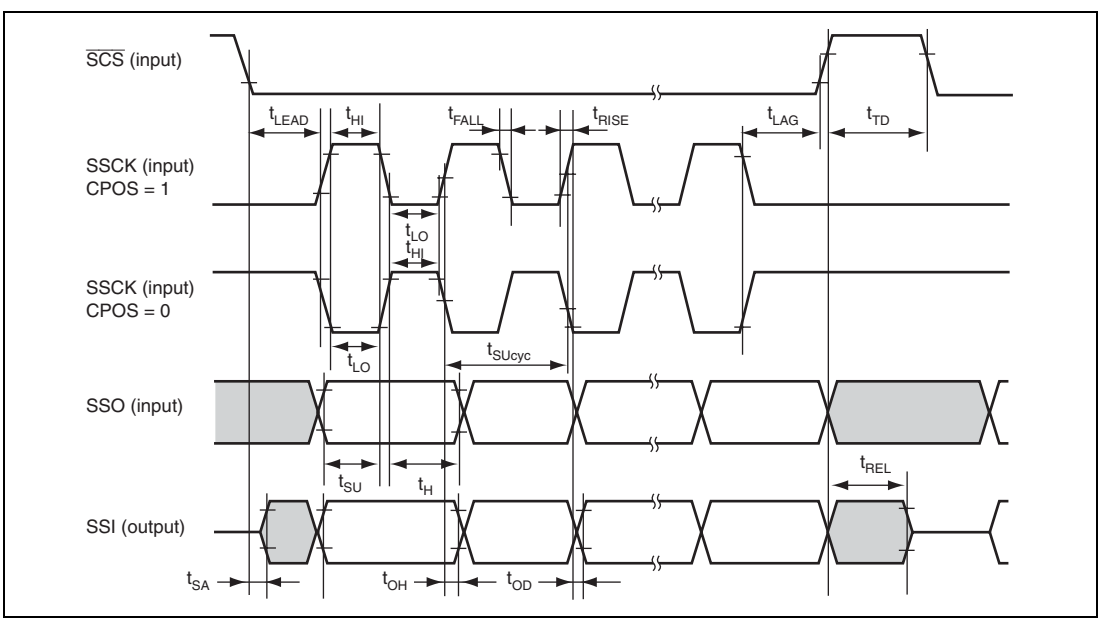


Figure 21.20 SSU Timing (Slave, CPHS = 0)

21.4 A/D Conversion Characteristics

Table 21.7 A/D Conversion Characteristics

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Pϕ = 8 to 35 MHz,
T_a = –40°C to +85°C (wide-range specifications)

Item	Min.	Typ.	Max.	Unit
Resolution	10	10	10	Bit
Conversion time	7.4	—	200	μs
Analog input capacitance	—	—	20	pF
Permissible signal source impedance	—	—	5	kΩ
Nonlinearity error	—	—	±3.5	LSB
Offset error	—	—	±3.5	LSB
Full-scale error	—	—	±3.5	LSB
Quantization error	—	±0.5	—	LSB
Absolute accuracy	—	—	±4.0	LSB

21.5 Flash Memory Characteristics

Table 21.8 Flash Memory Characteristics

Conditions: V_{CC} = 4.5 V to 5.5 V, AV_{CC0} = 4.5 V to 5.5 V, AV_{CC1} = 4.5 V to 5.5 V,
V_{SS} = AV_{SS} = 0 V, Iϕ = 8 to 48 MHz, Pϕ = 8 to 35 MHz,
T_a = 0°C to +85°C (wide-range specifications)

Item	Symbol	Min.	Typ.	Max.	Unit	Test Condition
Programming time*1*2*4	t_P	—	3	30	ms/128 bytes	
Erase time*1*2*4	t_E	—	80	800	ms/4-kbyte block	
		—	500	5000	ms/32-kbyte block	
		—	1000	10000	ms/64-kbyte block	
Programming time (total) *1*2*4	Σt_P	—	5	15	s/256 kbytes	T_a = 25°C, memory filled with 0.
Erase time (total) *1*2*4	Σt_E	—	5	15	s/256 kbytes	T_a = 25°C
Programming/erase time (total) *1*2*4	Σt_{PE}	—	10	30	s/256 kbytes	T_a = 25°C
Number of programming	N_{WEC}	100*3	—	—	Times	
Data retention time*4	t_{DRP}	10	—	—	Year	

Notes: 1. Programming time and erase time depend on data in the flash memory.
2. Programming time and erase time do not include time for data transfer.
3. All the characteristics after programming are guaranteed within this value (guaranteed value is from 1 to Min. value).
4. Characteristics when programming is performed within the Min. value

Appendix

A. Port States in Each Pin State

Table A.1 Port States in Each Pin State

Port Name	MCU Operating Mode	Reset	Software Standby Mode
Port 1	All	Hi-Z	Keep
Port 2	All	Hi-Z	Keep
Port 3	All	Hi-Z	Keep
Port 4	All	Hi-Z	Hi-Z
Port 5	All	Hi-Z	Hi-Z
Port 6	All	Hi-Z	Keep
Port A	All	Hi-Z	Keep
Port D	All	Hi-Z	Keep
Port H	All	Hi-Z	Keep
Port J	All	Hi-Z	Keep
Port K	All	Hi-Z	Keep

B. Product Lineup

Product Classification	Product Model	Marking	Package	Package Code
H8SX/1527R	R5F61527R	R5F61527R	QFP-100	PRQP0100KB-A (FP-100M)
H8SX/1525R	R5F61525R	R5F61525R		

C. Package Dimensions

For the package dimensions, data in the Renesas IC Package General Catalog has priority.

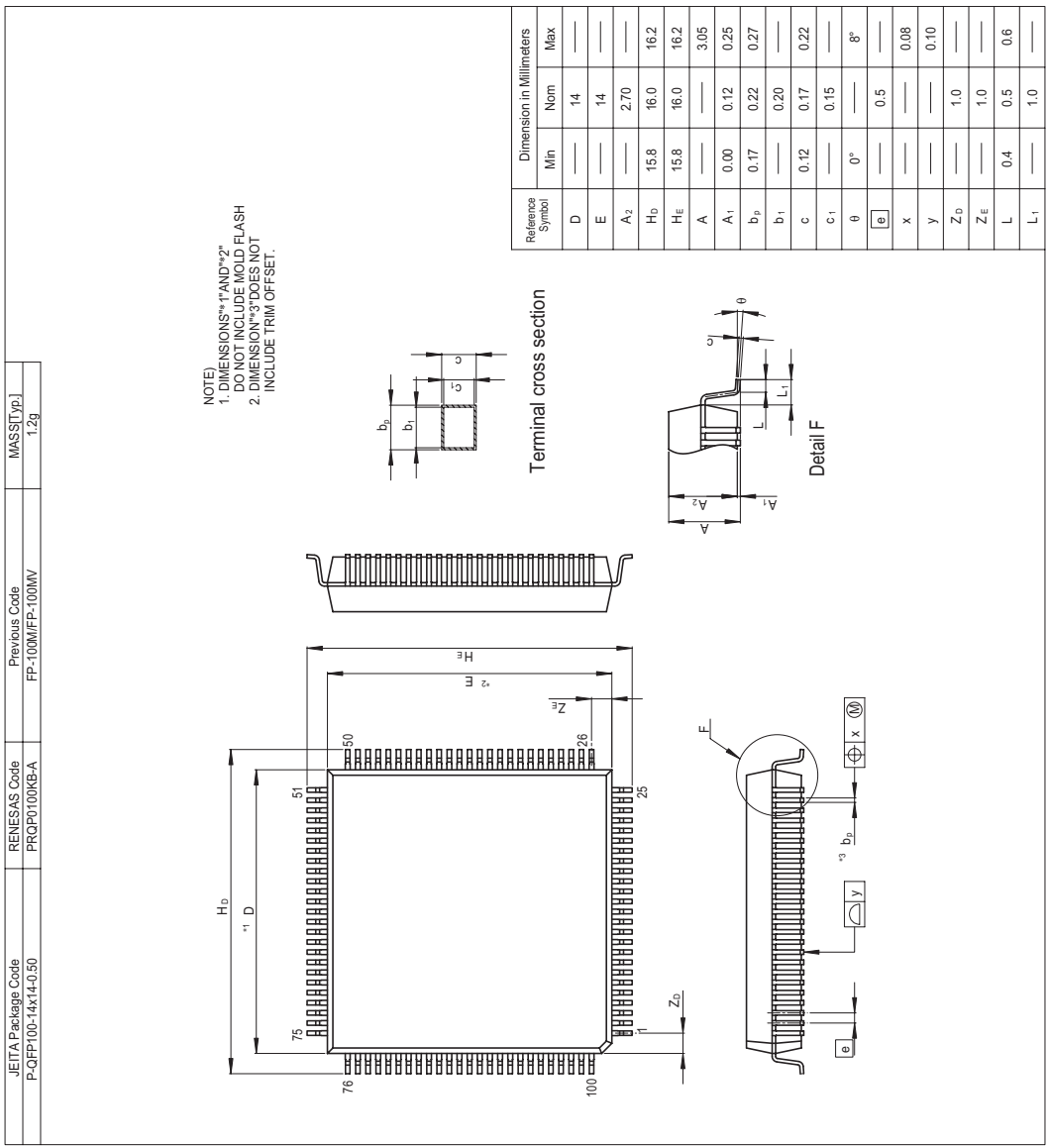


Figure C.1 Package Dimensions (FP-100M)

Index

Numerics

0-output/1-output 302
16-bit timer pulse unit (TPU) 251

A

A/D conversion accuracy..... 573
A/D converter 559
Absolute accuracy..... 573
Absolute maximum ratings..... 757
AC characteristics..... 760
Address error 77
Address error exception handling 78
Address map 70
Address modes..... 158
All-module-clock-stop mode 688, 696
Asynchronous mode 406
AT-cut parallel-resonance type..... 681
Available output signal and settings
in each port 239

B

Bϕ clock output control 701
Bit rate 399
Block diagram..... 2
Block structure..... 587
Block transfer mode..... 164
Boot mode 584, 613
Burst mode..... 169
Bus access modes 168
Bus arbitration 131
Bus configuration 127
Bus controller (BSC) 125
Bus-released state 63

C

CAN interface 453
Can sleep mode 500
Clock 409
Clock pulse generator 677
Clock synchronous communication
mode 550
Clocked synchronous mode 423
Communications protocol 646
Controller area network (RCAN-ET)..... 449
CPU priority control function
over DMAC 118
Crystal resonator 681
Cycle stealing mode 168

D

Data direction register 212
Data register 212
DC characteristics 758
Direct convention..... 431
DMA controller (DMAC) 133
Double-buffered structure 406
Download pass/fail result parameter 601
Dual address mode 158

E

Electrical characteristics 757
Error protection..... 638
Error signal 431
Exception handling 71
Exception handling by illegal
instruction 82
Exception handling vector table..... 72
Exception-handling state..... 63
Extended repeat area 156

Extended repeat area function 170
External bus clock (Bϕ) 677
External clock 682
External interrupts 101
External trigger input 572

F

Flash erase block select parameter 610
Flash memory 581
Flash multipurpose address area parameter 608
Flash multipurpose data destination parameter 609
Flash pass and fail parameter 602
Flash program/erase frequency parameter 607
Free-running count operation 300
Frequency divider 682
Full-scale error 573

G

General illegal instruction 82
General registers 29

H

Halt mode 499
Hardware protection 637

I

I/O ports 205
ID code 417
ID Reorder 464
Illegal instruction 82
Input buffer control register 213
Input capture function 303
Internal bus 129
Internal interrupts 102
Internal peripheral bus 127
Internal system bus 1 127
Interrupt 79
Interrupt control mode 0 110
Interrupt control mode 2 112
Interrupt controller 85
Interrupt exception handling 80
Interrupt exception handling sequence ... 114
Interrupt exception handling vector table 103
Interrupt response times 115
Interrupt sources 101
Interrupt sources and vector address offsets 103
Interval timer mode 373
Inverse convention 432
IRQn interrupts 101

L

Local acceptance filter mask (LAFM) 461

M

Mailbox 452
Mailbox control 453
Mailbox structure 456
Mark state 406, 443
MCU operating modes 65
Memory MAT configuration 586
Message control field 457
Message data fields 462
Message receive sequence 506
Message transmission sequence 504
Micro processor interface (MPI) 452
Mode 1 69
Mode 2 69
Mode 3 69

Mode pin 65
Module stop mode 695
Multi-clock function 128, 695
Multiprocessor bit 417
Multiprocessor communication function 417

N

NMI interrupts 101
Nonlinearity error 573
Non-overlapping operation 359
Normal transfer mode 162

O

Offset addition 172
Offset error 573
On-board programming 613
On-board programming mode 613
On-chip baud rate generator 409
Open-drain control register 215
Oscillator 681
Output buffer control 216
Output trigger 358
Overflow 372

P

Package 1
Package dimensions 777
Parity bit 406
Periodic count operation 300
Peripheral module clock (Pϕ) 128, 677
Phase counting mode 318
Pin assignments 4
Pin configuration in each operating mode 6
Pin functions 10
PLL circuit 682
Port function controller 243
Port H realtime input data register 215
Port register 213
Port states in each pin state 775
Power-down modes 687
Processing states 63
Product lineup 776
Program execution state 63
Program stop state 63
Programmable pulse generator (PPG) 345
Programmer mode 584, 644
Programming/erasing interface 588
Programming/erasing interface parameters 599
Programming/erasing interface register 592
Protection 637
Pull-up MOS control register 214

Q

Quantization error 573

R

RAM 579
RCAN-ET bit rate calculation 474
RCAN-ET interrupt sources 510
RCAN-ET memory map 454
RCAN-ET reset sequence 498
Reconfiguration of Mailbox 508
Register addresses 704
Register bits 723
Register configuration in each port 210
Register states in each operating mode 743
Registers
  ABACK0 ..... 491, 704, 723, 743
  ADCR 567, 720, 740, 754
  ADCSR 565, 720, 740, 754
  ADDR 564, 720, 740, 753
  BCR0 ..... 471, 704, 723, 743

BCR1 .....471, 704, 723, 743
BCR2 .....126, 718, 737, 751
BRR 399, 718, 737, 751
CCR 30
CPUPCR.....88, 719, 739, 753
DACR.....151, 716, 733, 750
DBSR.....141, 716, 732, 750
DDAR.....138, 716, 732, 750
DDR.....212, 715, 731, 748
DMDR 142, 716, 732, 750
DMRSR 157, 717, 736, 750
DOFR 139, 716, 732, 750
DPFR 601
DR 212, 720, 739, 753
DSAR 137, 716, 732, 750
DTCR 140, 716, 732, 750
EXR 32
FCCS 592, 718, 738, 752
FEBS 610
FECS 595, 718, 738, 752
FKEY.....596, 718, 738, 752
FMATS.....597, 718, 738, 752
FMPAR 608
FMPDR 609
FPCS.....595, 718, 738, 752
FPEFEQ..... 607
FPFR..... 602
FTDAR.....598, 718, 738, 752
GSR 469, 704, 723, 743
ICR 213, 715, 731, 749
IER.....92, 719, 739, 753
IMR 483, 704, 723, 743
INTCR.....87, 719, 739, 753
IPR.....90, 717, 736, 751
IRR 476, 704, 723, 743
ISCRH 94, 717, 737, 751
ISCRL.....94, 717, 737, 751
ISR.....99, 719, 739, 753
MAC 33
MBIMR0 .....494, 704, 723, 743
MCR 463, 704, 723, 743
MDCR..... 66, 718, 737, 751
MSTPCRA..... 691, 718, 737, 751
MSTPCRB 691, 718, 737, 751
MSTPCRC 694, 718, 737, 751
NDERH..... 347, 720, 740, 753
NDERL 347, 720, 740, 753
NDRH 350, 720, 740, 753
NDRL 350, 720, 740, 753
ODR..... 215, 716, 732, 749
PC 30
PCR..... 214, 353, 720, 740, 753
PFCR9..... 243, 716, 732, 749
PFCRA..... 245, 716, 732, 749
PFCRB..... 247, 716, 732, 749
PHRTIDR 215, 712, 727, 746
PMR..... 354, 720, 740, 753
PODRH..... 349, 720, 740, 753
PODRL 349, 720, 740, 753
PORT 213, 719, 739, 753
RAMER 612, 718, 737, 751
RCANMON 710, 726, 745
RDR 381, 718, 738, 752
REC..... 484, 704, 723, 743
RFPR0..... 493, 704, 723, 743
RSR.....381
RSTCSR 371, 720, 740, 754
RXPR0 ..... 492, 704, 723, 743
SBR.....32
SBYCR 689, 718, 737, 751
SCKCR 678, 718, 737, 751
SCMR 398, 718, 738, 752
SCR..... 385, 718, 738, 751
SMR..... 382, 718, 737, 751
SSCR2..... 528, 711, 726, 745
SSCRH..... 520, 711, 726, 745
SSCRL 522, 711, 726, 745
SSER..... 524, 711, 726, 745
SSIER 100, 716, 732, 750
SSMR..... 523, 711, 726, 745

SSR 390, 718, 738, 752
SSRDR 531, 711, 726, 745
SSSR..... 525, 711, 726, 745
SSTDR..... 530, 711, 726, 745
SSTRSR..... 533
SYSCR 67, 718, 737, 751
TCNT..... 296, 368, 720, 721,
..... 740, 741, 754
TCR 265, 721, 741, 754
TCSR 369, 720, 740, 754
TDR 381, 718, 738, 752
TEC..... 484, 704, 723, 743
TGR 296, 721, 741, 754
TIER 290, 721, 741, 754
TIOR..... 272, 721, 741, 754
TMDR..... 270, 721, 741, 754
TSR.....292, 382, 721, 741, 754
TSTR 297, 721, 741, 754
TSYR..... 298, 721, 741, 754
TXACK0 ..... 490, 704, 723, 743
TXCR0 ..... 489, 704, 723, 743
TXPR0..... 486, 704, 723, 743
TXPR1 ..... 486, 704, 723, 743
UMSR0..... 495, 704, 723, 743
VBR..... 32
Repeat transfer mode 163
Reset 74
Reset exception handling..... 74
Reset state 63
Resolution..... 573

S

Sample-and-hold circuit..... 571
Scan mode 569
Serial communication interface (SCI) 377
Single address mode 160
Single mode 568
Sleep mode 499, 688, 696
Slot illegal instruction..... 82
Smart card interface 430
Software protection..... 638
Software standby mode..... 688, 697
SSU mode 538
Stack status after exception handling..... 83
Standard serial communication interface
specifications for boot mode 644
Start bit..... 406
State transitions..... 64
Stop bit..... 406
Synchronous clearing..... 305
Synchronous presetting 305
Synchronous serial communication unit
(SSU) 515
System clock (Iϕ) 128, 677

T

Test mode settings..... 502
The address map for each mailbox..... 455
Time quanta is defined..... 471
Toggle output..... 302
Trace exception handling..... 76
Transfer clock 534
Transfer modes..... 162
Transmit/receive data..... 406
Trap instruction exception handling..... 81

U

User boot MAT 586
User boot mode 584, 627
User MAT 586
User program mode..... 584, 617

V

Vector table address 72
Vector table address offset..... 72

W

Watchdog timer (WDT)..... 367
Watchdog timer mode 372
Write data buffer function..... 130
Write data buffer function for external data bus 130

Renesas 32-Bit CISC Microcomputer
Hardware Manual

H8SX/1520R Group

Publication Date: Rev.1.00, Mar. 06, 2006
Published by: Sales Strategic Planning Div.
Renesas Technology Corp.
Edited by: Customer Support Department
Global Strategic Communication Div.
Renesas Solutions Corp.

© 2006. Renesas Technology Corp., All rights reserved. Printed in Japan.

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

Colophon 6.0

H8SX/1520R Group
Hardware Manual

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ09B0282-0100