

The sales of these products are limited for China, Hong Kong, and India.

# R7F0C01072DNP, R7F0C010B2DFP-C

User's Manual: Hardware

16-Bit Single-Chip Microcontrollers

All information contained in these materials, including products and product specifications, represents information on the product at the time of publication and is subject to change by Renesas Electronics Corp. without notice. Please review the latest information published by Renesas Electronics Corp. through various means, including the Renesas Electronics Corp. website (<http://www.renesas.com>).

## Notice

1. 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.
2. 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.
3. 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.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.  
"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 etc.  
"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.  

Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). 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 for which it is not intended. 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 which the product is not intended by Renesas Electronics.
6. 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.
7. 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 systems manufactured by you.
8. 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.
9. 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. You should not use Renesas Electronics products or 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. When exporting the Renesas Electronics 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.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
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.

## NOTES FOR CMOS DEVICES

- (1) **VOLTAGE APPLICATION WAVEFORM AT INPUT PIN:** Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (MAX) and VIH (MIN) due to noise, etc., the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (MAX) and VIH (MIN).
- (2) **HANDLING OF UNUSED INPUT PINS:** Unconnected CMOS device inputs can be cause of malfunction. If an input pin is unconnected, it is possible that an internal input level may be generated due to noise, etc., causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using pull-up or pull-down circuitry. Each unused pin should be connected to VDD or GND via a resistor if there is a possibility that it will be an output pin. All handling related to unused pins must be judged separately for each device and according to related specifications governing the device.
- (3) **PRECAUTION AGAINST ESD:** A strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it when it has occurred. Environmental control must be adequate. When it is dry, a humidifier should be used. It is recommended to avoid using insulators that easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors should be grounded. The operator should be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with mounted semiconductor devices.
- (4) **STATUS BEFORE INITIALIZATION:** Power-on does not necessarily define the initial status of a MOS device. Immediately after the power source is turned ON, devices with reset functions have not yet been initialized. Hence, power-on does not guarantee output pin levels, I/O settings or contents of registers. A device is not initialized until the reset signal is received. A reset operation must be executed immediately after power-on for devices with reset functions.
- (5) **POWER ON/OFF SEQUENCE:** In the case of a device that uses different power supplies for the internal operation and external interface, as a rule, switch on the external power supply after switching on the internal power supply. When switching the power supply off, as a rule, switch off the external power supply and then the internal power supply. Use of the reverse power on/off sequences may result in the application of an overvoltage to the internal elements of the device, causing malfunction and degradation of internal elements due to the passage of an abnormal current. The correct power on/off sequence must be judged separately for each device and according to related specifications governing the device.
- (6) **INPUT OF SIGNAL DURING POWER OFF STATE :** Do not input signals or an I/O pull-up power supply while the device is not powered. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Input of signals during the power off state must be judged separately for each device and according to related specifications governing the device.

# How to Use This Manual

## Readers

This manual is intended for user engineers who wish to understand the functions of the R7F0C01072DNP and R7F0C010B2DFP-C and design and develop application systems and programs for these devices.

The target products are as follows.

- 24-pin product: R7F0C01072DNP
- 32-pin product: R7F0C010B2DFP-C

## Purpose

This manual is intended to give users an understanding of the functions described in the **Organization** below.

## Organization

The R7F0C01072DNP, R7F0C010B2DFP-C manual is separated into two parts: this manual and the software edition (common to the RL78 Family).

<b>R7F0C01072DNP, R7F0C010B2DFP-C User's Manual Hardware</b>
--

<b>RL78 Family User's Manual Software</b>
---

- |  |   |
|--|---|
| <ul style="list-style-type: none"><li>• Pin functions</li><li>• Internal block functions</li><li>• Interrupts</li><li>• Other on-chip peripheral functions</li><li>• Electrical specifications</li></ul> | <ul style="list-style-type: none"><li>• CPU functions</li><li>• Instruction set</li><li>• Explanation of each instruction</li></ul> |
|--|---|

## How to Read This Manual

It is assumed that the readers of this manual have general knowledge of electrical engineering, logic circuits, and microcontrollers.

- To gain a general understanding of functions:  
→ Read this manual in the order of the **CONTENTS**.
- How to interpret the register format:  
→ For a bit number enclosed in angle brackets, the bit name is defined as a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler.
- To know details of the R7F0C01072DNP and R7F0C010B2DFP-C instructions:  
→ Refer to the separate document **RL78 Family User's Manual: Software (R01US0015E)**.

<b>Conventions</b>	Data significance:	Higher digits on the left and lower digits on the right
	Active low representations:	$\overline{\text{xxx}}$ (overscore over pin and signal name)
	<b>Note:</b>	Footnote for item marked with <b>Note</b> in the text
	<b>Caution:</b>	Information requiring particular attention
	<b>Remark:</b>	Supplementary information
	Numerical representations:	Binary            ...xxxx or xxxxB
		Decimal           ...xxxx
		Hexadecimal     ...xxxxH

**Related Documents**           The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

**Documents Related to Devices**

Document Name	Document No.
R7F0C01072DNP, R7F0C010B2DFP-C User's Manual: Hardware	This manual
RL78 Family User's Manual: Software	R01US0015E

**Documents Related to Flash Memory Programming**

Document Name	Document No.
PG-FP5 Flash Memory Programmer User's Manual	R20UT0008E

**Other Documents**

Document Name	Document No.
Renesas MPUs & MCUs RL78 Family	R01CP0003E
Semiconductor Package Mount Manual	<b>Note</b>
Quality Grades on NEC Semiconductor Devices	C11531E
Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD)	C11892E
Semiconductor Reliability Handbook	R51ZZ0001E

**Note** See the "Semiconductor Package Mount Manual" website (<http://www.renesas.com/products/package/manual/index.jsp>).

**Caution** The related documents listed above are subject to change without notice. Be sure to use the latest version of each document when designing.

All trademarks and registered trademarks are the property of their respective owners. EEPROM is a trademark of Renesas Electronics Corporation.

SuperFlash is a registered trademark of Silicon Storage Technology, Inc. in several countries including the United States and Japan.

Caution: This product uses SuperFlash® technology licensed from Silicon Storage Technology, Inc.
--

# CONTENTS

<b>CHAPTER 1 OUTLINE</b> .....	<b>1</b>
<b>1.1 Features</b> .....	<b>1</b>
<b>1.2 List of Part Numbers</b> .....	<b>2</b>
<b>1.3 Pin Configuration (Top View)</b> .....	<b>3</b>
1.3.1 24-pin products.....	3
1.3.2 32-pin products.....	4
<b>1.4 Pin Identification</b> .....	<b>5</b>
<b>1.5 Block Diagram</b> .....	<b>6</b>
1.5.1 24-pin products.....	6
1.5.2 32-pin products.....	7
<b>1.6 Outline of Functions</b> .....	<b>8</b>
<b>CHAPTER 2 PIN FUNCTIONS</b> .....	<b>10</b>
<b>2.1 Pin Function List</b> .....	<b>10</b>
2.1.1 24-pin products.....	11
2.1.2 32-pin products.....	12
<b>2.2 Functions Other Than Port Pins</b> .....	<b>13</b>
2.2.1 With functions for each product .....	13
2.2.2 Function descriptions .....	14
<b>2.3 Description of Pin Functions</b> .....	<b>16</b>
2.3.1 P10 to P17 (port 1) .....	16
2.3.2 P20 to P27 (port 2) .....	16
2.3.3 P30 to P35 (port 3) .....	17
2.3.4 P40 (port 4) .....	18
2.3.5 P60, P61 (port 6) .....	19
2.3.6 P120, P122 (port 12) .....	19
2.3.7 P137 (port 13) .....	20
2.3.8 VDD, VSS.....	20
2.3.9 RESET .....	20
2.3.10 REGC.....	20
<b>2.4 Connection of Unused Pins</b> .....	<b>21</b>
<b>CHAPTER 3 CPU ARCHITECTURE</b> .....	<b>24</b>
<b>3.1 Memory Space</b> .....	<b>24</b>
3.1.1 Internal program memory space.....	28
3.1.2 Mirror area.....	30
3.1.3 Internal data memory space .....	32

3.1.4	Special function register (SFR) area .....	32
3.1.5	Extended special function register (2nd SFR: 2nd Special Function Register) area .....	32
3.1.6	Data memory addressing .....	33
<b>3.2</b>	<b>Processor Registers.....</b>	<b>34</b>
3.2.1	Control registers .....	34
3.2.2	General-purpose registers.....	36
3.2.3	ES and CS registers.....	37
3.2.4	Special function registers (SFRs) .....	38
3.2.5	Extended special function registers (2nd SFRs: 2nd Special Function Registers) .....	42
<b>3.3</b>	<b>Instruction Address Addressing.....</b>	<b>47</b>
3.3.1	Relative addressing .....	47
3.3.2	Immediate addressing .....	47
3.3.3	Table indirect addressing .....	48
3.3.4	Register direct addressing.....	49
<b>3.4</b>	<b>Addressing for Processing Data Addresses .....</b>	<b>50</b>
3.4.1	Implied addressing .....	50
3.4.2	Register addressing .....	50
3.4.3	Direct addressing .....	51
3.4.4	Short direct addressing .....	52
3.4.5	SFR addressing.....	53
3.4.6	Register indirect addressing.....	54
3.4.7	Based addressing.....	55
3.4.8	Based indexed addressing .....	59
3.4.9	Stack addressing.....	60
<b>CHAPTER 4</b>	<b>PORT FUNCTIONS .....</b>	<b>64</b>
<b>4.1</b>	<b>Port Functions .....</b>	<b>64</b>
<b>4.2</b>	<b>Port Configuration.....</b>	<b>64</b>
4.2.1	Port 1.....	65
4.2.2	Port 2.....	70
4.2.3	Port 3.....	73
4.2.4	Port 4.....	79
4.2.5	Port 6.....	80
4.2.6	Port 12.....	81
4.2.7	Port 13.....	82
<b>4.3</b>	<b>Registers Controlling Port Function .....</b>	<b>83</b>
4.3.1	Port mode registers (PMxx).....	85
4.3.2	Port registers (Pxx).....	86
4.3.3	Pull-up resistor option registers (PUxx) .....	87
4.3.4	Port mode control register 1 (PMC1) (24-pin products only) .....	88
4.3.5	A/D port configuration register (ADPC) .....	89

<b>4.4 Port Function Operations .....</b>	<b>90</b>
4.4.1 Writing to I/O port .....	90
4.4.2 Reading from I/O port .....	90
4.4.3 Operations on I/O port .....	90
<b>4.5 Settings of Port Related Register When Using Alternate Function .....</b>	<b>91</b>
<b>4.6 Cautions When Using Port Function .....</b>	<b>94</b>
4.6.1 Cautions on 1-bit manipulation instruction for port register n (Pn) .....	94
4.6.2 Cautions on specifying the pin settings .....	95
<b>CHAPTER 5 CLOCK GENERATOR .....</b>	<b>96</b>
<b>5.1 Functions of Clock Generator .....</b>	<b>96</b>
<b>5.2 Configuration of Clock Generator .....</b>	<b>97</b>
<b>5.3 Registers Controlling Clock Generator .....</b>	<b>99</b>
5.3.1 Clock operation mode control register (CMC) .....	99
5.3.2 System clock control register (CKC) .....	101
5.3.3 Clock operation status control register (CSC) .....	102
5.3.4 Oscillation stabilization time counter status register (OSTC) .....	103
5.3.5 Oscillation stabilization time select register (OSTS) .....	105
5.3.6 Peripheral enable registers 0, 1 (PER0, PER1) .....	107
5.3.7 High-speed on-chip oscillator frequency select register (HOCODIV) .....	109
5.3.8 High-speed on-chip oscillator trimming register (HIOTRM) .....	110
<b>5.4 System Clock Oscillator .....</b>	<b>111</b>
5.4.1 X1 oscillator .....	111
5.4.2 High-speed on-chip oscillator .....	114
5.4.3 Low-speed on-chip oscillator .....	114
<b>5.5 Clock Generator Operation .....</b>	<b>115</b>
<b>5.6 Controlling Clock .....</b>	<b>117</b>
5.6.1 Example of setting high-speed on-chip oscillator .....	117
5.6.2 Example of setting X1 oscillation clock .....	118
5.6.3 CPU clock status transition diagram .....	119
5.6.4 Condition before changing CPU clock and processing after changing CPU clock .....	123
5.6.5 Time required for switchover of CPU clock and system clock .....	124
5.6.6 Conditions before clock oscillation is stopped .....	124
<b>5.7 Resonator and Oscillator Constants .....</b>	<b>125</b>
<b>CHAPTER 6 TIMER ARRAY UNIT .....</b>	<b>127</b>
<b>6.1 Functions of Timer Array Unit .....</b>	<b>129</b>
6.1.1 Independent channel operation function .....	129
6.1.2 Simultaneous channel operation function .....	130
6.1.3 8-bit timer operation function (channels 1 and 3 only) .....	131
<b>6.2 Configuration of Timer Array Unit .....</b>	<b>132</b>



6.2.1	Timer count register mn (TCRmn)	136
6.2.2	Timer data register mn (TDRmn)	137
<b>6.3</b>	<b>Registers Controlling Timer Array Unit</b>	<b>138</b>
6.3.1	Peripheral enable register 0 (PER0)	139
6.3.2	Timer clock select register m (TPSm)	140
6.3.3	Timer mode register mn (TMRmn)	143
6.3.4	Timer status register mn (TSRmn)	148
6.3.5	Timer channel enable status register m (TEm)	149
6.3.6	Timer channel start register m (TSm)	150
6.3.7	Timer channel stop register m (TTm)	151
6.3.8	Timer input select register 0 (TIS0)	152
6.3.9	Timer output enable register m (TOEm)	153
6.3.10	Timer output register m (TOm)	154
6.3.11	Timer output level register m (TOLm)	155
6.3.12	Timer output mode register m (TOMm)	156
6.3.13	Noise filter enable register 1 (NFEN1)	157
6.3.14	Port mode registers 1, 3 (PM1, PM3)	158
<b>6.4</b>	<b>Basic Rules of Timer Array Unit</b>	<b>159</b>
6.4.1	Basic rules of simultaneous channel operation function	159
6.4.2	Basic rules of 8-bit timer operation function (channels 1 and 3 only)	161
<b>6.5</b>	<b>Operation of Counter</b>	<b>162</b>
6.5.1	Count clock (fTCLK)	162
6.5.2	Start timing of counter	164
6.5.3	Operation of counter	165
<b>6.6</b>	<b>Channel Output (TOmn pin) Control</b>	<b>170</b>
6.6.1	TOmn pin output circuit configuration	170
6.6.2	TOmn pin output setting	171
6.6.3	Cautions on channel output operation	172
6.6.4	Collective manipulation of TOmn bit	177
6.6.5	Timer interrupt and TOmn pin output at operation start	178
<b>6.7</b>	<b>Timer Input (TImn) Control</b>	<b>179</b>
6.7.1	TImn input circuit configuration	179
6.7.2	Noise filter	179
6.7.3	Cautions on channel input operation	180
<b>6.8</b>	<b>Independent Channel Operation Function of Timer Array Unit</b>	<b>181</b>
6.8.1	Operation as interval timer/square wave output	181
6.8.2	Operation as external event counter	187
6.8.3	Operation as input pulse interval measurement	192
6.8.4	Operation as input signal high-/low-level width measurement	196
6.8.5	Operation as delay counter	200
<b>6.9</b>	<b>Simultaneous Channel Operation Function of Timer Array Unit</b>	<b>205</b>

6.9.1	Operation as one-shot pulse output function .....	205
6.9.2	Operation as PWM function.....	212
6.9.3	Operation as multiple PWM output function .....	219
<b>6.10</b>	<b>Cautions When Using Timer Array Unit .....</b>	<b>227</b>
6.10.1	Cautions when using timer output .....	227
<b>CHAPTER 7</b>	<b>CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER.....</b>	<b>228</b>
<b>7.1</b>	<b>Functions of Clock Output/Buzzer Output Controller .....</b>	<b>228</b>
<b>7.2</b>	<b>Configuration of Clock Output/Buzzer Output Controller .....</b>	<b>230</b>
<b>7.3</b>	<b>Registers Controlling Clock Output/Buzzer Output Controller .....</b>	<b>230</b>
7.3.1	Clock output select registers n (CKSn).....	230
7.3.2	Registers controlling port functions of pins to be used for clock or buzzer output .....	232
<b>7.4</b>	<b>Operations of Clock Output/Buzzer Output Controller .....</b>	<b>233</b>
7.4.1	Operation as output pin .....	233
<b>7.5</b>	<b>Cautions of Clock Output/Buzzer Output Controller .....</b>	<b>233</b>
<b>CHAPTER 8</b>	<b>WATCHDOG TIMER .....</b>	<b>234</b>
<b>8.1</b>	<b>Functions of Watchdog Timer.....</b>	<b>234</b>
<b>8.2</b>	<b>Configuration of Watchdog Timer .....</b>	<b>235</b>
<b>8.3</b>	<b>Register Controlling Watchdog Timer.....</b>	<b>236</b>
8.3.1	Watchdog timer enable register (WDTE).....	236
<b>8.4</b>	<b>Operation of Watchdog Timer .....</b>	<b>237</b>
8.4.1	Controlling operation of watchdog timer .....	237
8.4.2	Setting overflow time of watchdog timer .....	238
8.4.3	Setting window open period of watchdog timer .....	239
8.4.4	Setting watchdog timer interval interrupt .....	240
<b>CHAPTER 9</b>	<b>A/D CONVERTER .....</b>	<b>241</b>
<b>9.1</b>	<b>Function of A/D Converter.....</b>	<b>241</b>
<b>9.2</b>	<b>Configuration of A/D Converter .....</b>	<b>243</b>
<b>9.3</b>	<b>Registers Controlling A/D Converter.....</b>	<b>245</b>
9.3.1	Peripheral enable register 0 (PER0).....	246
9.3.2	A/D converter mode register 0 (ADM0) .....	247
9.3.3	A/D converter mode register 1 (ADM1) .....	257
9.3.4	A/D converter mode register 2 (ADM2) .....	258
9.3.5	12-bit A/D conversion result register (ADCR) .....	261
9.3.6	8-bit A/D conversion result register (ADCRH) .....	262
9.3.7	Analog input channel specification register (ADS).....	263
9.3.8	Conversion result comparison upper limit setting register (ADUL) .....	265
9.3.9	Conversion result comparison lower limit setting register (ADLL) .....	265

9.3.10 A/D test register (ADTES) .....	266
9.3.11 Registers controlling port function of analog input pins .....	267
<b>9.4 A/D Converter Conversion Operations .....</b>	<b>268</b>
<b>9.5 Input Voltage and Conversion Results .....</b>	<b>270</b>
<b>9.6 A/D Converter Operation Modes .....</b>	<b>271</b>
9.6.1 Software trigger mode (select mode, sequential conversion mode) .....	271
9.6.2 Software trigger mode (select mode, one-shot conversion mode) .....	272
9.6.3 Software trigger mode (scan mode, sequential conversion mode) .....	273
9.6.4 Software trigger mode (scan mode, one-shot conversion mode) .....	274
9.6.5 Hardware trigger no-wait mode (select mode, sequential conversion mode) .....	275
9.6.6 Hardware trigger no-wait mode (select mode, one-shot conversion mode) .....	276
9.6.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode) .....	277
9.6.8 Hardware trigger no-wait mode (scan mode, one-shot conversion mode) .....	278
9.6.9 Hardware trigger wait mode (select mode, sequential conversion mode) .....	279
9.6.10 Hardware trigger wait mode (select mode, one-shot conversion mode) .....	280
9.6.11 Hardware trigger wait mode (scan mode, sequential conversion mode) .....	281
9.6.12 Hardware trigger wait mode (scan mode, one-shot conversion mode) .....	282
<b>9.7 A/D Converter Setup Flowchart .....</b>	<b>283</b>
9.7.1 Setting up software trigger mode .....	284
9.7.2 Setting up hardware trigger no-wait mode .....	285
9.7.3 Setting up hardware trigger wait mode .....	286
9.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode) .....	287
9.7.5 Setting up test mode .....	288
<b>9.8 SNOOZE Mode Function .....</b>	<b>289</b>
<b>9.9 How to Read A/D Converter Characteristics Table .....</b>	<b>293</b>
<b>9.10 Cautions for A/D Converter .....</b>	<b>295</b>
 <b>CHAPTER 10 D/A CONVERTER .....</b>	 <b>299</b>
<b>10.1 Function of D/A Converter .....</b>	<b>299</b>
<b>10.2 Configuration of D/A Converter .....</b>	<b>300</b>
<b>10.3 Configuration of A/D Converter .....</b>	<b>301</b>
10.3.1 A/D port configuration register (ADPC) .....	301
10.3.2 Peripheral enable register 1 (PER1) .....	302
10.3.3 D/A converter mode register (DAM) .....	303
10.3.4 D/A conversion value setting register i (DACSi) (i = 0, 1) .....	303
10.3.5 Port mode register 2 (PM2) .....	304
<b>10.4 Operations of D/A Converter .....</b>	<b>305</b>
10.4.1 Operation in normal mode .....	305
10.4.2 Operation in real-time output mode .....	306
<b>10.5 Cautions for D/A Converter .....</b>	<b>307</b>

<b>CHAPTER 11 SERIAL ARRAY UNIT</b> .....	<b>308</b>
<b>11.1 Functions of Serial Array Unit</b> .....	<b>309</b>
11.1.1 3-wire serial I/O (CSI00).....	309
11.1.2 UART (UART0) .....	310
<b>11.2 Configuration of Serial Array Unit</b> .....	<b>311</b>
11.2.1 Shift register .....	313
11.2.2 Lower 8/9 bits of the serial data register mn (SDRmn).....	313
<b>11.3 Registers Controlling Serial Array Unit</b> .....	<b>315</b>
11.3.1 Peripheral enable register 0 (PER0).....	316
11.3.2 Serial clock select register m (SPSm) .....	317
11.3.3 Serial mode register mn (SMRmn) .....	318
11.3.4 Serial communication operation setting register mn (SCRmn) .....	319
11.3.5 Higher 7 bits of the serial data register mn (SDRmn).....	322
11.3.6 Serial flag clear trigger register mn (SIRmn) .....	323
11.3.7 Serial status register mn (SSRmn).....	324
11.3.8 Serial channel start register m (SSm).....	326
11.3.9 Serial channel stop register m (STm) .....	327
11.3.10 Serial channel enable status register m (SEm) .....	328
11.3.11 Serial output enable register m (SOEm).....	329
11.3.12 Serial output register m (SOM).....	330
11.3.13 Serial output level register m (SOLm) .....	331
11.3.14 Serial standby control register m (SSCm) .....	332
11.3.15 Input switch control register (ISC) .....	333
11.3.16 Noise filter enable register 0 (NFEN0).....	334
11.3.17 Port mode register 3 (PM3) .....	335
<b>11.4 Operation Stop Mode</b> .....	<b>336</b>
11.4.1 Stopping the operation by units .....	337
11.4.2 Stopping the operation by channels .....	338
<b>11.5 Operation of 3-Wire Serial I/O (CSI00) Communication</b> .....	<b>339</b>
11.5.1 Master transmission .....	341
11.5.2 Master reception.....	350
11.5.3 Master transmission/reception.....	359
11.5.4 Slave transmission .....	368
11.5.5 Slave reception.....	377
11.5.6 Slave transmission/reception.....	383
11.5.7 SNOOZE mode function (CSI00) .....	392
11.5.8 Calculating transfer clock frequency.....	396
11.5.9 Procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication ..	398
<b>11.6 Clock Synchronous Serial Communication with Slave Select Input Function</b> .....	<b>399</b>
11.6.1 Slave transmission .....	402
11.6.2 Slave reception.....	412

11.6.3	Slave transmission/reception.....	419
11.6.4	Calculating transfer clock frequency.....	429
11.6.5	Procedure for processing errors that occurred during slave select input function communication.....	431
<b>11.7</b>	<b>Operation of UART (UART0) Communication .....</b>	<b>432</b>
11.7.1	UART transmission .....	434
11.7.2	UART reception.....	444
11.7.3	SNOOZE mode function.....	451
11.7.4	Calculating baud rate .....	457
11.7.5	Procedure for processing errors that occurred during UART (UART0) communication .....	461
<b>CHAPTER 12</b>	<b>SERIAL INTERFACE IICA .....</b>	<b>462</b>
<b>12.1</b>	<b>Functions of Serial Interface IICA.....</b>	<b>462</b>
<b>12.2</b>	<b>Configuration of Serial Interface IICA .....</b>	<b>465</b>
<b>12.3</b>	<b>Registers Controlling Serial Interface IICA.....</b>	<b>468</b>
12.3.1	Peripheral enable register 0 (PER0).....	468
12.3.2	IICA control register n0 (IICCTLn0) .....	469
12.3.3	IICA status register n (IICSn).....	474
12.3.4	IICA flag register n (IICFn).....	476
12.3.5	IICA control register n1 (IICCTLn1) .....	478
12.3.6	IICA low-level width setting register n (IICWLn) .....	480
12.3.7	IICA high-level width setting register n (IICWHn) .....	480
12.3.8	Port mode register 6 (PM6) .....	481
<b>12.4</b>	<b>I<sup>2</sup>C Bus Mode Functions .....</b>	<b>482</b>
12.4.1	Pin configuration.....	482
12.4.2	Setting transfer clock by using IICWLn and IICWHn registers.....	483
<b>12.5</b>	<b>I<sup>2</sup>C Bus Definitions and Control Methods .....</b>	<b>485</b>
12.5.1	Start conditions.....	485
12.5.2	Addresses .....	486
12.5.3	Transfer direction specification.....	486
12.5.4	Acknowledge (ACK) .....	487
12.5.5	Stop condition.....	488
12.5.6	Wait .....	489
12.5.7	Canceling wait .....	491
12.5.8	Interrupt request (INTIICAn) generation timing and wait control.....	492
12.5.9	Address match detection method .....	493
12.5.10	Error detection.....	493
12.5.11	Extension code.....	493
12.5.12	Arbitration .....	494
12.5.13	Wake up function.....	496
12.5.14	Communication reservation.....	499
12.5.15	Cautions .....	503

12.5.16	Communication operations.....	504
12.5.17	Timing of I <sup>2</sup> C interrupt request (INTIICAn) occurrence.....	511
<b>12.6</b>	<b>Timing Charts .....</b>	<b>532</b>
<b>CHAPTER 13</b>	<b>DMA CONTROLLER .....</b>	<b>547</b>
<b>13.1</b>	<b>Functions of DMA Controller .....</b>	<b>547</b>
<b>13.2</b>	<b>Configuration of DMA Controller .....</b>	<b>548</b>
13.2.1	DMA SFR address register n (DSAn).....	548
13.2.2	DMA RAM address register n (DRAn).....	549
13.2.3	DMA byte count register n (DBCn).....	550
<b>13.3</b>	<b>Registers Controlling DMA Controller .....</b>	<b>551</b>
13.3.1	DMA mode control register n (DMCn).....	552
13.3.2	DMA operation control register n (DRCn).....	554
<b>13.4</b>	<b>Operation of DMA Controller.....</b>	<b>555</b>
13.4.1	Operation procedure .....	555
13.4.2	Transfer mode.....	556
13.4.3	Termination of DMA transfer .....	556
<b>13.5</b>	<b>Example of Setting of DMA Controller .....</b>	<b>557</b>
13.5.1	CSI consecutive transmission .....	557
13.5.2	Consecutive capturing of A/D conversion results .....	559
13.5.3	UART consecutive reception + ACK transmission.....	561
13.5.4	Holding DMA transfer pending by DWAITn bit .....	563
13.5.5	Forced termination by software .....	564
<b>13.6</b>	<b>Cautions on Using DMA Controller .....</b>	<b>566</b>
<b>CHAPTER 14</b>	<b>EVENT LINK CONTROLLER (ELC).....</b>	<b>568</b>
<b>14.1</b>	<b>Functions of ELC.....</b>	<b>568</b>
<b>14.2</b>	<b>Configuration of ELC .....</b>	<b>568</b>
<b>14.3</b>	<b>Registers Controlling ELC.....</b>	<b>569</b>
14.3.1	Event output destination select register n (ELSELRn) (n = 00 to 09) .....	569
<b>14.4</b>	<b>Operation.....</b>	<b>571</b>
<b>CHAPTER 15</b>	<b>INTERRUPT FUNCTIONS.....</b>	<b>572</b>
<b>15.1</b>	<b>Interrupt Function Types .....</b>	<b>572</b>
<b>15.2</b>	<b>Interrupt Sources and Configuration .....</b>	<b>572</b>
<b>15.3</b>	<b>Registers Controlling Interrupt Functions.....</b>	<b>576</b>
15.3.1	Interrupt request flag registers (IF0L, IF0H, IF1L) .....	578
15.3.2	Interrupt mask flag registers (MK0L, MK0H, MK1L) .....	579
15.3.3	Priority specification flag registers (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L) .....	580

15.3.4 External interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0).....	581
15.3.5 Program status word (PSW).....	582
<b>15.4 Interrupt Servicing Operations .....</b>	<b>583</b>
15.4.1 Maskable interrupt request acknowledgment .....	583
15.4.2 Software interrupt request acknowledgment .....	586
15.4.3 Multiple interrupt servicing.....	586
15.4.4 Interrupt request hold .....	590
<b>CHAPTER 16 STANDBY FUNCTION .....</b>	<b>591</b>
<b>16.1 Standby Function .....</b>	<b>591</b>
<b>16.2 Registers Controlling Standby Function .....</b>	<b>592</b>
16.2.1 Oscillation stabilization time counter status register (OSTC).....	593
16.2.2 Oscillation stabilization time select register (OSTS).....	594
<b>16.3 Standby Function Operation .....</b>	<b>595</b>
16.3.1 HALT mode .....	595
16.3.2 STOP mode.....	599
16.3.3 SNOOZE mode .....	604
<b>CHAPTER 17 RESET FUNCTION.....</b>	<b>606</b>
<b>17.1 Timing of Reset Operation .....</b>	<b>608</b>
<b>17.2 Register for Confirming Reset Source.....</b>	<b>614</b>
17.2.1 Reset control flag register (RESF).....	614
<b>CHAPTER 18 POWER-ON-RESET CIRCUIT .....</b>	<b>617</b>
<b>18.1 Functions of Power-on-reset Circuit .....</b>	<b>617</b>
<b>18.2 Configuration of Power-on-reset Circuit.....</b>	<b>618</b>
<b>18.3 Operation of Power-on-reset Circuit .....</b>	<b>618</b>
<b>18.4 Cautions for Power-on-reset Circuit.....</b>	<b>622</b>
<b>CHAPTER 19 VOLTAGE DETECTOR .....</b>	<b>724</b>
<b>19.1 Functions of Voltage Detector .....</b>	<b>724</b>
<b>19.2 Configuration of Voltage Detector .....</b>	<b>725</b>
<b>19.3 Registers Controlling Voltage Detector .....</b>	<b>725</b>
19.3.1 Voltage detection register (LVIM) .....	726
19.3.2 Voltage detection level register (LVIS) .....	727
<b>19.4 Operation of Voltage Detector .....</b>	<b>729</b>
19.4.1 When used as reset mode.....	729
19.4.2 When used as interrupt mode .....	731
19.4.3 When used as interrupt and reset mode .....	733

19.5 Cautions for Voltage Detector.....	739
<b>CHAPTER 20 SAFETY FUNCTIONS.....</b>	<b>641</b>
20.1 Overview of Safety Functions.....	641
20.2 Registers Used by Safety Functions.....	642
20.3 Operation of Flash memory CRC operation function (high-speed CRC) .....	642
20.3.1 Flash memory CRC control register (CRC0CTL) .....	643
20.3.2 Flash memory CRC operation result register (PGCRCL) .....	643
20.3.3 Operation flow .....	644
20.4 CRC operation function (general-purpose CRC) .....	645
20.4.1 CRC input register (CRCIN) .....	645
20.4.2 CRC data register (CRCD).....	646
20.4.3 Operation flow .....	646
20.5 RAM parity error detection function.....	647
20.5.1 RAM parity error control register (RPECTL).....	647
20.6 RAM guard function .....	649
20.6.1 Invalid memory access detection control register (IAWCTL).....	649
20.7 SFR guard function .....	650
20.7.1 Invalid memory access detection control register (IAWCTL).....	650
20.8 Invalid memory access detection function.....	651
20.8.1 Invalid memory access detection control register (IAWCTL).....	652
20.9 Frequency detection function .....	653
20.9.1 Timer input select register 0 (TIS0) .....	654
20.10 A/D test function .....	655
<b>CHAPTER 21 REGULATOR .....</b>	<b>659</b>
21.1 Regulator Overview.....	659
<b>CHAPTER 22 OPTION BYTE.....</b>	<b>660</b>
22.1 Functions of Option Bytes .....	660
22.1.1 User option byte (000C0H to 000C2H).....	660
22.1.2 On-chip debug option byte (000C3H).....	660
22.2 Format of User Option Byte .....	661
22.3 Format of On-chip Debug Option Byte.....	664
22.4 Setting of Option Byte.....	665
<b>CHAPTER 23 FLASH MEMORY .....</b>	<b>666</b>
23.1 Serial Programming Using Flash Memory Programmer .....	667
23.1.1 Programming environment .....	669
23.1.2 Communication mode .....	669



<b>23.2 Serial Programming Using External Device (that Incorporates UART)</b> .....	<b>670</b>
23.2.1 Programming environment .....	670
23.2.2 Communication mode .....	671
<b>23.3 Connection of Pins on Board</b> .....	<b>672</b>
23.3.1 P40/TOOL0 pin .....	672
23.3.2 RESET pin.....	672
23.3.3 Port pins .....	673
23.3.4 REGC pin .....	673
23.3.5 X1 and X2 pins .....	673
23.3.6 Power supply .....	673
<b>23.4 Data Flash</b> .....	<b>674</b>
23.4.1 Data flash overview .....	674
23.4.2 Register controlling data flash memory .....	675
23.4.3 Procedure for accessing data flash memory .....	676
<b>23.5 Programming Method</b> .....	<b>677</b>
23.5.1 Controlling flash memory.....	677
23.5.2 Flash memory programming mode.....	678
23.5.3 Selecting communication mode.....	679
23.5.4 Communication commands .....	680
23.5.5 Description of signature data.....	682
<b>23.6 Security Settings</b> .....	<b>683</b>
<b>23.7 Flash Memory Programming by Self-programming</b> .....	<b>685</b>
23.7.1 Flash shield window function.....	687
<b>23.8 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)</b> .....	<b>688</b>
<b>CHAPTER 24 ON-CHIP DEBUG FUNCTION</b> .....	<b>689</b>
<b>24.1 Connecting E1 On-chip Debugging Emulator to R7F0C010</b> .....	<b>689</b>
<b>24.2 On-chip Debug Security ID</b> .....	<b>690</b>
<b>24.3 Securing of User Resources</b> .....	<b>690</b>
<b>CHAPTER 25 BCD CORRECTION CIRCUIT</b> .....	<b>692</b>
<b>25.1 BCD Correction Circuit Function</b> .....	<b>692</b>
<b>25.2 Registers Used by BCD Correction Circuit</b> .....	<b>692</b>
25.2.1 BCD correction result register (BCDADJ).....	692
<b>25.3 BCD Correction Circuit Operation</b> .....	<b>693</b>
<b>CHAPTER 26 INSTRUCTION SET</b> .....	<b>695</b>
<b>26.1 Conventions Used in Operation List</b> .....	<b>696</b>
26.1.1 Operand identifiers and specification methods.....	696
26.1.2 Description of operation column .....	697

26.1.3 Description of flag operation column .....	698
26.1.4 PREFIX instruction .....	698
<b>26.2 Operation List .....</b>	<b>699</b>
<b>CHAPTER 27 ELECTRICAL SPECIFICATIONS .....</b>	<b>717</b>
<b>27.1 Absolute Maximum Ratings .....</b>	<b>718</b>
<b>27.2 Oscillator Characteristics .....</b>	<b>720</b>
27.2.1 X1 oscillator characteristics .....	720
27.2.2 On-chip oscillator characteristics .....	720
<b>27.3 DC Characteristics .....</b>	<b>721</b>
27.3.1 Pin characteristics .....	721
27.3.2 Supply current characteristics .....	724
<b>27.4 AC Characteristics .....</b>	<b>727</b>
<b>27.5 Peripheral Functions Characteristics.....</b>	<b>731</b>
27.5.1 Serial array unit .....	731
27.5.2 Serial interface IICA .....	737
27.5.3 On-chip debug (UART).....	738
<b>27.6 Analog Characteristics .....</b>	<b>738</b>
27.6.1 A/D converter characteristics.....	738
27.6.2 Temperature sensor/internal reference voltage characteristics .....	741
27.6.3 D/A converter .....	741
27.6.4 POR circuit characteristics .....	742
27.6.5 LVD circuit characteristics .....	743
<b>27.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics.....</b>	<b>744</b>
<b>27.8 Flash Memory Programming Characteristics.....</b>	<b>744</b>
<b>27.9 Timing for Switching Flash Memory Programming Modes.....</b>	<b>745</b>
<b>CHAPTER 28 PACKAGE DRAWINGS .....</b>	<b>746</b>
<b>28.1 24-pin Products .....</b>	<b>746</b>
<b>28.2 32-pin Products .....</b>	<b>747</b>
<b>APPENDIX A REVISION HISTORY .....</b>	<b>748</b>
<b>A.1 Major Revisions in This Edition.....</b>	<b>748</b>

## CHAPTER 1 OUTLINE

### 1.1 Features

- <R> ○ Minimum instruction execution time can be changed from high speed (0.03125  $\mu$ s: @ 32 MHz operation with high-speed on-chip oscillator) to low-speed (1  $\mu$ s: @ 1 MHz operation with high-speed on-chip oscillator)
- General-purpose registers: 8 bits  $\times$  32 registers (8 bits  $\times$  8 registers  $\times$  4 banks)
- ROM: 16 KB, RAM: 1.5 KB, Data flash: 2 KB
- High-speed on-chip oscillator
  - Select from 32 MHz (TYP.), 24 MHz (TYP.), 16 MHz (TYP.), 12 MHz (TYP.), 8 MHz (TYP.), 4 MHz (TYP.), and 1 MHz (TYP.)
- On-chip single-power supply flash memory (with prohibition of block erase/writing function)
- Self-programming
- On-chip debug function
- On-chip power-on-reset (POR) circuit and voltage detector (LVD)
- <R> ○ On-chip watchdog timer (operable with the dedicated low-speed on-chip oscillator)
- On-chip clock output/buzzer output controller
- On-chip BCD adjustment
- I/O ports: 26 or 28 (N-ch open drain: 2)
- Timer
  - 16-bit timer: TAU: 4 channels
  - Watchdog timer: 1 channel
- Serial interface
  - CSI
  - UART
  - I<sup>2</sup>C
- 8/12-bit resolution A/D converter ( $V_{DD} = 2.7$  to 3.6 V): 6 or 8 channels
- Standby function: HALT, STOP, SNOOZE mode
- On-chip D/A converter
- DMA controller: 2 channels
- On-chip event link controller (ELC)
- Power supply voltage:  $V_{DD} = 2.7$  to 3.6 V
- Operating ambient temperature:  $T_A = -40$  to  $+85^\circ\text{C}$

**Remarks** The functions mounted depend on the product. See **1.6 Outline of Functions**.

○ ROM, RAM capacities

<R>

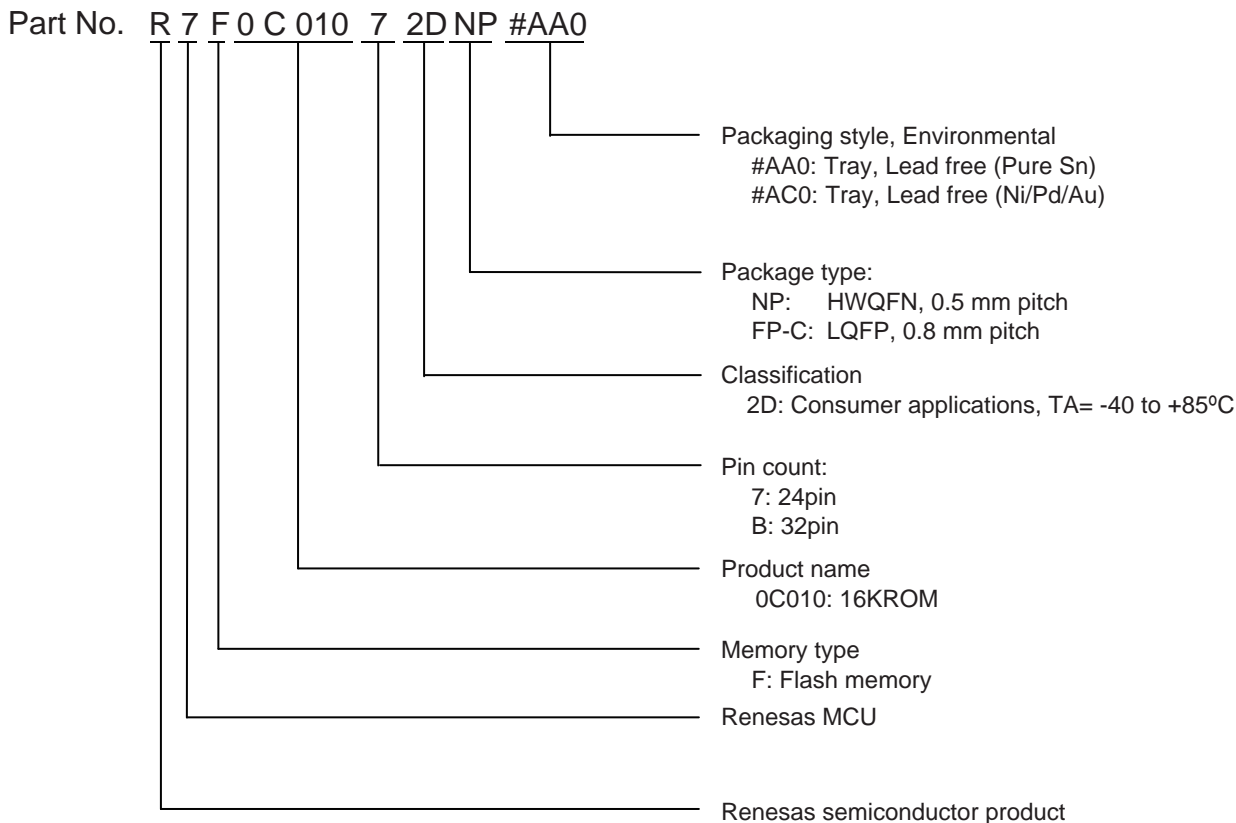
Flash ROM	Data Flash	RAM	24 Pins	32 Pins
16 KB	2 KB	1.5 KB	R7F0C01072DNP	R7F0C010B2DFP-C

**Note** This is about 0.5 KB when the self-programming function and data flash function are used. (For details, see **CHAPTER 3**)

### 1.2 List of Part Numbers

<R>

Figure 1 - 1 Part Number, Memory Size, and Package



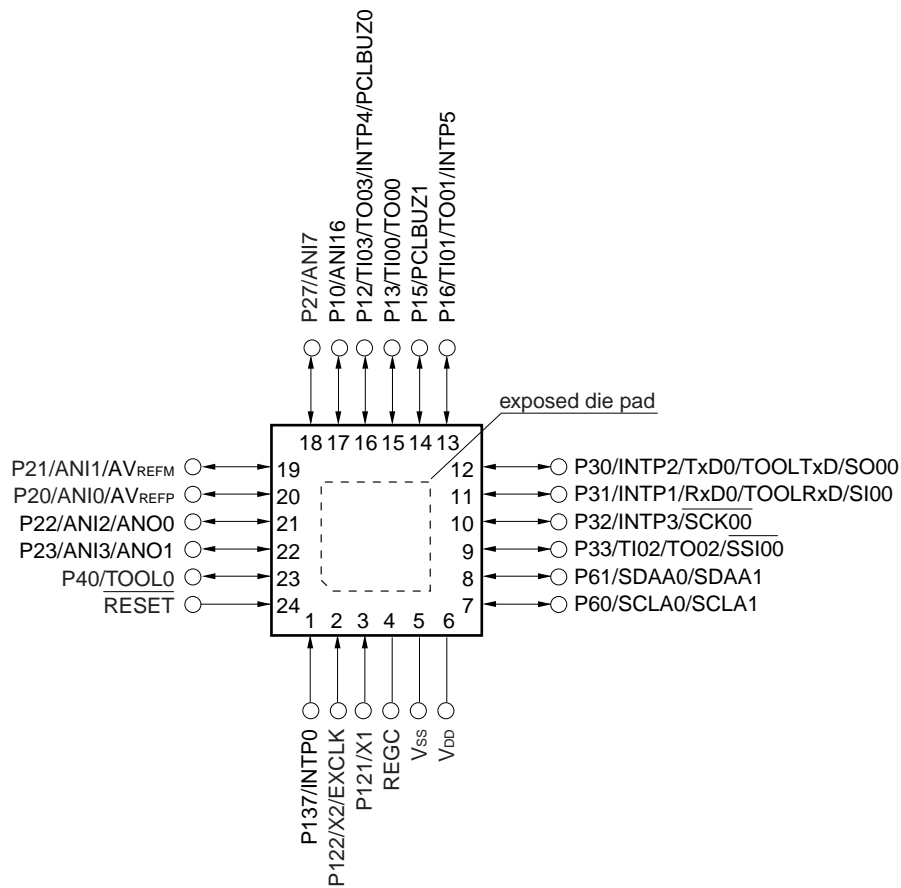
<R>

Pin Count	Package	Data Flash	Packaging style, Environmental	Part Number
24 pins	24-pin plastic WQFN (4 × 4)	2KB	Tray, Lead free (Pure Sn)	R7F0C01072DNP#AA0
			Tray, Lead free (Ni/Pd/Au)	R7F0C01072DNP#AC0
32 pins	32-pin plastic LQFP (7 × 7)		Tray, Lead free (Pure Sn)	R7F0C010B2DNP#AA0
			Tray, Lead free (Ni/Pd/Au)	R7F0C010B2DNP#AC0

### 1.3 Pin Configuration (Top View)

#### 1.3.1 24-pin products

- 24-pin plastic WQFN (4 × 4)

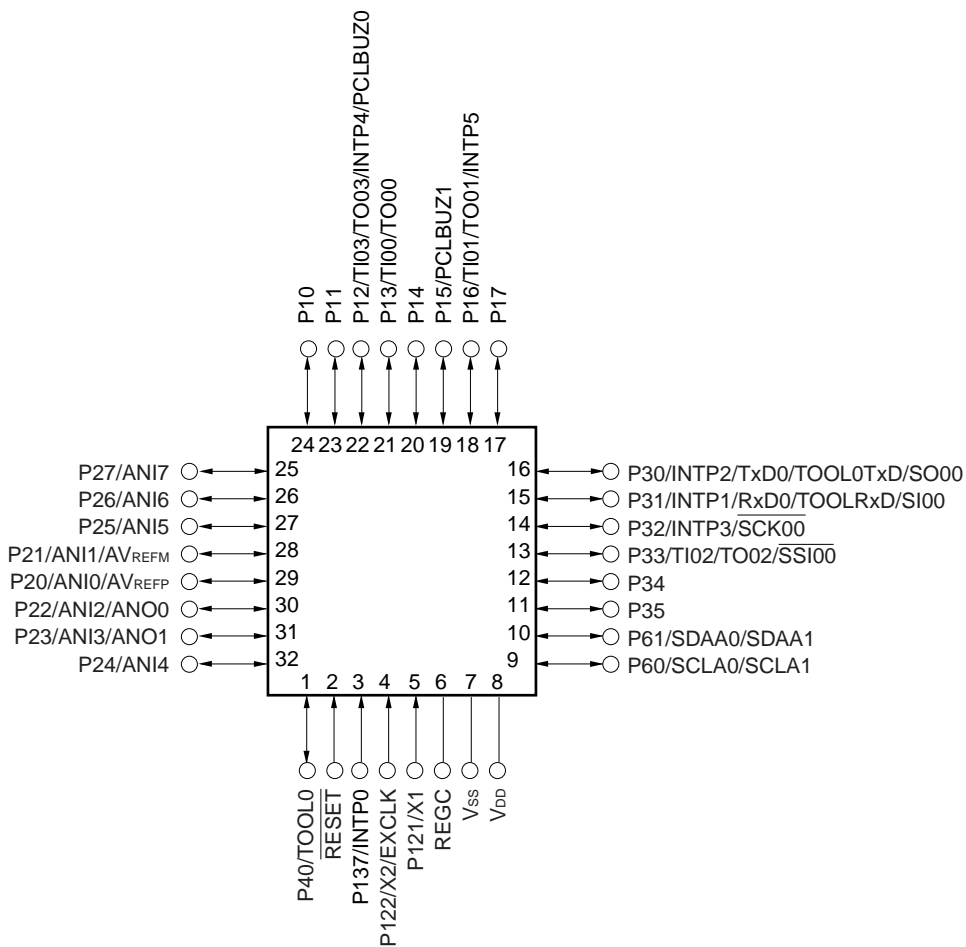


**Caution** Connect the REGC pin to V<sub>ss</sub> via a capacitor (0.47 to 1  $\mu$ F).

**Remark** For pin identification, see 1.4 Pin Identification.

### 1.3.2 32-pin products

- 32-pin plastic LQFP (7 × 7)



**Caution** Connect the REGC pin to V<sub>SS</sub> via a capacitor (0.47 to 1  $\mu$ F).

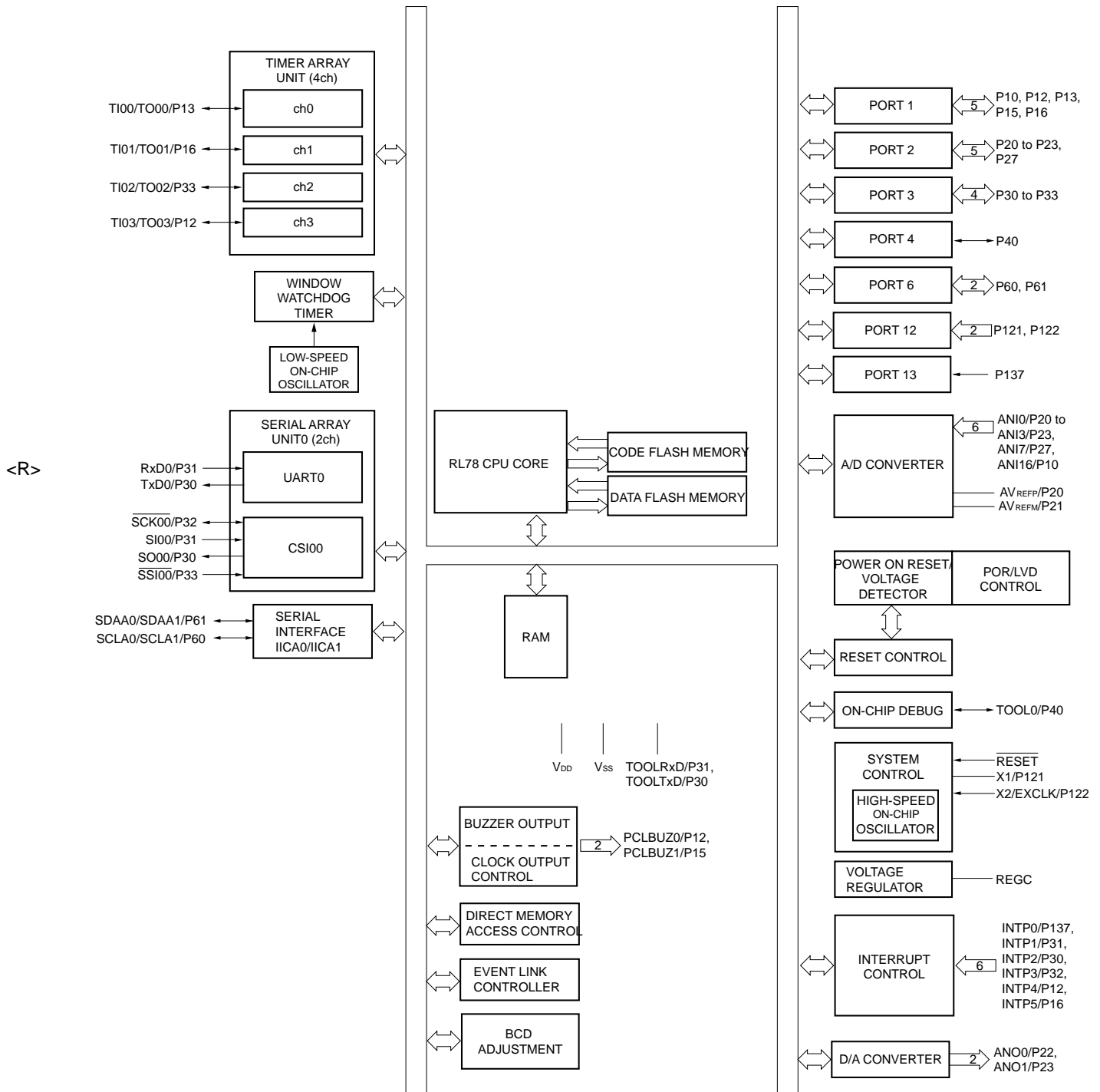
**Remark** For pin identification, see 1.4 Pin Identification.

## 1.4 Pin Identification

ANI0 to ANI7, ANI16:	Analog input	RxD0:	Receive data
ANO0, ANO1:	Analog output	$\overline{\text{SCK00}}$ :	Serial clock input/output
AVREFM:	Analog reference voltage minus	SCLA0, SCLA1:	Serial clock input/output
AVREFP:	Analog reference voltage plus	SDAA0, SDAA1:	Serial data input/output
EXCLK:	External clock input (main system clock)	SI00:	Serial data input
INTP0 to INTP5:	External Interrupt Input	SO00:	Serial data output
P10 to P17:	Port 1	$\overline{\text{SSI00}}$ :	Serial interface chip select input
P20 to P27:	Port 2	TI00 to TI03:	Timer input
P30 to P35:	Port 3	TO00 to TO03:	Timer output
P40:	Port 4	TOOL0:	Data input/output for tool
P60, P61:	Port 6	TOOLRxD, TOOLTxD:	Data input/output for external device
P121, P122:	Port 12	TxD0:	Transmit data
P137:	Port 13	V <sub>DD</sub> :	Power supply
PCLBUZ0, PCLBUZ1:	Programmable clock output/buzzer output	V <sub>SS</sub> :	Ground
REGC:	Regulator capacitance	X1, X2:	Crystal oscillator (main system clock)
$\overline{\text{RESET}}$ :	Reset		

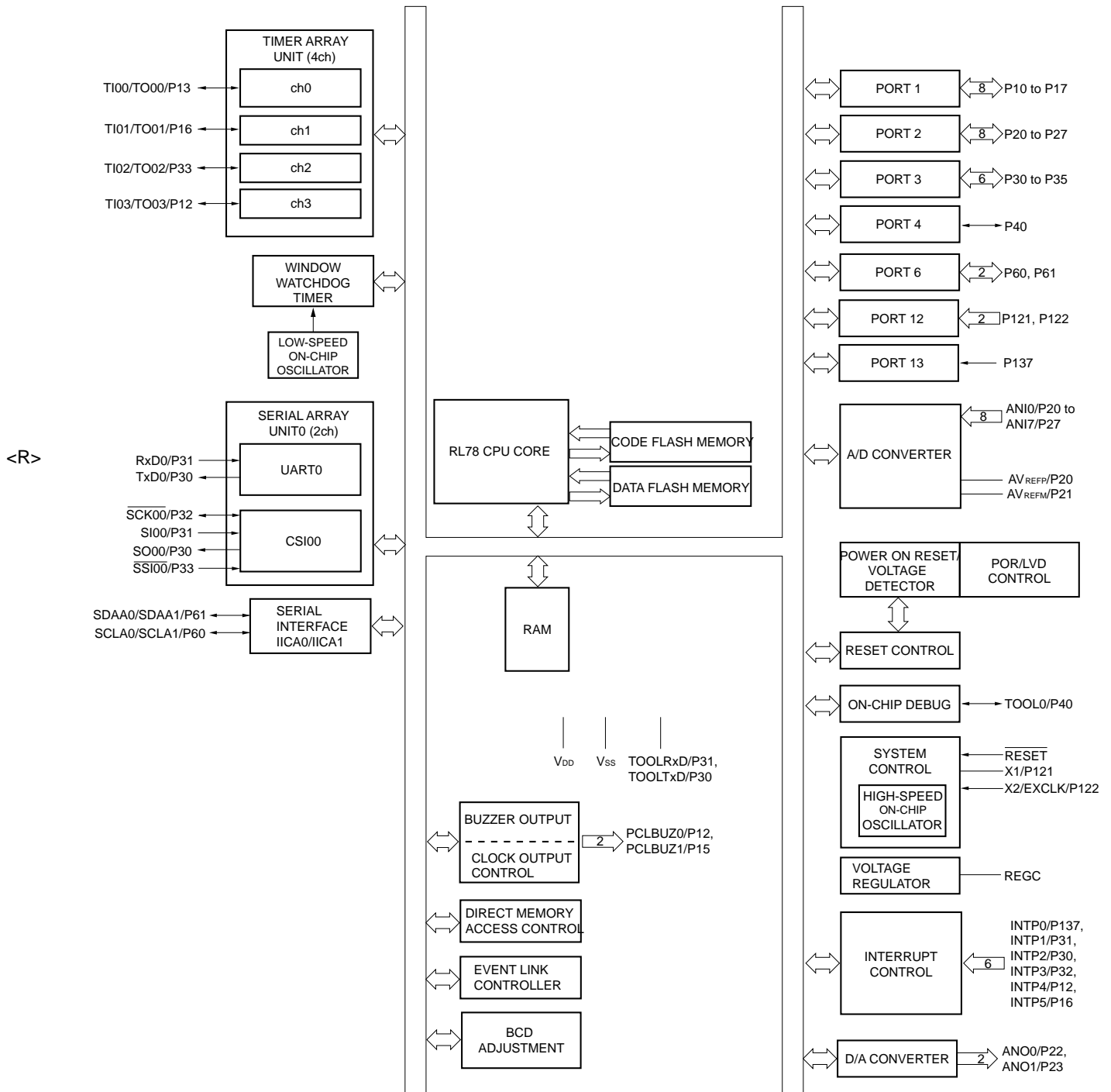
### 1.5 Block Diagram

#### 1.5.1 24-pin products





1.5.2 32-pin products



## 1.6 Outline of Functions

(1/2)

Item		24-pin	32-pin
		R7F0C01072DNP	R7F0C010B2DFP-C
Code flash memory		16 KB	
Data flash memory		2 KB	
RAM		1.5 KB <sup>Note</sup>	
Memory space		1 MB	
Main system clock	High-speed system clock	X1 (crystal/ceramic) oscillation, external main system clock input (EXCLK) 1 to 20 MHz: $V_{DD} = 2.7$ to $3.6$ V	
	High-speed on-chip oscillator clock ( $f_{IH}$ )	High-speed operation: 32 MHz ( $V_{DD} = 2.7$ to $3.6$ V)	
Low-speed on-chip oscillator clock		15 kHz (TYP.): $V_{DD} = 2.7$ to $3.6$ V	
General-purpose registers		8 bits $\times$ 32 registers (8 bits $\times$ 8 registers $\times$ 4 banks)	
Minimum instruction execution time		0.03125 $\mu$ s (High-speed on-chip oscillator: $f_{IH} = 32$ MHz operation)	
		0.05 $\mu$ s (High-speed system clock: $f_{MX} = 20$ MHz operation)	
<R>		Instruction set	
		<ul style="list-style-type: none"> <li>• Data transfer (8/16 bits)</li> <li>• Adder and subtractor/logical operation (8/16 bits)</li> <li>• Multiplication (8 bits <math>\times</math> 8 bits)</li> <li>• Rotate, barrel shift, and bit manipulation (set, reset, test, and boolean operation), etc.</li> </ul>	
I/O port	Total	20	28
	CMOS I/O	15	23
	CMOS input	3	3
	N-ch O.D I/O (6 V tolerance)	2	2
Timer	16-bit timer	4 channels (TAU)	
	Watchdog timer	1 channel	
	Timer output	4 PWM outputs: 3	
<R>	Clock output/buzzer output		2
	<ul style="list-style-type: none"> <li>• 2.44 kHz, 4.88 kHz, 9.77 kHz, 1.25 MHz, 2.5 MHz, 5 MHz, 10 MHz (Main system clock: <math>f_{MAIN} = 20</math> MHz operation)</li> </ul>		
8/12-bit resolution A/D converter		6 channels	8 channels
D/A converter		2 channels	
Serial interface		• CSI: 1 channel/UART: 1 channel	
		I <sup>2</sup> C bus	1 channel (2 slave addresses)
DMA controller		2 channels	
Event link controller (ELC)		Event input: 10, Event trigger output: 3	
Vectored interrupt sources	Internal	12	
	External	6	

**Note** This is about 0.5 KB when the self-programming function and data flash function are used. (For details, see **CHAPTER 3**)

(2/2)

Item	24-pin	32-pin
	R7F0C01072DNP	R7F0C010B2DFP-C
Reset	<ul style="list-style-type: none"> <li>• Reset by <math>\overline{\text{RESET}}</math> pin</li> <li>• Internal reset by watchdog timer</li> <li>• Internal reset by power-on-reset</li> <li>• Internal reset by voltage detector</li> <li>• Internal reset by illegal instruction execution<sup>Note</sup></li> <li>• Internal reset by RAM parity error</li> <li>• Internal reset by illegal-memory access</li> </ul>	
Power-on-reset circuit	<ul style="list-style-type: none"> <li>• Power-on-reset: 1.51 <math>\pm</math>0.03 V</li> <li>• Power-down-reset: 1.50 <math>\pm</math>0.03 V</li> </ul>	
Voltage detector	2.75 V to 3.13 V (4 stages)	
On-chip debug function	Provided	
Power supply voltage	$V_{\text{DD}} = 2.7$ to 3.6 V	
Operating ambient temperature	$T_{\text{A}} = -40$ to +85 °C	

**Note** The illegal instruction is generated when instruction code FFH is executed.

Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

## CHAPTER 2 PIN FUNCTIONS

### 2.1 Pin Function List

Pin I/O buffer power supplies are unique for all products. The relationship between these power supplies and the pins is shown below.

<R> The input and output, buffer, and pull-up resistor settings for each port are also valid for the alternate function.

**Table 2-1. Pin I/O Buffer Power Supplies**

Power Supply	Corresponding Pins
V <sub>DD</sub>	All pins

## 2.1.1 24-pin products

Function Name	I/O	Function	After Reset	Alternate Function
P10	I/O	Port 1. 5-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	ANI16
P12				TI03/TO03/INTP4/PCLBUZ0
P13				TI00/TO00
P15				PCLBUZ1
P16				TI01/TO01/INTP5
P20				I/O
P21	ANI1/AV <sub>REFM</sub>			
P22	ANI2/ANO0			
P23	ANI3/ANO1			
P27	ANI7			
P30	I/O	Port 3. 4-bit I/O port. Input/output can be specified. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	
P31				INTP1/RxD0/ TOOLRxD/SI00
P32				INTP3/ $\overline{\text{SCK00}}$
P33				TI02/TO02/ $\overline{\text{SSI00}}$
P40	I/O	Port 4. 1-bit I/O port. Input/output can be specified. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	TOOL0
P60	I/O	Port 6. 2-bit I/O port. Output of P60 and P61 can be set to N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units.	Input port	SCLA0/SCLA1
P61				SDAA0/SDAA1
P121	Input	Port 12. 2-bit input port.	Input port	X1
P122				X2/EXCLK
P137	Input	Port 13. 1-bit input only port.	Input port	INTP0

## 2.1.2 32-pin products

Function Name	I/O	Function	After Reset	Alternate Function
P10	I/O	Port 1. 8-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	–
P11				–
P12				TI03/TO03/INTP4/ PCLBUZ0
P13				TI00/TO00
P14				–
P15				PCLBUZ1
P16				TI01/TO01/INTP5
P17				–
P20	I/O	Port 2. 8-bit I/O port. Input/output can be specified in 1-bit units.	Analog input port	ANI0/AV <sub>REFP</sub>
P21				ANI1/AV <sub>REFM</sub>
P22				ANI2/ANO0
P23				ANI3/ANO1
P24				ANI4
P25				ANI5
P26				ANI6
P27				ANI7
P30	I/O	Port 3. 6-bit I/O port. Input/output can be specified in 1-bit units. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	INTP2/TxD0/ TOOLTxD/SO00
P31				INTP1/RxD0/ TOOLRxD/SI00
P32				INTP3/SCK00
P33				TI02/TO02/SSI00
P34				–
P35				–
P40	I/O	Port 4. 1-bit I/O port. Input/output can be specified. Use of an on-chip pull-up resistor can be specified by a software setting at input port.	Input port	TOOL0
P60	I/O	Port 6. 2-bit I/O port. Output of P60 and P61 can be set to N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units.	Input port	SCLA0/SCLA1
P61				SDAA0/SDAA1
P121	Input	Port 12. 2-bit input port.	Input port	X1
P122				X2/EXCLK
P137	Input	Port 13. 1-bit input only port.	Input port	INTP0

&lt;R&gt;

## 2.2 Functions Other Than Port Pins

### 2.2.1 With functions for each product

Function Name	32-pin	24-pin
ANI0	√	√
ANI1	√	√
ANI2	√	√
ANI3	√	√
ANI4	√	–
ANI5	√	–
ANI6	√	–
ANI7	√	√
ANI16	–	√
ANO0	√	√
ANO1	√	√
INTP0	√	√
INTP1	√	√
INTP2	√	√
INTP3	√	√
INTP4	√	√
INTP5	√	√
PCLBUZ0	√	√
PCLBUZ1	√	√
REGC	√	√
RESET	√	√
RxD0	√	√
SCK00	√	√
SCLA0	√	√
SCLA1	√	√
SDAA0	√	√
SDAA1	√	√
SI00	√	√
SO00	√	√
SSI00	√	√
TI00	√	√
TI01	√	√
TI02	√	√
TI03	√	√
TO00	√	√
TO01	√	√
TO02	√	√
TO03	√	√
TxD0	√	√

Function Name	32-pin	24-pin
X1	√	√
X2	√	√
EXCLK	√	√
V <sub>DD</sub>	√	√
V <sub>SS</sub>	√	√
TOOLRxD	√	√
TOOLTxD	√	√
TOOL0	√	√

&lt;R&gt;

## 2.2.2 Function descriptions

(1/2)

Function Name	I/O	Function
ANI0	Input	A/D converter analog input
ANI1		
ANI2		
ANI3		
ANI4		
ANI5		
ANI6		
ANI7		
ANI16		
ANO0	Output	D/A converter output
ANO1		
INTP0	Input	External interrupt input
INTP1		
INTP2		
INTP3		
INTP4		
INTP5		
PCLBUZ0	Output	Clock output/buzzer output
PCLBUZ1		
REGC	–	Connecting regulator output stabilization capacitance for internal operation. Connect to V <sub>SS</sub> via a capacitor (0.47 to 1 $\mu$ F).
RESET	Input	External reset input
RxD0	Input	Serial data input to UART0
SCK00	I/O	Clock I/O for CSI00
SCLA0	I/O	Clock I/O for I <sup>2</sup> C
SCLA1		
SDAA0	I/O	Serial data I/O for I <sup>2</sup> C
SDAA1		
SI00	Input	Serial data input to CSI00
SO00	Output	Serial data output from CSI00
SSI00	Input	Chip select input to CSI00
TI00	Input	External count clock input to 16-bit timer 00
TI01		External count clock input to 16-bit timer 01
TI02		External count clock input to 16-bit timer 02
TI03		External count clock input to 16-bit timer 03
TO00	Output	16-bit timer 00 output
TO01		16-bit timer 01 output
TO02		16-bit timer 02 output
TO03		16-bit timer 03 output
TxD0		Serial data output from UART0



(2/2)

Function Name	I/O	Function
X1	–	Resonator connection for main system clock
X2	–	
EXCLK	Input	External clock input for main system clock
V <sub>DD</sub>	–	Positive power supply for all pins
V <sub>SS</sub>	–	Ground potential for all pins
TOOLRxD	Input	UART reception pin for the external device connection used during flash memory programming
TOOLTxD	Output	UART transmission pin for the external device connection used during flash memory programming
TOOL0	I/O	Data I/O for flash Memory programmer/debugger

&lt;R&gt;

**Caution** After reset release, the relationships between P40/TOOL0 and the operating mode are as follows.

**Table 2-2. Relationships Between P40/TOOL0 and Operation Mode After Reset Release**

P40/TOOL0	Operating Mode
EV <sub>DD</sub>	Normal operation mode
0 V	Flash memory programming mode

For details, see 23.5 Programming Method.

**Remark** Use bypass capacitors (about 0.1  $\mu$ F) as noise and latch up countermeasures with relatively thick wires at the shortest distance to VDD to VSS lines.

## 2.3 Description of Pin Functions

**Remark** The pins mounted depend on the product. See 1.3 **Pin Configuration (Top View)** and 2.1 **Pin Function List**.

### 2.3.1 P10 to P17 (port 1)

P10 to P17 function as an I/O port. These pins also function as timer I/O, external interrupt request input, and clock/buzzer output.

Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 1 (PU1).

The following operation modes can be specified in 1-bit units.

#### (1) Port mode

P10 to P17 function as an I/O port. P10 to P17 can be set to input or output port in 1-bit units using port mode register 1 (PM1).

#### (2) Control mode

P10 to P17 function as timer I/O, external interrupt request input, and clock/buzzer output.

##### (a) TI00, TI01, TI03

These are the pins for inputting an external count clock/capture trigger to 16-bit timers 00, 01, and 03.

##### (b) TO00, TO01, TO03

These are the timer output pins of 16-bit timers 00, 01, and 03.

##### (c) INTP4, INTP5

These are the external interrupt request input pins for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.

##### (d) PCLBUZ0, PCLBUZ1

These are the clock/buzzer output pins.

##### (e) ANI16 (24-pin products only)

This is an analog input pin (ANI16) of A/D converter. See 9.10 (5) **Analog input (ANIn) pins**.

### 2.3.2 P20 to P27 (port 2)

P20 to P27 function as an I/O port. These pins also function as A/D converter analog input and reference voltage input, and D/A converter output.

Setting digital or analog to each pin can be done in A/D port configuration register (ADPC).

The following operation modes can be specified in 1-bit units.

#### (1) Port mode

P20 to P27 function as an I/O port. P20 to P27 can be set to input or output port in 1-bit units using port mode register 2 (PM2).

#### (2) Control mode

P20 to P27 function as A/D converter analog input, reference voltage input and D/A converter output.

**(a) ANI0 to ANI7**

These are the analog input pins (ANI0 to ANI7) of A/D converter. See **9.10 (5) Analog input (ANIn) pins**.

**(b) AVREFP**

This is a pin that inputs the A/D converter reference potential (+ side).

**(c) AVREFM**

This is a pin that inputs the A/D converter reference potential (–side).

**(d) ANO0, ANO1**

These are the D/A converter output pins.

**2.3.3 P30 to P35 (port 3)**

P30 to P35 function as an I/O port. These pins also function as external interrupt request input, serial interface data I/O, clock I/O, chip select input, programming UART I/O, and timer I/O.

Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 3 (PU3).

The following operation modes can be specified in 1-bit units.

**(1) Port mode**

P30 to P35 function as an I/O port. P30 and P31 can be set to input or output port in 1-bit units using port mode register 3 (PM3).

**(2) Control mode**

P30 to P35 function as external interrupt request input, serial interface data I/O, clock I/O, chip select input, programming UART I/O, and timer I/O.

**(a) INTP1, INTP2, INTP3**

These are the external interrupt request input pins for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.

**(b) TxD0**

This is a serial data output pin of serial data interface UART0.

**(c) RxD0**

This is a serial data input pin of serial data interface UART0.

**(d) SI00**

This is a serial data input pin of serial interface CSI00.

**(e) SO00**

This is a serial data output pin of serial interface CSI00.

**(f)  $\overline{\text{SCK00}}$** 

This is a serial clock I/O pin of serial interface CSI00.

**(g)  $\overline{\text{SSI00}}$** 

This is a chip select input pin of serial interface CSI00.

**(h) TI02**

This is a pin for inputting an external count clock/capture trigger to 16-bit timer 02.

**(i) TO02**

This is a timer output pin from 16-bit timer 02.

**(j) TOOLTxD**

This is the UART serial data output pin for the external device connection used during flash memory programming.

**(k) TOOLRxD**

This is the UART serial data input pin for the external device connection used during flash memory programming.

**<R> 2.3.4 P40 (port 4)**

P40 to P47 function as an I/O port. These pins also function as data I/O for a flash memory programmer/debugger, serial interface data I/O.

Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 4 (PU4).

Be sure to connect an external pull-up resistor to P40 when on-chip debugging is enabled (by using an option byte).

The following operation modes can be specified in 1-bit units.

**(1) Port mode**

P40 to P47 function as an I/O port. P40 to P47 can be set to input or output port in 1-bit units using port mode register 4 (PM4).

**(2) Control mode**

P40 to P47 function as data I/O for a flash memory programmer/debugger.

**(a) TOOL0**

This is a data I/O pin for a flash memory programmer/debugger.

Be sure to pull up this pin externally when on-chip debugging is enabled (pulling it down is prohibited).

**Caution** After reset release, the relationships between P40/TOOL0 and the operating mode are as follows.

**Table 2-3. Relationships Between P40/TOOL0 and Operation Mode After Reset Release**

P40/TOOL0	Operating Mode
EV <sub>DD</sub>	Normal operation mode
0 V	Flash memory programming mode

**For details, see 23.5 Programming Method.**

**<R> Remark** Use bypass capacitors (about 0.1 μF) as noise and latch up countermeasures with relatively thick wires at the shortest distance to VDD to VSS lines.

### 2.3.5 P60, P61 (port 6)

P60 and P61 function as an I/O port. These pins also function as serial interface data I/O and clock I/O. Output of P60 and P61 is N-ch open-drain output (6 V tolerance).

The following operation modes can be specified in 1-bit units.

#### (1) Port mode

P60 and P61 function as an I/O port. P60 to P67 can be set to input port or output port in 1-bit units using port mode register 6 (PM6).

#### (2) Control mode

P60 and P61 function as serial interface data I/O, clock I/O, chip select input, and timer I/O.

##### (a) SCLA0, SCLA1

These are the serial clock I/O pins of serial interface IICA0 and IICA1.

##### (b) SDAA0, SDAA1

These are the serial data I/O pins of serial interface IICA0 and IICA1.

### 2.3.6 P120, P122 (port 12)

P120 and P122 function as an input port. These pins also function as connecting resonator for main system clock and external clock input for main system clock.

The following operation modes can be specified in 1-bit units.

#### (1) Port mode

P120 and P122 function as an input port.

#### (2) Control mode

P120 and P122 function as connecting resonator for main system clock and external clock input for main system clock.

##### (a) X1, X2

These are the pins for connecting a resonator for main system clock.

##### (b) EXCLK

This is an external clock input pin for main system clock.

### 2.3.7 P137 (port 13)

P137 functions as an input port. P137 pin also functions as external interrupt request input.

#### (1) Port mode

P137 functions as an input port.

#### (2) Control mode

P137 functions as external interrupt request input.

##### (a) INTPO

This is an external interrupt request input pin for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified.

### 2.3.8 V<sub>DD</sub>, V<sub>SS</sub>

#### (1) V<sub>DD</sub>

V<sub>DD</sub> is the positive power supply pin.

#### (2) V<sub>SS</sub>

V<sub>SS</sub> is the ground potential pin. .

**Remark** Use bypass capacitors (about 0.1  $\mu\text{F}$ ) as noise and latch up countermeasures with relatively thick wires at the shortest distance to V<sub>DD</sub> to V<sub>SS</sub> line.

### 2.3.9 RESET

This is the active-low system reset input pin.

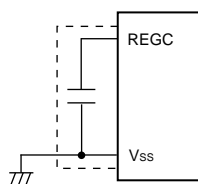
When the external reset pin is not used, connect this pin directly or via a resistor to V<sub>DD</sub>.

When the external reset pin is used, design the circuit based on V<sub>DD</sub>.

### 2.3.10 REGC

This is the pin for connecting regulator output stabilization capacitance for internal operation. Connect this pin to V<sub>SS</sub> via a capacitor (0.47 to 1  $\mu\text{F}$ ).

Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.



**Caution** Keep the wiring length as short as possible for the broken-line part in the above figure.

&lt;R&gt;

## 2.4 Connection of Unused Pins

Tables 2 - 3 and 2 - 4 show the Connection of Unused Pins.

**Table 2-4. Connection of Unused Pins**

Pin Name	I/O Circuit Type	I/O	Recommended Connection of Unused Pins
P10/ANI16 <sup>Note 1</sup>	11-U	I/O	Input: Independently connect to V <sub>DD</sub> or V <sub>SS</sub> via a resistor. Output: Leave open.
P11 <sup>Note 2</sup>	8-R		
P12/TI03/TO03/INTP4/ PCLBUZ0			
P13/TI00/TO00			
P14 <sup>Note 2</sup>			
P15/PCLBUZ1			
P16/TI01/TO01/INTP5			
P17 <sup>Note 2</sup>			
P20/ANI0/AV <sub>REFP</sub>			
P21/ANI1/AV <sub>REFM</sub>			
P22/ANI2/ANO0	44		
P23/ANI3/ANO1			
P24/ANI4 <sup>Note 2</sup>	11-G		
P25/ANI5 <sup>Note 2</sup>			
P26/ANI6 <sup>Note 2</sup>			
P27/ANI7			
P30/INTP2/TxD0 /TOOLTxD/SO00	8-R		
P31/INTP1/RxD0/ TOOLRxD/SI00			
P32/INTP3/SCK00			
P33/TI02/TO02/SSI00			
P34 <sup>Note 2</sup>			
P35 <sup>Note 2</sup>			
P40/TOOL0			
P60/SCLA0/SCLA1		13-R	
P61/SDAA0/SDAA1			
P121/X1	37-C	Input	Independently connect to V <sub>DD</sub> or V <sub>SS</sub> via a resistor.
P122/X2/EXCLK			
P137/INTP0	2	Input	Independently connect to V <sub>DD</sub> or V <sub>SS</sub> via a resistor.
RESET	2	Input	Connect directly or via a resistor to V <sub>DD</sub> .
REGC	–	–	Connect to V <sub>SS</sub> via capacitor (0.47 to 1 μF).

**Notes** 1. 24-pin products only for ANI16

2. 32-pin products only

Figure 2-1. Pin I/O Circuit List (1/2)

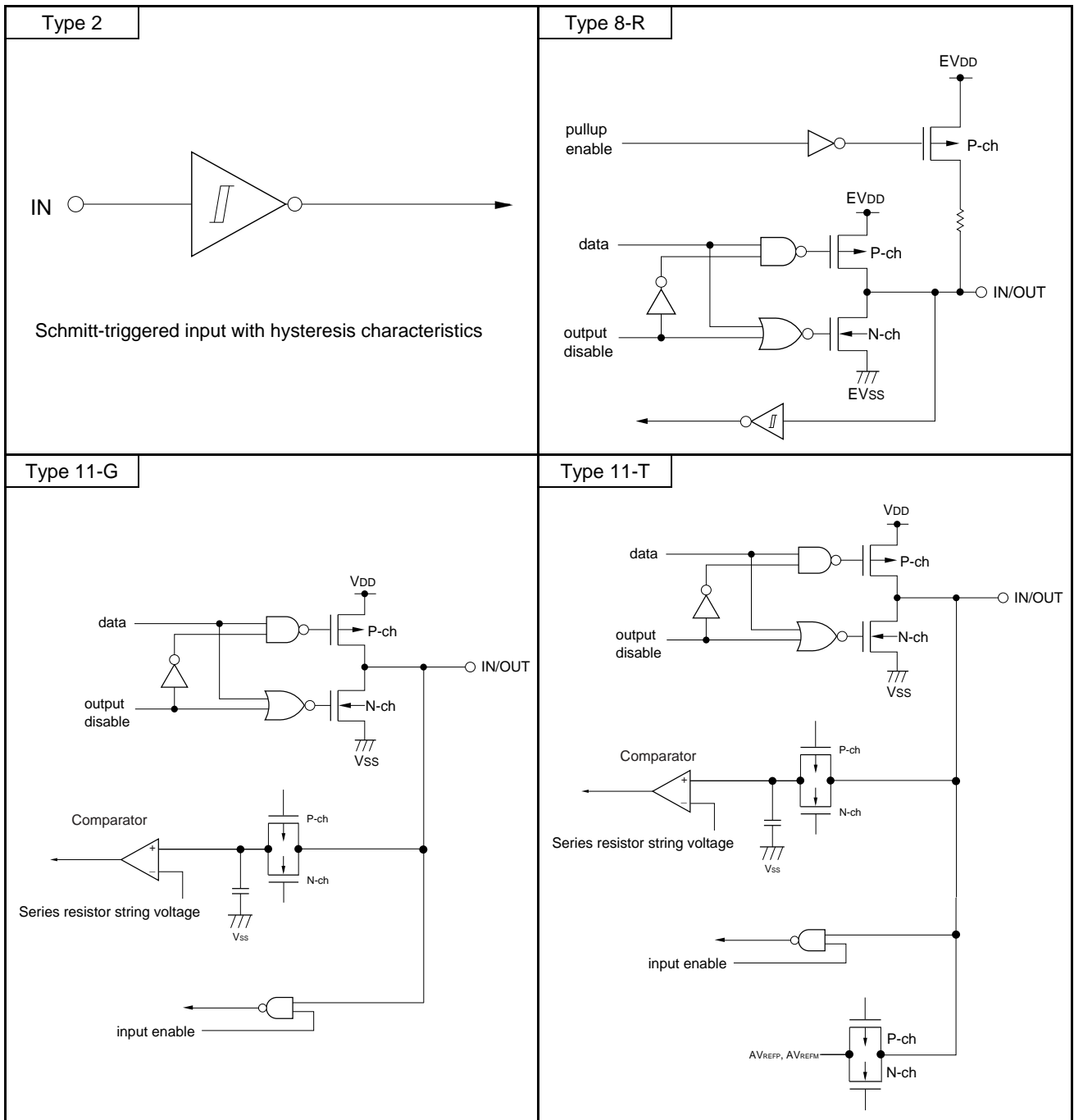
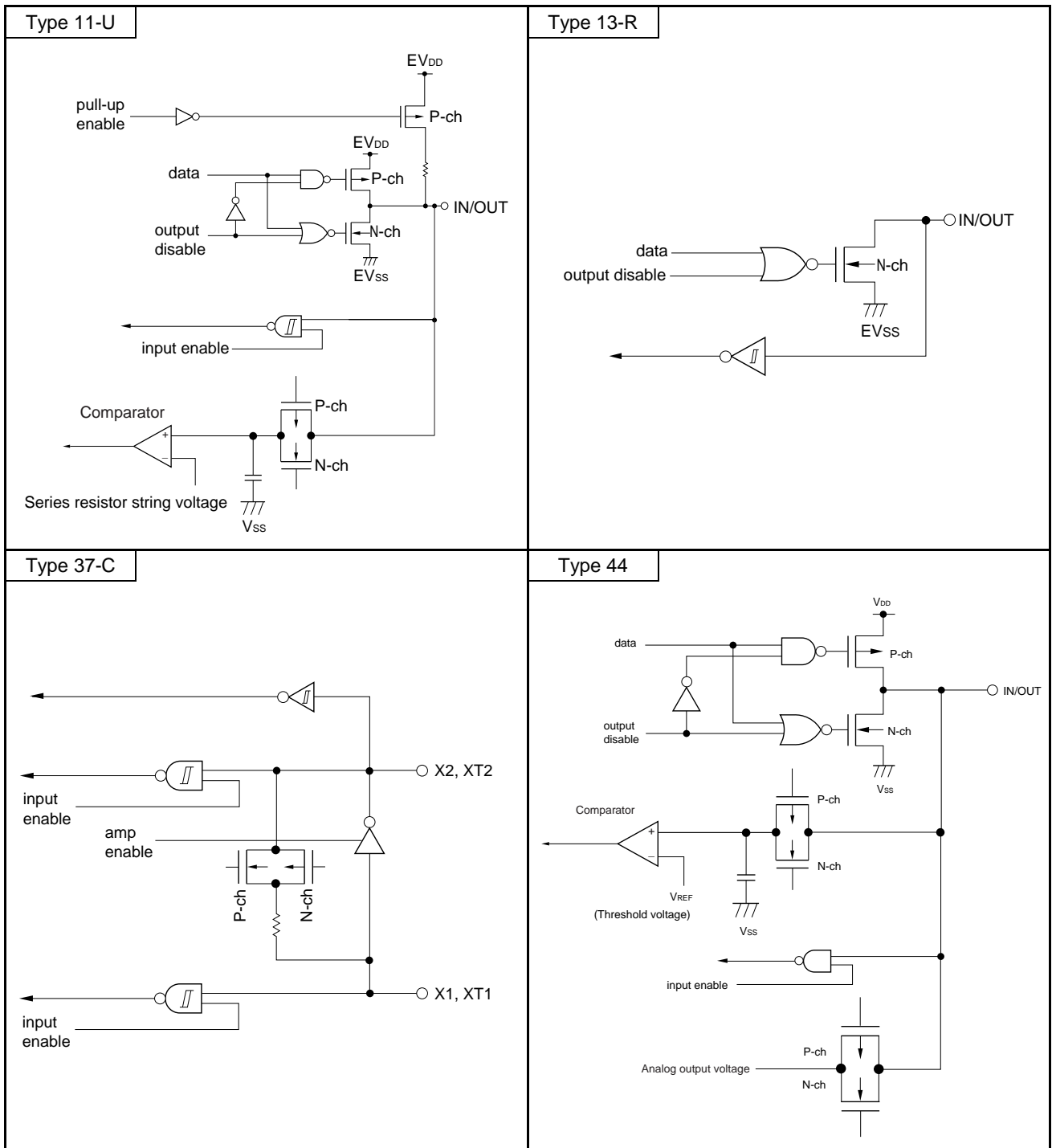




Figure 2-1. Pin I/O Circuit List (2/2)



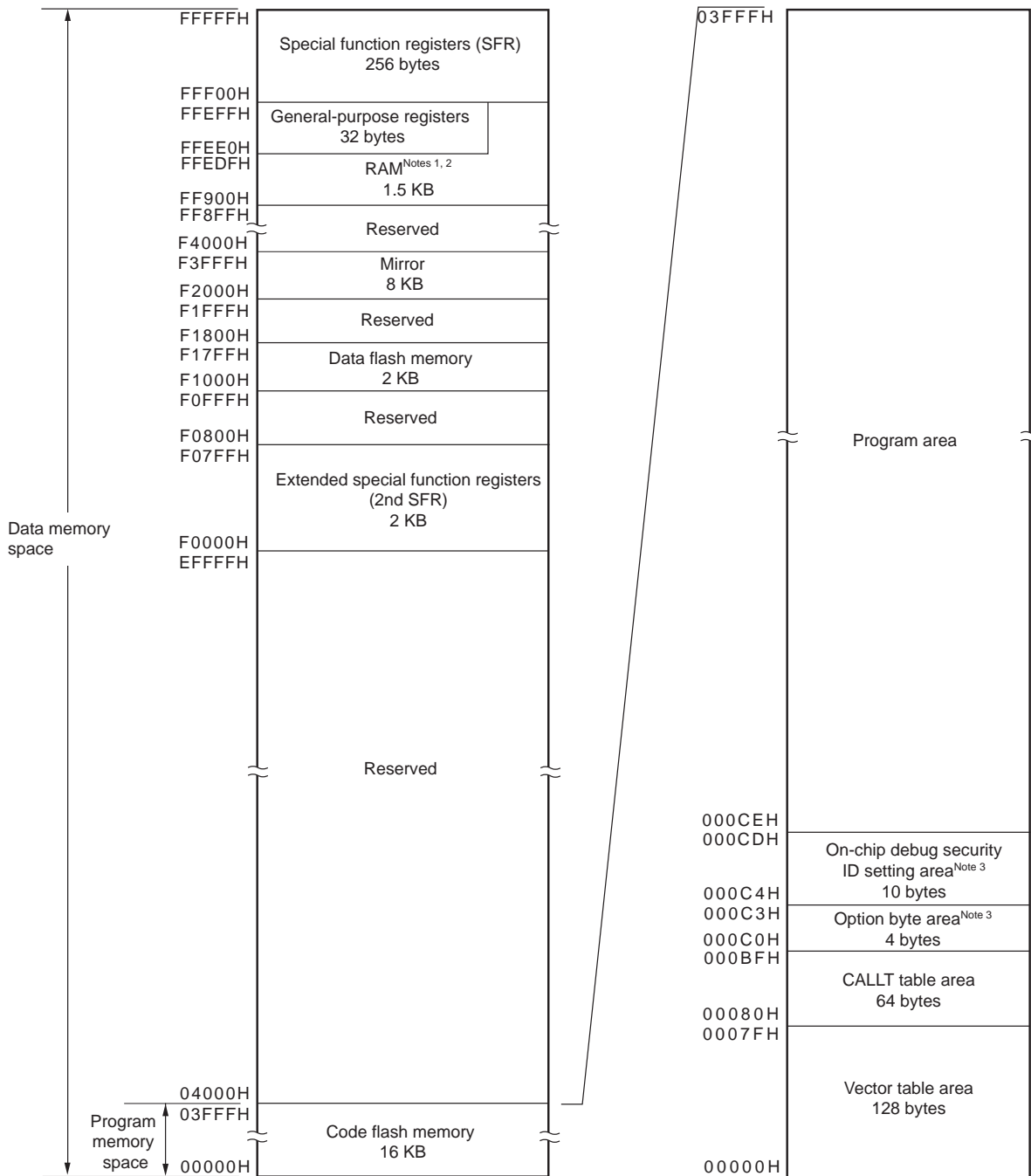
## CHAPTER 3 CPU ARCHITECTURE

### 3.1 Memory Space

Products in the R7F0C010 can access a 1 MB address space. Figure 3-1 shows the memory maps.

<R>

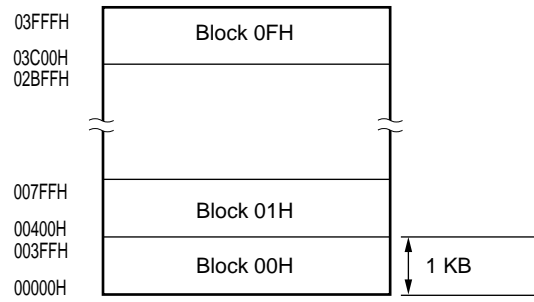
Figure 3-1. Memory Map



- Notes**
- Do not allocate RAM addresses which are used as a stack area, a data buffer, a branch destination of vector interrupt processing to the area FFE20H to FFEFFH and FF900H to FFC80H when performing self-programming.
  - Instructions can be executed from the RAM area excluding the general-purpose register area.
  - Set the option bytes to 000C0H to 000C3H, and the on-chip debug security IDs to 000C4H to 000CDH.

**Caution** While RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed and the RAM area + 10 bytes when instructions are fetched from RAM areas, respectively.  
 Reset signal generation sets RAM parity error resets to enabled (RPERDIS = 0). For details, see 22.5 RAM Parity Error Detection Function.

**Remark** The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see **Table 3-1 Correspondence Between Address Values and Block Numbers in Flash Memory.**



Correspondence between the address values and block numbers in the flash memory are shown below.

**Table 3-1. Correspondence Between Address Values and Block Numbers in Flash Memory**

Address Value	Block Number
00000H to 003FFH	00H
00400H to 007FFH	01H
00800H to 00BFFH	02H
00C00H to 00FFFH	03H
01000H to 013FFH	04H
01400H to 017FFH	05H
01800H to 01BFFH	06H
01C00H to 01FFFH	07H
02000H to 023FFH	08H
02400H to 027FFH	09H
02800H to 02BFFH	0AH
02C00H to 02FFFH	0BH
03000H to 033FFH	0CH
03400H to 037FFH	0DH
03800H to 03BFFH	0EH
03C00H to 03FFFH	0FH

### 3.1.1 Internal program memory space

The internal program memory space stores the program and table data.

The R7F0C010 products incorporate internal ROM (flash memory), as shown below.

**Table 3-2. Internal ROM Capacity**

Part Number	Internal ROM	
	Structure	Capacity
R7F0C010	Flash memory	16384 × 8 bits (00000H to 03FFFH)

The internal program memory space is divided into the following areas.

#### (1) Vector table area

The 128-byte area 00000H to 0007FH is reserved as a vector table area. The program start addresses for branch upon reset or generation of each interrupt request are stored in the vector table area. Furthermore, the interrupt jump address is a 64 K address of 00000H to 0FFFFH, because the vector code is assumed to be 2 bytes.

Of the 16-bit address, the lower 8 bits are stored at even addresses and the higher 8 bits are stored at odd addresses.

Table 3-3. Vector Table

Vector Table Address	Interrupt Source
0000H	RESET, POR, LVD, WDT, TRAP, IAW, RPE
0004H	INTWDTI
0006H	INTLVI
0008H	INTP0
000AH	INTP1
000CH	INTP2
000EH	INTP3
0010H	INTP4
0012H	INTP5
0014H	INTAD
0016H	INTIICA0
0018H	INTFL
001AH	INTDMA0
001CH	INTDMA1
001EH	INTST0/INTCSI00
0020H	INTSR0
0022H	INTSRE0
	INTTM01H
0028H	INTTM03H
002AH	INTIICA1
002CH	INTTM00
002EH	INTTM01
0030H	INTTM02
0032H	INTTM03

**(2) CALLT instruction table area**

The 64-byte area 00080H to 000BFH can store the subroutine entry address of a 2-byte call instruction (CALLT). Set the subroutine entry address to a value in a range of 00000H to 0FFFFH (because an address code is of 2 bytes).

**(3) Option byte area**

A 4-byte area of 000C0H to 000C3H can be used as an option byte area. For details, see **CHAPTER 22 OPTION BYTE**.

**(4) On-chip debug security ID setting area**

A 10-byte area of 000C4H to 000CDH can be used as an on-chip debug security ID setting area. For details, see **CHAPTER 24 ON-CHIP DEBUG FUNCTION**.

**3.1.2 Mirror area**

The R7F0C010 mirror the code flash area of 02000H to 03FFFH, F2000H to F3FFFH.

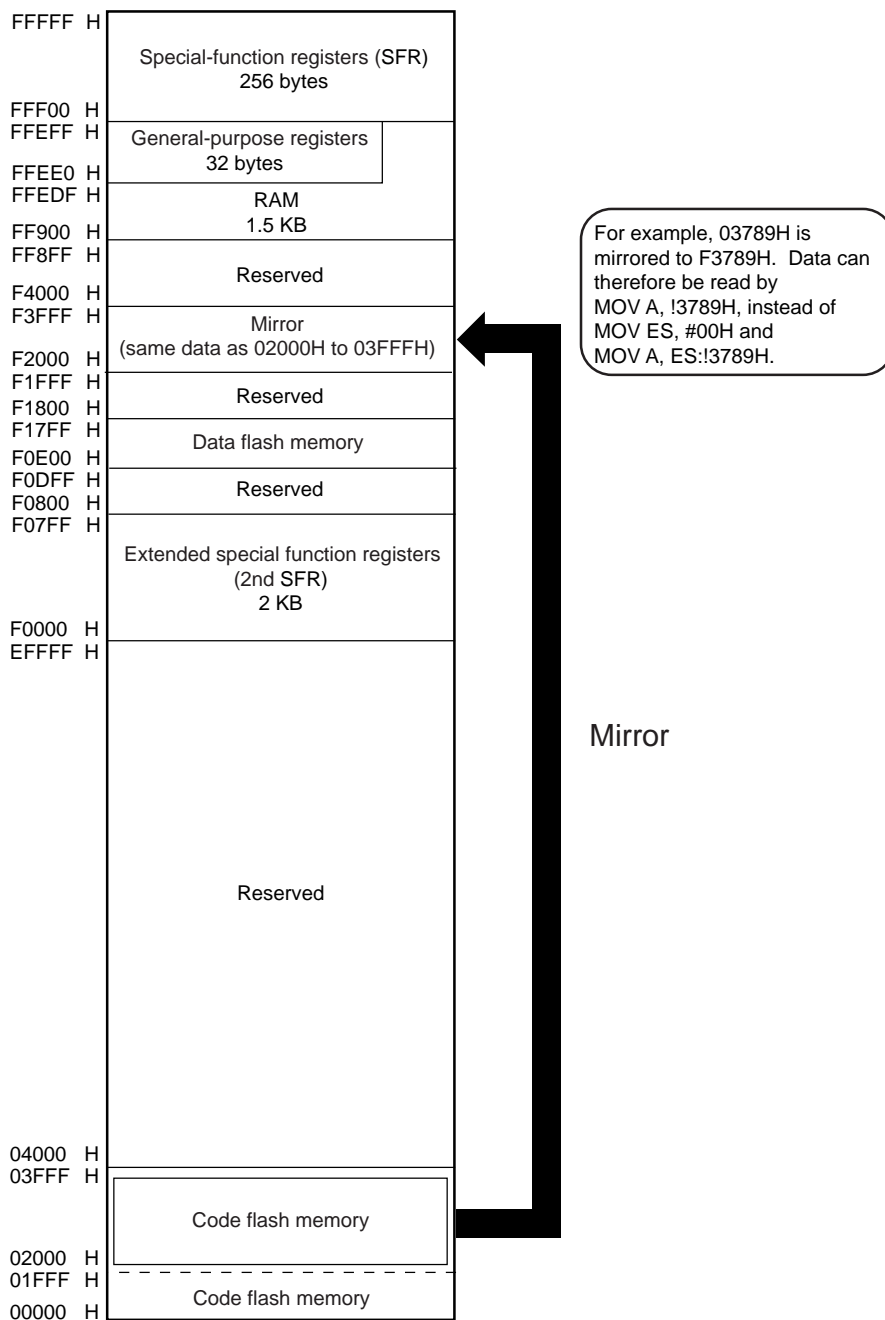
By reading data from F2000H to F3FFFH, an instruction that does not have the ES register as an operand can be used, and thus the contents of the code flash can be read with the shorter code. However, the code flash area is not mirrored to the SFR, extended SFR, RAM, and use prohibited areas.

See **3.1 Memory Space** for the mirror area of each product.

The mirror area can only be read and no instruction can be fetched from this area.

The following show examples.

**Example** R7F0C010 (Flash memory: 16 KB, RAM: 1.5 KB)



The PMC register is described below.



- **Processor mode control register (PMC)**

This register sets the flash memory space for mirroring to area from F2000H to F3FFFH.

The PMC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

**Figure 3-2. Format of Processor Mode Control Register (PMC)**

Address: FFFFEH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	<0>
PMC	0	0	0	0	0	0	0	MAA

MAA	Selection of flash memory space for mirroring to area from F2000H to F3FFFH
0	02000H to 03FFFH is mirrored to F2000H to F3FFFH
1	Setting prohibited

**Cautions 1. Be sure to clear bit 0 (MAA) of this register to 0 (default value).**

**2. After setting the PMC register, wait for at least one instruction and access the mirror area.**

<R>

### 3.1.3 Internal data memory space

The R7F0C010 products incorporate the following RAMs.

**Table 3-4. Internal RAM Capacity**

Part Number	Internal RAM
R7F0C010	1536 × 8 bits (FF900H to FFEFFH)

The internal RAM can be used as a data area and a program area where instructions are fetched (it is prohibited to use the general-purpose register area for fetching instructions). Four general-purpose register banks consisting of eight 8-bit registers per bank are assigned to the 32-byte area of FFEE0H to FFEFFH of the internal RAM area.

The internal RAM is used as stack memory.

**Cautions 1. It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.**

**2. When self-programming is performed or the data flash memory is rewritten, the stack used for each library and the RAM address used for the data buffer and DMA transfer should not be set to the RAM area of the following products. For details, refer to RL78 Family Flash Self Programming Library Type01 User's Manual and RL78 Family Data Flash Library Type04 User's Manual.**

**R7F0C010: FFE20H to FFEFFH**

**3. Use of the RAM areas of the following products is prohibited, because these areas are used for self-programming library and data flash library. (Refer to Figure 3-1 Memory Map)**

**R7F0C010: FF900H to FFC80H**

### 3.1.4 Special function register (SFR) area

On-chip peripheral hardware special function registers (SFRs) are allocated in the area FFF00H to FFFFFH (see **Table 3-5 in 3.2.4 Special function registers (SFRs)**).

**Caution Do not access addresses to which SFRs are not assigned.**

### 3.1.5 Extended special function register (2nd SFR: 2nd Special Function Register) area

On-chip peripheral hardware special function registers (2nd SFRs) are allocated in the area F0000H to F07FFH (see **Table 3-6 in 3.2.5 Extended Special function registers (2nd SFRs: 2nd Special Function Registers)**).

SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

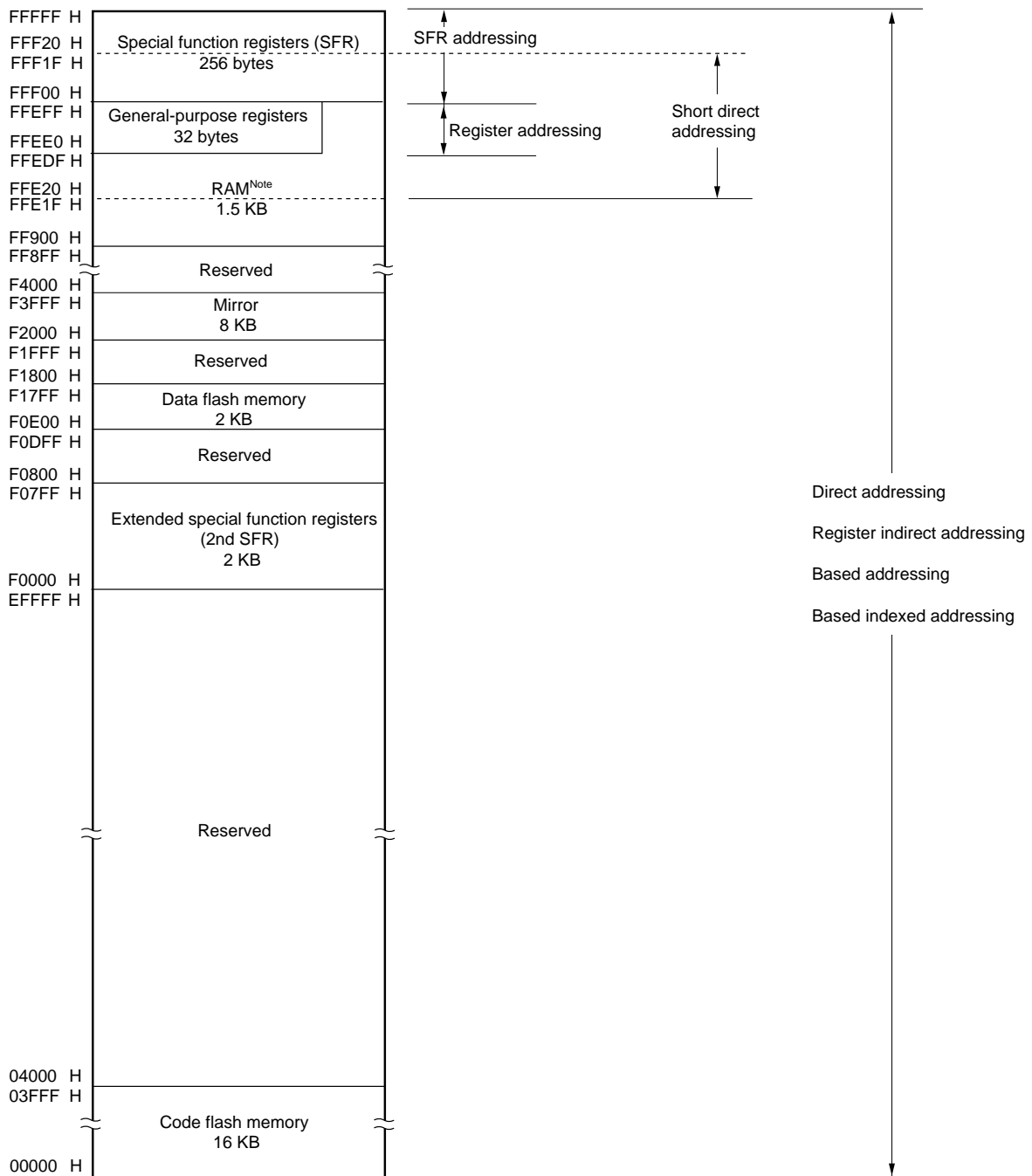
**Caution Do not access addresses to which extended SFRs are not assigned.**

### 3.1.6 Data memory addressing

Addressing refers to the method of specifying the address of the instruction to be executed next or the address of the register or memory relevant to the execution of instructions.

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the R7F0C010, based on operability and other considerations. In particular, special addressing methods designed for the functions of the special function registers (SFR) and general-purpose registers are available for use. Figures 3-3 shows correspondence between data memory and addressing. For details of each addressing, see **3.4 Addressing for Processing Data Addresses**.

<R> **Figure 3-3. Correspondence Between Data Memory and Addressing**



**Note** FFE20H to FFEFFH and FF900H to FFC80H area used by the self-programming libraries cannot be used when the self-programming function and data flash function are used.

## 3.2 Processor Registers

The R7F0C010 products incorporate the following processor registers.

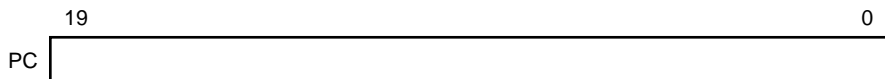
### 3.2.1 Control registers

The control registers control the program sequence, statuses and stack memory. The control registers consist of a program counter (PC), a program status word (PSW) and a stack pointer (SP).

#### (1) Program counter (PC)

The program counter is a 20-bit register that holds the address information of the next program to be executed. In normal operation, PC is automatically incremented according to the number of bytes of the instruction to be fetched. When a branch instruction is executed, immediate data and register contents are set. Reset signal generation sets the reset vector table values at addresses 0000H and 0001H to the 16 lower-order bits of the program counter. The four higher-order bits of the program counter are cleared to 0000.

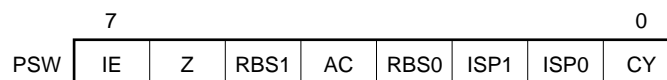
**Figure 3-4. Format of Program Counter**



#### (2) Program status word (PSW)

The program status word is an 8-bit register consisting of various flags set/reset by instruction execution. Program status word contents are stored in the stack area upon vectored interrupt request is acknowledged or PUSH PSW instruction execution and are restored upon execution of the RETB, RETI and POP PSW instructions. Reset signal generation sets the PSW register to 06H.

**Figure 3-5. Format of Program Status Word**



##### (a) Interrupt enable flag (IE)

This flag controls the interrupt request acknowledge operations of the CPU.

When 0, the IE flag is set to the interrupt disabled (DI) state, and all maskable interrupt requests are disabled.

When 1, the IE flag is set to the interrupt enabled (EI) state and maskable interrupt request acknowledgment is controlled with an in-service priority flag (ISP1, ISP0), an interrupt mask flag for various interrupt sources, and a priority specification flag.

The IE flag is reset (0) upon DI instruction execution or interrupt acknowledgment and is set (1) upon EI instruction execution.

##### (b) Zero flag (Z)

When the operation or comparison result is zero or equal, this flag is set (1). It is reset (0) in all other cases.

##### (c) Register bank select flags (RBS0, RBS1)

These are 2-bit flags to select one of the four register banks.

In these flags, the 2-bit information that indicates the register bank selected by SEL RBn instruction execution is stored.

**(d) Auxiliary carry flag (AC)**

If the operation result has a carry from bit 3 or a borrow at bit 3, this flag is set (1). It is reset (0) in all other cases.

**(e) In-service priority flags (ISP1, ISP0)**

This flag manages the priority of acknowledgeable maskable vectored interrupts. Vectored interrupt requests specified lower than the value of ISP0 and ISP1 flags by the priority specification flag registers (PRn0L, PRn0H, PRn1L) (see 15.3.3) cannot be acknowledged. Actual vectored interrupt request acknowledgment is controlled by the interrupt enable flag (IE).

**Remark** n = 0, 1

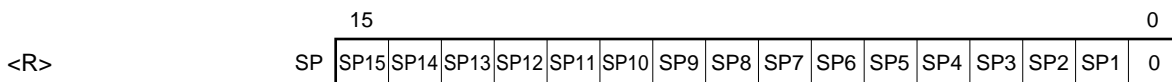
**(f) Carry flag (CY)**

This flag stores overflow and underflow upon add/subtract instruction execution. It stores the shift-out value upon rotate instruction execution and functions as a bit accumulator during bit operation instruction execution.

**(3) Stack pointer (SP)**

This is a 16-bit register to hold the start address of the memory stack area. Only the internal RAM area can be set as the stack area.

**Figure 3-6. Format of Stack Pointer**



<R> In stack addressing through a stack pointer, the SP is decremented ahead of write (save) to the stack memory and is incremented after read (restore) from the stack memory.

**Cautions 1. Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.**

**2. It is prohibited to use the general-purpose register space (FFEE0H to FFEFFH) for fetching instructions or as a stack area.**

**3. When self-programming is performed or the data flash memory is rewritten, the stack used for each library and the RAM address used for the data buffer and DMA transfer should not be set to the RAM area of the following products. For details, refer to RL78 Family Flash Self Programming Library Type01 User's Manual and RL78 Family Data Flash Library Type04 User's Manual.**

**R7F0C010: FFE20H to FFEFFH**

**4. Use of the RAM areas of the following products is prohibited, because these areas are used for self-programming library and data flash library. (Refer to Figure 3-1 Memory Map)**

**R7F0C010: FF900H to FFC80H**

**3.2.2 General-purpose registers**

General-purpose registers are mapped at particular addresses (FFEE0H to FFEFFH) of the data memory. The general-purpose registers consists of 4 banks, each bank consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H).

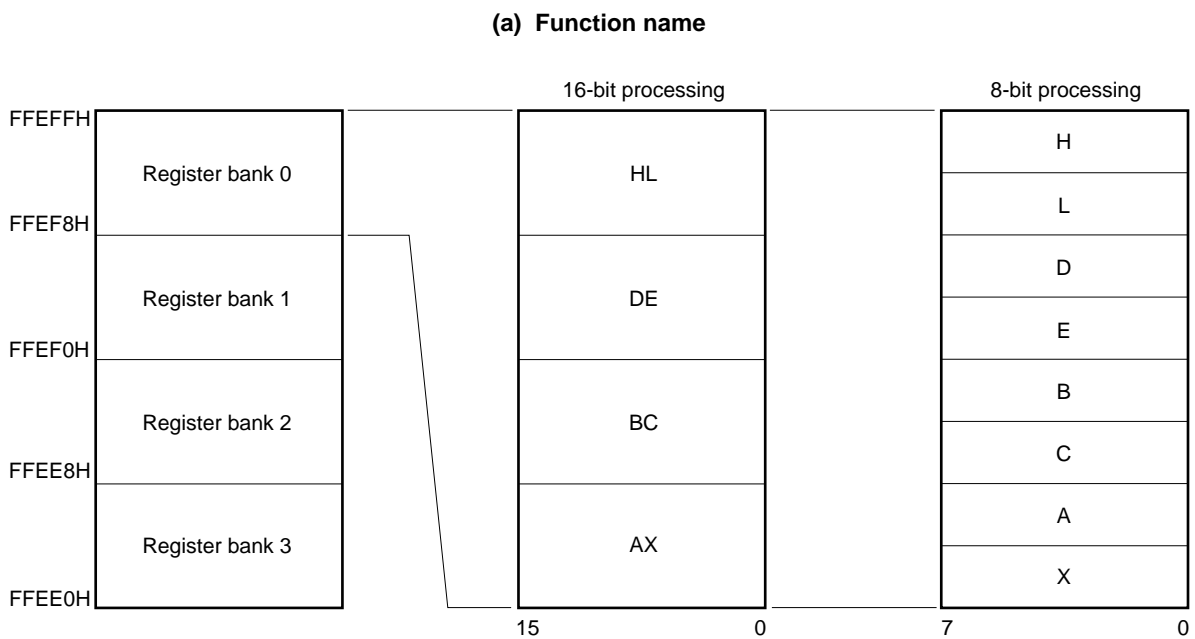
Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

<R>

Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupt processing for each bank.

**Caution** It is prohibited to use the general-purpose register (FFEE0H to FFEFFH) space for fetching instructions or as a stack area.

**Figure 3-7. Configuration of General-Purpose Registers**

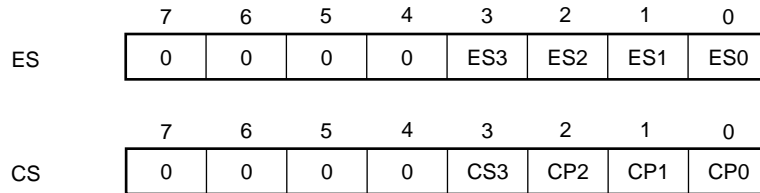


### 3.2.3 ES and CS registers

The ES register and CS register are used to specify the higher address for data access and when a branch instruction is executed (register direct addressing), respectively.

The default value of the ES register after reset is 0FH, and that of the CS register is 00H.

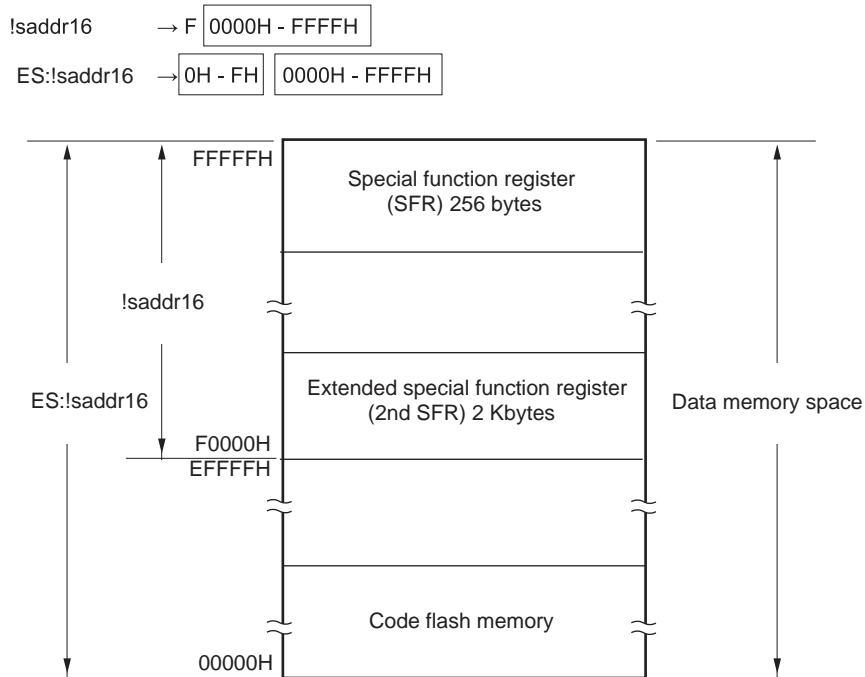
**Figure 3-8. Configuration of ES and CS Registers**



Though the data area which can be accessed with 16-bit addresses is the 64 Kbytes from F0000H to FFFFFH, using the ES register as well extends this to the 1 Mbyte from 00000H to FFFFFH.

**Figure 3-9. Extension of Data Area Which Can Be Accessed**

<R>



### 3.2.4 Special function registers (SFRs)

Unlike a general-purpose register, each SFR has a special function.

SFRs are allocated to the FFF00H to FFFFFH area.

SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the SFR type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation

Describe as follows for the 1-bit manipulation instruction operand (sfr.bit).

When the bit name is defined: <Bit name>

When the bit name is not defined: <Register name>, <Bit number> or <Address>, <Bit number>

- 8-bit manipulation

Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (sfr). This manipulation can also be specified with an address.

- 16-bit manipulation

Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (sfrp). When specifying an address, describe an even address.

Table 3-5 gives a list of the SFRs. The meanings of items in the table are as follows.

- Symbol

Symbol indicating the address of a special function register. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.

- R/W

Indicates whether the corresponding SFR can be read or written.

R/W: Read/write enable

R: Read only

W: Write only

- Manipulable bit units

“√” indicates the manipulable bit unit (1, 8, or 16). “—” indicates a bit unit for which manipulation is not possible.

- After reset

Indicates each register status upon reset signal generation.

**Caution** Do not access addresses to which extended SFRs are not assigned.

**Remark** For extended SFRs (2nd SFRs), see 3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers).

<R>



Table 3-5. Special Function Register (SFR) List (1/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
FFF01H	Port register 1	P1		R/W	√	√	–	00H
FFF02H	Port register 2	P2		R/W	√	√	–	00H
FFF03H	Port register 3	P3		R/W	√	√	–	00H
FFF04H	Port register 4	P4		R/W	√	√	–	00H
FFF06H	Port register 6	P6		R/W	√	√	–	00H
FFF0CH	Port register 12	P12		R/W	√	√	–	Undefined
FFF0DH	Port register 13	P13		R/W	√	√	–	Undefined
FFF10H	Serial data register 00	TXD0/ SIO00	SDR00	R/W	–	√	√	0000H
FFF11H		–			–	–		
FFF12H	Serial data register 01	RXD0/ SIO01	SDR01	R/W	–	√	√	0000H
FFF13H		–			–	–		
FFF18H	Timer data register 00	TDR00		R/W	–	–	√	0000H
FFF19H								
FFF1AH	Timer data register 01	TDR01L	TDR01	R/W	–	√	√	00H
FFF1BH		TDR01H			–	√	00H	
FFF1EH	12-bit A/D conversion result register	ADCR		R	–	–	√	0000H
FFF1FH	8-bit A/D conversion result register	ADCRH		R	–	√	–	00H
FFF21H	Port mode register 1	PM1		R/W	√	√	–	FFH
FFF22H	Port mode register 2	PM2		R/W	√	√	–	FFH
FFF23H	Port mode register 3	PM3		R/W	√	√	–	FFH
FFF24H	Port mode register 4	PM4		R/W	√	√	–	FFH
FFF26H	Port mode register 6	PM6		R/W	√	√	–	FFH
FFF30H	A/D converter mode register 0	ADM0		R/W	√	√	–	00H
FFF31H	Analog input channel specification register	ADS		R/W	√	√	–	00H
FFF32H	A/D converter mode register 1	ADM1		R/W	√	√	–	00H
FFF38H	External interrupt rising edge enable register 0	EGP0		R/W	√	√	–	00H
FFF39H	External interrupt falling edge enable register 0	EGN0		R/W	√	√	–	00H
FFF50H	IICA shift register 0	IICA0		R/W	–	√	–	00H
FFF51H	IICA status register 0	IICS0		R	√	√	–	00H
FFF52H	IICA flag register 0	IICF0		R/W	√	√	–	00H
FFF54H	IICA shift register 1	IICA1		R/W	–	√	–	00H
FFF55H	IICA status register 1	IICS1		R	√	√	–	00H
FFF56H	IICA flag register 1	IICF1		R/W	√	√	–	00H
FFF58H	D/A conversion value setting register 0	DACS0		R/W	–	–	√	0000H
FFF59H								
FFF5AH	D/A conversion value setting register 1	DACS1		R/W	–	–	√	0000H
FFF5BH								
FFF5CH	D/A converter mode register	DAM		R/W	√	√	–	00H

Table 3-5. Special Function Register (SFR) (2/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
FFF64H	Timer data register 02	TDR02		R/W	–	–	√	0000H
FFF65H								
FFF66H	Timer data register 03	TDR03L	TDR03	R/W	–	√	√	00H
FFF67H		TDR03H			–	√	00H	
FFFA0H	Clock operation mode control register	CMC		R/W	–	√	–	00H
FFFA1H	Clock operation status control register	CSC		R/W	√	√	–	C0H
FFFA2H	Oscillation stabilization time counter status register	OSTC		R	√	√	–	00H
FFFA3H	Oscillation stabilization time select register	OSTS		R/W	–	√	–	07H
FFFA4H	System clock control register	CKC		R/W	√	√	–	00H
FFFA5H	Clock output select register 0	CKS0		R/W	√	√	–	00H
FFFA6H	Clock output select register 1	CKS1		R/W	√	√	–	00H
FFFA8H	Reset control flag register	RESF		R	–	√	–	Undefined <sup>Note 1</sup>
FFFA9H	Voltage detection register	LVIM		R/W	√	√	–	00H <sup>Note 2</sup>
FFFAAH	Voltage detection level register	LVIS		R/W	√	√	–	00H/01H/81H <sup>Note 3</sup>
FFFABH	Watchdog timer enable register	WDTE		R/W	–	√	–	1AH/9AH <sup>Note 4</sup>
FFFACh	CRC input register	CRCIN		R/W	–	√	–	00H
FFFB0H	DMA SFR address register 0	DSA0		R/W	–	√	–	00H
FFFB1H	DMA SFR address register 1	DSA1		R/W	–	√	–	00H
FFFB2H	DMA RAM address register 0L	DRA0L	DRA0	R/W	–	√	√	00H
FFFB3H	DMA RAM address register 0H	DRA0H		R/W	–	√	00H	
FFFB4H	DMA RAM address register 1L	DRA1L	DRA1	R/W	–	√	√	00H
FFFB5H	DMA RAM address register 1H	DRA1H		R/W	–	√	00H	
FFFB6H	DMA byte count register 0L	DBC0L	DBC0	R/W	–	√	√	00H
FFFB7H	DMA byte count register 0H	DBC0H		R/W	–	√	00H	
FFFB8H	DMA byte count register 1L	DBC1L	DBC1	R/W	–	√	√	00H
FFFB9H	DMA byte count register 1H	DBC1H		R/W	–	√	00H	
FFFB AH	DMA mode control register 0	DMC0		R/W	√	√	–	00H
FFFB BH	DMA mode control register 1	DMC1		R/W	√	√	–	00H
FFFB CH	DMA operation control register 0	DRC0		R/W	√	√	–	00H
FFFB DH	DMA operation control register 1	DRC1		R/W	√	√	–	00H

- Notes**
1. The reset value of the RESF register varies depending on the reset source.
  2. The reset value of the LVIM register varies depending on the reset source.
  3. The reset value of the LVIS register varies depending on the reset source and the setting of the option byte.
  4. The reset value of the WDTE register is determined by the setting of the option byte.

Table 3-5. Special Function Register (SFR) List (3/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
FFFA8H	Reset control flag register	RESF		R	–	√	–	Undefined Note 1
FFFA9H	Voltage detection register	LVIM		R/W	√	√	–	00H <sup>Note 2</sup>
FFFAAH	Voltage detection level register	LVIS		R/W	√	√	–	00H/01H/81H Note 3
FFFABH	Watchdog timer enable register	WDTE		R/W	–	√	–	9AH/1AH Note 4
FFFACH	CRC input register	CRCIN		R/W	–	√	–	00H
FFFE0H	Interrupt request flag register 0L	IF0L	IF0	R/W	√	√	√	00H
FFFE1H	Interrupt request flag register 0H	IF0H		R/W	√	√		00H
FFFE2H	Interrupt request flag register 1L	IF1L		R/W	√	√	–	00H
FFFE4H	Interrupt mask flag register 0L	MK0L	MK0	R/W	√	√	√	FFH
FFFE5H	Interrupt mask flag register 0H	MK0H		R/W	√	√		FFH
FFFE6H	Interrupt mask flag register 1L	MK1L		R/W	√	√	–	FFH
FFFE8H	Priority specification flag register 00L	PR00L	PR00	R/W	√	√	√	FFH
FFFE9H	Priority specification flag register 00H	PR00H		R/W	√	√		FFH
FFFEAH	Priority specification flag register 01L	PR01L		R/W	√	√	–	FFH
FFFECH	Priority specification flag register 10L	PR10L	PR10	R/W	√	√	√	FFH
FF FEDH	Priority specification flag register 10H	PR10H		R/W	√	√		FFH
FF FEEH	Priority specification flag register 11L	PR11L		R/W	√	√	–	FFH
FF FFEH	Processor mode control register	PMC		R/W	√	√	–	00H

- Notes**
1. The reset value of the RESF register varies depending on the reset source.
  2. The reset value of the LVIM register varies depending on the reset source.
  3. The reset value of the LVIS register varies depending on the reset source and the setting of the option byte.
  4. The reset value of the WDTE register is determined by the setting of the option byte.

**Remark** For extended SFRs (2nd SFRs), see **Table 3-6 Extended Special Function Register (2nd SFR) List**.

### 3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)

Unlike a general-purpose register, each extended SFR (2nd SFR) has a special function.

Extended SFRs are allocated to the F0000H to F07FFH area. SFRs other than those in the SFR area (FFF00H to FFFFFH) are allocated to this area. An instruction that accesses the extended SFR area, however, is 1 byte longer than an instruction that accesses the SFR area.

Extended SFRs can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulable bit units, 1, 8, and 16, depend on the SFR type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation

Describe as follows for the 1-bit manipulation instruction operand (!addr16.bit)

When the bit name is defined: <Bit name>

When the bit name is not defined: <Register name>.<Bit number> or <Address>.<Bit number>

- 8-bit manipulation

Describe the symbol defined by the assembler for the 8-bit manipulation instruction operand (!addr16). This manipulation can also be specified with an address.

- 16-bit manipulation

Describe the symbol defined by the assembler for the 16-bit manipulation instruction operand (!addr16). When specifying an address, describe an even address.

Table 3-6 gives a list of the extended SFRs. The meanings of items in the table are as follows.

- Symbol

Symbol indicating the address of an extended SFR. It is a reserved word in the assembler, and is defined as an sfr variable using the #pragma sfr directive in the compiler. When using the assembler, debugger, and simulator, symbols can be written as an instruction operand.

- R/W

Indicates whether the corresponding extended SFR can be read or written.

R/W: Read/write enable

R: Read only

W: Write only

- Manipulable bit units

“√” indicates the manipulable bit unit (1, 8, or 16). “–” indicates a bit unit for which manipulation is not possible.

- After reset

Indicates each register status upon reset signal generation.

**Caution** Do not access addresses to which extended SFRs (2nd SFRs) are not assigned.

**Remark** For SFRs in the SFR area, see 3.2.4 Special function registers (SFRs).

<R>

Table 3-6. Extended Special Function Register (2nd SFR) List (1/4)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F0010H	A/D converter mode register 2	ADM2		R/W	√	√	–	00H
F0011H	Conversion result comparison upper limit setting register	ADUL		R/W	–	√	–	FFH
F0012H	Conversion result comparison lower limit setting register	ADLL		R/W	–	√	–	00H
F0013H	A/D test register	ADTES		R/W	–	√	–	00H
F0031H	Pull-up resistor option register 1	PU1		R/W	√	√	–	00H
F0033H	Pull-up resistor option register 3	PU3		R/W	√	√	–	00H
F0034H	Pull-up resistor option register 4	PU4		R/W	√	√	–	01H
F0061H	Port mode control register 1	PMC1		R/W	√	√	–	FFH
F0070H	Noise filter enable register 0	NFEN0		R/W	√	√	–	00H
F0071H	Noise filter enable register 1	NFEN1		R/W	√	√	–	00H
F0073H	Input switch control register	ISC		R/W	√	√	–	00H
F0074H	Timer input select register 0	TIS0		R/W	–	√	–	00H
F0076H	A/D port configuration register	ADPC		R/W	–	√	–	00H
F0078H	Invalid memory access detection control register	IAWCTL		R/W	–	√	–	00H
F007AH	Peripheral enable register 1	PER1		R/W	√	√	–	00H
F0090H	Data flash control register	DFLCTL		R/W	√	√	–	00H
F00A0H	High-speed on-chip oscillator trimming register	HIOTRM		R/W	–	√	–	Undefined Note
F00A8H	High-speed on-chip oscillator frequency select register	HOCODIV		R/W	–	√	–	Undefined
F00F0H	Peripheral enable register 0	PER0		R/W	√	√	–	00H
F00F5H	RAM parity error control register	RPECTL		R/W	√	√	–	00H
F00FEH	BCD adjust result register	BCDADJ		R	–	√	–	Undefined
F0100H	Serial status register 00	SSR00L	SSR00	R	–	√	√	0000H
F0101H		–			–			
F0102H	Serial status register 01	SSR01L	SSR01	R	–	√	√	0000H
F0103H		–			–			
F0108H	Serial flag clear trigger register 00	SIR00L	SIR00	R/W	–	√	√	0000H
F0109H		–			–			
F010AH	Serial flag clear trigger register 01	SIR01L	SIR01	R/W	–	√	√	0000H
F010BH		–			–			

**Note** The value after a reset is adjusted at shipment.

Table 3-6. Extended Special Function Register (2nd SFR) List (2/4)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F0110H	Serial mode register 00	SMR00		R/W	-	-	√	0020H
F0111H								
F0112H	Serial mode register 01	SMR01		R/W	-	-	√	0020H
F0113H								
F0118H	Serial communication operation setting register 00	SCR00		R/W	-	-	√	0087H
F0119H								
F011AH	Serial communication operation setting register 01	SCR01		R/W	-	-	√	0087H
F011BH								
F0120H	Serial channel enable status register 0	SE0L	SE0	R	√	√	√	0000H
F0121H		-			-			
F0122H	Serial channel start register 0	SS0L	SS0	R/W	√	√	√	0000H
F0123H		-			-			
F0124H	Serial channel stop register 0	ST0L	ST0	R/W	√	√	√	0000H
F0125H		-			-			
F0126H	Serial clock select register 0	SPS0L	SPS0	R/W	-	√	√	0000H
F0127H		-			-			
F0128H	Serial output register 0	SO0		R/W	-	-	√	0303H
F0129H								
F012AH	Serial output enable register 0	SOE0L	SOE0	R/W	√	√	√	0000H
F012BH		-			-			
F0134H	Serial output level register 0	SOL0L	SOL0	R/W	-	√	√	0000H
F0135H		-			-			
F0138H	Serial standby control register 0	SSC0L	SSC0	R/W	-	√	√	0000H
		-			-			
F0180H	Timer counter register 00	TCR00		R	-	-	√	FFFFH
F0181H								
F0182H	Timer counter register 01	TCR01		R	-	-	√	FFFFH
F0183H								
F0184H	Timer counter register 02	TCR02		R	-	-	√	FFFFH
F0185H								
F0186H	Timer counter register 03	TCR03		R	-	-	√	FFFFH
F0187H								
F0190H	Timer mode register 00	TMR00		R/W	-	-	√	0000H
F0191H								
F0192H	Timer mode register 01	TMR01		R/W	-	-	√	0000H
F0193H								
F0194H	Timer mode register 02	TMR02		R/W	-	-	√	0000H
F0195H								
F0196H	Timer mode register 03	TMR03		R/W	-	-	√	0000H
F0197H								

Table 3-6. Extended Special Function Register (2nd SFR) List (3/4)

Address	Extended Special Function Register (2nd SFR) Name	Symbol		R/W	Manipulable Bit Range			After Reset
					1-bit	8-bit	16-bit	
F01A0H	Timer status register 00	TSR00L	TSR00	R	–	√	√	0000H
F01A1H		–			–			
F01A2H	Timer status register 01	TSR01L	TSR01	R	–	√	√	0000H
F01A3H		–			–			
F01A4H	Timer status register 02	TSR02L	TSR02	R	–	√	√	0000H
F01A5H		–			–			
F01A6H	Timer status register 03	TSR03L	TSR03	R	–	√	√	0000H
F01A7H		–			–			
F01B0H	Timer channel enable status register 0	TE0L	TE0	R	√	√	√	0000H
F01B1H		–			–			
F01B2H	Timer channel start register 0	TS0L	TS0	R/W	√	√	√	0000H
F01B3H		–			–			
F01B4H	Timer channel stop register 0	TT0L	TT0	R/W	√	√	√	0000H
F01B5H		–			–			
F01B6H	Timer clock select register 0	TPS0		R/W	–	–	√	0000H
F01B7H								
F01B8H	Timer output register 0	TO0L	TO0	R/W	–	√	√	0000H
F01B9H		–			–			
F01BAH	Timer output enable register 0	TOE0L	TOE0	R/W	√	√	√	0000H
F01BBH		–			–			
F01BCH	Timer output level register 0	TOL0L	TOL0	R/W	–	√	√	0000H
F01BDH		–			–			
F01BEH	Timer output mode register 0	TOM0L	TOM0	R/W	–	√	√	0000H
F01BFH		–			–			
F0230H	IICA control register 00	IICCTL00		R/W	√	√	–	00H
F0231H	IICA control register 01	IICCTL01		R/W	√	√	–	00H
F0232H	IICA low-level width setting register 0	IICWL0		R/W	–	√	–	FFH
F0233H	IICA high-level width setting register 0	IICWH0		R/W	–	√	–	FFH
F0234H	Slave address register 0	SVA0		R/W	–	√	–	00H
F0238H	IICA control register 10	IICCTL10		R/W	√	√	–	00H
F0239H	IICA control register 11	IICCTL11		R/W	√	√	–	00H
F023AH	IICA low-level width setting register 1	IICWL1		R/W	–	√	–	FFH
F023BH	IICA high-level width setting register 1	IICWH1		R/W	–	√	–	FFH
F023CH	Slave address register 1	SVA1		R/W	–	√	–	00H
F02F0H	Flash memory CRC control register	CRC0CTL		R/W	√	√	–	00H
F02F2H	Flash memory CRC operation result register	PGCRCL		R/W	–	–	√	0000H
F02FAH	CRC data register	CRCD		R/W	–	–	√	0000H

**Table 3-6. Extended Special Function Register (2nd SFR) List (4/4)**

Address	Extended Special Function Register (2nd SFR) Name	Symbol	R/W	Manipulable Bit Range			After Reset
				1-bit	8-bit	16-bit	
F0300H	Event output destination select register 00	ELSELR00	R/W	√	√	–	00H
F0301H	Event output destination select register 01	ELSELR01	R/W	√	√	–	00H
F0302H	Event output destination select register 02	ELSELR02	R/W	√	√	–	00H
F0303H	Event output destination select register 03	ELSELR03	R/W	√	√	–	00H
F0304H	Event output destination select register 04	ELSELR04	R/W	√	√	–	00H
F0305H	Event output destination select register 05	ELSELR05	R/W	√	√	–	00H
F0306H	Event output destination select register 06	ELSELR06	R/W	√	√	–	00H
F0307H	Event output destination select register 07	ELSELR07	R/W	√	√	–	00H
F0308H	Event output destination select register 08	ELSELR08	R/W	√	√	–	00H
F0309H	Event output destination select register 09	ELSELR09	R/W	√	√	–	00H

**Remark** For SFRs in the SFR area, see **Table 3-5 Special Function Register List**.



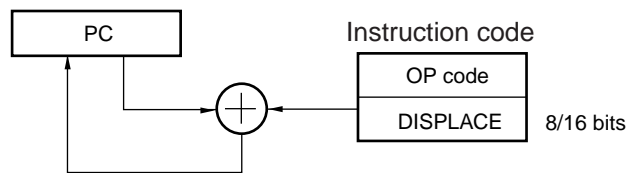
### 3.3 Instruction Address Addressing

#### 3.3.1 Relative addressing

##### [Function]

Relative addressing stores in the program counter (PC) the result of adding a displacement value included in the instruction word (signed complement data:  $-128$  to  $+127$  or  $-32768$  to  $+32767$ ) to the program counter (PC)'s value (the start address of the next instruction), and specifies the program address to be used as the branch destination. Relative addressing is applied only to branch instructions.

Figure 3-10. Outline of Relative Addressing



#### 3.3.2 Immediate addressing

##### [Function]

Immediate addressing stores immediate data of the instruction word in the program counter, and specifies the program address to be used as the branch destination.

For immediate addressing, CALL !!addr20 or BR !!addr20 is used to specify 20-bit addresses and CALL !addr16 or BR !addr16 is used to specify 16-bit addresses. 0000 is set to the higher 4 bits when specifying 16-bit addresses.

Figure 3-11. Example of CALL !!addr20/BR !!addr20

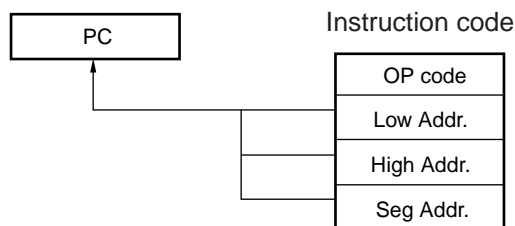
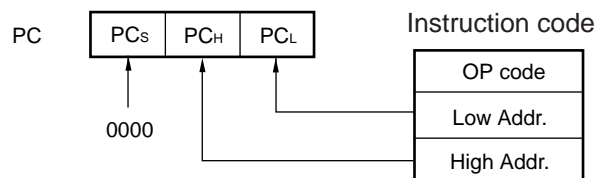


Figure 3-12. Example of CALL !addr16/BR !addr16



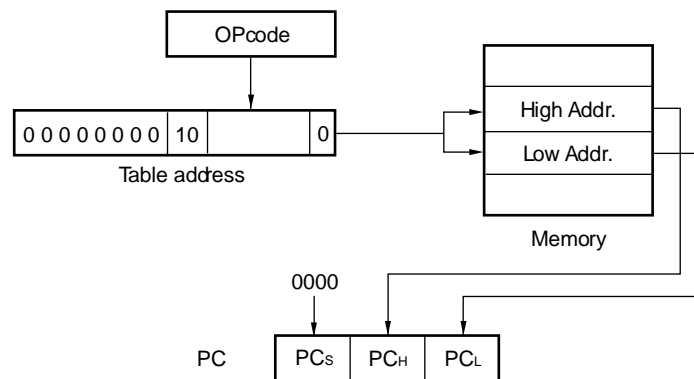
### 3.3.3 Table indirect addressing

#### [Function]

Table indirect addressing specifies a table address in the CALLT table area (0080H to 00BFH) with the 5-bit immediate data in the instruction word, stores the contents at that table address and the next address in the program counter (PC) as 16-bit data, and specifies the program address. Table indirect addressing is applied only for CALLT instructions.

In the RL78 microcontrollers, branching is enabled only to the 64 KB space from 00000H to 0FFFFH.

Figure 3-13. Outline of Table Indirect Addressing

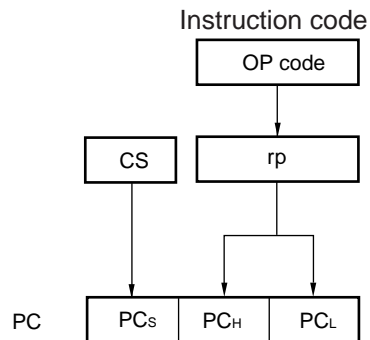


### 3.3.4 Register direct addressing

**[Function]**

Register direct addressing stores in the program counter (PC) the contents of a general-purpose register pair (AX/BC/DE/HL) and CS register of the current register bank specified with the instruction word as 20-bit data, and specifies the program address. Register direct addressing can be applied only to the CALL AX, BC, DE, HL, and BR AX instructions.

**Figure 3-14. Outline of Register Direct Addressing**



### 3.4 Addressing for Processing Data Addresses

#### 3.4.1 Implied addressing

##### [Function]

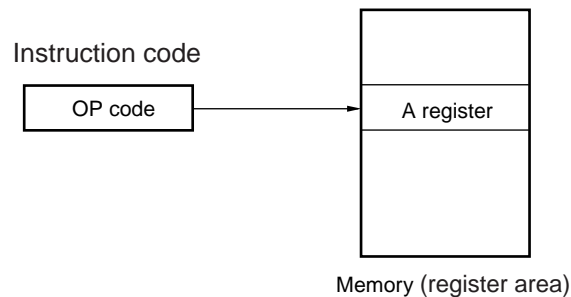
Instructions for accessing registers (such as accumulators) that have special functions are directly specified with the instruction word, without using any register specification field in the instruction word.

##### [Operand format]

Because implied addressing can be automatically employed with an instruction, no particular operand format is necessary.

Implied addressing can be applied only to MULU X.

**Figure 3-15. Outline of Implied Addressing**



#### 3.4.2 Register addressing

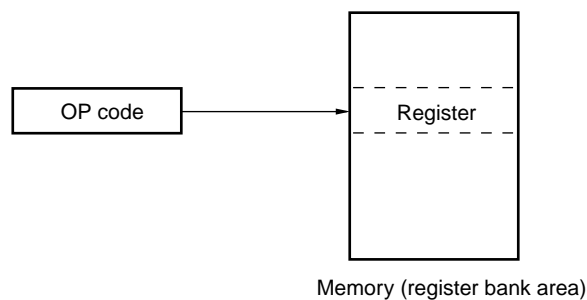
##### [Function]

Register addressing accesses a general-purpose register as an operand. The instruction word of 3-bit long is used to select an 8-bit register and the instruction word of 2-bit long is used to select a 16-bit register.

##### [Operand format]

Identifier	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

**Figure 3-16. Outline of Register Addressing**



3.4.3 Direct addressing

[Function]

Direct addressing uses immediate data in the instruction word as an operand address to directly specify the target address.

[Operand format]

Identifier	Description
!addr16	Label or 16-bit immediate data (only the space from F0000H to FFFFFH is specifiable)
ES:!addr16	Label or 16-bit immediate data (higher 4-bit addresses are specified by the ES register)

Figure 3-17. Example of !addr16

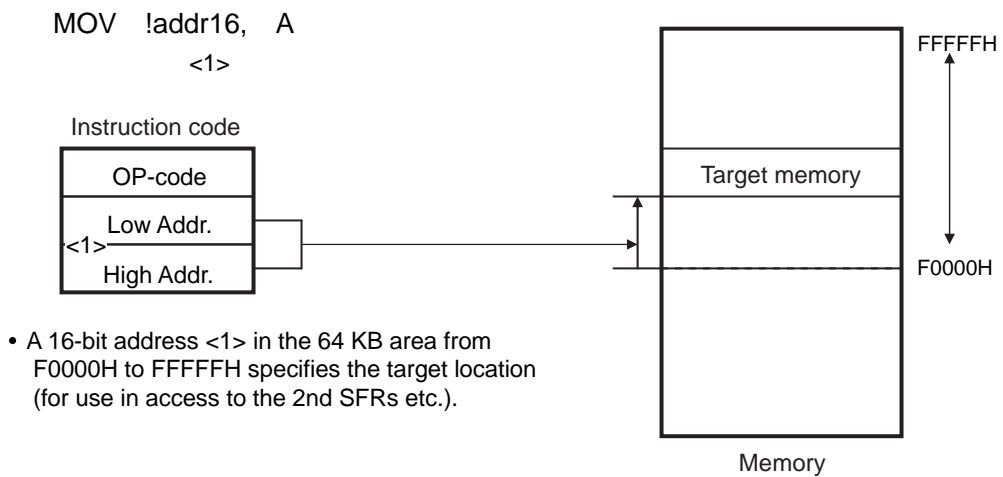
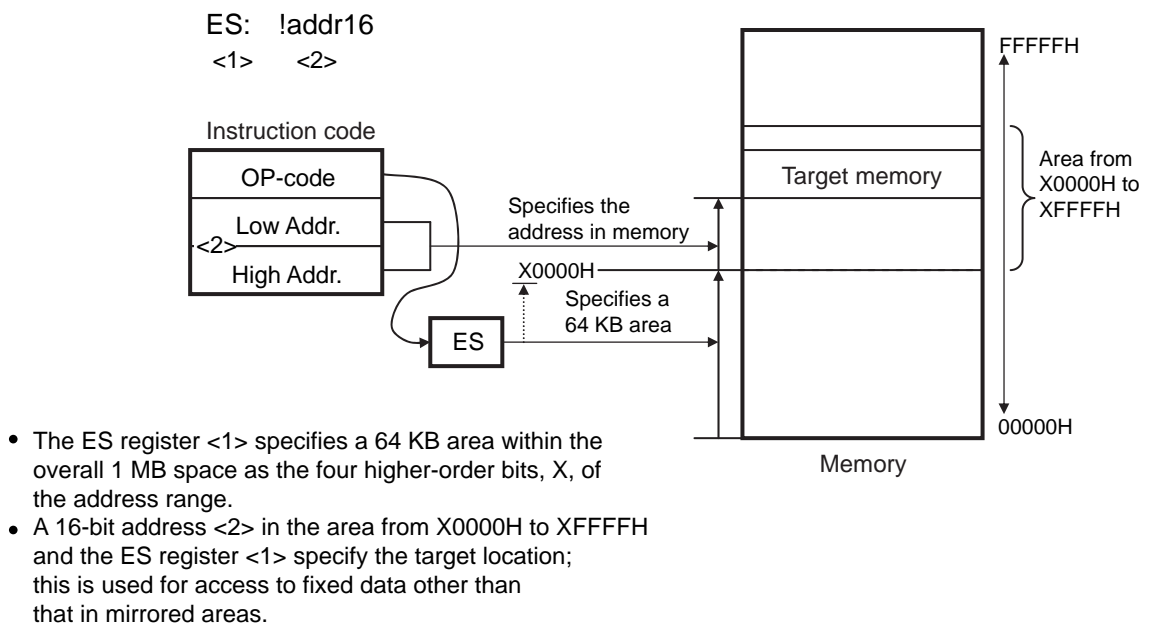


Figure 3-18. Example of ES:!addr16



### 3.4.4 Short direct addressing

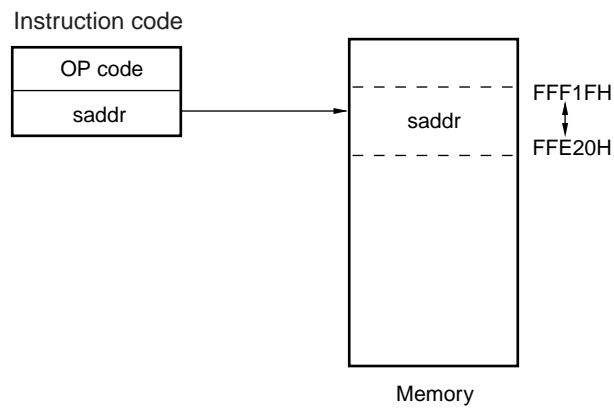
#### [Function]

Short direct addressing directly specifies the target addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFE20H to FFF1FH.

#### [Operand format]

Identifier	Description
SADDR	Label, or FFE20H to FFF1FH immediate data
SADDRP	Label, or FFE20H to FFF1FH immediate data (even address only)

**Figure 3-19. Outline of Short Direct Addressing**



**Remark** SADDR and SADDRP are used to describe the values of addresses FE20H to FF1FH with 16-bit immediate data (higher 4 bits of actual address are omitted), and the values of addresses FFE20H to FFF1FH with 20-bit immediate data.

Regardless of whether SADDR or SADDRP is used, addresses within the space from FFE20H to FFF1FH are specified for the memory.

### 3.4.5 SFR addressing

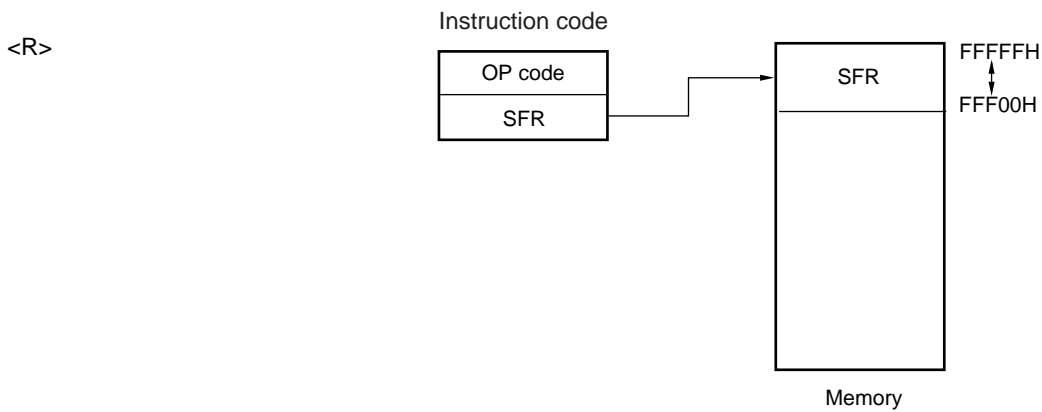
#### [Function]

SFR addressing directly specifies the target SFR addresses using 8-bit data in the instruction word. This type of addressing is applied only to the space from FFF00H to FFFFFH.

#### [Operand format]

Identifier	Description
SFR	SFR name
SFRP	16-bit manipulatable SFR name (even address)

Figure 3-20. Outline of SFR Addressing



3.4.6 Register indirect addressing

[Function]

Register indirect addressing directly specifies the target addresses using the contents of the register pair specified with the instruction word as an operand address.

[Operand format]

Identifier	Description
-	[DE], [HL] (only the space from F0000H to FFFFFH is specifiable)
-	ES:[DE], ES:[HL] (higher 4-bit addresses are specified by the ES register)

Figure 3-21. Example of [DE], [HL]

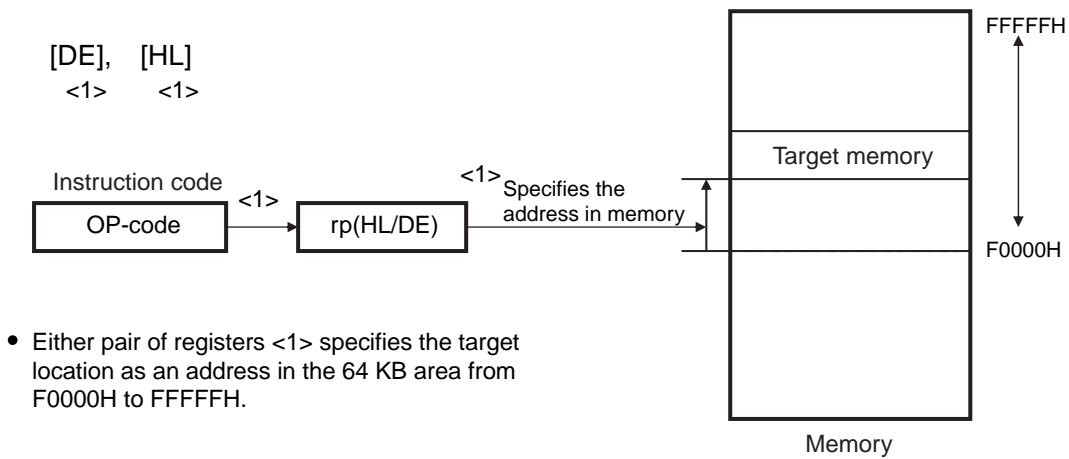
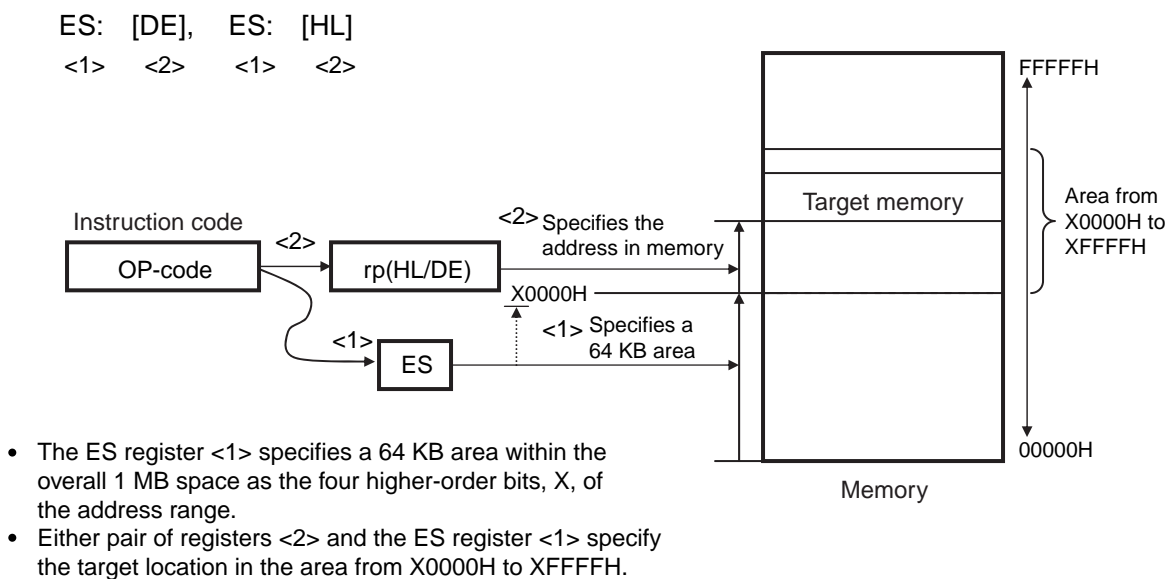


Figure 3-22. Example of ES:[DE], ES:[HL]





### 3.4.7 Based addressing

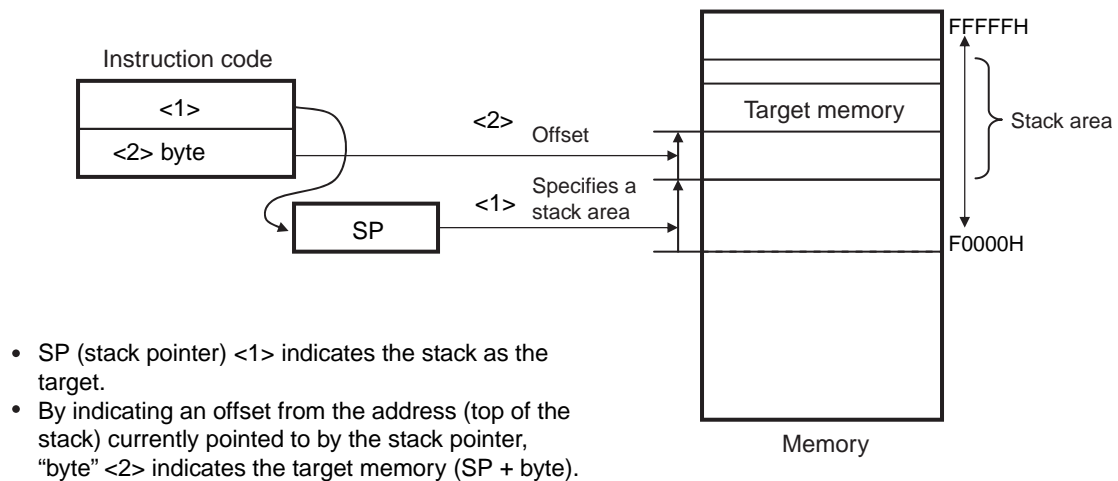
#### [Function]

Based addressing uses the contents of a register pair specified with the instruction word or 16-bit immediate data as a base address, and 8-bit immediate data or 16-bit immediate data as offset data. The sum of these values is used to specify the target address.

#### [Operand format]

Identifier	Description
–	[HL + byte], [DE + byte], [SP + byte] (only the space from F0000H to FFFFFH is specifiable)
–	word[B], word[C] (only the space from F0000H to FFFFFH is specifiable)
–	word[BC] (only the space from F0000H to FFFFFH is specifiable)
–	ES:[HL + byte], ES:[DE + byte] (higher 4-bit addresses are specified by the ES register)
–	ES:word[B], ES:word[C] (higher 4-bit addresses are specified by the ES register)
–	ES:word[BC] (higher 4-bit addresses are specified by the ES register)

Figure 3-23. Example of [SP+byte]



- SP (stack pointer)  $\langle 1 \rangle$  indicates the stack as the target.
- By indicating an offset from the address (top of the stack) currently pointed to by the stack pointer, "byte"  $\langle 2 \rangle$  indicates the target memory (SP + byte).

Figure 3-24. Example of [HL+byte], [DE+byte]

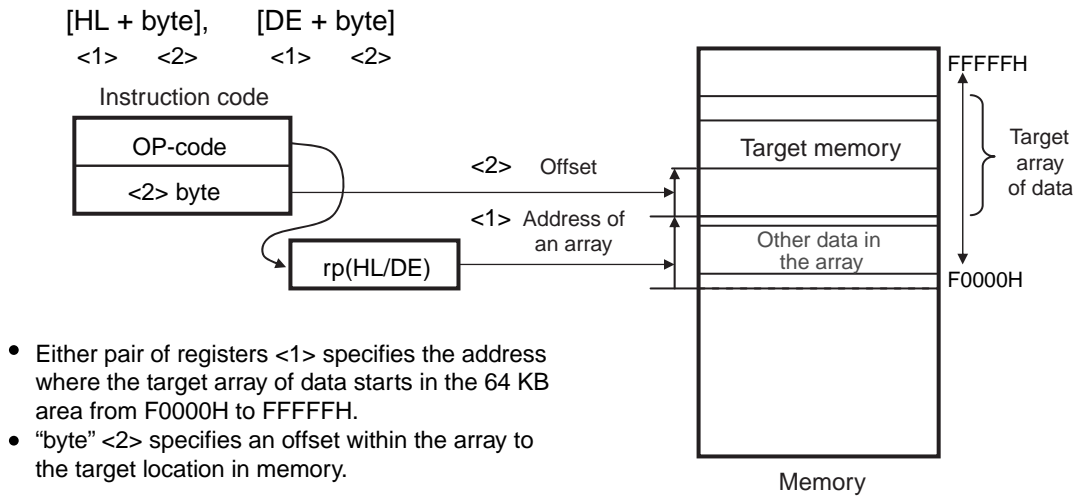


Figure 3-25. Example of word[B], word[C]

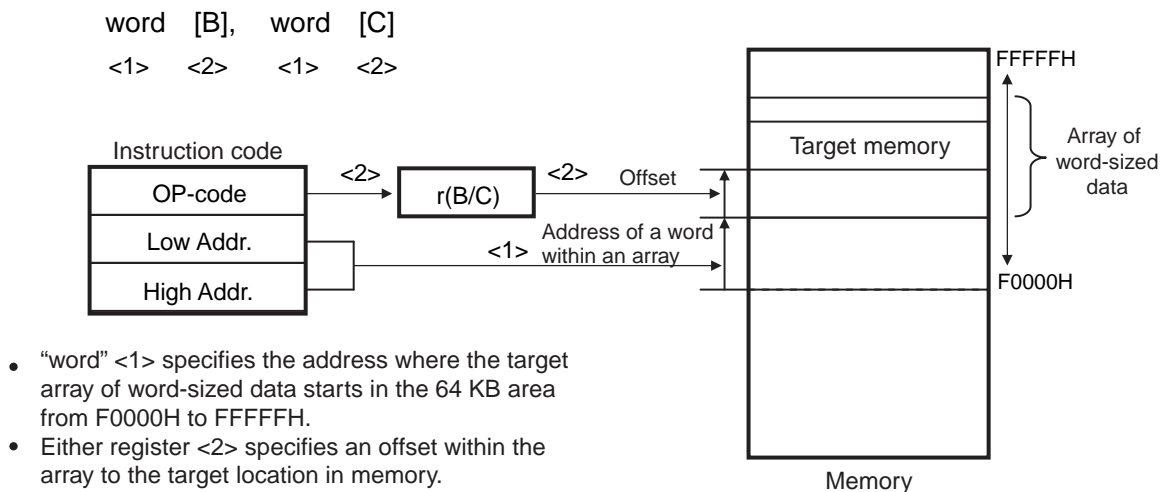


Figure 3-26. Example of word[BC]

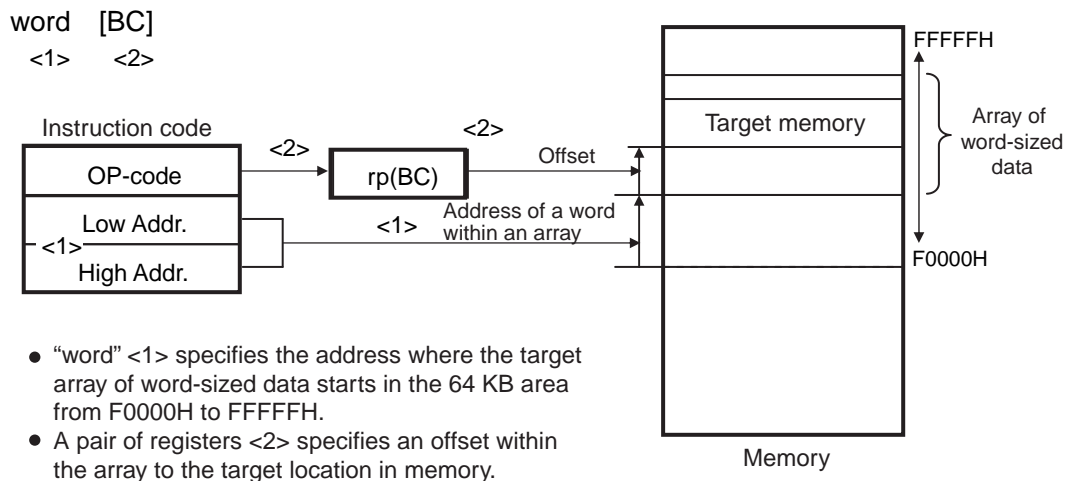
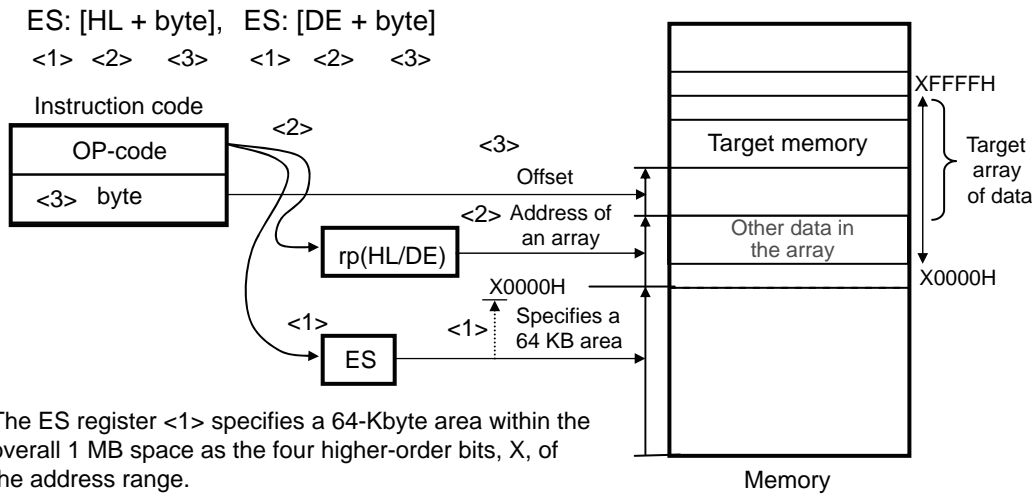
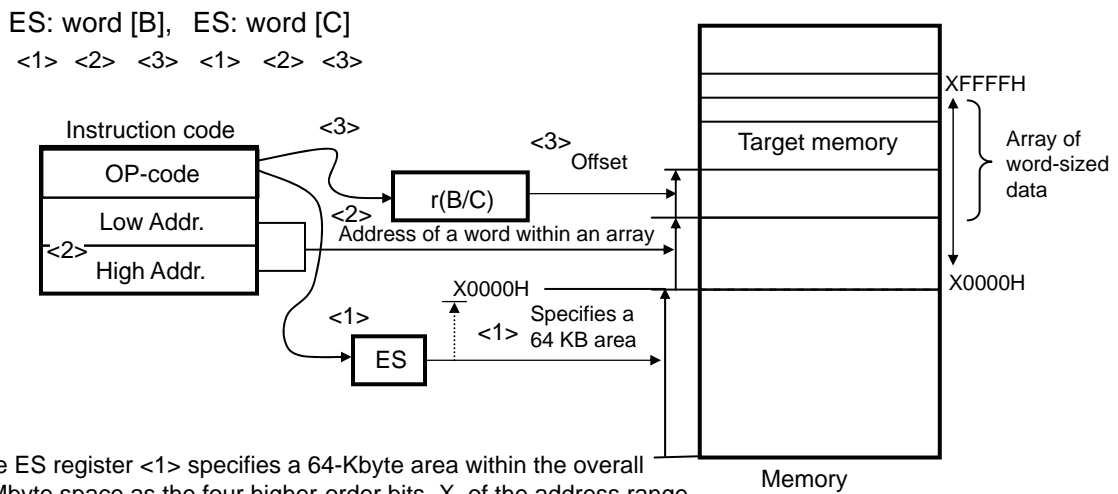


Figure 3-27. Example of ES:[HL+byte], ES:[DE+byte]



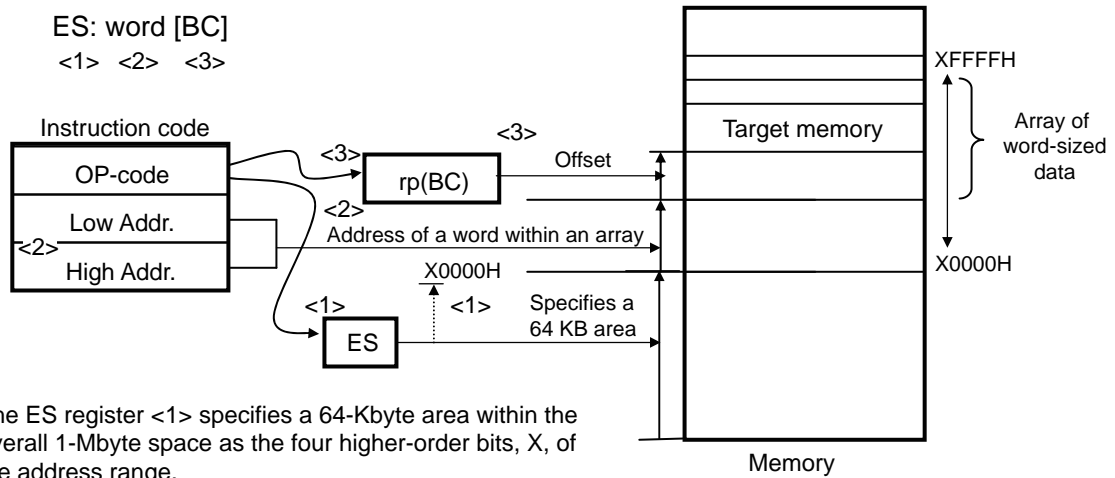
- The ES register <1> specifies a 64-Kbyte area within the overall 1 MB space as the four higher-order bits, X, of the address range.
- Either pair of registers <2> specifies the address where the target array of data starts in the 64-Kbyte area specified in the ES register <1>.
- "byte" <3> specifies an offset within the array to the target location in memory.

Figure 3-28. Example of ES:word[B], ES:word[C]



- The ES register <1> specifies a 64-Kbyte area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- "word" <2> specifies the address where the target array of word-sized data starts in the 64-Kbyte area specified in the ES register <1>.
- Either register <3> specifies an off set within the array to the target location in memory.

Figure 3-29. Example of ES:word[BC]



- The ES register <1> specifies a 64-Kbyte area within the overall 1-Mbyte space as the four higher-order bits, X, of the address range.
- "word" <2> specifies the address where the target array of word-sized data starts in the 64 KB area specified in the ES register <1>.
- A pair of registers <3> specifies an offset within the array to the target location in memory.

3.4.8 Based indexed addressing

[Function]

Based indexed addressing uses the contents of a register pair specified with the instruction word as the base address, and the content of the B register or C register similarly specified with the instruction word as offset address. The sum of these values is used to specify the target address.

[Operand format]

Identifier	Description
-	[HL+B], [HL+C] (only the space from F0000H to FFFFFH is specifiable)
-	ES:[HL+B], ES:[HL+C] (higher 4-bit addresses are specified by the ES register)

Figure 3-30. Example of [HL+B], [HL+C]

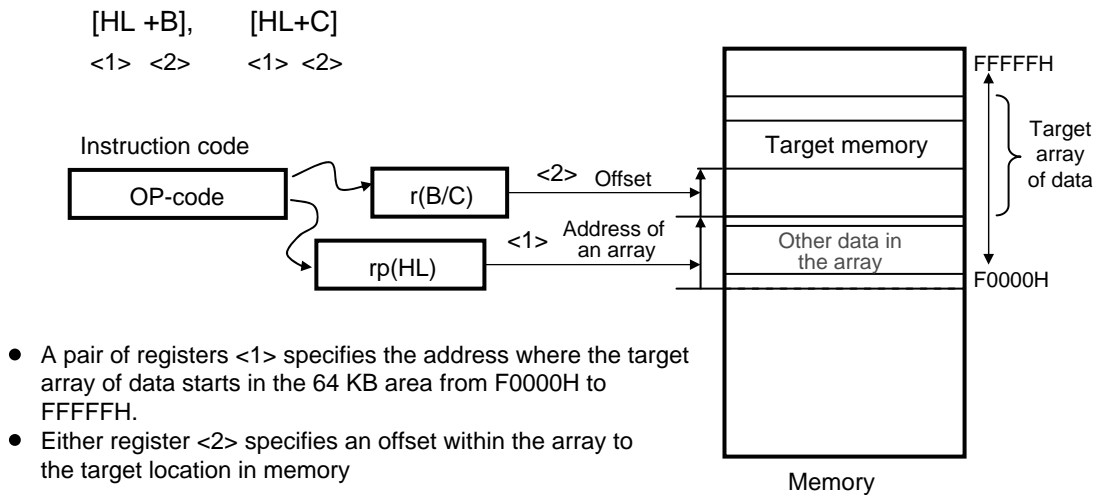
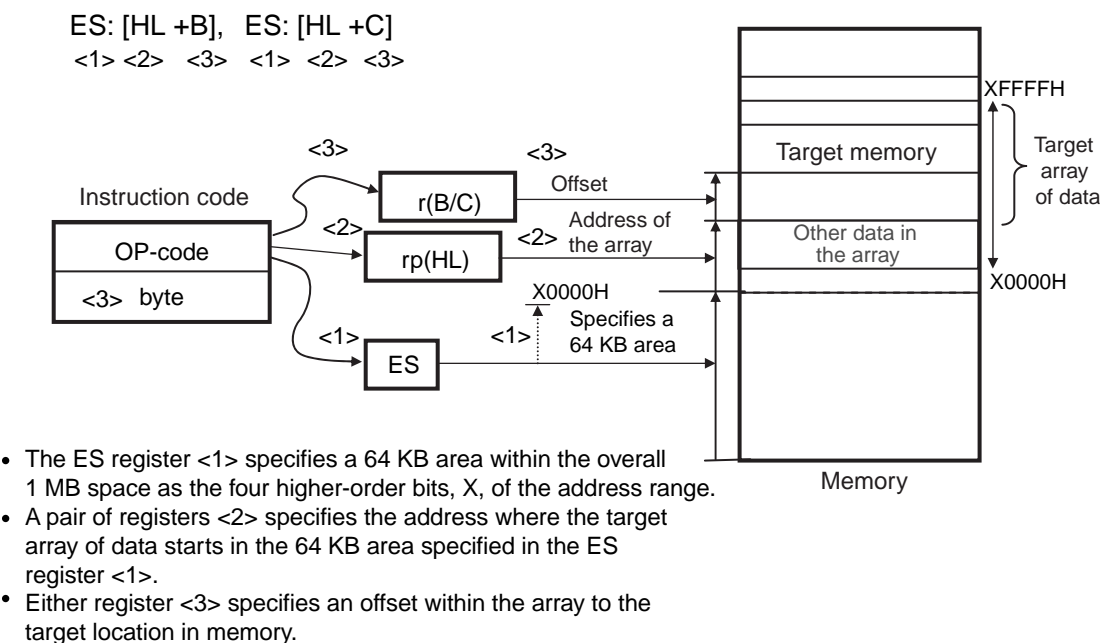


Figure 3-31. Example of ES:[HL+B], ES:[HL+C]



### 3.4.9 Stack addressing

#### [Function]

The stack area is indirectly addressed with the stack pointer (SP) values. This addressing is automatically employed when the PUSH, POP, subroutine call, and return instructions are executed or the register is saved/restored upon generation of an interrupt request.

Only the internal RAM area can be set as the stack area.

#### [Description format]

Identifier	Description
–	PUSH PSW AX/BC/DE/HL POP PSW AX/BC/DE/HL CALL/CALLT RET BRK RETB (Interrupt request generated) RETI

The data saved/restored by each stack operation is shown in Figures 3-32 to 3-37.

Figure 3-32. Example of PUSH rp

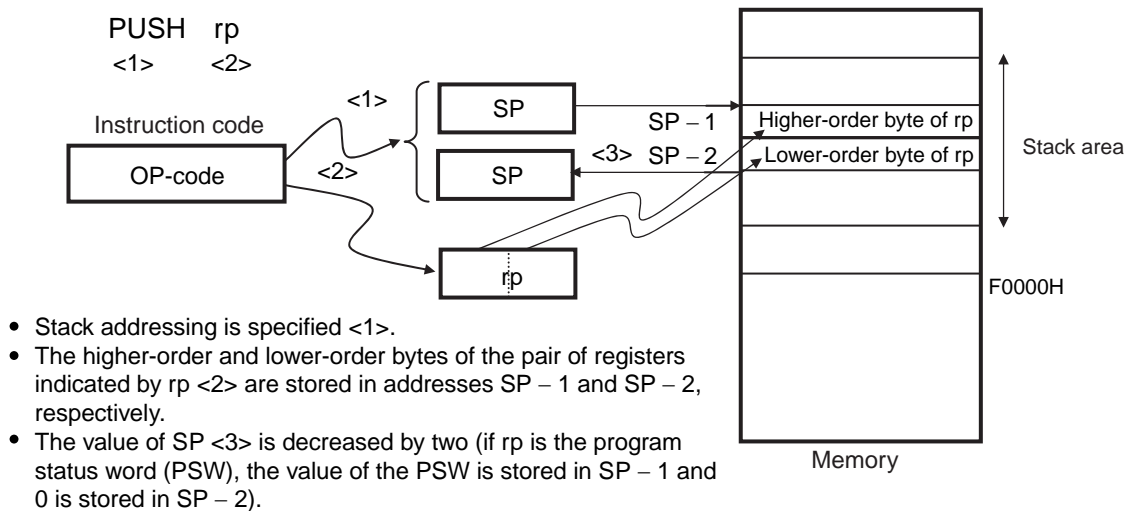
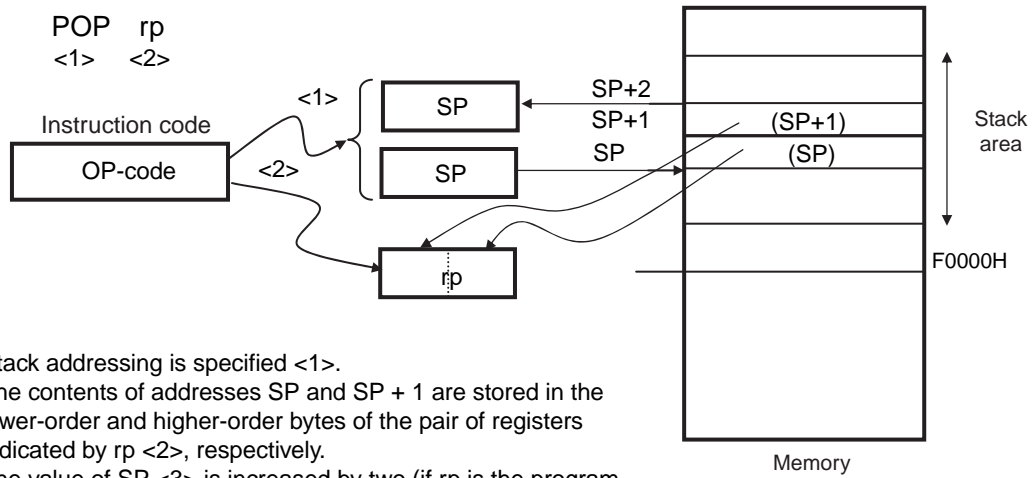
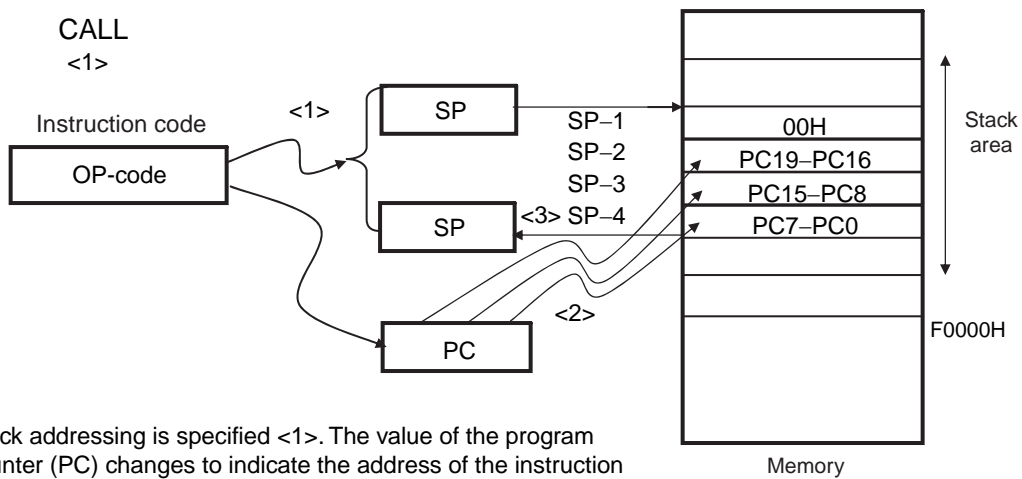


Figure 3-33. Example of POP



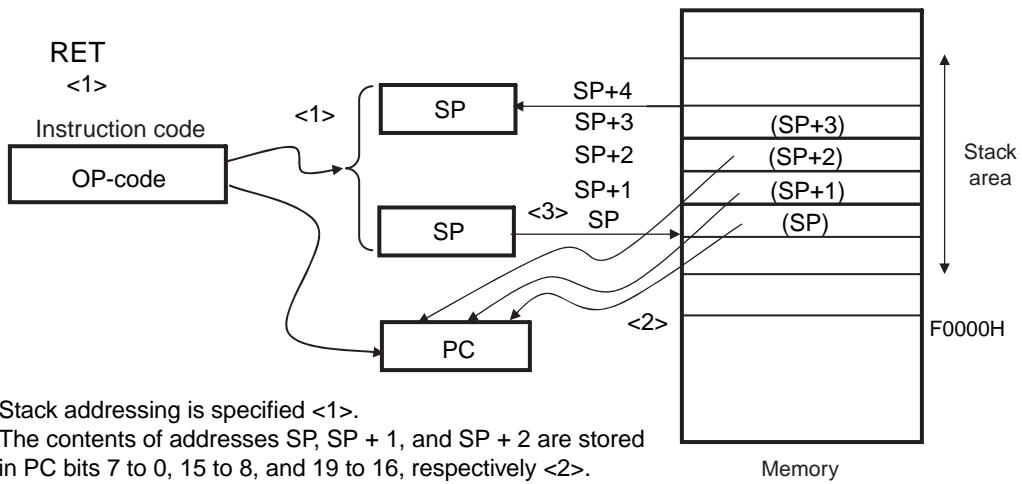
- Stack addressing is specified <1>.
- The contents of addresses SP and SP + 1 are stored in the lower-order and higher-order bytes of the pair of registers indicated by rp <2>, respectively.
- The value of SP <3> is increased by two (if rp is the program status word (PSW), the content of address SP + 1 is stored in the PSW).

Figure 3-34. Example of CALL, CALLT



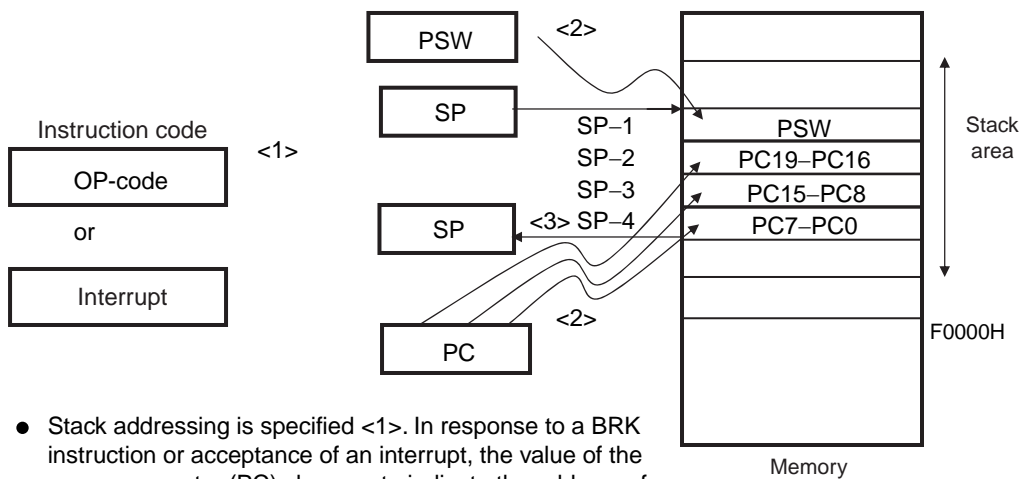
- Stack addressing is specified <1>. The value of the program counter (PC) changes to indicate the address of the instruction following the CALL instruction.
- 00H, the values of PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP - 1, SP - 2, SP - 3, and SP - 4, respectively <2>.
- The value of the SP <3> is decreased by 4.

Figure 3-35. Example of RET



- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, and SP + 2 are stored in PC bits 7 to 0, 15 to 8, and 19 to 16, respectively <2>.
- The value of SP <3> is increased by four.

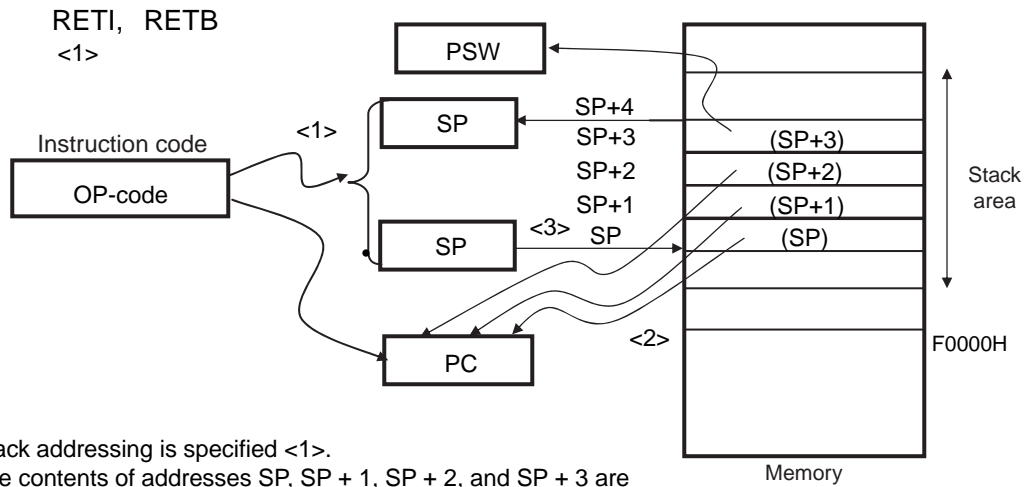
Figure 3-36. Example of Interrupt, BRK



- Stack addressing is specified <1>. In response to a BRK instruction or acceptance of an interrupt, the value of the program counter (PC) changes to indicate the address of the next instruction.
- The values of the PSW, PC bits 19 to 16, 15 to 8, and 7 to 0 are stored in addresses SP - 1, SP - 2, SP - 3, and SP - 4, respectively <2>.
- The value of the SP <3> is decreased by 4.



Figure 3-37. Example of RETI, RETB



- Stack addressing is specified <1>.
- The contents of addresses SP, SP + 1, SP + 2, and SP + 3 are stored in PC bits 7 to 0, 15 to 8, 19 to 16, and the PSW, respectively <2>.
- The value of SP <3> is increased by four.

## CHAPTER 4 PORT FUNCTIONS

### 4.1 Port Functions

The R7F0C010 microcontrollers are provided with digital I/O ports, which enable variety of control operations.

In addition to the function as digital I/O ports, these ports have several alternate functions. For details of the alternate functions, see **CHAPTER 2 PIN FUNCTIONS**.

### 4.2 Port Configuration

Ports include the following hardware.

**Table 4-1. Port Configuration**

Item	Configuration
Control registers	Port mode registers (PM1 to PM4, PM6) Port registers (P1 to P4, P6, P12, P13) Pull-up resistor option registers ( PU1, PU3, PU4) Port mode control register 1 (PMC1) A/D port configuration register (ADPC)
Port	<ul style="list-style-type: none"> <li>• 24-pin products: Total: 20 (CMOS I/O: 15, CMOS input: 3, N-ch open drain I/O: 2)</li> <li>• 32-pin products: Total: 28 (CMOS I/O: 23, CMOS input: 3, N-ch open drain I/O: 2)</li> </ul>
Pull-up resistor	<ul style="list-style-type: none"> <li>• 24-pin products: Total:10</li> <li>• 32-pin products: Total:15</li> </ul>

**Caution** Most of the following descriptions in this chapter use the 32-pin products as an example.

### 4.2.1 Port 1

Port 1 is an I/O port with an output latch. Port 1 can be set to the input mode or output mode in 1-bit units using port mode register 1 (PM1). When the P10 to P17 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 1 (PU1).

This port can also be used for timer I/O, external interrupt request input, and clock/buzzer output.

Reset signal generation sets port 1 to input mode.

**Table 4-2. Settings of Registers When Using Port 1**

Port	I/O	PM1×	Alternate Function Setting	Remark
P10	Input	1	×	CMOS input
	Output	0	×	CMOS output
P11	Input	1	×	CMOS input
	Output	0	×	CMOS output
P12	Input	1	×	CMOS input
	Output	0	PCLBUZ0 output = 0 <sup>Note 1</sup> TO03 output = 0 <sup>Note 3</sup>	CMOS output
P13	Input	1	×	CMOS input
	Output	0	TO00 output = 0 <sup>Note 3</sup>	CMOS output
P14	Input	1	×	CMOS input
	Output	0	×	CMOS output
P15	Input	1	×	CMOS input
	Output	0	PCLBUZ1 output = 0 <sup>Note 2</sup>	CMOS output
P16	Input	1	×	CMOS input
	Output	0	TO01 output = 0 <sup>Note 3</sup>	CMOS output
P17	Input	1	×	CMOS input
	Output	0	×	CMOS output

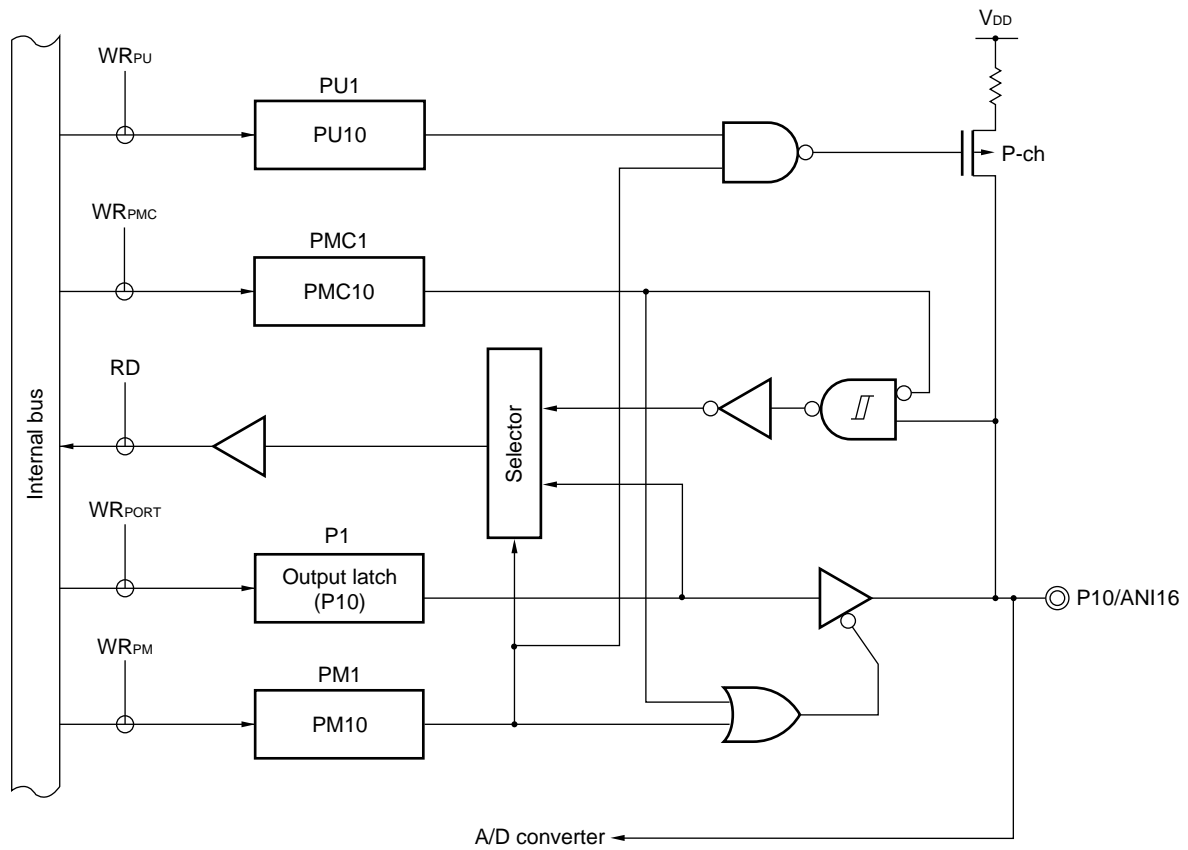
- Notes**
1. To use P12/TI03/TO03/INTP4/PCLBUZ0 as a general-purpose port, set bit 7 (PCLOE0) of clock output select register 0 (CKS0) to "0", which is the same as their default status setting.
  2. To use P15/PCLBUZ1 as a general-purpose port, set bit 7 (PCLOE1) of clock output select register 1 (CKS1) to "0", which is the same as their default status setting.
  3. To use P12/TI03/TO03/INTP4/PCLBUZ0, P13/TI00/TO00, or P16/TI01/TO01/INTP5 as a general-purpose port, set bits 0, 1, and 3 (TO00, TO01, TO03) of timer output register 0 (TO0) and bits 0, 1, and 3 (TOE00, TOE01, TOE03) of timer output enable register 0 (TOE0) to "0", which is the same as their default status setting.

**Remark** ×: don't care

PM1×: Port mode register 1

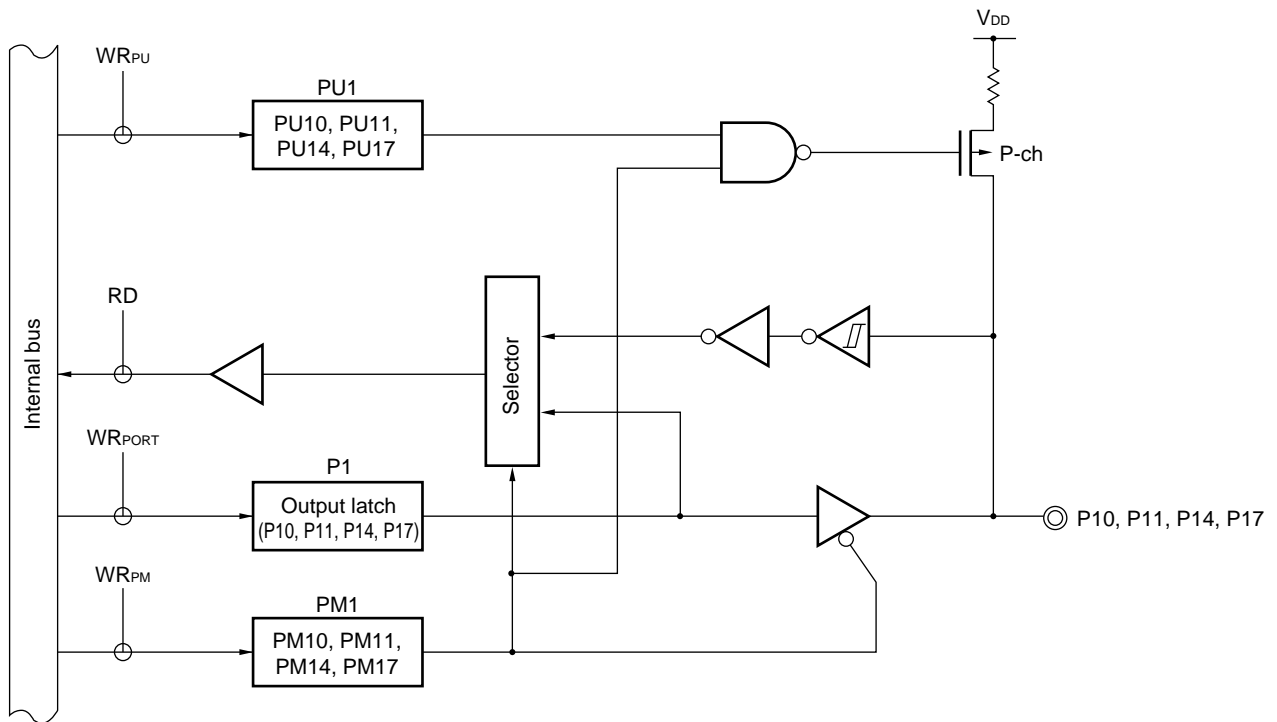
For example, figures 4-1 to 4-4 show block diagrams of port 1.

Figure 4-1. Block Diagrams of P10 (24-pin products)



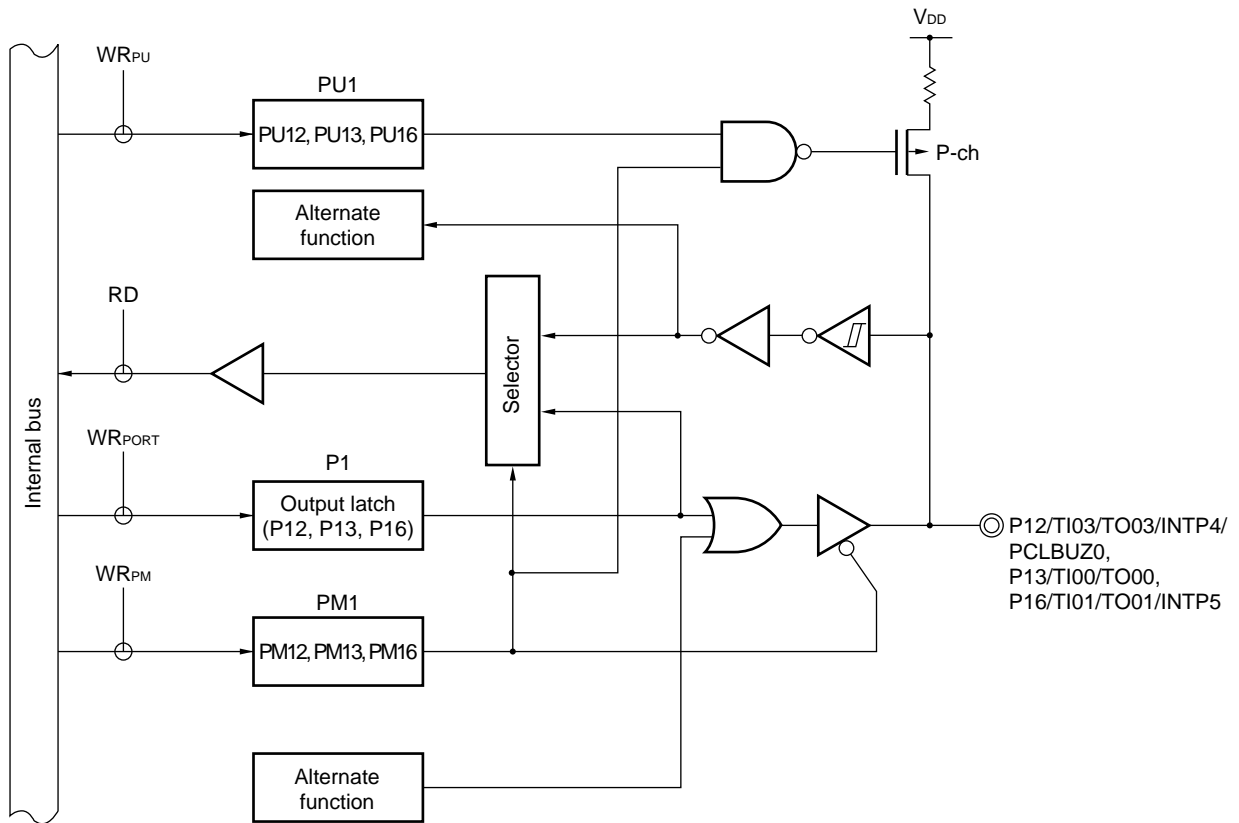
- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- PMC1: Port mode control register 1
- WR<sub>xx</sub>: Write signal

Figure 4-2. Block Diagram of P10, P11, P14, and P17 (32-pin products)



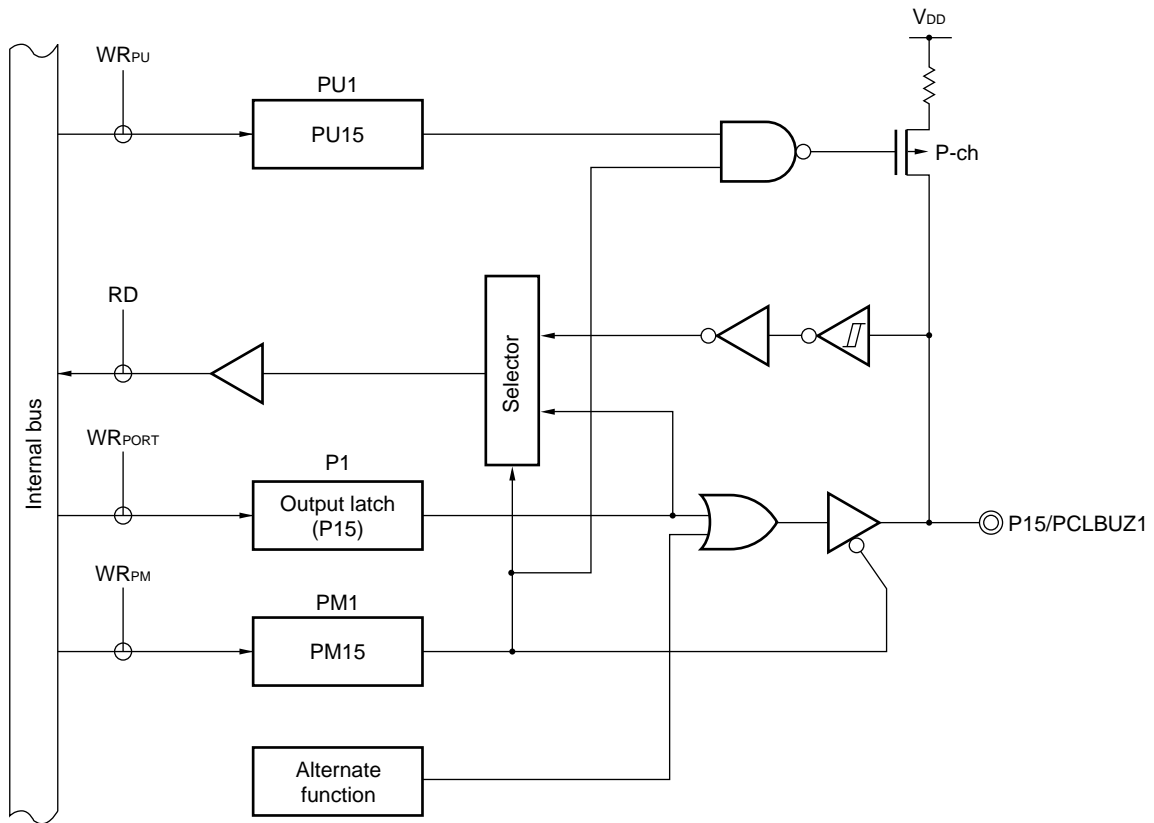
- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- WR<sub>xx</sub>: Write signal

Figure 4-3. Block Diagram of P12, P13, and P16



- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- WR<sub>xx</sub>: Write signal

Figure 4-4. Block Diagrams of P15



- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- $WR_{xx}$ : Write signal

### 4.2.2 Port 2

Port 2 is an I/O port with an output latch. Port 2 can be set to the input mode or output mode in 1-bit units using port mode register 2 (PM2).

This port can also be used for A/D converter analog input, (+ side and – side) reference voltage input, and D/A converter output.

To use P20/ANI0, P21/ANI1, P22/ANI2/ANO0, P23/ANI3/ANO1, P24/ANI4 to P27/ANI7 as digital input pins or digital output pins, use these pins as following order.

P27 → P26 → P25 → P21 → P20 → P22 → P23 → P24

To use P20 and P21 as A/D converter analog input and (+ side and – side) reference voltage input when P20/ANI0, P21/ANI1, P22/ANI2/ANO0, P23/ANI3/ANO1, P24/ANI4 to P27/ANI7 is used as analog I/O pins, use these pins starting from the next pin of P20 or P21.

In other case, use these pins as following order.

P24 → P23 → P22 → P20 → P21 → P25 → P26 → P27

**Table 4-3. Settings of Registers When Using Port 2**

Name	I/O	PM2n	ADPC	Alternate Function Setting	Remark
P2n	Input	1	1	–	To use P2n as a port, use these pins from a higher bit.
	Output	0	1		

**Remarks 1.** PM2x: Port mode register 2

ADPC: A/D port configuration register

2. n = 0 to 7

**Table 4-4. Setting Functions of P20/ANI0, P21/ANI1, P24/ANI4 to P27/ANI7 Pins**

ADPC Register	PM2 Register	ADS Register	P20/ANI0, P21/ANI1, P24/ANI4 to P27/ANI7 Pins
Digital I/O selection	Input mode	–	Digital input
	Output mode	–	Digital output
Analog input selection	Input mode	Selects ANI.	Analog input (to be converted)
		Does not select ANI.	Analog input (not to be converted)
	Output mode	Selects ANI.	Setting prohibited
		Does not select ANI.	

P20/ANI0, P21/ANI1, P24/ANI4 to P27/ANI7 are set in the analog input mode when the reset signal is generated.



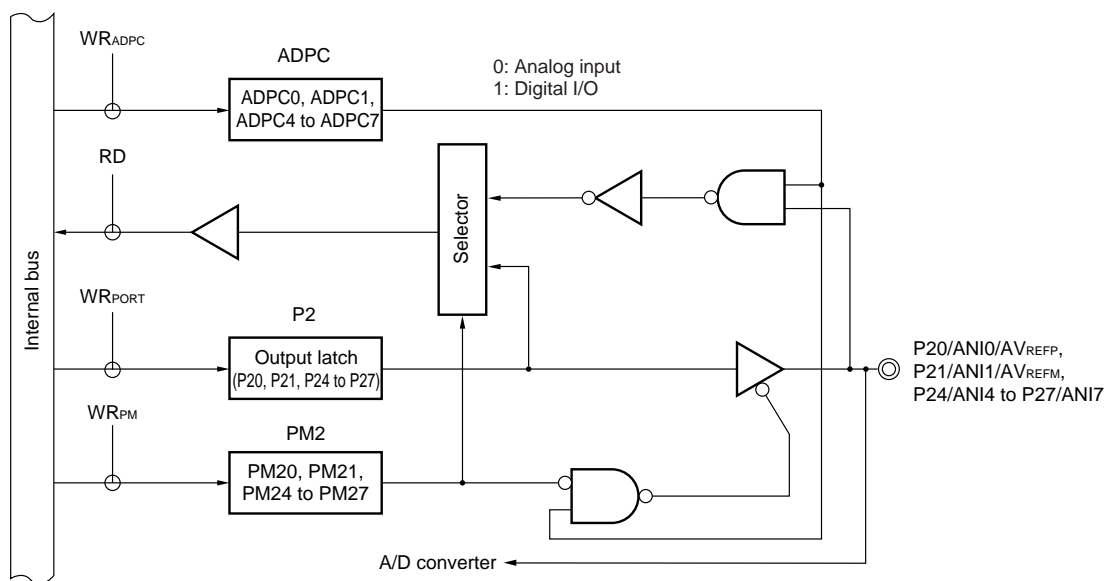
**Table 4-5. Setting Functions of P22/ANI2/ANO0, P23/ANI3/ANO1 Pins**

ADPC Register	PM2 Register	DAM Register	ADS Register	P22/ANI2/ANO0, P23/ANI3/ANO1 Pins
Digital I/O selection	Input mode	–	–	Digital input
	Output mode	–	–	Digital output
Analog input selection	Input mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	Analog output
		Stops D/A conversion operation	Selects ANI.	Analog input (to be converted)
			Does not select ANI.	Analog input (not to be converted)
	Output mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	
Stops D/A conversion operation	Selects ANI.			
	Does not select ANI.			

P22/ANI2/ANO0 and P23/ANI3/ANO1 are set in the analog input mode when the reset signal is generated.

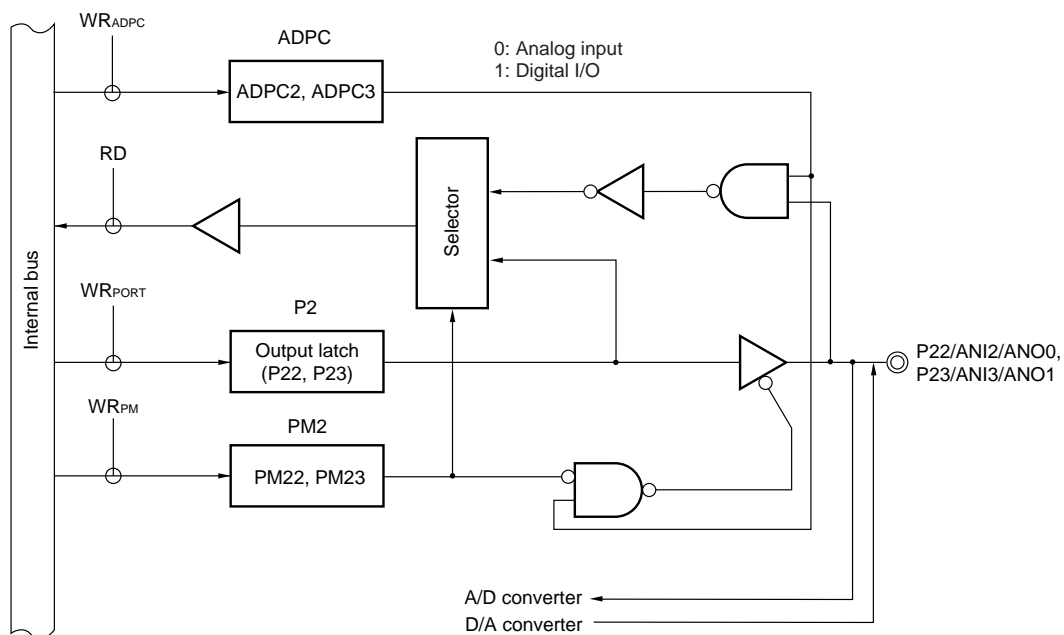
For example, figures 4-5 and 4-6 show block diagrams of port 2 for 32-pin products.

**Figure 4-5. Block Diagram of P20, P21, P24 to P27**



- ADPC: A/D port configuration register
- P2: Port register 2
- PM2: Port mode register 2
- RD: Read signal
- WR<sub>xx</sub>: Write signal

**Figure 4-6. Block Diagram of P22 and P23**



- P2: Port register 2
- PM2: Port mode register 2
- RD: Read signal
- WR<sub>xx</sub>: Write signal

### 4.2.3 Port 3

Port 3 is an I/O port with an output latch. Port 3 can be set to the input mode or output mode in 1-bit units using port mode register 3 (PM3). When the P30 to P35 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3).

This port can also be used for external interrupt request input, serial interface data I/O, clock I/O, programming UART transmission/reception, timer I/O, and chip select input.

Reset signal generation sets P30 to P35 to input mode.

**Table 4-6. Settings of Registers When Using Port 3**

Name	I/O	PM3 <sub>x</sub>	Alternate Function Setting	Remark
P30	Input	1	×	
	Output	0	SO00/TxD0 output = 1 <sup>Note 1</sup>	
P31	Input	1	×	
	Output	0	×	
P32	Input	1	×	
	Output	0	SCK00 output = 1 <sup>Note 1</sup>	
P33	Input	1	×	
	Output	0	TO02 output = 0 <sup>Note 2</sup>	
P34, P35	Input	1	×	
	Output	0	×	

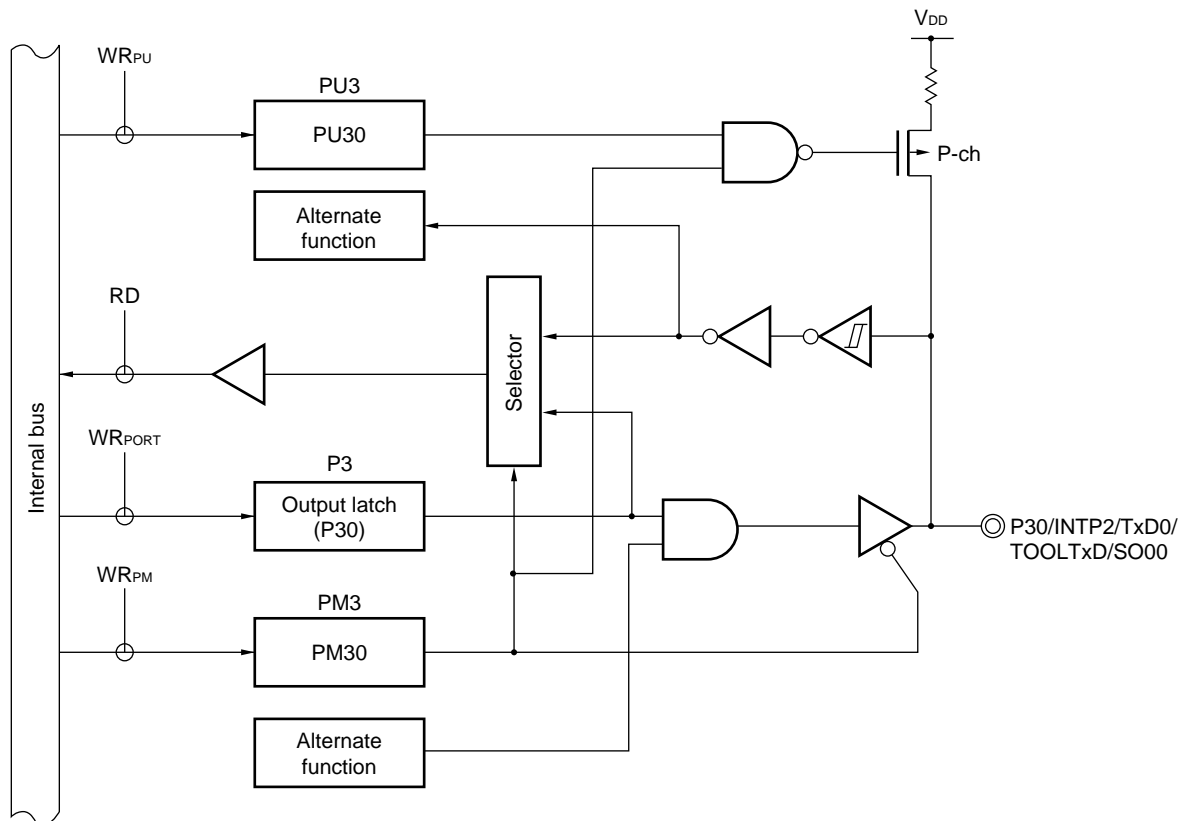
- Notes**
- To use P30/INTP2/TxD0/TOOLTxD/SO00, P32/INTP3/SCK00 as a general-purpose port, set bit 0 (SE0) of serial channel enable status register 0 (SE0), bit 0 (SO00) of serial output register 0 (SO0) and bit 0 (SOE00) of serial output enable register 0 (SOE0) to the default status.
  - To use P33/TI02/TO02 as a general-purpose port, set bit 2 (TO02) of timer output register 0 (TO0) and bit 2 (TOE02) of timer output enable register 0 (TOE0) to "0", which is the same as their default status setting.

**Remark** ×: don't care

PM3<sub>x</sub>: Port mode register 3

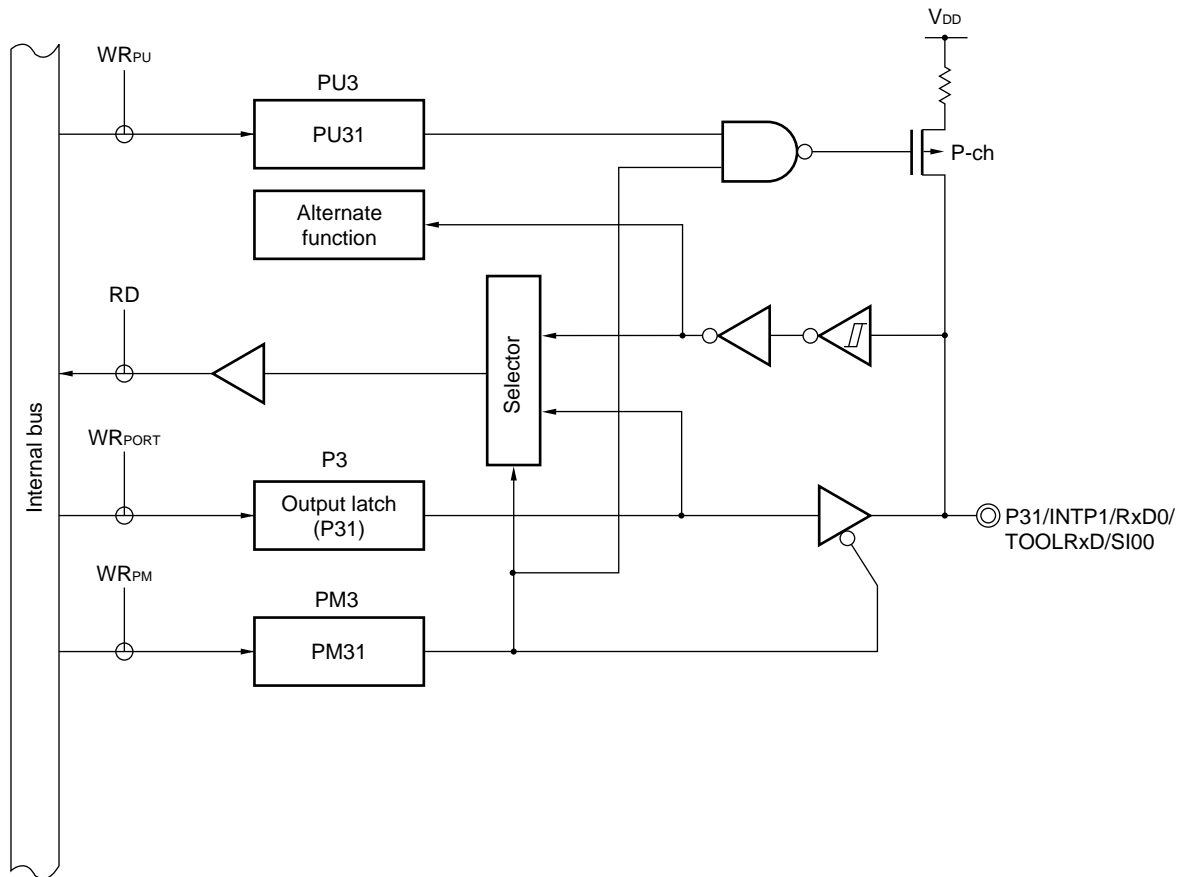
For example, figures 4-7 to 4-11 show block diagrams of port 3 for 32-pin products.

Figure 4-7. Block Diagram of P30



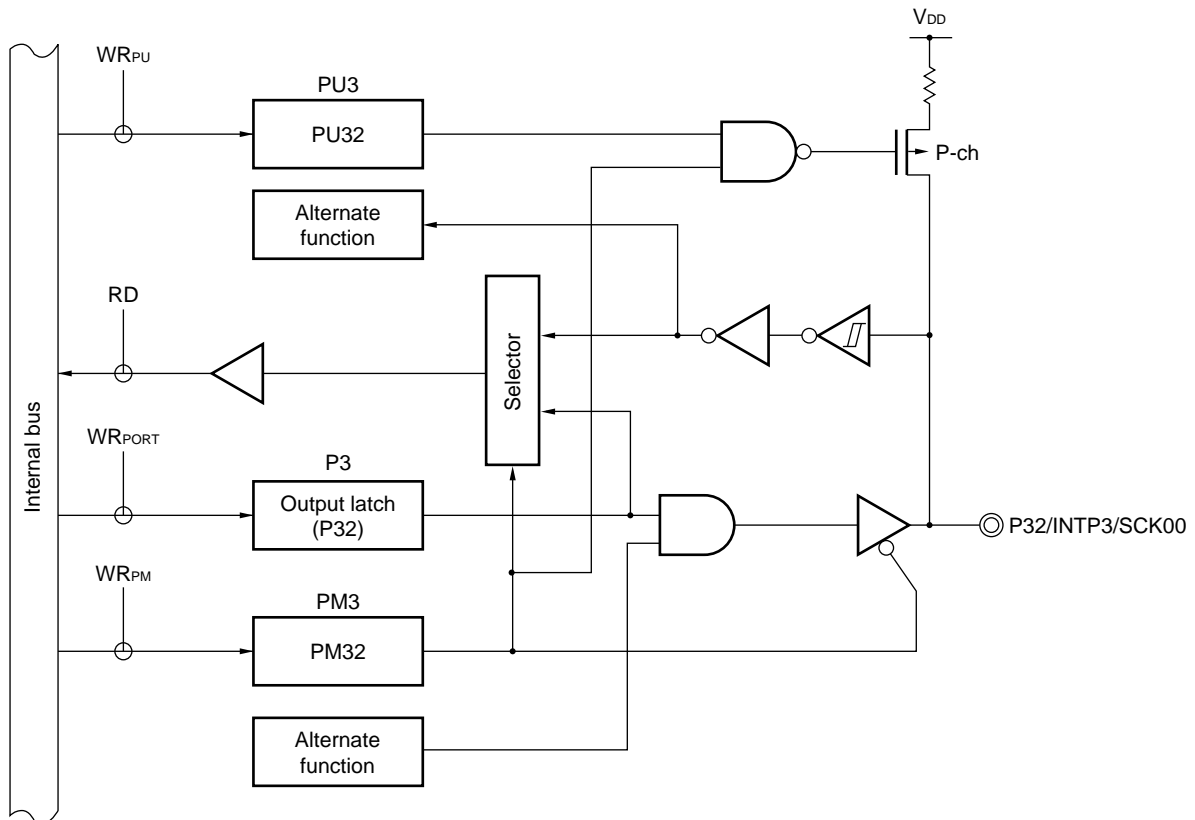
- P3: Port register 3
- PU3: Pull-up resistor option register 3
- PM3: Port mode register 3
- RD: Read signal
- $WR_{xx}$ : Write signal

Figure 4-8. Block Diagram of P31



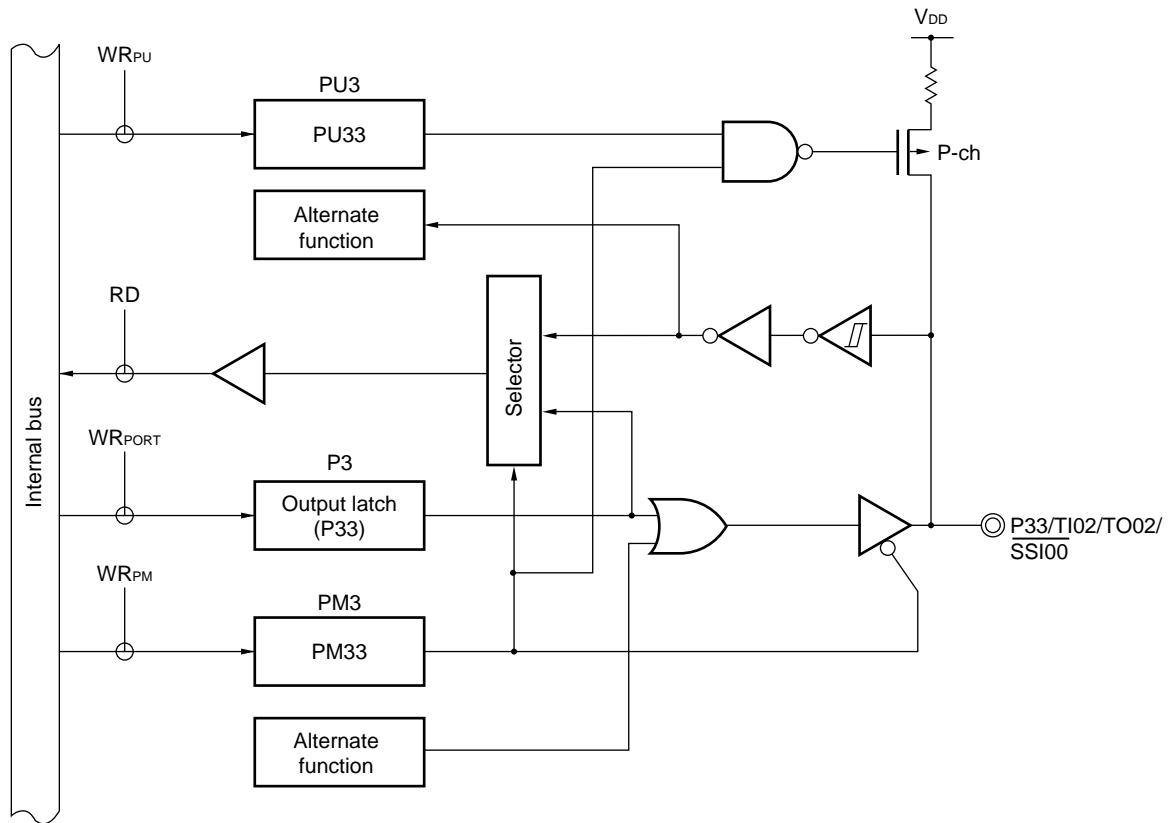
- P3: Port register 3  
 PU3: Pull-up resistor option register 3  
 PM3: Port mode register 3  
 RD: Read signal  
 WR<sub>xx</sub>: Write signal

Figure 4-9. Block Diagram of P32



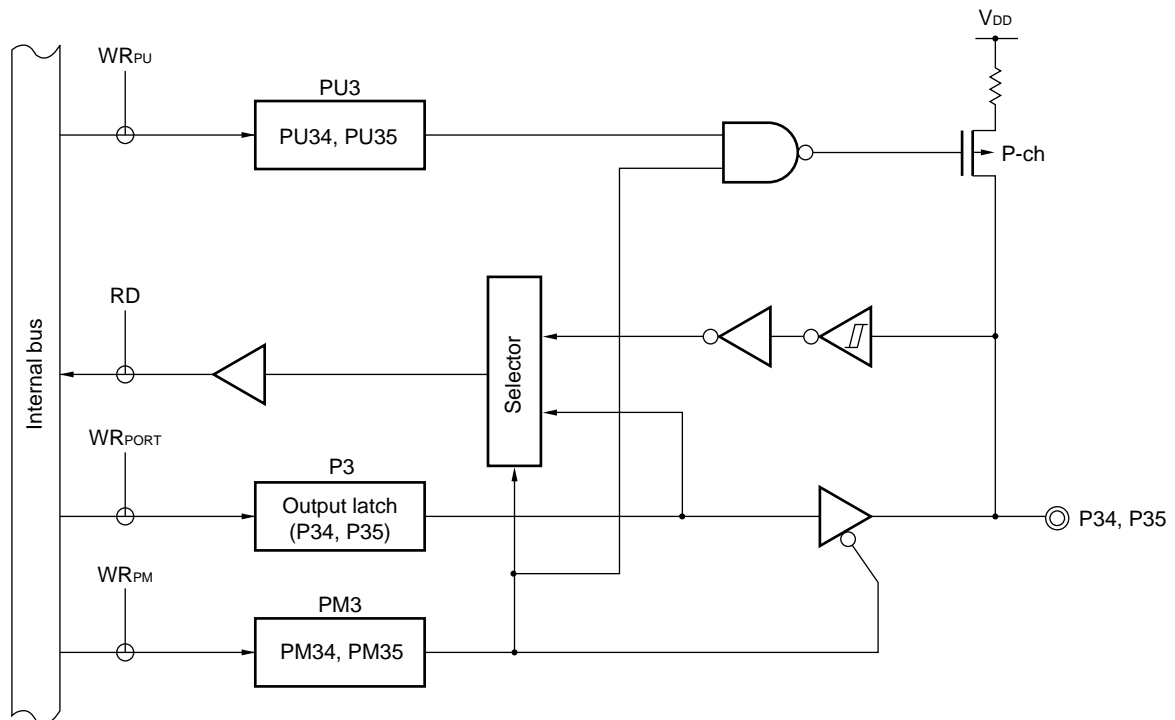
- P3: Port register 3
- PU3: Pull-up resistor option register 3
- PM3: Port mode register 3
- RD: Read signal
- WR<sub>xx</sub>: Write signal

Figure 4-10. Block Diagram of P33



- P3: Port register 3
- PU3: Pull-up resistor option register 3
- PM3: Port mode register 3
- RD: Read signal
- WR<sub>xx</sub>: Write signal

Figure 4-11. Block Diagram of P34 and P35



- P3: Port register 3
- PU3: Pull-up resistor option register 3
- PM3: Port mode register 3
- RD: Read signal
- WR<sub>xx</sub>: Write signal



4.2.4 Port 4

Port 4 is an I/O port with an output latch. Port 4 can be set to the input mode or output mode using port mode register 4 (PM4). When the P40 pin is used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 4 (PU4).

This port can also be used for data I/O for a flash memory programmer/debugger. Reset signal generation sets port 4 to input mode.

Table 4-7. Settings of Registers When Using Port 4

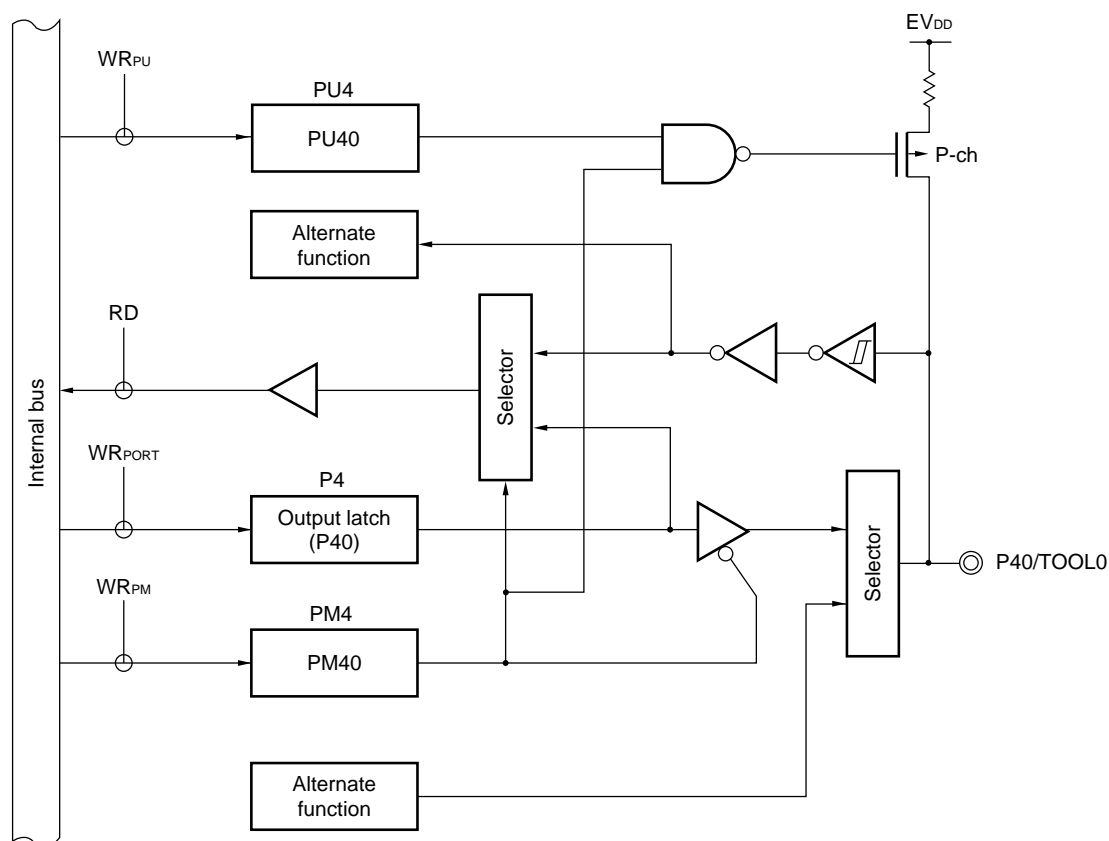
Name	I/O	PM4x	Alternate Function Setting	Remark
P40	Input	1	x	
	Output	0	x	

**Caution** When a tool is connected, the P40 pin cannot be used as a port pin.

**Remark** x: don't care  
 PM4x: Port mode register 4

Figure 4-12 shows a block diagram of port 4.

Figure 4-12. Block Diagram of P40



- P4: Port register 4
- PU4: Pull-up resistor option register 4
- PM4: Port mode register 4
- RD: Read signal
- WR<sub>xx</sub>: Write signal

4.2.5 Port 6

Port 6 is an I/O port with an output latch. Port 6 can be set to the input mode or output mode in 1-bit units using port mode register 6 (PM6).

The output of the P60 and P61 pins is N-ch open-drain output (6 V tolerance).

This port can also be used for serial interface data I/O and clock I/O.

Reset signal generation sets port 6 to input mode.

Table 4-8. Settings of Registers When Using Port 6

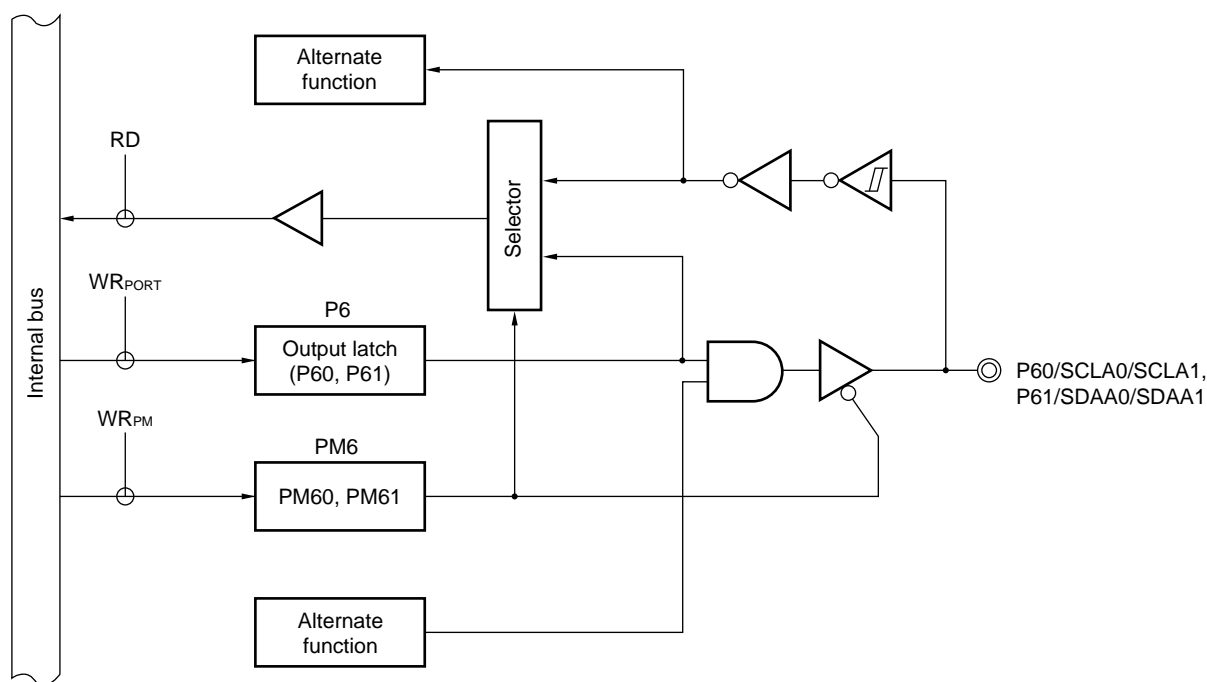
Name	I/O	PM6 <sub>x</sub>	Alternate Function Setting	Remark
P60	Input	1	SCLA0 output = 0 <sup>Note</sup>	
	Output	0		
P61	Input	1	SDAA0 output = 0 <sup>Note</sup>	
	Output	0		

**Note** Stop the operation of serial interface IICA when using P60/SCLA0/SCLA1 and P61/SDAA0/SDAA1 as general-purpose ports.

**Remark** x: don't care  
 PM6<sub>x</sub>: Port mode register 6

Figure 4-13 shows a block diagram of port 6

Figure 4-13. Block Diagram of P60 and P61



P6: Port register 6  
 PM6: Port mode register 6  
 RD: Read signal  
 WR<sub>xx</sub>: Write signal

4.2.6 Port 12

P121 and P122 are 2-bit input only ports.

This port can also be used for connecting resonator for main system clock and external clock input for main system clock.

Reset signal generation sets port 12 to input mode.

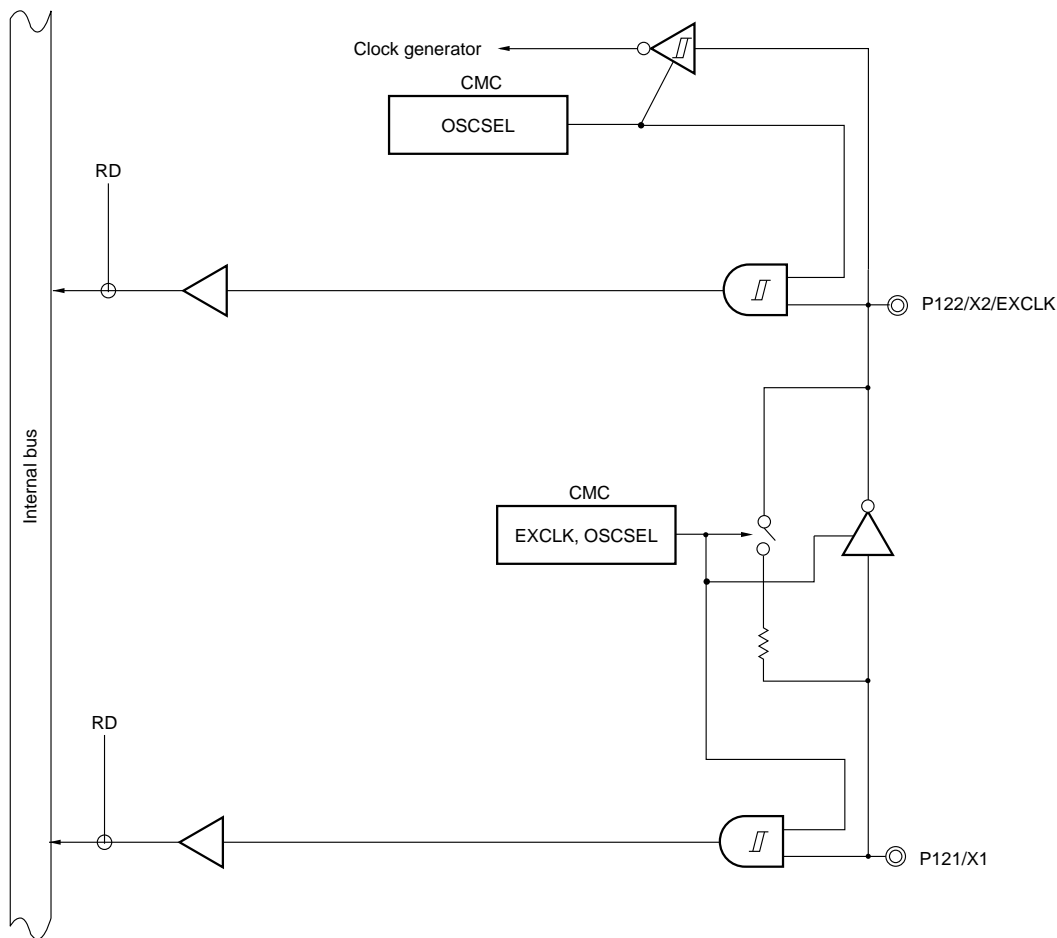
Table 4-12. Settings of Registers When Using Port 12

Name	I/O	Alternate Function Setting	Remark
P121	Input	OSCSEL bit of CMC register = 0 or EXCLK bit = 1	
P122	Input	OSCSEL bit of CMC register = 0	

**Caution** The function setting on P121 and P122 is available only once after the reset release. The port once set for connection to an X1 oscillator, external clock input cannot be used as an input port unless the reset is performed.

Figure 4-14 shows a block diagram of port 12.

Figure 4-14. Block Diagram of P121 and P122



CMC: Clock operation mode control register

RD: Read signal

**4.2.7 Port 13**

P137 is a 1-bit input-only port.

P137 is fixed to an input ports.

This port can also be used for external interrupt request input.

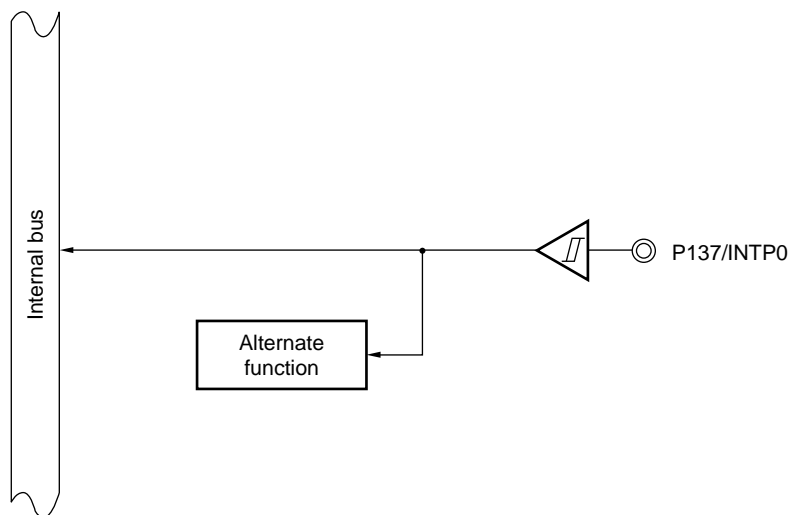
**Table 4-10. Settings of Registers When Using Port 13**

Name	I/O	Alternate Function Setting	Remark
P137	Input	x	

**Remark** x: don't care

Figure 4-15 shows a block diagram of port 13.

**Figure 4-15. Block Diagram of P137**



### 4.3 Registers Controlling Port Function

Port functions are controlled by the following registers.

- Port mode registers (PMxx)
- Port registers (Pxx)
- Pull-up resistor option registers (PUxx)
- Port mode control register 1 (PMC1)
- A/D port configuration register (ADPC)

<R>

**Caution** Which registers and bits are included depends on the product. For registers and bits mounted on each product, see Tables 4 - 11. Be sure to set bits that are not mounted to their initial values.

**Table 4-11. PMxx, Pxx, PUxx, PMC1 Registers and Bits Mounted on Each Product (1/2)**

Port		Bit Name				32 Pin	24 Pin
		PMxx Register	Pxx Register	PUxx Register	PMC1 Register <sup>Note</sup>		
Port 1	0	PM10	P10	PU10	PMC10 <sup>Note</sup>	√	√
	1	PM11	P11	PU11	–	√	–
	2	PM12	P12	PU12	–	√	√
	3	PM13	P13	PU13	–	√	√
	4	PM14	P14	PU14	–	√	–
	5	PM15	P15	PU15	–	√	√
	6	PM16	P16	PU16	–	√	√
	7	PM17	P17	PU17	–	√	–
Port 2	0	PM20	P20	–	–	√	√
	1	PM21	P21	–	–	√	√
	2	PM22	P22	–	–	√	√
	3	PM23	P23	–	–	√	√
	4	PM24	P24	–	–	√	–
	5	PM25	P25	–	–	√	–
	6	PM26	P26	–	–	√	–
	7	PM27	P27	–	–	√	√
Port 3	0	PM30	P30	PU30	–	√	√
	1	PM31	P31	PU31	–	√	√
	2	PM32	P32	PU32	–	√	√
	3	PM33	P33	PU33	–	√	√
	4	PM34	P34	PU34	–	√	–
	5	PM35	P35	PU35	–	√	–
	6	–	–	–	–	–	–
	7	–	–	–	–	–	–

**Note** 24-pin products only.

Table 4-11. PMxx, Pxx, PUxx, PMC1 Registers and Bits Mounted on Each Product (2/2)

Port		Bit Name				32 Pin	24 Pin
		PMxx Register	Pxx Register	PUxx Register	PMC1 Register		
Port 4	0	PM40	P40	PU40	–	√	√
	1	–	–	–	–	–	–
	2	–	–	–	–	–	–
	3	–	–	–	–	–	–
	4	–	–	–	–	–	–
	5	–	–	–	–	–	–
	6	–	–	–	–	–	–
	7	–	–	–	–	–	–
Port 6	0	PM60	P60	–	–	√	√
	1	PM61	P61	–	–	√	√
	2	–	–	–	–	–	–
	3	–	–	–	–	–	–
	4	–	–	–	–	–	–
	5	–	–	–	–	–	–
	6	–	–	–	–	–	–
	7	–	–	–	–	–	–
Port 12	0	–	–	–	–	–	–
	1	–	P121	–	–	√	√
	2	–	P122	–	–	√	√
	3	–	–	–	–	–	–
	4	–	–	–	–	–	–
	5	–	–	–	–	–	–
	6	–	–	–	–	–	–
	7	–	–	–	–	–	–
Port 13	0	–	–	–	–	–	–
	1	–	–	–	–	–	–
	2	–	–	–	–	–	–
	3	–	–	–	–	–	–
	4	–	–	–	–	–	–
	5	–	–	–	–	–	–
	6	–	–	–	–	–	–
	7	–	P137	–	–	√	√

### 4.3.1 Port mode registers (PMxx)

These registers specify input or output mode for the port in 1-bit units.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

When port pins are used as alternate-function pins, set the port mode register by referencing **4.5 Settings of Port Related Register When Using Alternate Function**.

**Figure 4-16. Format of Port Mode Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PM1	PM17 <sup>Note</sup>	PM16	PM15	PM14 <sup>Note</sup>	PM13	PM12	PM11 <sup>Note</sup>	PM10	FFF21H	FFH	R/W
PM2	PM27	PM26 <sup>Note</sup>	PM25 <sup>Note</sup>	PM24 <sup>Note</sup>	PM23	PM22	PM21	PM20	FFF22H	FFH	R/W
PM3	1	1	PM35 <sup>Note</sup>	PM34 <sup>Note</sup>	PM33	PM32	PM31	PM30	FFF23H	FFH	R/W
PM4	1	1	1	1	1	1	1	PM40	FFF24H	FFH	R/W
PM6	1	1	1	1	1	1	PM61	PM60	FFF26H	FFH	R/W
PMmn	Pmn pin I/O mode selection (m = 1 to 4, 6; n = 0 to 7)										
0	Output mode (output buffer on)										
1	Input mode (output buffer off)										

**Note** These are not provided in 24-pin products.

**Caution** Be sure to set bits 6 and 7 of the PM3 register, bits 1 to 7 of the PM4 register, and bits 2 to 7 of the PM6 register to “1”.

### 4.3.2 Port registers (Pxx)

These registers set the output latch value of a port.

If the data is read in the input mode, the pin level is read. If it is read in the output mode, the output latch value is read<sup>Note</sup>.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Note** If P20 to P27 are set up as analog inputs of the A/D converter, when a port is read while in the input mode, 0 is always returned, not the pin level.

**Figure 4-17. Format of Port Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W	
P1	P17 <sup>Note 3</sup>	P16	P15	P14 <sup>Note 3</sup>	P13	P12	P11 <sup>Note 3</sup>	P10	FFF01H	00H (output latch)	R/W	
P2	P27	P26 <sup>Note 3</sup>	P25 <sup>Note 3</sup>	P24 <sup>Note 3</sup>	P23	P22	P21	P20	FFF02H	00H (output latch)	R/W	
P3	0	0	P35 <sup>Note 3</sup>	P34 <sup>Note 3</sup>	P33	P32	P31	P30	FFF03H	00H (output latch)	R/W	
P4	0	0	0	0	0	0	0	P40	FFF04H	00H (output latch)	R/W	
P6	0	0	0	0	0	0	P61	P60	FFF06H	00H (output latch)	R/W	
P12	0	0	0	0	0	P122	P121	0	FFF0CH	Undefined	R/W <sup>Note 1</sup>	
P13	P137	0	0	0	0	0	0	0	FFF0DH	<b>Note 2</b>	R/W <sup>Note 1</sup>	
	Pmn	Output data control (in output mode)				Input data read (in input mode)						
	0	Output 0				Input low level						
	1	Output 1				Input high level						

- Notes**
1. P121, P122, and P137 are read-only.
  2. P137: Undefined
  3. These are not provided in 24-pin products.

**Remark** m = 1 to 4, 6, 12, 13; n = 0 to 7



### 4.3.3 Pull-up resistor option registers (PUxx)

These registers specify whether the on-chip pull-up resistors are to be used or not. On-chip pull-up resistors can be used in 1-bit units only for the bits set to input mode ( $PM_{mn} = 1$ ) for the pins to which the use of an on-chip pull-up resistor has been specified in these registers. On-chip pull-up resistors cannot be connected to bits set to output mode and bits used as alternate-function output pins and analog setting ( $PMC = 1$ ), regardless of the settings of these registers.

These registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H (Only PU4 is set to 01H).

**Figure 4-18. Format of Pull-up Resistor Option Register**

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PU1	PU17 <sup>Note</sup>	PU16	PU15	PU14 <sup>Note</sup>	PU13	PU12	PU11 <sup>Note</sup>	PU10	F0031H	00H	R/W
PU3	0	0	PU35 <sup>Note</sup>	PU34 <sup>Note</sup>	PU33	PU32	PU31	PU30	F0033H	00H	R/W
PU4	0	0	0	0	0	0	0	PU40	F0034H	01H	R/W

PU <sub>mn</sub>	P <sub>mn</sub> pin on-chip pull-up resistor selection (m = 0, 1, 4; n = 0 to 7)
0	On-chip pull-up resistor not connected
1	On-chip pull-up resistor connected

**Note** These are not provided in 24-pin products.

#### 4.3.4 Port mode control register 1 (PMC1) (24-pin products only)

This register sets the digital I/O/analog input in 1-bit units.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 4-19. Format of Port Mode Control Register

Symbol	7	6	5	4	3	2	1	0	Address	After reset	R/W
PMC1	1	1	1	1	1	1	1	PMC10	F0061H	FFH	R/W

PMC10	P10 pin digital I/O/analog input selection
0	Digital I/O (alternate function other than analog input)
1	Analog input

- Cautions**
1. Set the channel used for A/D conversion to the input mode by using port mode register 1 (PM1).
  2. Do not set the pin set by the PMC register as digital I/O by the analog input channel specification register (ADS).
  3. Be sure to set bits that are not mounted to their initial values.

<R>

### 4.3.5 A/D port configuration register (ADPC)

This register switches the ANI0/P20 to ANI7/P27 pins to digital I/O of port or analog input of A/D converter.

The ADPC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 4-20. Format of A/D Port Configuration Register (ADPC)**

Address: F0076H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADPC	ADPC7	ADPC6 <sup>Note</sup>	ADPC5 <sup>Note</sup>	ADPC4 <sup>Note</sup>	ADPC3	ADPC2	ADPC1	ADPC0
ADPCn	Analog input (A)/digital I/O (D) selection of P2n/ANIn							
0	Analog input (A) (default)							
1	Digital I/O (D)							

**Note** These are not provided in 24-pin products.

- Cautions**
1. Set the port to analog input by ADPC register to the input mode by using port mode register 2 (PM2).
  2. Do not set the pin set by the ADPC register as digital I/O by the analog input channel specification register (ADS).
  3. When using  $AV_{REFP}$  and  $AV_{REFM}$ , set ANI0 and ANI1 to analog input and set the port mode register to the input mode.

<R>

## 4.4 Port Function Operations

Port operations differ depending on whether the input or output mode is set, as shown below.

### 4.4.1 Writing to I/O port

#### (1) Output mode

A value is written to the output latch by a transfer instruction, and the output latch contents are output from the pin.

Once data is written to the output latch, it is retained until data is written to the output latch again.

The data of the output latch is cleared when a reset signal is generated.

#### (2) Input mode

A value is written to the output latch by a transfer instruction, but since the output buffer is off, the pin status does not change. Therefore, byte data can be written to the ports used for both input and output.

Once data is written to the output latch, it is retained until data is written to the output latch again.

The data of the output latch is cleared when a reset signal is generated.

### 4.4.2 Reading from I/O port

#### (1) Output mode

The output latch contents are read by a transfer instruction. The output latch contents do not change.

#### (2) Input mode

The pin status is read by a transfer instruction. The output latch contents do not change.

### 4.4.3 Operations on I/O port

#### (1) Output mode

An operation is performed on the output latch contents, and the result is written to the output latch. The output latch contents are output from the pins.

Once data is written to the output latch, it is retained until data is written to the output latch again.

The data of the output latch is cleared when a reset signal is generated.

#### (2) Input mode

The pin level is read and an operation is performed on its contents. The result of the operation is written to the output latch, but since the output buffer is off, the pin status does not change. Therefore, byte data can be written to the ports used for both input and output.

The data of the output latch is cleared when a reset signal is generated.

#### 4.5 Settings of Port Related Register When Using Alternate Function

To use the alternate function of a port pin, set the port mode register, and output latch as shown in **Table 4-12**.

**Caution** If the output function of an alternate function is assigned to a pin that is also used as an output pin, the output of the unused alternate function must be set to its initial state. See 4.6.2 for details about the applicable units and how to handle such pins.

**Table 4-12. Settings of Port Related Register When Using Alternate Function (1/2)**

Pin Name	Alternate Function		PM <sub>xx</sub>	P <sub>xx</sub>
	Name	I/O		
P12	TI03	Input	1	×
	TO03	Output	0	0
	INTP4	Input	1	×
	PCLBUZ0	Output	0	0
P13	TI00	Input	1	×
	TO00	Output	0	0
P15	PCLBUZ1	Output	0	0
P16	TI01	Input	1	×
	TO01	Output	0	0
	INTP5	Input	1	×
P20 <sup>Note</sup>	ANI0 <sup>Note</sup>	Input	1	×
	AV <sub>REFP</sub> <sup>Note</sup>	Input	1	×
P21 <sup>Note</sup>	ANI1 <sup>Note</sup>	Input	1	×
	AV <sub>REFM</sub> <sup>Note</sup>	Input	1	×
P22 <sup>Note</sup>	ANI2 <sup>Note</sup>	Input	1	×
	ANO0	Output	1	×
P23 <sup>Note</sup>	ANI3 <sup>Note</sup>	Input	1	×
	ANO1	Output	1	×
P24 to P27 <sup>Note</sup>	ANI4 to ANI7 <sup>Note</sup>	Input	1	×

- Remarks 1.** ×: don't care  
 PM<sub>xx</sub>: Port mode register  
 P<sub>xx</sub>: Port output latch
- 2.** The relationship between pins and their alternate functions shown in this table indicates the relationship when a 32-pin product is used. In other products, alternate functions might be assigned to different pins, but even in this case, the PM<sub>xx</sub> and P<sub>xx</sub> set in the same way.

(**Note** is described on next page.)

**Table 4-12. Settings of Port Related Register When Using Alternate Function (2/2)**

Pin Name	Alternate Function		PM <sub>xx</sub>	P <sub>xx</sub>
	Name	I/O		
P30	INTP2	Input	1	×
	TxD0	Output	0	1
	TOOLTxD	Output	0	1
	SO00	Output	0	1
P31	INTP1	Input	1	×
	RxD0	Input	1	×
	TOOLRxD	Input	1	×
	SI00	Input	1	×
P32	INTP3	Input	1	×
	SCK00	Input	1	×
		Output	0	1
P33	TI02	Input	1	×
	TO02	Output	0	0
	SSI00	Input	1	×
P40	TOOL0	I/O	×	×
P60	SCLA0	I/O	0	1
	SCLA1	I/O	0	1
P61	SDAA0	I/O	0	1
	SDAA1	I/O	0	1
P137	INTP0	Input	–	×

**Remarks 1.** ×: don't care

PM<sub>xx</sub>: Port mode register

P<sub>xx</sub>: Port output latch

2. The relationship between pins and their alternate functions shown in this table indicates the relationship when a 32-pin product is used. In other products, alternate functions might be assigned to different pins, but even in this case, the PM<sub>xx</sub> and P<sub>xx</sub> set in the same way.

**Note** The functions of the ANI0/P20 to ANI7/P27 pins can be selected by using the A/D port configuration register (ADPC), analog input channel specification register (ADS), and port mode register 2 (PM2).

**Table 4-13. Setting Functions of ANI0/P20, ANI1/P21, and ANI4/P24 to ANI7/P27 Pins**

ADPC Register	PM2 Register	ADS Register	ANI0/P20, ANI1/P21, ANI4/P24 to ANI7/P27 Pins
Digital I/O selection	Input mode	×	Digital input
	Output mode	×	Digital output
Analog input selection	Input mode	Selects ANI.	Analog input (to be converted)
		Does not select ANI.	Analog input (not to be converted)
	Output mode	Selects ANI.	Setting prohibited
		Does not select ANI.	

**Remark** ×: don't care

**Table 4-14. Setting Functions of P22/ANI2/ANO0 and P23/ANI3/ANO1 Pins**

ADPC Register	PM2 Register	DAM Register	ADS Register	P22/ANI2/ANO0, P23/ANI3/ANO1 Pins
Digital I/O selection	Input mode	–	–	Digital input
	Output mode	–	–	Digital output
Analog input selection	Input mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	Analog output
		Stops D/A conversion operation	Selects ANI.	Analog input (to be converted)
			Does not select ANI.	Analog input (not to be converted)
	Output mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	
Stops D/A conversion operation		Selects ANI.		
		Does not select ANI.		

The functions of the ANI16/P10 pin of 24-pin product can be selected by using the port mode control register 1 (PMC1), analog input channel specification register (ADS), and port mode register 1 (PM1).

## 4.6 Cautions When Using Port Function

### 4.6.1 Cautions on 1-bit manipulation instruction for port register n (Pn)

When a 1-bit manipulation instruction is executed on a port that provides both input and output functions, the output latch value of an input port that is not subject to manipulation may be written in addition to the targeted bit.

Therefore, it is recommended to rewrite the output latch when switching a port from input mode to output mode.

<Example> When P10 is an output port, P11 to P17 are input ports (all pin statuses are high level), and the port latch value of port 1 is 00H, if the output of output port P10 is changed from low level to high level via a 1-bit manipulation instruction, the output latch value of port 1 is FFH.

Explanation: The targets of writing to and reading from the Pn register of a port whose PMnm bit is 1 are the output latch and pin status, respectively.

A 1-bit manipulation instruction is executed in the following order in the R7F0C010.

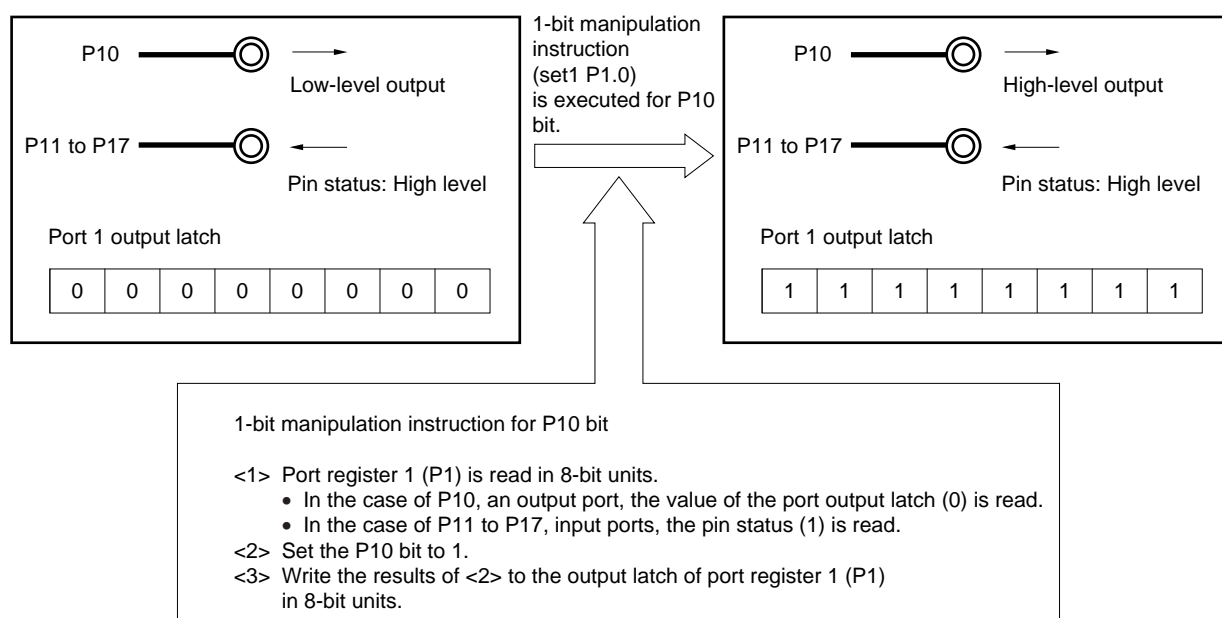
- <1> The Pn register is read in 8-bit units.
- <2> The targeted one bit is manipulated.
- <3> The Pn register is written in 8-bit units.

In step <1>, the output latch value (0) of P10, which is an output port, is read, while the pin statuses of P11 to P17, which are input ports, are read. If the pin statuses of P11 to P17 are high level at this time, the read value is FEH.

The value is changed to FFH by the manipulation in <2>.

FFH is written to the output latch by the manipulation in <3>.

**Figure 4-21. Bit Manipulation Instruction (P10)**





#### 4.6.2 Cautions on specifying the pin settings

If the output function of an alternate function is assigned to a pin that is also used as an output pin, the output of the unused alternate function must be set to its initial state so as to prevent conflicting outputs. For details about the alternate output function, see **4.5 Settings of Port Related Register When Using Alternate Function**.

No specific setting is required for input pins because the output function of their alternate functions is disabled (the buffer output is Hi-Z).

The following indicates the specific targets and the method of processing.

**Table 4-15. Handling of Unused Alternate Functions**

Affected Unit	Output or I/O Pins of Unused Alternate Functions	Handling of Unused Alternate Functions
Timer array units	TO0n	Make sure that bit n (TO0n) of timer output register 0 (TO0) and bit n (TOE0n) of timer output enable register 0 (TOE0) are set to their initial value (0).
Clock/buzzer output circuit	PCLBUZn	Make sure that bit 7 (PCLOEn) of clock output select register n (CKSn) is set to its initial value (0).
Serial array units	SCK00, SO00, TxD0	Make sure that bit 0 (SE0) of serial channel enable status register 0 (SE0), bit 0 (SO0) of serial output register 0 (SO0), and bit 0 (SOE0) of serial output enable register 0 (SOE0) are set to their initial value (1 for SO00 and 0 for others) <sup>Note</sup> .
IICA	SCLAn, SDAAn	Disable the IICA operation by setting bit 7 (IICEn) of the IICCTLn0 register to 0.

**Example:** P12/TI03/TO03/INTP4/PCLBUZ0 pin

##### (1) When the pin is used as PCLBUZ0 output

- P12: Specify the output mode by setting PM12 of port mode register 1 to 0.  
 TI03, INTP4: These are input pins, so this note does not apply.  
 TO03: This is an output pin, so set TO01 and TOE01 of timer array unit 0 to 0.

##### (2) When the pin is used as TO03 output

- P12: Specify the output mode by setting PM12 of port mode register 1 to 0.  
 PCLBUZ0: This is an output pin, so set PCLOE0 of clock/buzzer output to 0, respectively.  
 TI03: This is an input pin, so this note does not apply.

Like TxD0 when using the P30/INTP2/TxD0/TOOLTxD/SO00 pin as the SO00 output pin, changing the operation mode does not enable alternate functions assigned to pins on the same serial channel, and this note does not apply to such pins. (If the CSI function is specified (MD002 = MD001 = 0), the pin does not function as a UART pin, and therefore TxD0 output is invalid.)

Disabling the unused functions, including blocks that are only used for input or do not have I/O, is recommended to lower power consumption.

## CHAPTER 5 CLOCK GENERATOR

## 5.1 Functions of Clock Generator

The clock generator generates the clock to be supplied to the CPU and peripheral hardware.

The following three kinds of system clocks and clock oscillators are selectable.

## (1) Main system clock

## &lt;1&gt; X1 oscillator

This circuit oscillates a clock of  $f_x = 1$  to 20 MHz by connecting a resonator to X1 and X2.

Oscillation can be stopped by executing the STOP instruction or setting of the MSTOP bit (bit 7 of the clock operation status control register (CSC)).

## &lt;2&gt; High-speed on-chip oscillator

The frequency at which to oscillate can be selected from among  $f_{IH} = 32, 24, 16, 12, 8, 4,$  or 1 MHz (typ.) by using the option byte (000C2H). After a reset release, the CPU always starts operating with this high-speed on-chip oscillator clock. Oscillation can be stopped by executing the STOP instruction or setting the

<R> HIOSTOP bit (bit 0 of the CSC register).

An external main system clock ( $f_{EX} = 1$  to 20 MHz) can also be supplied from the EXCLK/X2/P122 pin. An external main system clock input can be disabled by executing the STOP instruction or setting of the MSTOP bit.

As the main system clock, a high-speed system clock (X1 clock or external main system clock) or high-speed on-chip oscillator clock can be selected by setting of the MCM0 bit (bit 4 of the system clock control register (CKC)).

The frequency specified by using an option byte can be changed by using the high-speed on-chip oscillator frequency select register (HOCODIV). For details about the frequency, see Figure 5 - 10 Format of High-speed onchip oscillator frequency select register (HOCODIV).

The frequencies that can be specified for the high-speed on-chip oscillator by using the option byte and the highspeed on-chip oscillator frequency select register (HOCODIV) are shown below.

Power Supply Voltage	Oscillation Frequency (MHz)						
	1	4	8	12	16	24	32
$2.7\text{ V} \leq \text{VDD} \leq 3.6\text{ V}$	√	√	√	√	√	√	√

<R>

## (2) Low-speed on-chip oscillator clock (Low-speed On-chip oscillator)

This circuit oscillates a clock of  $f_{IL} = 15$  kHz (TYP.).

The low-speed on-chip oscillator clock cannot be used as the CPU clock.

Only the following peripheral hardware runs on the low-speed on-chip oscillator clock.

- Watchdog timer

This clock operates when bit 4 (WDTON) of the option byte (000C0H) is set to 1.

However, when  $\text{WDTON} = 1$  and bit 0 (WDSTBYON) of the option byte (000C0H) is 0, oscillation of the low-speed on-chip oscillator stops if the HALT or STOP instruction is executed.

**Remark**  $f_x$ : X1 clock oscillation frequency  
 $f_{IH}$ : High-speed on-chip oscillator clock frequency  
 $f_{EX}$ : External main system clock frequency  
 $f_{IL}$ : Low-speed on-chip oscillator clock frequency

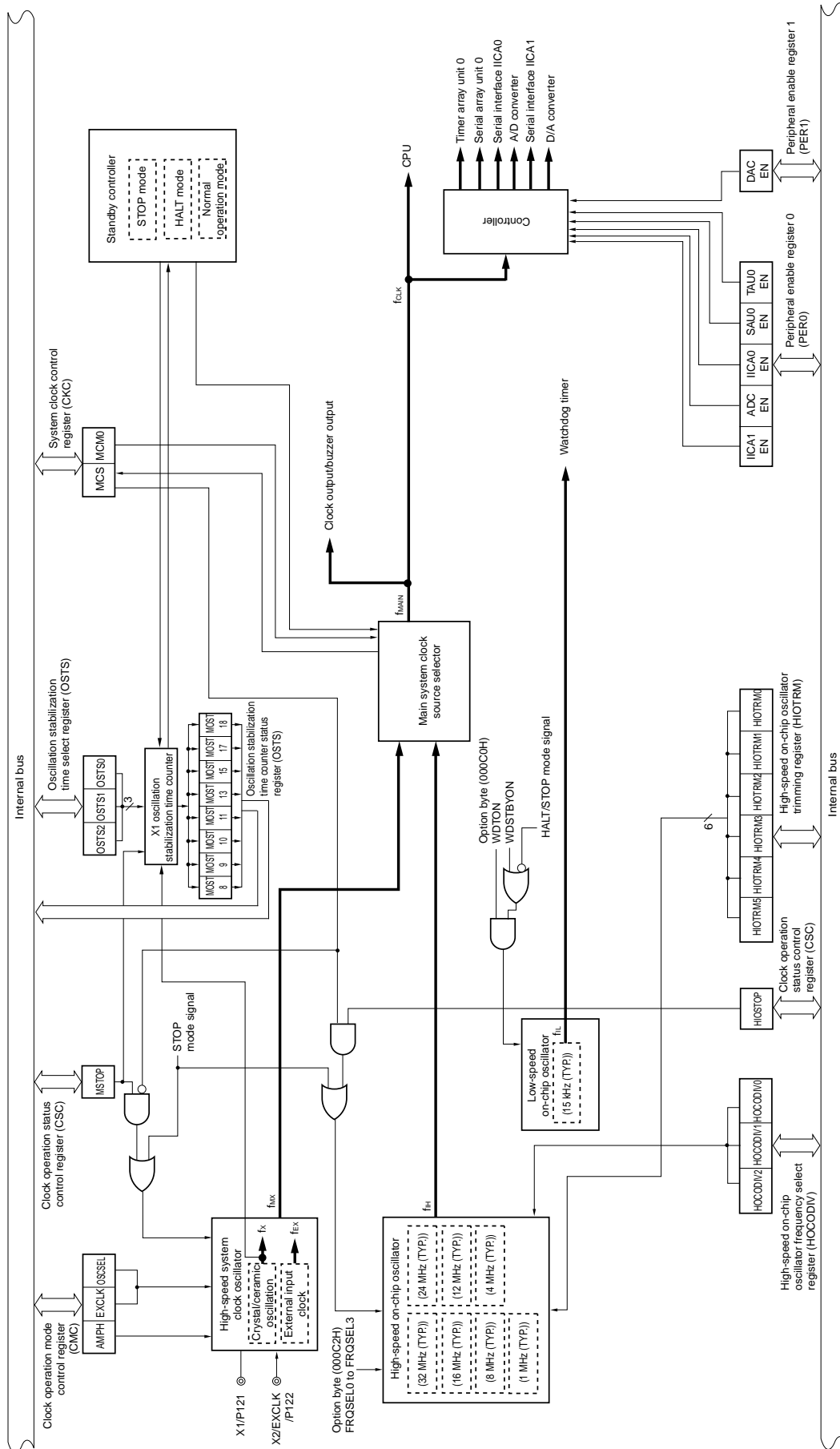
## 5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

**Table 5-1. Configuration of Clock Generator**

Item	Configuration
Control registers	Clock operation mode control register (CMC) System clock control register (CKC) Clock operation status control register (CSC) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) Peripheral enable registers 0, 1 (PER0, PER1) High-speed on-chip oscillator frequency select register (HOCODIV) High-speed on-chip oscillator trimming register (HIOTRM)
Oscillators	X1 oscillator High-speed on-chip oscillator Low-speed on-chip oscillator

Figure 5-1. Block Diagram of Clock Generator



(Remark is listed on the next page after next.)

<b>Remark</b>	<b>fx:</b>	X1 clock oscillation frequency
	<b>f<sub>1H</sub>:</b>	High-speed on-chip oscillator clock frequency
	<b>f<sub>EX</sub>:</b>	External main system clock frequency
	<b>f<sub>MX</sub>:</b>	High-speed system clock frequency
	<b>f<sub>MAIN</sub>:</b>	Main system clock frequency
	<b>f<sub>CLK</sub>:</b>	CPU/peripheral hardware clock frequency
	<b>f<sub>1L</sub>:</b>	Low-speed on-chip oscillator clock frequency

### 5.3 Registers Controlling Clock Generator

The following nine registers are used to control the clock generator.

- Clock operation mode control register (CMC)
- System clock control register (CKC)
- Clock operation status control register (CSC)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)
- Peripheral enable registers 0, 1 (PER0, PER1)
- High-speed on-chip oscillator frequency select register (HOCODIV)
- High-speed on-chip oscillator trimming register (HIOTRM)

<R>

**Caution** Which registers and bits are included depends on the product. Be sure to set registers and bits that are not mounted in a product to their initial values.

#### 5.3.1 Clock operation mode control register (CMC)

This register is used to set the operation mode of the X1/P121 and X2/EXCLK/P122 pins, and to select a gain of the oscillator.

The CMC register can be written only once by an 8-bit memory manipulation instruction after reset release. This register can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 5-2. Format of Clock Operation Mode Control Register (CMC)

Address: FFFA0H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL	0	0	0	0	0	AMPH
	EXCLK	OSCSEL	High-speed system clock pin operation mode		X1/P121 pin		X2/EXCLK/P122 pin	
	0	0	Input port mode		Input port			
	0	1	X1 oscillation mode		Crystal/ceramic resonator connection			
	1	0	Input port mode		Input port			
	1	1	External clock input mode		Input port		External clock input	
	AMPH	Control of X1 clock oscillation frequency						
	0	$1 \text{ MHz} \leq f_x \leq 10 \text{ MHz}$						
	1	$10 \text{ MHz} < f_x \leq 20 \text{ MHz}$						

- Cautions**
1. The CMC register can be written only once after reset release, by an 8-bit memory manipulation instruction. When using the CMC register with its initial value (00H), be sure to set the register to 00H after a reset ends in order to prevent malfunction due to a program loop. Such a malfunction becomes unrecoverable when a value other than 00H is mistakenly written..
  2. After reset release, set the CMC register before X1 oscillation is started as set by the clock operation status control register (CSC).
  3. Specify the settings for the AMPH bit while  $f_{IH}$  is selected as  $f_{CLK}$  after a reset ends (before  $f_{CLK}$  is switched to  $f_{MX}$ ).
  4. Although the maximum system clock frequency is 32 MHz, the maximum frequency of the X1 oscillator is 20 MHz.

**Remark**  $f_x$ : X1 clock frequency

### 5.3.2 System clock control register (CKC)

This register is used to select a CPU/peripheral hardware clock and a main system clock.  
The CKC register can be set by a 1-bit or 8-bit memory manipulation instruction.  
Reset signal generation sets this register to 00H.

**Figure 5-3. Format of System Clock Control Register (CKC)**

Address: FFFA4H After reset: 00H R/W<sup>Note</sup>

Symbol	7	6	<5>	<4>	3	2	1	0
CKC	0	0	MCS	MCM0	0	0	0	0

MCS	Status of Main system clock ( $f_{MAIN}$ )
0	High-speed on-chip oscillator clock ( $f_{IH}$ )
1	High-speed system clock ( $f_{MX}$ )

MCM0	Main system clock ( $f_{MAIN}$ ) operation control
0	Selects the high-speed on-chip oscillator clock ( $f_{IH}$ ) as the main system clock ( $f_{MAIN}$ )
1	Selects the high-speed system clock ( $f_{MX}$ ) as the main system clock ( $f_{MAIN}$ )

**Note** Bit 5 is read-only.

**Remark**  $f_{IH}$ : High-speed on-chip oscillator clock frequency  
 $f_{MX}$ : High-speed system clock frequency  
 $f_{MAIN}$ : Main system clock frequency

**Caution** Be sure to set bits 0 to 3, 6, and 7 to "0".

### 5.3.3 Clock operation status control register (CSC)

This register is used to control the operations of the high-speed system clock and high-speed on-chip oscillator clock (except the low-speed on-chip oscillator clock).

The CSC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to C0H.

**Figure 5-4. Format of Clock Operation Status Control Register (CSC)**

Address: FFFA1H After reset: C0H R/W

Symbol	<7>	6	5	4	3	2	1	<0>
CSC	MSTOP	0	0	0	0	0	0	HIOSTOP

MSTOP	High-speed system clock operation control		
	X1 oscillation mode	External clock input mode	Input port mode
0	X1 oscillator operating	External clock from EXCLK pin is valid	Input port
1	X1 oscillator stopped	External clock from EXCLK pin is invalid	

HIOSTOP	High-speed on-chip oscillator clock operation control
0	High-speed on-chip oscillator operating
1	High-speed on-chip oscillator stopped

- Cautions 1. After reset release, set the clock operation mode control register (CMC) before setting the CSC register.**
- Set the oscillation stabilization time select register (OSTS) before setting the MSTOP bit to 0 after releasing reset. Note that if the OSTS register is being used with its default settings, the OSTS register is not required to be set here.**
  - To start X1 oscillation as set by the MSTOP bit, check the oscillation stabilization time of the X1 clock by using the oscillation stabilization time counter status register (OSTC).**
  - Do not stop the clock selected for the CPU peripheral hardware clock ( $f_{CLK}$ ) with the OSC register.**
  - The setting of the flags of the register to stop clock oscillation (invalidate the external clock input) and the condition before clock oscillation is to be stopped are as Table 5-2.**

**Table 5-2. Stopping Clock Method**

Clock	Condition Before Stopping Clock (Invalidating External Clock Input)	Setting of CSC Register Flags
X1 clock	CPU and peripheral hardware clocks operate with a clock other than the high-speed system clock. (MCS = 0)	MSTOP = 1
External main system clock		
High-speed on-chip oscillator clock	CPU and peripheral hardware clocks operate with a clock other than the high-speed on-chip oscillator clock. (MCS = 1)	HIOSTOP = 1



### 5.3.4 Oscillation stabilization time counter status register (OSTC)

This is the register that indicates the count status of the X1 clock oscillation stabilization time counter.

The X1 clock oscillation stabilization time can be checked in the following case,

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock is being used as the CPU clock.
- If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.

The OSTC register can be read by a 1-bit or 8-bit memory manipulation instruction.

When reset signal is generated, the STOP instruction and MSTOP (bit 7 of clock operation status control register (CSC)) = 1 clear the OSTC register to 00H.

**Remark** The oscillation stabilization time counter starts counting in the following cases.

- When oscillation of the X1 clock starts (EXCLK = 0 → MSTOP = 0)
- When the STOP mode is released

Figure 5-5. Format of Oscillation Stabilization Time Counter Status Register (OSTC)

Address: FFFA2H After reset: 00H R

Symbol	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18

MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18	Oscillation stabilization time status		
									fx = 10 MHz	fx = 20 MHz
0	0	0	0	0	0	0	0	2 <sup>9</sup> /fx max.	25.6 μs max.	12.8 μs max.
1	0	0	0	0	0	0	0	2 <sup>9</sup> /fx min.	25.6 μs min.	12.8 μs min.
1	1	0	0	0	0	0	0	2 <sup>9</sup> /fx min.	51.2 μs min.	25.6 μs min.
1	1	1	0	0	0	0	0	2 <sup>10</sup> /fx min.	102 μs min.	51.2 μs min.
1	1	1	1	0	0	0	0	2 <sup>11</sup> /fx min.	204 μs min.	102 μs min.
1	1	1	1	1	0	0	0	2 <sup>13</sup> /fx min.	819 μs min.	409 μs min.
1	1	1	1	1	1	0	0	2 <sup>15</sup> /fx min.	3.27 ms min.	1.63 ms min.
1	1	1	1	1	1	1	0	2 <sup>17</sup> /fx min.	13.1 ms min.	6.55 ms min.
1	1	1	1	1	1	1	1	2 <sup>18</sup> /fx min.	26.2 ms min.	13.1 ms min.

<R>

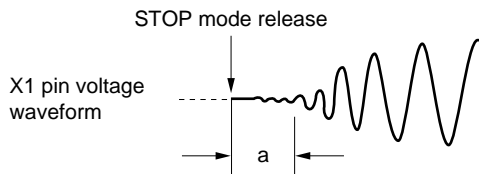
**Cautions 1.** After the above time has elapsed, the bits are set to 1 in order from the MOST8 bit and remain 1.

**2.** The oscillation stabilization time counter counts up to the oscillation stabilization time set by the oscillation stabilization time select register (OSTS).

In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the count value which is to be checked by the OSTC register.

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock is being used as the CPU clock.
- If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.  
(Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is released.)

**3.** The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).



**Remark** fx: X1 clock oscillation frequency

### 5.3.5 Oscillation stabilization time select register (OSTS)

This register is used to select the X1 clock oscillation stabilization wait time when the STOP mode is released.

When the X1 clock is selected as the CPU clock, the operation automatically waits for the time set using the OSTS register after the STOP mode is released.

When the CPU clock is switched from the high-speed on-chip oscillator clock to the X1 clock, or when STOP mode is entered while the high-speed on-chip oscillator is used as the CPU clock and the X1 clock is also oscillating, and then STOP mode is released. The oscillation stabilization time can be checked up to the time set using the OSTC register.

The OSTS register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets the OSTS register to 07H.

Figure 5-6. Format of Oscillation Stabilization Time Select Register (OSTS)

Address: FFFA3H After reset: 07H R/W

Symbol	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0	Oscillation stabilization time selection	
			$f_x = 10 \text{ MHz}$	$f_x = 20 \text{ MHz}$
0	0	0	$2^8/f_x$	25.6 $\mu\text{s}$
0	0	1	$2^9/f_x$	51.2 $\mu\text{s}$
0	1	0	$2^{10}/f_x$	102 $\mu\text{s}$
0	1	1	$2^{11}/f_x$	204 $\mu\text{s}$
1	0	0	$2^{13}/f_x$	819 $\mu\text{s}$
1	0	1	$2^{15}/f_x$	3.27 ms
1	1	0	$2^{17}/f_x$	13.1 ms
1	1	1	$2^{18}/f_x$	26.2 ms

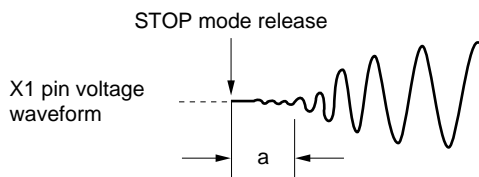
**Cautions 1. Change the setting of the OSTS register before setting the MSTOP bit of the clock operation status control register (CSC) to 0.**

**2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the OSTS register.**

In the following cases, set the oscillation stabilization time of the OSTS register to the value greater than the count value which is to be checked by the OSTC register after the oscillation starts.

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock is being used as the CPU clock.
- If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating. (Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after the STOP mode is released.)

**3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts (“a” below).**



**Remark**  $f_x$ : X1 clock oscillation frequency

### 5.3.6 Peripheral enable registers 0, 1 (PER0, PER1)

These registers are used to enable or disable supplying the clock to the peripheral hardware. Clock supply to the hardware that is not used is also stopped so as to decrease the power consumption and noise.

To use the peripheral functions below, which are controlled by this register, set (1) the bit corresponding to each function before specifying the initial settings of the peripheral functions.

- Serial interface IICA1
- A/D converter
- D/A converter
- Serial interface IICA0
- Serial array unit 0
- Timer array unit 0

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (1/2)**

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

IICA1EN	Control of serial interface IICA1 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the serial interface IICA1 cannot be written.</li> <li>• The serial interface IICA1 is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the serial interface IICA1 can be read and written.</li> </ul>

ADCEN	Control of A/D converter input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the A/D converter cannot be written.</li> <li>• The A/D converter is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the A/D converter can be read and written.</li> </ul>

IICA0EN	Control of serial interface IICA0 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the serial interface IICA0 cannot be written.</li> <li>• The serial interface IICA0 is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the serial interface IICA0 can be read and written.</li> </ul>

**Caution** Be sure to clear bits 1, 3 and 7 to "0".

Figure 5-7. Format of Peripheral Enable Register 0 (PER0) (2/2)

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

SAU0EN	Control of serial array unit 0 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>SFR used by the serial array unit 0 cannot be written.</li> <li>The serial array unit 0 is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>SFR used by the serial array unit 0 can be read and written.</li> </ul>

TAU0EN	Control of timer array unit 0 input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>SFR used by timer array unit 0 cannot be written.</li> <li>Timer array unit 0 is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>SFR used by timer array unit 0 can be read and written.</li> </ul>

**Caution** Be sure to clear bits 1, 3 and 7 to "0".

Figure 5-8. Format of Peripheral Enable Register 1 (PER1)

Address: F007AH After reset: 00H R/W

Symbol	<7>	6	5	4	3	1	1	0
PER1	DACEN	0	0	0	0	0	0	0

DACEN	Control of D/A converter input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>SFR used by the D/A converter cannot be written.</li> <li>The D/A converter is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>SFR used by the D/A converter can be read and written.</li> </ul>

**Caution** Be sure to clear bits 0 to 6 to "0".

&lt;R&gt;

### 5.3.7 High-speed on-chip oscillator frequency select register (HOCODIV)

The frequency of the high-speed on-chip oscillator which is set by an option byte (000C2H) can be changed by using high-speed on-chip oscillator frequency select register (HOCODIV). However, the selectable frequency depends on the FRQSEL3 bit of the option byte (000C2H).

The HOCODIV register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H).

**Figure 5-9. Format of High-speed On-chip Oscillator Frequency Select Register (HOCODIV)**

Address: F00A8H After reset: the value set by FRQSEL2 to FRQSEL0 of the option byte (000C2H) R/W

Symbol	7	6	5	4	3	2	1	0
HOCODIV	0	0	0	0	0	HOCODIV2	HOCODIV1	HOCODIV0

HOCODIV2	HOCODIV1	HOCODIV0	High-Speed On-Chip Oscillator Clock Frequency	
			FRQSEL3 Bit is 0	FRQSEL3 Bit of is 1
0	0	0	24 MHz	32 MHz
0	0	1	12 MHz	16 MHz
0	1	0	6 MHz	8 MHz
0	1	1	3 MHz	4 MHz
1	0	0	Setting prohibited	2 MHz
1	0	1	Setting prohibited	1 MHz
Other than above			Setting prohibited	

**Cautions** 1. Set the HOCODIV register within the operable voltage range of the flash operation mode set in the option byte (000C2H) before and after the frequency change.

Option Byte (000C2H) Value		Flash Operation Mode	Operating Frequency Range	Operating Voltage Range
CMODE1	CMODE0			
1	0	LS (low-speed main) mode	1 to 8 MHz	2.7 to 3.6 V
1	1	HS (high-speed main) mode	1 to 32 MHz	2.7 to 3.6 V

- Set the HOCODIV register with the high-speed on-chip oscillator clock ( $f_{IH}$ ) selected as the CPU/peripheral hardware clock ( $f_{CLK}$ ).
- After the frequency is changed with the HOCODIV register, the frequency is switched after the following transition time has elapsed.
  - Operation for up to three clocks at the pre-change frequency
  - CPU/peripheral hardware clock wait at the post-change frequency for up to three clocks

### 5.3.8 High-speed on-chip oscillator trimming register (HIOTRM)

This register is used to adjust the accuracy of the high-speed on-chip oscillator.

With self-measurement of the high-speed on-chip oscillator frequency via a timer using high-accuracy external clock input (timer array unit), and so on, the accuracy can be adjusted.

The HIOTRM register can be set by an 8-bit memory manipulation instruction.

**Caution** The frequency will vary if the temperature and V<sub>DD</sub> pin voltage change after accuracy adjustment. When the temperature and V<sub>DD</sub> voltage change, accuracy adjustment must be executed regularly or before the frequency accuracy is required.

Figure 5-10. Format of High-Speed On-Chip Oscillator Trimming Register (HIOTRM)

Address: F00A0H After reset: **Note** R/W

Symbol	7	6	5	4	3	2	1	0
HIOTRM	0	0	HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0

HIOTRM5	HIOTRM4	HIOTRM3	HIOTRM2	HIOTRM1	HIOTRM0	High-speed on-chip oscillator
0	0	0	0	0	0	Minimum speed
0	0	0	0	0	1	↑ ↓
0	0	0	0	1	0	
0	0	0	0	1	1	
0	0	0	1	0	0	
• • •						
1	1	1	1	1	0	↓
1	1	1	1	1	1	

<R> **Note** The value after reset is adjusted at shipment.

**Remark 1.** The HIOTRM register can be used to adjust the high-speed on-chip oscillator clock to an accuracy within about 0.05%.

**Remark 2.** For the usage example of the HIOTRM register, see the application note for RL78 MCU Series High-speed On-chip Oscillator (HOCO) Clock Frequency Correction (R01AN0464).



## 5.4 System Clock Oscillator

### 5.4.1 X1 oscillator

The X1 oscillator oscillates with a crystal resonator or ceramic resonator (1 to 20 MHz) connected to the X1 and X2 pins.

An external clock can also be input. In this case, input the clock signal to the EXCLK pin.

To use the X1 oscillator, set bits 7 and 6 (EXCLK, OSCSEL) of the clock operation mode control register (CMC) as follows.

- Crystal or ceramic oscillation: EXCLK, OSCSEL = 0, 1
- External clock input: EXCLK, OSCSEL = 1, 1

When the X1 oscillator is not used, set the input port mode (EXCLK, OSCSEL = 0, 0).

When the pins are not used as input port pins, either, see **Table 2-4 Connection of Unused Pins**.

Figure 5-11 shows an example of the external circuit of the X1 oscillator.

**Figure 5-11. Example of External Circuit of X1 Oscillator**

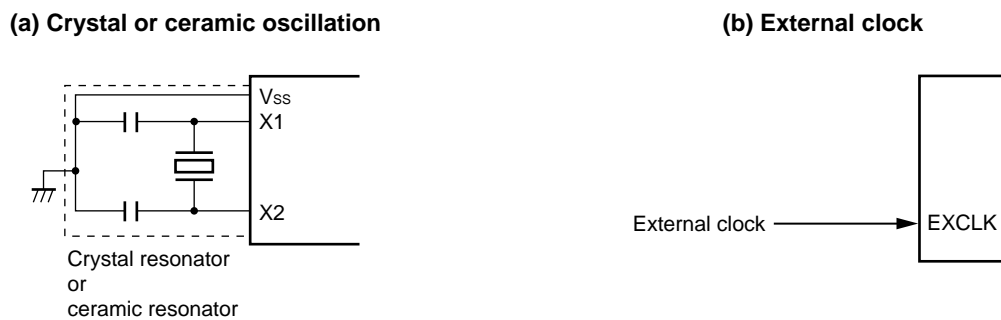
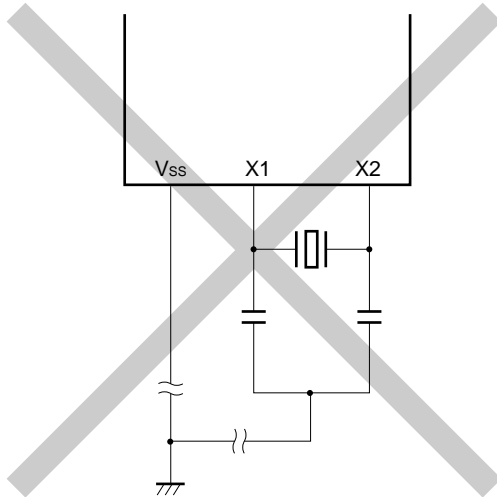


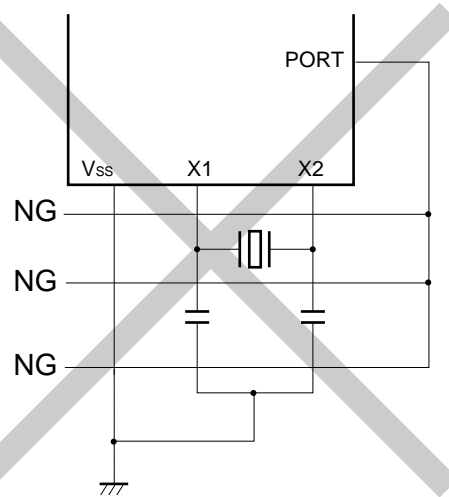
Figure 5-12 shows examples of incorrect resonator connection.

**Figure 5-12. Examples of Incorrect Resonator Connection (1/2)**

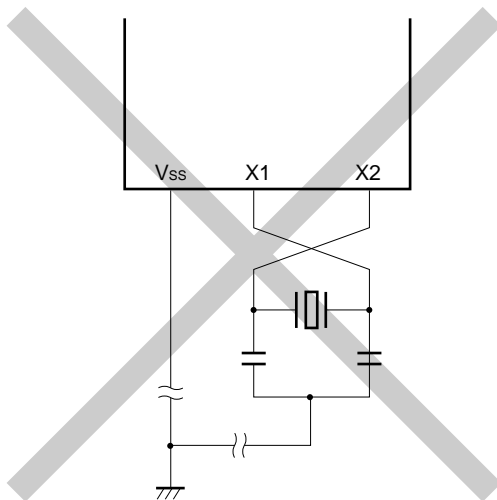
**(a) Too long wiring**



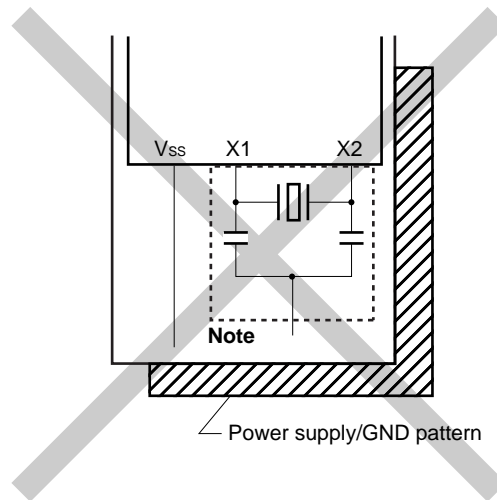
**(b) Crossed signal line**



**(c) The X1 and X2 signal line wires cross.**



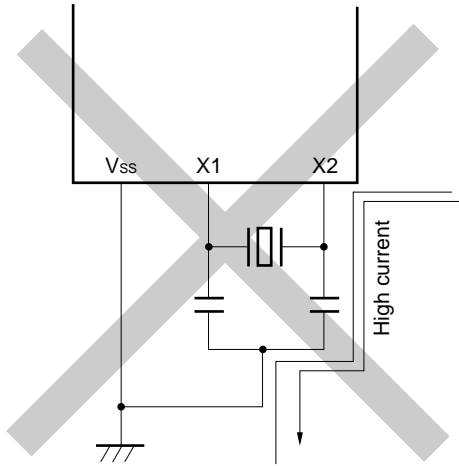
**(d) A power supply/GND pattern exists under the X1 and X2 wires.**



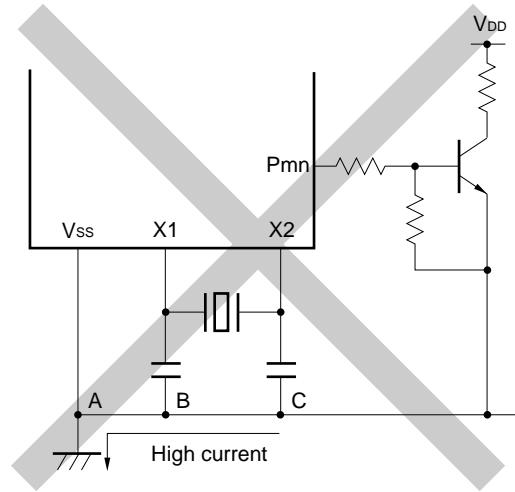
**Note** Do not place a power supply/GND pattern under the wiring section (section indicated by a broken line in the figure) of the X1 and X2 pins and the resonators in a multi-layer board or double-sided board. Do not configure a layout that will cause capacitance elements and affect the oscillation characteristics.

Figure 5-12. Examples of Incorrect Resonator Connection (2/2)

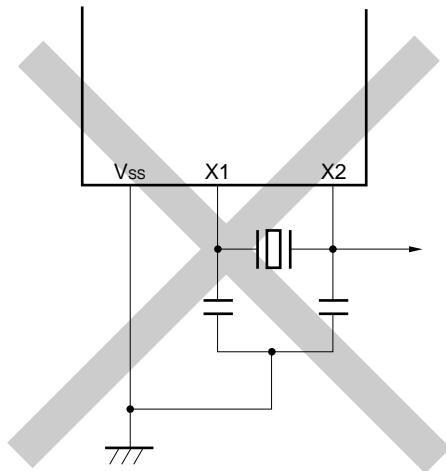
(e) Wiring near high alternating current



(f) Current flowing through ground line of oscillator (potential at points A, B, and C fluctuates)



(g) Signals are fetched



&lt;R&gt;

**5.4.2 High-speed on-chip oscillator**

The high-speed on-chip oscillator is incorporated in the R7F0C010. The frequency can be selected from among 32, 24, 16, 12, 8, 4, or 1 MHz by using the option byte (000C2H). Oscillation can be controlled by bit 0 (HIOSTOP) of the clock operation status control register (CSC). The high-speed on-chip oscillator automatically starts oscillating after reset release.

**5.4.3 Low-speed on-chip oscillator**

The low-speed on-chip oscillator is incorporated in the R7F0C010.

The low-speed on-chip oscillator clock is used only as the watchdog timer. The low-speed on-chip oscillator clock cannot be used as the CPU clock.

This clock operates when bit 4 (WDTON) of the option byte (000C0H) is set to 1.

&lt;R&gt;

If the watchdog timer operates in HALT, STOP or SNOOZE mode at WDSTBYON = 0, oscillation of the low-speed on-chip oscillator is stopped. While the watchdog timer operates, the low-speed on-chip oscillator clock does not stop even if the program freezes.

## 5.5 Clock Generator Operation

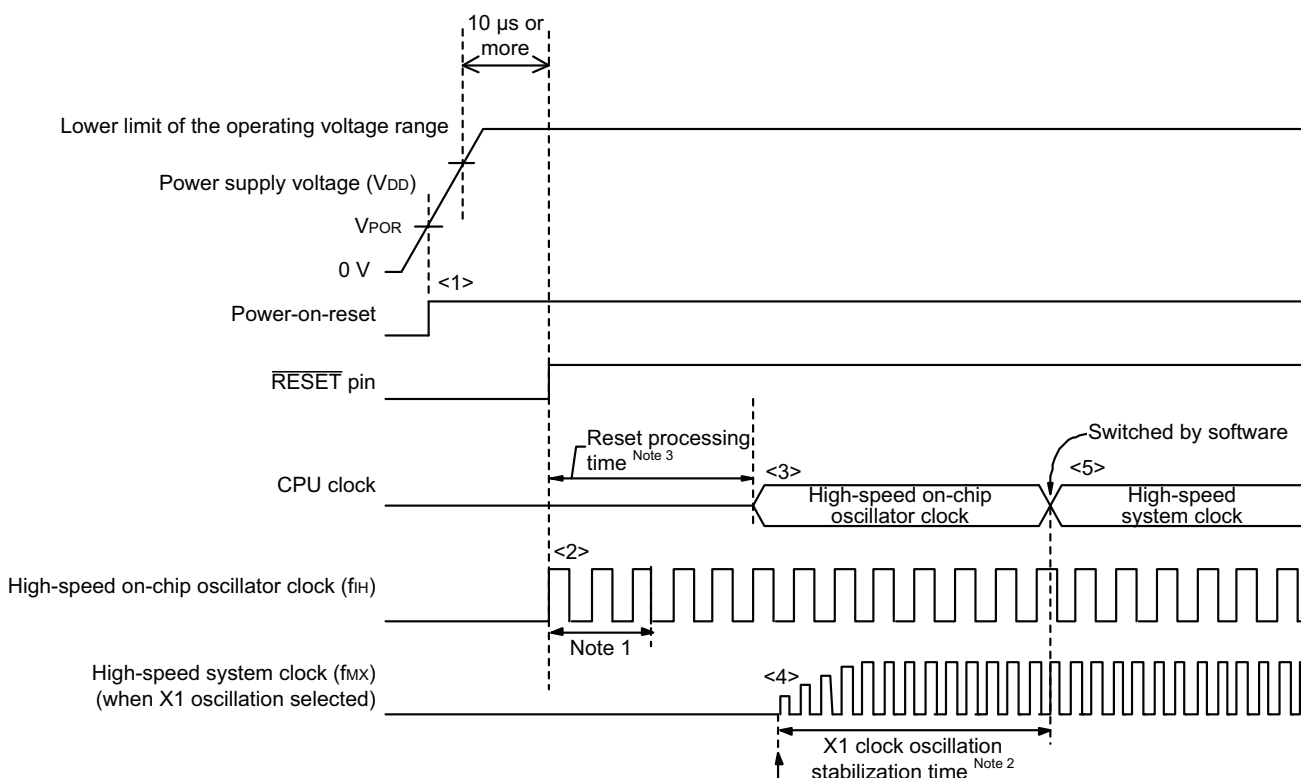
The clock generator generates the following clocks and controls the operation modes of the CPU, such as standby mode (see **Figure 5-1**).

- Main system clock  $f_{\text{MAIN}}$ 
  - High-speed system clock  $f_{\text{MX}}$ 
    - X1 clock  $f_x$
    - External main system clock  $f_{\text{EX}}$
  - High-speed on-chip oscillator clock  $f_{\text{IH}}$
- Low-speed on-chip oscillator clock  $f_{\text{IL}}$
- CPU/peripheral hardware clock  $f_{\text{CLK}}$

The CPU starts operation when the high-speed on-chip oscillator starts outputting after a reset release in the R7F0C010.

When the power supply voltage is turned on, the clock generator operation is shown in Figure 5-13.

Figure 5-13. Clock Generator Operation When Power Supply Voltage Is Turned On



- <1> When the power is turned on, an internal reset signal is generated by the power-on-reset (POR) circuit. Note that the reset state is maintained after a reset by the voltage detection circuit or an external reset until the voltage reaches the range of operating voltage described in **27.4 AC Characteristics (the above figure is an example when the external reset is in use)**.
- <2> When the power supply voltage exceeds 1.51 V (TYP.), the reset is released and the high-speed on-chip oscillator automatically starts oscillation.
- <3> The CPU starts operation on the high-speed on-chip oscillator clock after waiting for the voltage to stabilize and a reset processing have been performed after reset release.
- <4> Set the start of oscillation of the X1 clock via software (see **5.6.2 Example of setting X1 oscillation clock**).
- <5> When switching the CPU clock to the X1 or XT1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see **5.6.2 Example of setting X1 oscillation clock**).

- Notes**
1. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.
  2. When releasing a reset, confirm the oscillation stabilization time for the X1 clock using the oscillation stabilization time counter status register (OSTC).
  3. For the reset processing time, see **CHAPTER 18 POWER-ON-RESET CIRCUIT**.

**Caution** It is not necessary to wait for the oscillation stabilization time when an external clock input from the EXCLK pin is used.

## 5.6 Controlling Clock

### 5.6.1 Example of setting high-speed on-chip oscillator

After a reset release, the CPU/peripheral hardware clock ( $f_{CLK}$ ) always starts operating with the high-speed on-chip oscillator clock. The frequency of the high-speed on-chip oscillator can be selected from 32, 24, 16, 12, 8, 4, and 1 MHz by using FRQSEL0 to FRQSEL3 of the option byte (000C2H).

[Option byte setting]

Address: 000C2H

Option byte (000C2H)	7	6	5	4	3	2	1	0
	CMODE1	CMODE0			FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0
	0/1	0/1	1	0	0/1	0/1	0/1	0/1

CMODE1	CMODE0	Setting of flash operation mode	
1	0	LS (low speed main) mode	$V_{DD} = 2.7\text{ V to }3.6\text{ V @ }1\text{ MHz to }8\text{ MHz}$
1	1	HS (high speed main) mode	$V_{DD} = 2.7\text{ V to }3.6\text{ V @ }1\text{ MHz to }32\text{ MHz}$
Other than above		Setting prohibited	

FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of the high-speed on-chip oscillator
1	0	0	0	32 MHz
0	0	0	0	24 MHz
1	0	0	1	16 MHz
0	0	0	1	12 MHz
1	0	1	0	8 MHz
1	0	1	1	4 MHz
1	1	0	1	1 MHz
Other than above				Setting prohibited

[High-speed on-chip oscillator frequency select register (HOCODIV) setting]

Address: F00A8H

HOCODIV	7	6	5	4	3	2	1	0
	0	0	0	0	0	HOCODIV2	HOCODIV1	HOCODIV0

HOCODIV2	HOCODIV1	HOCODIV0	Selection of high-speed on-chip oscillator clock frequency	
			FRQSEL3 Bit is 0	FRQSEL3 Bit of is 1
0	0	0	24 MHz	32 MHz
0	0	1	12 MHz	16 MHz
0	1	0	6 MHz	8 MHz
0	1	1	3 MHz	4 MHz
1	0	0	Setting prohibited	2 MHz
1	0	1	Setting prohibited	1 MHz
Other than above			Setting prohibited	

### 5.6.2 Example of setting X1 oscillation clock

After a reset release, the CPU/peripheral hardware clock ( $f_{CLK}$ ) always starts operating with the high-speed on-chip oscillator clock. To subsequently change the clock to the X1 oscillation clock, set the oscillator and start oscillation by using the oscillation stabilization time select register (OSTS) and clock operation mode control register (CMC) and clock operation status control register (CSC) and wait for oscillation to stabilize by using the oscillation stabilization time counter status register (OSTC). After the oscillation stabilizes, set the X1 oscillation clock to  $f_{CLK}$  by using the system clock control register (CKC).

[Register settings] Set the register in the order of <1> to <5> below.

<1> Set (1) the OSCSEL bit of the CMC register, except for the cases where  $f_x > 10$  MHz, in such cases set (1) the AMPH bit, to operate the X1 oscillator.

	7	6	5	4	3	2	1	0
CMC	EXCLK	OSCSEL						AMPH <sup>Note</sup>
	0	1	0	0	0	0	0	0/1

**Note** Set AMPH bit to 0 when X1 clock is 10 MHz or below.

<2> Using the OSTS register, select the oscillation stabilization time of the X1 oscillator at releasing of the STOP mode.

Example: Setting values when a wait of at least 102.4  $\mu$ s is set based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTS						OSTS2	OSTS1	OSTS0
	0	0	0	0	0	0	1	0

<3> Clear (0) the MSTOP bit of the CSC register to start oscillating the X1 oscillator.

	7	6	5	4	3	2	1	0
CSC	MSTOP							HIOSTOP
	0	1	0	0	0	0	0	0

<4> Use the OSTC register to wait for oscillation of the X1 oscillator to stabilize.

Example: Wait until the bits reach the following values when a wait of at least 102.4  $\mu$ s is set based on a 10 MHz resonator.

	7	6	5	4	3	2	1	0
OSTC	MOST8	MOST9	MOST10	MOST11	MOST13	MOST15	MOST17	MOST18
	1	1	1	0	0	0	0	0

<5> Use the MCM0 bit of the CKC register to specify the X1 oscillation clock as the CPU/peripheral hardware clock.

	7	6	5	4	3	2	1	0
CKC			MCS	MCM0				
	0	0	0	1	0	0	0	0



5.6.3 CPU clock status transition diagram

Figure 5-14 shows the CPU clock status transition diagram of this product.

Figure 5-14. CPU Clock Status Transition Diagram

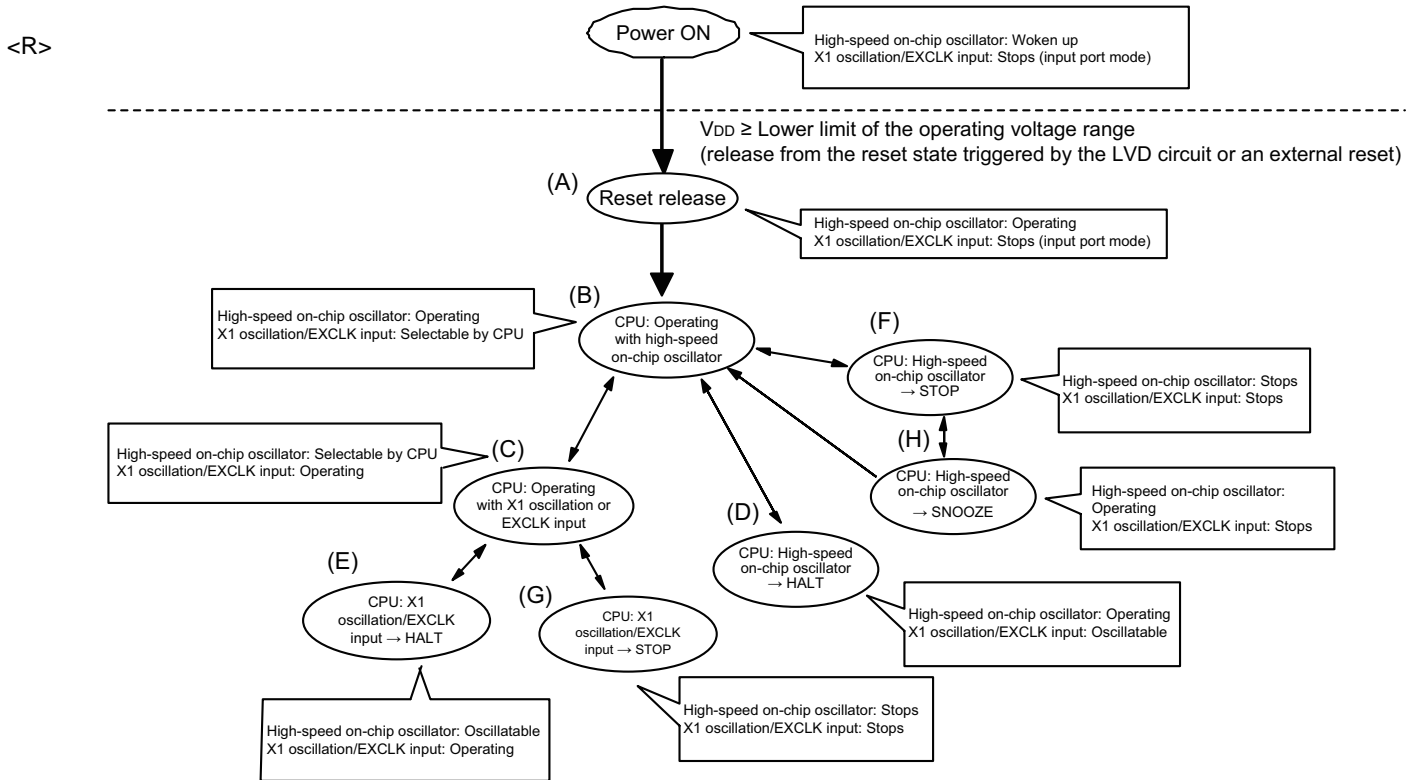


Table 5-3 shows transition of the CPU clock and examples of setting the SFR registers.

**Table 5-3. CPU Clock Transition and SFR Register Setting Examples (1/3)**

**(1) CPU operating with high-speed on-chip oscillator clock (B) after reset release (A)**

Status Transition	SFR Register Setting
(A) → (B)	SFR registers do not have to be set (default status after reset release).

**(2) CPU operating with high-speed system clock (C) after reset release (A)**

(The CPU operates with the high-speed on-chip oscillator clock immediately after a reset release (B).)

(Setting sequence of SFR registers) →

Setting Flag of SFR Register Status Transition	CMC Register <sup>Note</sup>			OSTS Register	CSC Register	OSTC Register	CKC Register
	EXCLK	OSCSEL	AMPH		MSTOP		MCM0
(A) → (B) → (C) (X1 clock: $1 \text{ MHz} \leq f_x \leq 10 \text{ MHz}$ )	0	1	0	<b>Note 2</b>	0	Must be checked	1
(A) → (B) → (C) (X1 clock: $10 \text{ MHz} < f_x \leq 20 \text{ MHz}$ )	0	1	1	<b>Note 2</b>	0	Must be checked	1
(A) → (B) → (C) (external main clock)	1	1	×	<b>Note 2</b>	0	Must not be checked	1

**Notes 1.** The clock operation mode control register (CMC) can be written only once by an 8-bit memory manipulation instruction after reset release.

**2.** Set the oscillation stabilization time as follows.

- Desired the oscillation stabilization time counter status register (OSTC) oscillation stabilization time  $\leq$  Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

**Caution** Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 27 ELECTRICAL SPECIFICATIONS).

**Remarks 1.** x: don't care

**2.** (A) to (H) in Table 5-3 correspond to (A) to (H) in Figure 5-5.

<R>

Table 5-3. CPU Clock Transition and SFR Register Setting Examples (2/3)

(3) CPU clock changing from high-speed on-chip oscillator clock (B) to high-speed system clock (C)

(Setting sequence of SFR registers) →

Setting Flag of SFR Register Status Transition	CMC Register <sup>Note 1</sup>			OSTS Register	CSC Register	OSTC Register	CKC Register
	EXCLK	OSCSEL	AMPH		MSTOP		
(B) → (C) (X1 clock: 1 MHz ≤ fX ≤ 10 MHz)	0	1	0	<b>Note 2</b>	0	Must be checked	1
(B) → (C) (X1 clock: 10 MHz < fX ≤ 20 MHz)	0	1	1	<b>Note 2</b>	0	Must be checked	1
(B) → (C) (external main clock)	1	1	×	<b>Note 2</b>	0	Must not be checked	1

Unnecessary if these registers are already set
 Unnecessary if the CPU is operating with the high-speed system clock

- Notes**
1. The clock operation mode control register (CMC) can be changed only once after reset release. This setting is not necessary if it has already been set.
  2. Set the oscillation stabilization time as follows.
    - Desired the oscillation stabilization time counter status register (OSTC) oscillation stabilization time ≤ Oscillation stabilization time set by the oscillation stabilization time select register (OSTS)

(4) CPU clock changing from high-speed system clock (C) to high-speed on-chip oscillator clock (B)

(Setting sequence of SFR registers) →

Setting Flag of SFR Register Status Transition	CSC Register	Oscillation accuracy stabilization time	CKC Register
	HIOSTOP		MCM0
(C) → (B)	0	18 μs to 65 μs	0

Unnecessary if the CPU is operating with the high-speed on-chip oscillator clock

- Remarks**
1. (A) to (H) in Tables 5 - 3 correspond to (A) to (H) in Figure 5 - 14
  2. The oscillation accuracy stabilization time changes according to the temperature conditions and the STOP mode period.

Table 5-3. CPU Clock Transition and SFR Register Setting Examples (3/3)

- (5) • HALT mode (D) set while CPU is operating with high-speed on-chip oscillator clock (B)  
 • HALT mode (E) set while CPU is operating with high-speed system clock (C)

Status Transition	Setting
(B) → (D) (C) → (E)	Executing HALT instruction

- (6) • STOP mode (F) set while CPU is operating with high-speed on-chip oscillator clock (B)  
 • STOP mode (G) set while CPU is operating with high-speed system clock (C)

(Setting sequence) →

Status Transition		Setting		
(B) → (F)		Stopping peripheral functions that cannot operate in STOP mode	–	Executing STOP instruction
(C) → (G)	In X1 oscillation		Sets the OSTS register	
	External main system clock		–	

(7) CPU changing from STOP mode (F) to SNOOZE mode (H)

For details about the setting for switching from the STOP mode to the SNOOZE mode, see 9.8 SNOOZE Mode Function, 11.5.7 SNOOZE mode function (CSI00) and 11.7.3 SNOOZE mode function.

**Remark** (A) to (H) in Table 5-3 correspond to (A) to (H) in Figure 5-14.

#### 5.6.4 Condition before changing CPU clock and processing after changing CPU clock

Condition before changing the CPU clock and processing after changing the CPU clock are shown below.

**Table 5-4. Changing CPU Clock**

CPU Clock		Condition Before Change	Processing After Change
Before Change	After Change		
High-speed on-chip oscillator clock	X1 clock	Stabilization of X1 oscillation • OSCSEL = 1, EXCLK = 0, MSTOP = 0 • After elapse of oscillation stabilization time	Operating current can be reduced by stopping high-speed on-chip oscillator (HIOSTOP = 1).
	External main system clock	Enabling input of external clock from the EXCLK pin • OSCSEL = 1, EXCLK = 1, MSTOP = 0	
<R> X1 clock	High-speed on-chip oscillator clock	Enabling oscillation of high-speed on-chip oscillator • HIOSTOP = 0 • After elapse of oscillation stabilization time	X1 oscillation can be stopped (MSTOP = 1).
	External main system clock	Transition not possible	–
<R> External main system clock	High-speed on-chip oscillator clock	Oscillation of high-speed on-chip oscillator • HIOSTOP = 0 • After elapse of oscillation stabilization time	External main system clock input can be disabled (MSTOP = 1).
	X1 clock	Transition not possible	–

### 5.6.5 Time required for switchover of CPU clock and system clock

By setting bit 4 (MCM0) of the system clock control register (CKC), the main system clock can be switched (between the high-speed on-chip oscillator clock and the high-speed system clock).

The actual switchover operation is not performed immediately after rewriting to the CKC register; operation continues on the pre-switchover clock for several clocks (see **Table 5-5** and **Table 5-6**).

Whether the main system clock is operating on the high-speed system clock or high-speed on-chip oscillator clock can be ascertained using bit 5 (MCS) of the CKC register.

When the CPU clock is switched, the peripheral hardware clock is also switched.

**Table 5-5. Maximum Time Required for System Clock Switchover**

Clock A	Switching Directions	Clock B	Remark
$f_{IH}$	↔	$f_{MX}$	See <b>Table 5-6</b>

**Table 5-6. Maximum Number of Clocks Required for  $f_{IH} \leftrightarrow f_{MX}$**

Set Value Before Switchover		Set Value After Switchover	
MCM0		MCM0	
		0 ( $f_{MAIN} = f_{IH}$ )	1 ( $f_{MAIN} = f_{MX}$ )
0 ( $f_{MAIN} = f_{IH}$ )	$f_{MX} \geq f_{IH}$		2 clock
	$f_{MX} < f_{IH}$		$2f_{IH}/f_{MX}$ clock
1 ( $f_{MAIN} = f_{MX}$ )	$f_{MX} \geq f_{IH}$	$2f_{MX}/f_{IH}$ clock	
	$f_{MX} < f_{IH}$	2 clock	

- Remarks 1.** The number of clocks listed in Table 5-6 is the number of CPU clocks before switchover.  
**2.** Calculate the number of clocks in Table 5-6 by removing the decimal portion.

**Example** When switching the main system clock from the high-speed system clock to the high-speed on-chip oscillator clock (@ oscillation with  $f_{IH} = 8$  MHz,  $f_{MX} = 10$  MHz)

$$2f_{MX}/f_{IH} = 2 (10/8) = 2.5 \rightarrow 3 \text{ clocks}$$

### 5.6.6 Conditions before clock oscillation is stopped

The following lists the register flag settings for stopping the clock oscillation (disabling external clock input) and conditions before the clock oscillation is stopped.

**Table 5-7. Conditions Before the Clock Oscillation Is Stopped and Flag Settings**

Clock	Conditions Before Clock Oscillation Is Stopped (External Clock Input Disabled)	Flag Settings of SFR Register
High-speed on-chip oscillator clock	MCS = 1 (The CPU is operating on a clock other than the high-speed on-chip oscillator clock.)	HIOSTOP = 1
X1 clock External main system clock	MCS = 0 (The CPU is operating on a clock other than the high-speed system clock.)	MSTOP = 1

&lt;R&gt;

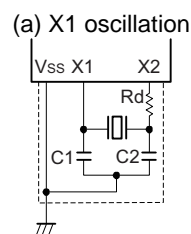
## 5.7 Resonator and Oscillator Constants

The resonators for which the operation is verified and their oscillator constants are shown below.

**Caution 1.** The constants for these oscillator circuits are reference values based on specific environments set up for evaluation by the manufacturers. For actual applications, request evaluation by the manufacturer of the oscillator circuit mounted on a board. Furthermore, if you are switching from a different product to this microcontroller, and whenever you change the board, again request evaluation by the manufacturer of the oscillator circuit mounted on the new board.

**Caution 2.** The oscillation voltage and oscillation frequency only indicate the oscillator characteristic. Use the RL78 microcontroller so that the internal operation conditions are within the specifications of the DC and AC characteristics.

Figure 5 - 17 Example of External Circuit



&lt;R&gt;

## (1) X1 oscillation

As of Feb 2014

Manufacturer	Resonator	Part Number	SMD/ Lead	Frequency (MHz)	Flash Operation Mode <sup>Note 1</sup>	Circuit Constants (Reference) <sup>Note 2</sup>			Voltage Range (V)			
						C1 (pF)	C2 (pF)	Rd (kΩ)	MIN.	MAX.		
Murata Manufacturing Co., Ltd.	Ceramic resonator	CSTCR4M00G55-R0	SMD	4.0	LS	(39)	(39)	0	2.7	5.5		
		CSTLS4M00G53-B0	Lead			(15)	(15)	0				
		CSTCR5M00G53-R0	SMD	5.0		(15)	(15)	0				
		CSTLS5M00G53-B0	Lead			(15)	(15)	0				
		CSTCR6M00G53-R0	SMD	6.0		(15)	(15)	0				
		CSTLS6M00G53-B0	Lead			(15)	(15)	0				
		CSTCE8M00G52-R0	SMD	8.0		(10)	(10)	0				
		CSTLS8M00G53-B0	Lead			(15)	(15)	0				
		CSTCR5M00G53-R0	SMD	5.0		HS	(15)	(15)	0	2.7	5.5	
		CSTLS5M00G53-B0	Lead				(15)	(15)	0			
		CSTCR6M00G53-R0	SMD	6.0			(15)	(15)	0			
		CSTLS6M00G53-B0	Lead				(15)	(15)	0			
		CSTCE8M00G52-R0	SMD	8.0			(10)	(10)	0			
		CSTLS8M00G53-B0	Lead				(15)	(15)	0			
		CSTCE10M0G52-R0	SMD	10.0			(10)	(10)	0			
		CSTLS10M0G53-B0	Lead				(15)	(15)	0			
		CSTCE16M0V53-R0	SMD	16.0			(15)	(15)	0			
		CSTLS16M0X51-B0	Lead				(5)	(5)	0			
CSTCE20M0V51-R0	SMD	20.0	HS	(5)	(5)		0	2.7	5.5			
CSTLS20M0X51-B0	Lead			(5)	(5)		0					
Nihon Dempa Kogyo Co., Ltd.	Ceramic resonator	NX3225HA <sup>Note 3</sup>		SMD	20		HS	Note 3			2.7	5.5

- Notes**
1. Set the flash operation mode by using the CMODE1 and CMODE0 bits of the option byte (000C2H).
  2. The reset value of the LVIM register varies depending on the reset source.
  3. When using these resonators, contact Nihon Dempa Kogyo Co., Ltd (<http://www.ndk.com/en>) for more information on matching.

**Remark** Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.  
 HS (High-speed main) mode: 2.7 V ≤ VDD ≤ 5.5 V@1 MHz to 24 MHz  
 LS (Low-speed main) mode: 2.7 V ≤ VDD ≤ 5.5 V@1 MHz to 8 MHz.



**CHAPTER 6 TIMER ARRAY UNIT**

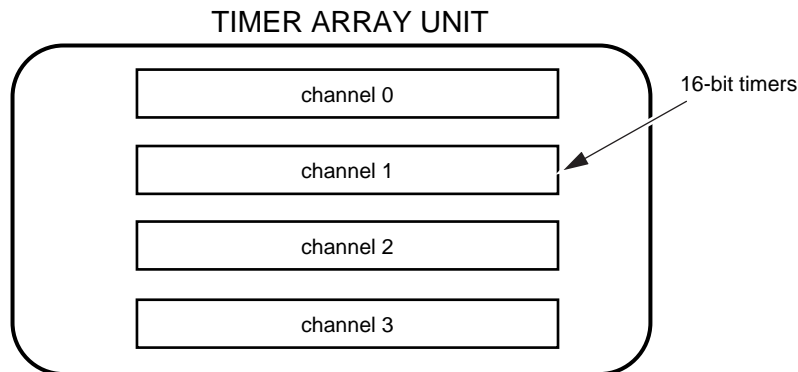
The number of units or channels of the timer array unit differs, depending on the product.

<R>	Units	Channels	24, 32-pin
	Unit 0	Channel 0	√
		Channel 1	√
		Channel 2	√
		Channel 3	√

<R> **Caution 1. Most of the following descriptions in this chapter use the 32-pin products as an example.**

The timer array unit has four 16-bit timers.

Each 16-bit timer is called a channel and can be used as an independent timer. In addition, two or more “channels” can be used to create a high-accuracy timer.



For details about each function, see the table below.

Independent Channel Operation Function	Simultaneous Channel Operation Function
<ul style="list-style-type: none"> <li>• Interval timer (→ see <b>6.8.1</b>)</li> <li>• Square wave output (→ see <b>6.8.1</b>)</li> <li>• External event counter (→ see <b>6.8.2</b>)</li> <li>• Input pulse interval measurement (→ see <b>6.8.3</b>)</li> <li>• Measurement of high-/low-level width of input signal (→ see <b>6.8.4</b>)</li> <li>• Delay counter (→ see <b>6.8.5</b>)</li> </ul>	<ul style="list-style-type: none"> <li>• One-shot pulse output (→ see <b>6.9.1</b>)</li> <li>• PWM output (→ see <b>6.9.2</b>)</li> <li>• Multiple PWM output (→ see <b>6.9.3</b>)</li> </ul>

<R>

It is possible to use the 16-bit timer of channels 1 and 3 of the unit 0 as two 8-bit timers (higher and lower). The functions that can use channels 1 and 3 as 8-bit timers are as follows:

- Interval timer (upper or lower 8-bit timer)/square wave output (lower 8-bit timer only)
- External event counter (lower 8-bit timer only)
- Delay counter (lower 8-bit timer only)

## 6.1 Functions of Timer Array Unit

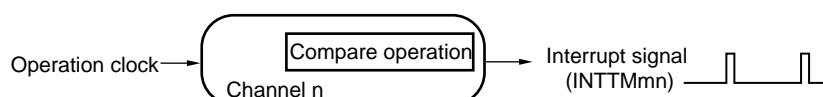
Timer array unit has the following functions.

### 6.1.1 Independent channel operation function

By operating a channel independently, it can be used for the following purposes without being affected by the operation mode of other channels.

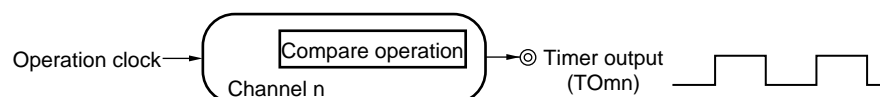
#### (1) Interval timer

Each timer of a unit can be used as a reference timer that generates an interrupt (INTTMmn) at fixed intervals.



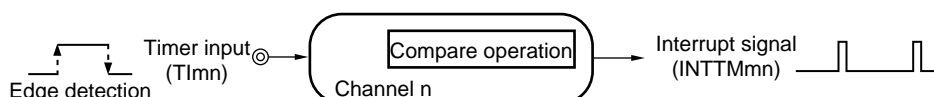
#### (2) Square wave output

A toggle operation is performed each time INTTMmn interrupt is generated and a square wave with a duty factor of 50% is output from a timer output pin (TOMn).



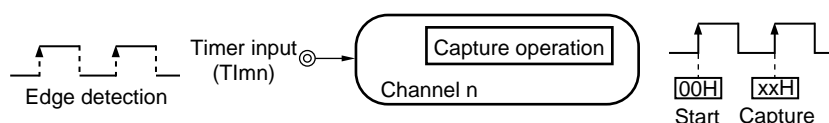
#### (3) External event counter

Each timer of a unit can be used as an event counter that generates an interrupt when the number of the valid edges of a signal input to the timer input pin (TImn) has reached a specific value.



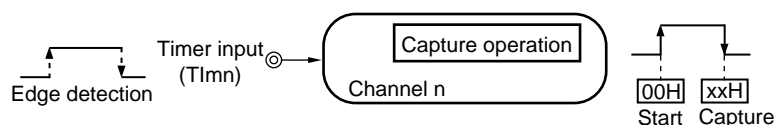
#### (4) Input pulse interval measurement

Counting is started by the valid edge of a pulse signal input to a timer input pin (TImn). The count value of the timer is captured at the valid edge of the next pulse. In this way, the interval of the input pulse can be measured.



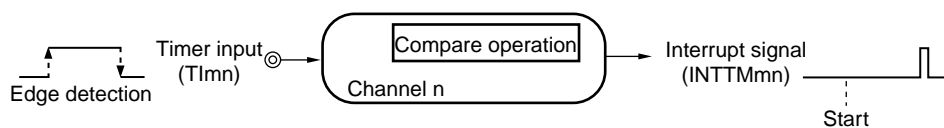
#### (5) Measurement of high-/low-level width of input signal

Counting is started by a single edge of the signal input to the timer input pin (TImn), and the count value is captured at the other edge. In this way, the high-level or low-level width of the input signal can be measured.



**(6) Delay counter**

Counting is started at the valid edge of the signal input to the timer input pin (TImn), and an interrupt is generated after any delay period.



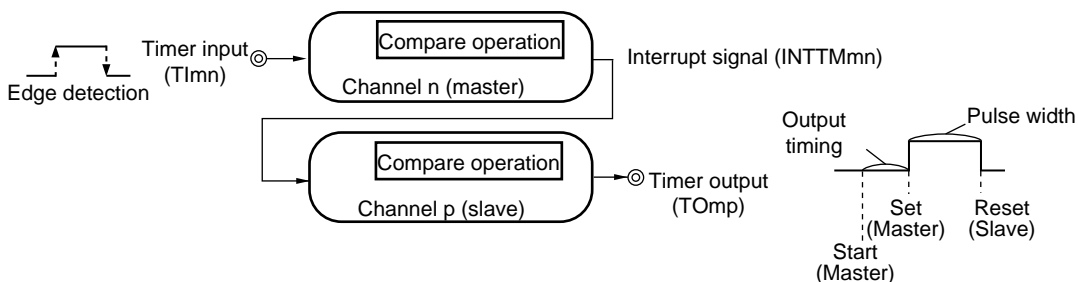
<R> **Remark 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**6.1.2 Simultaneous channel operation function**

By using the combination of a master channel (a reference timer mainly controlling the cycle) and slave channels (timers operating according to the master channel), channels can be used for the following purposes.

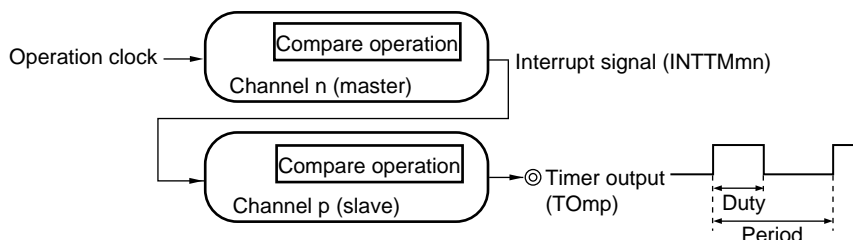
**(1) One-shot pulse output**

Two channels are used as a set to generate a one-shot pulse with a specified output timing and a specified pulse width.



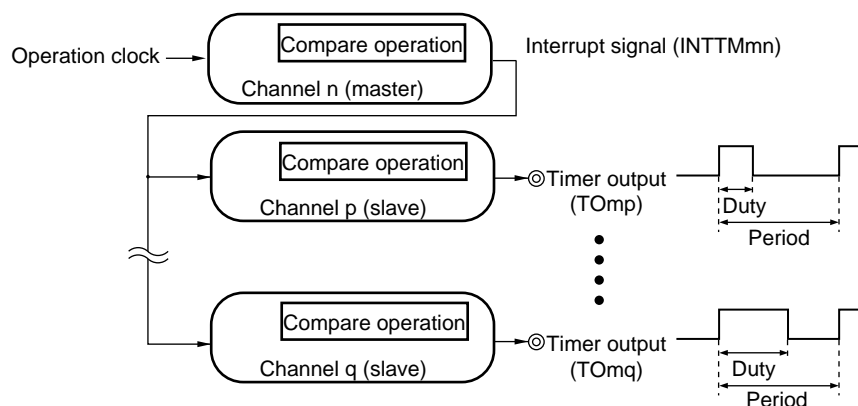
**(2) PWM (Pulse Width Modulation) output**

Two channels are used as a set to generate a pulse with a specified period and a specified duty factor.



**(3) Multiple PWM (Pulse Width Modulation) output**

By extending the PWM function and using one master channel and two or more slave channels, up to three types of PWM signals that have a specific period and a specified duty factor can be generated.



**Caution** For details about the rules of simultaneous channel operation function, see 6.4.1 Basic rules of simultaneous channel operation function.

**Remark** m: Unit number ( $m = 0$ ), n: Channel number ( $n = 0$  to 3),  
p, q: Slave channel number ( $n < p < q \leq 3$ )

**6.1.3 8-bit timer operation function (channels 1 and 3 only)**

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels. This function can only be used for channels 1 and 3.

**Caution** There are several rules for using 8-bit timer operation function.  
For details, see 6.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only).

## 6.2 Configuration of Timer Array Unit

Timer array unit includes the following hardware.

**Table 6-1. Configuration of Timer Array Unit**

Item	Configuration
Timer/counter	Timer count register mn (TCRmn)
Register	Timer data register mn (TDRmn)
Timer input	TI00 to TI03
Timer output	TO00 to TO03, output controller
Control registers	<Registers of unit setting block> <ul style="list-style-type: none"> <li>• Peripheral enable register 0 (PER0)</li> <li>• Timer clock select register m (TPSm)</li> <li>• Timer channel enable status register m (TEm)</li> <li>• Timer channel start register m (TSM)</li> <li>• Timer channel stop register m (TTm)</li> <li>• Timer input select register 0 (TIS0)</li> <li>• Timer output enable register m (TOEm)</li> <li>• Timer output register m (TOM)</li> <li>• Timer output level register m (TOLm)</li> <li>• Timer output mode register m (TOMm)</li> </ul> <hr/> <Registers of each channel> <ul style="list-style-type: none"> <li>• Timer mode register mn (TMRmn)</li> <li>• Timer status register mn (TSRmn)</li> <li>• Noise filter enable register 1 (NFEN1)</li> <li>• Port mode register (PMxx)<sup>Note</sup></li> <li>• Port register (Pxx)<sup>Note</sup></li> </ul>

<R>

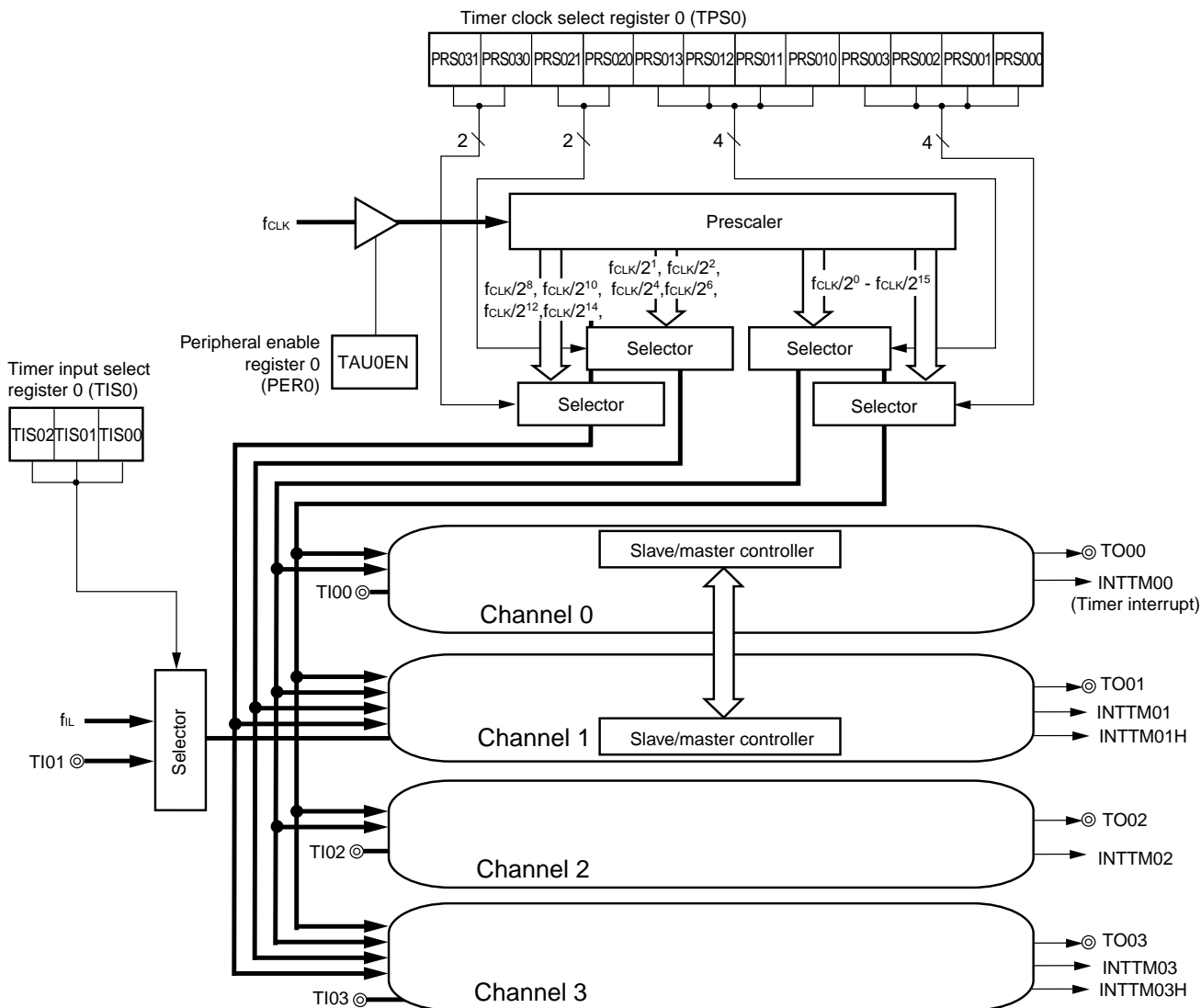
**Caution** The port mode registers (PMxx) and port registers (Pxx) to be set differ depending on the product. For details, see 4.5 Settings of Port Related Register When Using Alternate Function.

<R>

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-1 to Figure 6-5 shows the block diagrams of the timer array unit.

Figure 6-1. Entire Configuration of Timer Array Unit 0



**Remark** fiL: Low-speed on-chip oscillator clock frequency

Figure 6-2. Internal Block Diagram of Channel 0 of Timer Array Unit 0

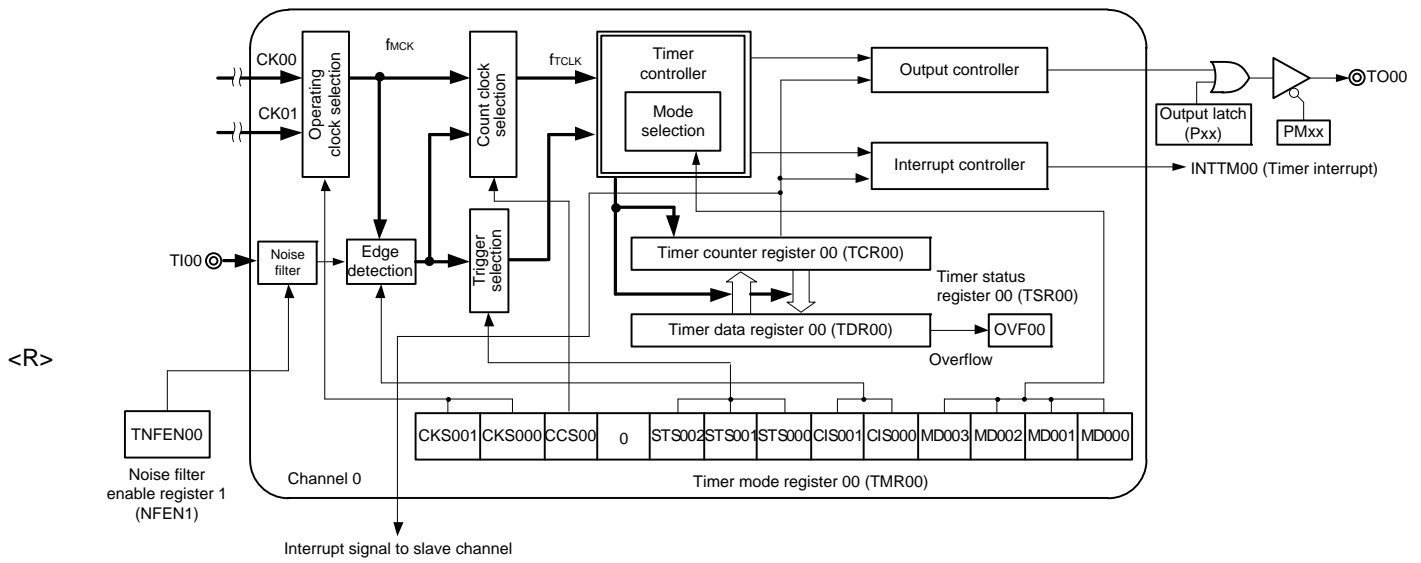
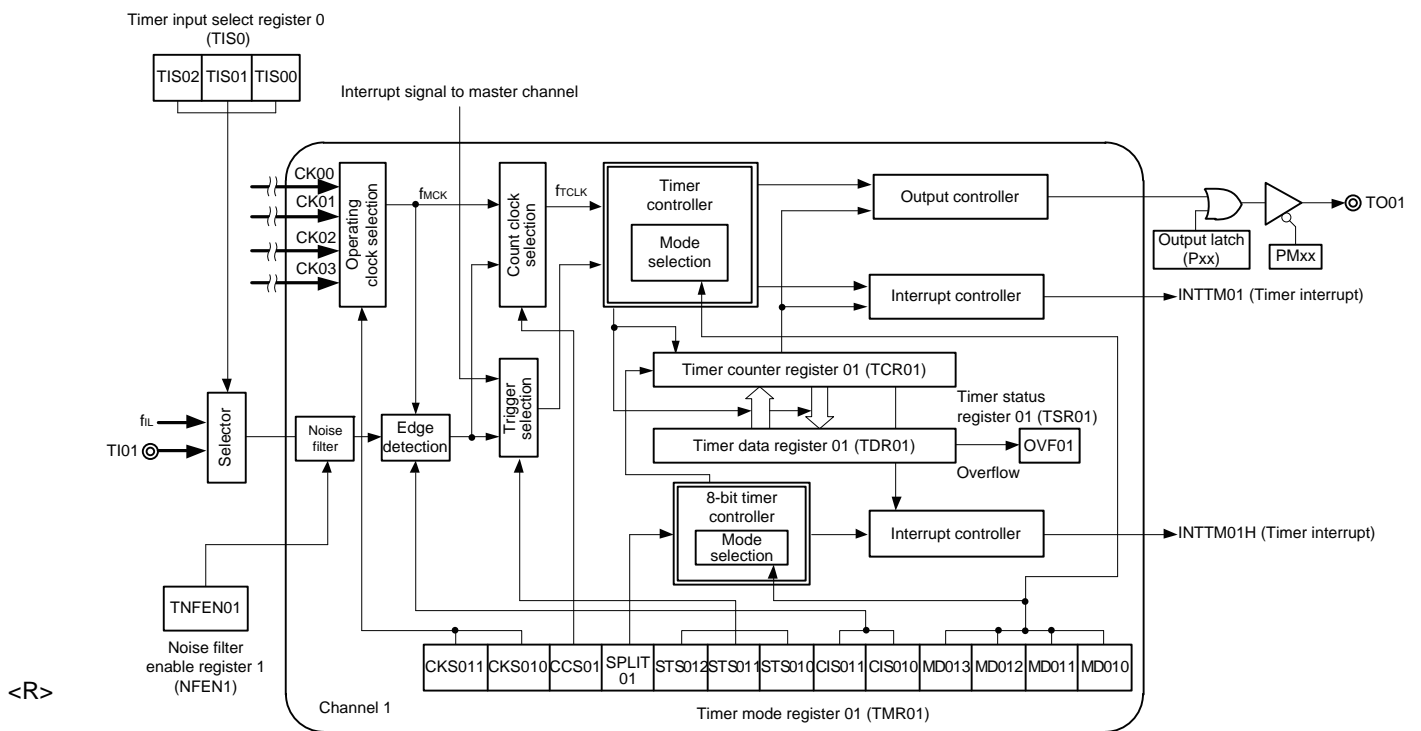


Figure 6-3. Internal Block Diagram of Channel 1 of Timer Array Unit 0





<R>

Figure 6-4. Internal Block Diagram of Channel 2 of Timer Array Unit 0

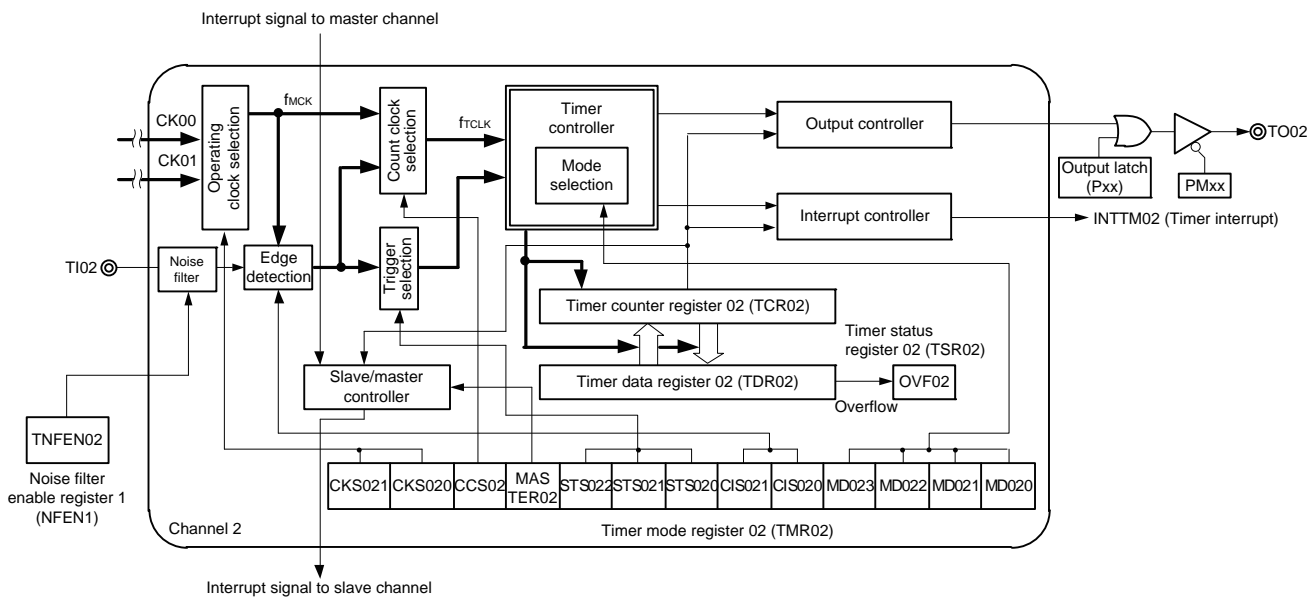
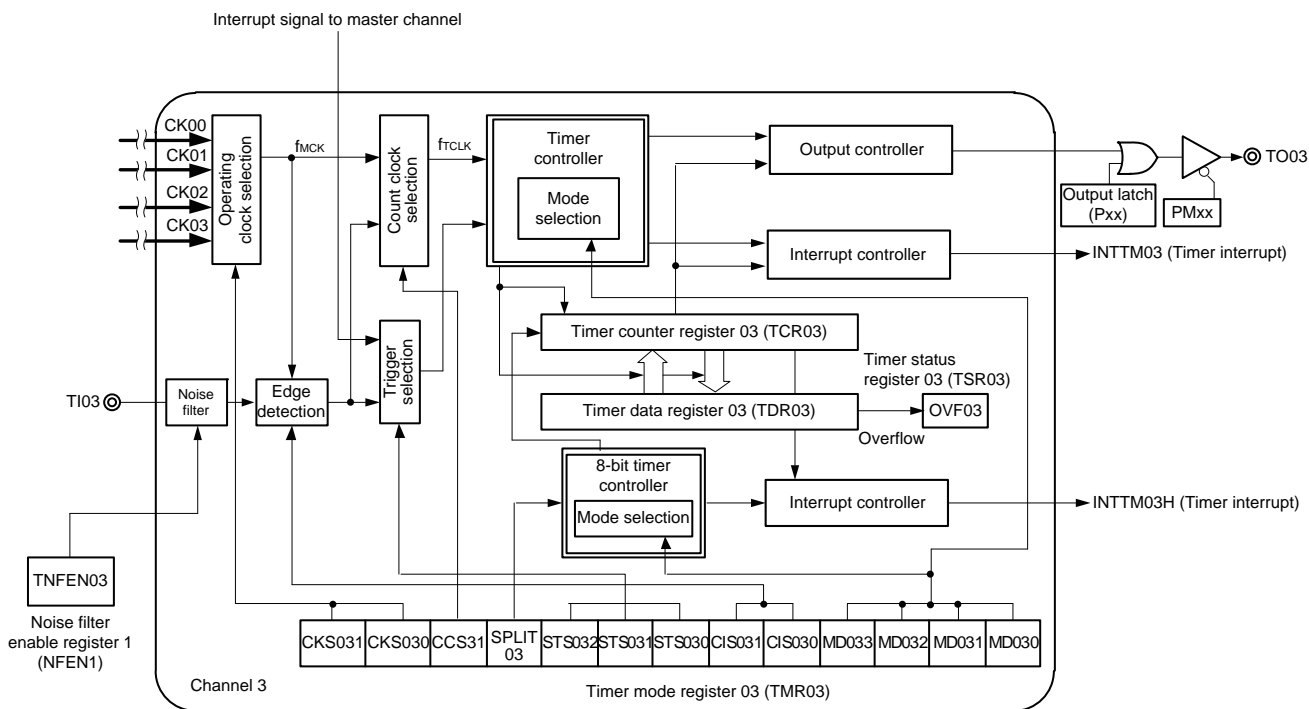


Figure 6-5. Internal Block Diagram of Channel 3 of Timer Array Unit 0



<R>

### 6.2.1 Timer count register mn (TCRmn)

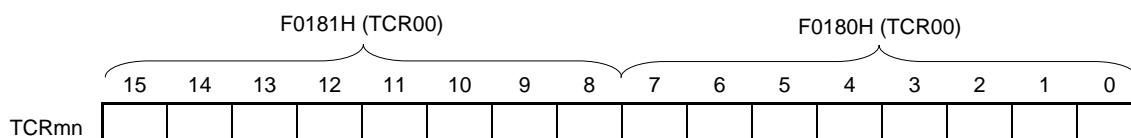
The TCRmn register is a 16-bit read-only register and is used to count clocks.

The value of this counter is incremented or decremented in synchronization with the rising edge of a count clock.

Whether the counter is incremented or decremented depends on the operation mode that is selected by the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn) (see 6.3.3 Timer mode register mn (TMRmn)).

**Figure 6-6. Format of Timer Count Register mn (TCRmn)**

Address: F0180H, F0181H (TCR00) to F0186H, F0187H (TCR03) After reset: FFFFH R



**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

The count value is set to FFFFH in the following cases.

- When the reset signal is generated
- When the TAUmEN bit of peripheral enable register 0 (PER0) is cleared
- When counting of the slave channel has been completed in the PWM output mode
- When counting of the slave channel has been completed in the delay count mode
- When counting of the master/slave channel has been completed in the one-shot pulse output mode
- When counting of the slave channel has been completed in the multiple PWM output mode

The count value is cleared to 0000H in the following cases.

- When the start trigger is input in the capture mode
- When capturing has been completed in the capture mode

**Caution** The count value is not captured to timer data register mn (TDRmn) even when the TCRmn register is read.

The TCRmn register read value differs as follows according to operation mode changes and the operating status.

**Table 6-2. Timer Count Register mn (TCRmn) Read Value in Various Operation Modes**

Operation Mode	Count Mode	Timer Count Register mn (TCRmn) Read Value <sup>Note</sup>			
		Value if the operation mode was changed after releasing reset	Value if the Operation was restarted after count operation paused (TTmn = 1)	Value if the operation mode was changed after count operation paused (TTmn = 1)	Value when waiting for a start trigger after one count
Interval timer mode	Count down	FFFFH	Value if stop	Undefined	–
Capture mode	Count up	0000H	Value if stop	Undefined	–
Event counter mode	Count down	FFFFH	Value if stop	Undefined	–
One-count mode	Count down	FFFFH	Value if stop	Undefined	FFFFH
Capture & one-count mode	Count up	0000H	Value if stop	Undefined	Capture value of TDRmn register + 1

**Note** This indicates the value read from the TCRmn register when channel n has stopped operating as a timer (TEmn = 0) and has been enabled to operate as a counter (TSmn = 1). The read value is held in the TCRmn register until the count operation starts.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.2.2 Timer data register mn (TDRmn)

This is a 16-bit register from which a capture function and a compare function can be selected.

The capture or compare function can be switched by selecting an operation mode by using the MDmn3 to MDmn0 bits of timer mode register mn (TMRmn).

The value of the TDRmn register can be changed at any time.

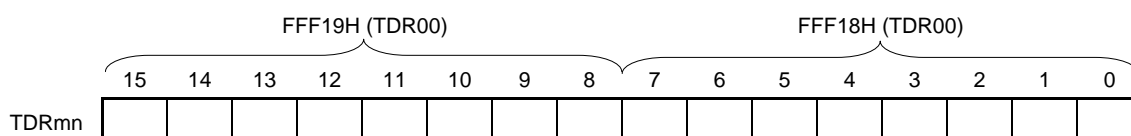
This register can be read or written in 16-bit units.

In addition, for the TDRm1 and TDRm3 registers, while in the 8-bit timer mode (when the SPLIT bits of timer mode registers m1 and m3 (TMRm1, TMRm3) are 1), it is possible to read and write the data in 8-bit units, with TDRm1H and TDRm3H used as the higher 8 bits, and TDRm1L and TDRm3L used as the lower 8 bits.

Reset signal generation clears this register to 0000H.

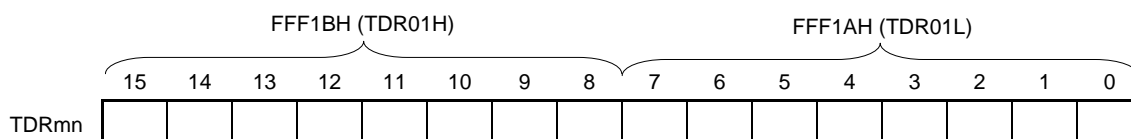
**Figure 6-7. Format of Timer Data Register mn (TDRmn) (n = 0, 2)**

Address: FFF18H, FFF19H (TDR00), FFF64H, FFF65H (TDR02) After reset: 0000H R/W



**Figure 6-8. Format of Timer Data Register mn (TDRmn) (n = 1, 3)**

Address: FFF1AH, FFF1BH (TDR01), FFF66H, FFF67H (TDR03) After reset: 0000H R/W



**(i) When timer data register mn (TDRmn) is used as compare register**

Counting down is started from the value set to the TDRmn register. When the count value reaches 0000H, an interrupt signal (INTTMmn) is generated. The TDRmn register holds its value until it is rewritten.

**Caution** The TDRmn register does not perform a capture operation even if a capture trigger is input, when it is set to the compare function.

**(ii) When timer data register mn (TDRmn) is used as capture register**

The count value of timer count register mn (TCRmn) is captured to the TDRmn register when the capture trigger is input.

A valid edge of the TImn pin can be selected as the capture trigger. This selection is made by timer mode register mn (TMRmn).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3 Registers Controlling Timer Array Unit

Timer array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Timer clock select register m (TPSm)
- Timer mode register mn (TMRmn)
- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSm)
- Timer channel stop register m (TTm)
- Timer input select register 0 (TIS0)
- Timer output enable register m (TOEm)
- Timer output register m (TOM)
- Timer output level register m (TOLm)
- Timer output mode register m (TOMm)
- Input switch control register (ISC)
- Noise filter enable register 1 (NFEN1)
- Port mode register (PMxx)
- Port register (Pxx)

<R>

**Caution** Which registers and bits are included depends on the product. Be sure to set bits that are not mounted to their initial values.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.1 Peripheral enable register 0 (PER0)

This registers is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the timer array unit 0 is used, be sure to set bit 0 (TAU0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 6-9. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

TAU0EN	Control of timer array 0 unit input clock
0	Stops supply of input clock. <ul style="list-style-type: none"> <li>• SFR used by the timer array unit 0 cannot be written.</li> <li>• The timer array unit 0 is in the reset status.</li> </ul>
1	Supplies input clock. <ul style="list-style-type: none"> <li>• SFR used by the timer array unit 0 can be read/written.</li> </ul>

<R>

**Cautions 1.** When setting the timer array unit, be sure to set the following registers first while the TAUmEN bit is set to 1. If TAUmEN = 0, writing to a control register of timer array unit is ignored, and all read values are default values (except for the timer input select register 0 (TIS0), input switch control register (ISC), noise filter enable register 1 (NFEN1), port mode registers 1, 3 (PM1, PM3), and port registers 1, 3 ( P1, P3)).

- Timer status register mn (TSRmn)
- Timer channel enable status register m (TEm)
- Timer channel start register m (TSM)
- Timer channel stop register m (TTm)
- Timer output enable register m (TOEm)
- Timer output register m (TOM)
- Timer output level register m (TOLm)
- Timer output mode register m (TOMm)

**2.** Be sure to clear bits 1, 3, and 7 to "0".

### 6.3.2 Timer clock select register m (TPSm)

<R> The TPSm register is a 16-bit register that is used to select two types or four types of operation clocks (CKm0, CKm1, CKm2, CKm3) that are commonly supplied to each channel from external prescaler. CKm1 is selected by using bits 7 to 4 of the TPSm register, and CKm0 is selected by using bits 3 to 0. In addition, for channel 1 and 3, CKm2 is selected by using bits 9 and 8 of the TPSm register, and CKm3 is selected by using bits 13 and 12.

Rewriting of the TPSm register during timer operation is possible only in the following cases.

If the PRSm00 to PRSm03 bits can be rewritten (n = 0 to 3):

All channels for which CKm0 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 0) are stopped (TEmn = 0).

If the PRSm10 to PRSm13 bits can be rewritten (n = 0 to 3):

All channels for which CKm1 is selected as the operation clock (CKSmn1, CKSmn0 = 0, 1) are stopped (TEmn = 0).

If the PRSm20 and PRSm21 bits can be rewritten (n = 1, 3):

All channels for which CKm2 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 0) are stopped (TEmn = 0).

If the PRSm30 and PRSm31 bits can be rewritten (n = 1, 3):

All channels for which CKm3 is selected as the operation clock (CKSmn1, CKSmn0 = 1, 1) are stopped (TEmn = 0).

The TPSm register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

Figure 6-10. Format of Timer Clock Select register m (TPSm) (1/2)

Address: F01B6H, F01B7H (TPS0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Selection of operation clock (CKmk) <sup>Note</sup> (k = 0, 1)	f <sub>CLK</sub>				
					f <sub>CLK</sub> = 2 MHz	f <sub>CLK</sub> = 5 MHz	f <sub>CLK</sub> = 10 MHz	f <sub>CLK</sub> = 20 MHz	f <sub>CLK</sub> = 32 MHz
0	0	0	0	f <sub>CLK</sub>	2 MHz	5 MHz	10 MHz	20 MHz	32 MHz
0	0	0	1	f <sub>CLK</sub> /2	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	0	1	0	f <sub>CLK</sub> /2 <sup>2</sup>	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
0	0	1	1	f <sub>CLK</sub> /2 <sup>3</sup>	250 kHz	625 kHz	1.25 MHz	2.5 MHz	4 MHz
0	1	0	0	f <sub>CLK</sub> /2 <sup>4</sup>	125 kHz	312.5 kHz	625 kHz	1.25 MHz	2 MHz
0	1	0	1	f <sub>CLK</sub> /2 <sup>5</sup>	62.5 kHz	156.2 kHz	312.5 kHz	625 kHz	1 MHz
0	1	1	0	f <sub>CLK</sub> /2 <sup>6</sup>	31.25 kHz	78.1 kHz	156.2 kHz	312.5 kHz	500 kHz
0	1	1	1	f <sub>CLK</sub> /2 <sup>7</sup>	15.62 kHz	39.1 kHz	78.1 kHz	156.2 kHz	250 kHz
1	0	0	0	f <sub>CLK</sub> /2 <sup>8</sup>	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
1	0	0	1	f <sub>CLK</sub> /2 <sup>9</sup>	3.91 kHz	9.76 kHz	19.5 kHz	39.1 kHz	62.5 kHz
1	0	1	0	f <sub>CLK</sub> /2 <sup>10</sup>	1.95 kHz	4.88 kHz	9.76 kHz	19.5 kHz	31.25 kHz
1	0	1	1	f <sub>CLK</sub> /2 <sup>11</sup>	976 Hz	2.44 kHz	4.88 kHz	9.76 kHz	15.63 kHz
1	1	0	0	f <sub>CLK</sub> /2 <sup>12</sup>	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	0	1	f <sub>CLK</sub> /2 <sup>13</sup>	244 Hz	610 Hz	1.22 kHz	2.44 kHz	3.91 kHz
1	1	1	0	f <sub>CLK</sub> /2 <sup>14</sup>	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz
1	1	1	1	f <sub>CLK</sub> /2 <sup>15</sup>	61 Hz	153 Hz	305 Hz	610 Hz	976 Hz

**Note** When changing the clock selected for f<sub>CLK</sub> (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH).

The timer array unit must also be stopped if the operating clock (fMCK) or the valid edge of the signal input from the TImn pin is selected.

**Cautions 1.** Be sure to clear bits 15, 14, 11, 10 to "0".

**2.** If f<sub>CLK</sub> (undivided) is selected as the operation clock (CKmk) and TDRnm is set to 0000H (n = 0, m = 0 to 3), interrupt requests output from timer array units are not detected.

**Remarks 1.** f<sub>CLK</sub>: CPU/peripheral hardware clock frequency

**2.** Waveform of the clock to be selected in the TPSm register which becomes high level for one period of f<sub>CLK</sub> from its rising edge (m = 1 to 15). For details, see 6.5.1 Count clock (f<sub>TCLK</sub>).

&lt;R&gt;

Figure 6-10. Format of Timer Clock Select register m (TPSm) (2/2)

Address: F01B6H, F01B7H (TPS0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TPSm	0	0	PRS m31	PRS m30	0	0	PRS m21	PRS m20	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRS m21	PRS m20	Selection of operation clock (CKm2) <sup>Note</sup>					
		$f_{CLK} = 2 \text{ MHz}$	$f_{CLK} = 5 \text{ MHz}$	$f_{CLK} = 10 \text{ MHz}$	$f_{CLK} = 20 \text{ MHz}$	$f_{CLK} = 32 \text{ MHz}$	
0	0	$f_{CLK}/2$	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	1	$f_{CLK}/2^2$	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
1	0	$f_{CLK}/2^4$	125 kHz	312.5 kHz	625 MHz	1.25 MHz	2 MHz
1	1	$f_{CLK}/2^6$	31.25 kHz	78.1 kHz	156.2 kHz	312.5 kHz	500 kHz

PRS m31	PRS m30	Selection of operation clock (CKm3) <sup>Note</sup>					
		$f_{CLK} = 2 \text{ MHz}$	$f_{CLK} = 5 \text{ MHz}$	$f_{CLK} = 10 \text{ MHz}$	$f_{CLK} = 20 \text{ MHz}$	$f_{CLK} = 32 \text{ MHz}$	
0	0	$f_{CLK}/2^8$	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
0	1	$f_{CLK}/2^{10}$	1.95 kHz	4.88 kHz	9.76 kHz	19.5 kHz	31.25 kHz
1	0	$f_{CLK}/2^{12}$	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	$f_{CLK}/2^{14}$	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz

**Note** When changing the clock selected for  $f_{CLK}$  (by changing the system clock control register (CKC) value), stop timer array unit (TTm = 00FFH).

&lt;R&gt;

The timer array unit must also be stopped if the operating clock ( $f_{MCK}$ ) specified by using the CKSmn0, and CKSmn1 bits or the valid edge of the signal input from the TImn pin is selected as the count clock ( $f_{TCLK}$ ).

**Caution** Be sure to clear bits 15, 14, 11, and 10 to "0".

By using channels 1 and 3 in the 8-bit timer mode and specifying CKm2 or CKm3 as the operation clock, the interval times shown in Table 6-3 can be achieved by using the interval timer function.

Table 6-3. Interval Times Available for Operation Clock CKSm2 or CKSm3

Clock		Interval Time <sup>Note</sup> ( $f_{CLK} = 32 \text{ MHz}$ )			
		10 $\mu\text{s}$	100 $\mu\text{s}$	1 ms	10 ms
CKm2	$f_{CLK}/2$	√	–	–	–
	$f_{CLK}/2^2$	√	–	–	–
	$f_{CLK}/2^4$	√	√	–	–
	$f_{CLK}/2^6$	√	√	–	–
CKm3	$f_{CLK}/2^8$	–	√	√	–
	$f_{CLK}/2^{10}$	–	√	√	–
	$f_{CLK}/2^{12}$	–	–	√	√
	$f_{CLK}/2^{14}$	–	–	√	√

**Note** The margin is within 5 %.

**Remarks 1.**  $f_{CLK}$ : CPU/peripheral hardware clock frequency

**2.** For details of a signal of  $f_{CLK}/2^j$  selected with the TPSm register, see **6.5.1 Count clock ( $f_{TCLK}$ )**.



### 6.3.3 Timer mode register mn (TMRmn)

The TMRmn register sets an operation mode of channel n. This register is used to select the operation clock ( $f_{MCK}$ ), select the count clock, select the master/slave, select the 16 or 8-bit timer (only for channels 1 and 3), specify the start trigger and capture trigger, select the valid edge of the timer input, and specify the operation mode (interval, capture, event counter, one-count, or capture and one-count).

Rewriting the TMRmn register is prohibited when the register is in operation (when  $TE_{mn} = 1$ ). However, bits 7 and 6 ( $CIS_{mn1}$ ,  $CIS_{mn0}$ ) can be rewritten even while the register is operating with some functions (when  $TE_{mn} = 1$ ) (for details, see **6.8 Independent Channel Operation Function of Timer Array Unit** and **6.9 Simultaneous Channel Operation Function of Timer Array Unit**).

The TMRmn register can be set by a 16-bit memory manipulation instruction.  
Reset signal generation clears this register to 0000H.

**Caution** The bits mounted depend on the channels in the bit 11 of TMRmn register.

**TMRm2, TMRm4, TMRm6: MASTERmn bit (n = 2)**

**TMRm1, TMRm3: SPLITmn bit (n = 1, 3)**

**TMRm0, TMRm5, TMRm7: Fixed to 0**

Figure 6-11. Format of Timer Mode Register mn (TMRmn) (1/4)

Address: F0190H, F0191H (TMR00) to F0196H, F0197H (TMR03) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 2)	CKS mn1	CKS mn0	0	CCS mn	MAST ERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 1, 3)	CKS mn1	CKS mn0	0	CCS mn	SPLIT mn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 0)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

CKS mn1	CKS mn0	Selection of operation clock ( $f_{mck}$ ) of channel n
0	0	Operation clock CKm0 set by timer clock select register m (TPSm)
0	1	Operation clock CKm2 set by timer clock select register m (TPSm)
1	0	Operation clock CKm1 set by timer clock select register m (TPSm)
1	1	Operation clock CKm3 set by timer clock select register m (TPSm)
Operation clock ( $f_{mck}$ ) is used by the edge detector. A count clock ( $f_{tclk}$ ) and a sampling clock are generated depending on the setting of the CCSmn bit.		
The operation clocks CKm2 and CKm3 can only be selected for channels 1 and 3.		

CCS mn	Selection of count clock ( $f_{tclk}$ ) of channel n
0	Operation clock ( $f_{mck}$ ) specified by the CKSmn0 and CKSmn1 bits
1	Valid edge of input signal input from the TImn pin In channel 5, Valid edge of input signal selected by TIS0
Count clock ( $f_{tclk}$ ) is used for the timer/counter, output controller, and interrupt controller.	

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Cautions 1.** Be sure to clear bits 13, 5, and 4 to "0".

- 2.** The timer array unit must be stopped (TTm = 00FFH) if the clock selected for  $f_{tclk}$  is changed (by changing the value of the system clock control register (CKC)), even if the operating clock specified by using the CKSmn0 and CKSmn1 bits ( $f_{mck}$ ) or the valid edge of the signal input from the TImn pin is selected as the count clock ( $f_{tclk}$ ).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-11. Format of Timer Mode Register mn (TMRmn) (2/4)

Address: F0190H, F0191H (TMR00) to F0196H, F0197H (TMR03) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 2)	CKS mn1	CKS mn0	0	CCS mn	MAST ERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 1, 3)	CKS mn1	CKS mn0	0	CCS mn	SPLIT mn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 0)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

(Bit 11 of TMRmn (n = 2))

MAS TER mn	Selection between using channel n independently or simultaneously with another channel(as a slave or master)	
0	Operates in independent channel operation function or as slave channel in simultaneous channel operation function.	
1	Operates as master channel in simultaneous channel operation function.	
<p>Only channel 2 can be set as a master channel (MASTERmn = 1).</p> <p>Channel 0 is fixed to 0 (channel 0 always operates as master regardless of the bit setting, because it is the highest channel).</p> <p>Clear the MASTERmn bit to 0 for a channel that is used with the independent channel operation function.</p>		

&lt;R&gt;

(Bit 11 of TMRmn (n = 1, 3))

SPLI Tmn	Selection of 8 or 16-bit timer operation for channels 1 and 3	
0	Operates as 16-bit timer. (Operates in independent channel operation function or as slave channel in simultaneous channel operation function.)	
1	Operates as 8-bit timer.	

STS mn2	STS mn1	STS mn0	Setting of start trigger or capture trigger of channel n
0	0	0	Only software trigger start is valid (other trigger sources are unselected).
0	0	1	Valid edge of the TImn pin input is used as both the start trigger and capture trigger.
0	1	0	Both the edges of the TImn pin input are used as a start trigger and a capture trigger.
1	0	0	Interrupt signal of the master channel is used (when the channel is used as a slave channel with the simultaneous channel operation function).
Other than above			Setting prohibited

**Note** Bit 11 is fixed at 0 of read only, write is ignored.**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**Figure 6-11. Format of Timer Mode Register mn (TMRmn) (3/4)**

Address: F0190H, F0191H (TMR00) to F0196H, F0197H (TMR03) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 2)	CKS mn1	CKS mn0	0	CCS mn	MAST ERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 1, 3)	CKS mn1	CKS mn0	0	CCS mn	SPLIT mn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 0)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

CIS mn1	CIS mn0	Selection of TImn pin input valid edge
0	0	Falling edge
0	1	Rising edge
1	0	Both edges (when low-level width is measured) Start trigger: Falling edge, Capture trigger: Rising edge
1	1	Both edges (when high-level width is measured) Start trigger: Rising edge, Capture trigger: Falling edge
If both the edges are specified when the value of the STSmn2 to STSmn0 bits is other than 010B, set the CISmn1 to CISmn0 bits to 10B.		

MD mn3	MD mn2	MD mn1	Operation mode of channel n	Corresponding function	Count operation of TCR
0	0	0	Interval timer mode	Interval timer/Square wave output/PWM output (master)	Counting down
0	1	0	Capture mode	Input pulse interval measurement	Counting up
0	1	1	Event counter mode	External event counter	Counting down
1	0	0	One-count mode	Delay counter/One-shot pulse output/PWM output (slave)	Counting down
1	1	0	Capture & one-count mode	Measurement of high-/low-level width of input signal	Counting up
Other than above			Setting prohibited		
The operation of each mode varies depending on MDmn0 bit (see next table).					

**Note** Bit 11 is fixed at 0 of read only, write is ignored.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-11. Format of Timer Mode Register mn (TMRmn) (4/4)

Address: F0190H, F0191H (TMR00) to F0196H, F0197H (TMR03) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 2)	CKS mn1	CKS mn0	0	CCS mn	MAST ERmn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 1, 3)	CKS mn1	CKS mn0	0	CCS mn	SPLIT mn	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TMRmn (n = 0)	CKS mn1	CKS mn0	0	CCS mn	0 <sup>Note 1</sup>	STS mn2	STS mn1	STS mn0	CIS mn1	CIS mn0	0	0	MD mn3	MD mn2	MD mn1	MD mn0

Operation mode (Value set by the MDmn3 to MDmn1 bits (see table shown in the previous page))	MD mn0	Setting of starting counting and interrupt
<ul style="list-style-type: none"> <li>Interval timer mode (0, 0, 0)</li> <li>Capture mode (0, 1, 0)</li> </ul>	0	Timer interrupt is not generated when counting is started (timer output does not change, either).
	1	Timer interrupt is generated when counting is started (timer output also changes).
<ul style="list-style-type: none"> <li>Event counter mode (0, 1, 1)</li> </ul>	0	Timer interrupt is not generated when counting is started (timer output does not change, either).
<ul style="list-style-type: none"> <li>One-count mode<sup>Note 2</sup> (1, 0, 0)</li> </ul>	0	Start trigger is invalid during counting operation. At that time, interrupt is not generated.
	1	Start trigger is valid during counting operation <sup>Note 3</sup> . At that time, interrupt is not generated.
<ul style="list-style-type: none"> <li>Capture &amp; one-count mode (1, 1, 0)</li> </ul>	0	Timer interrupt is not generated when counting is started (timer output does not change, either). Start trigger is invalid during counting operation. At that time interrupt is not generated.
Other than above		Setting prohibited

**Notes 1.** Bit 11 is fixed at 0 of read only, write is ignored.

**2.** In one-count mode, interrupt output (INTTMmn) when starting a count operation and TOMn output are not controlled.

**3.** If the start trigger (TSmn = 1) is issued during operation, the counter is initialized, and recounting is started (does not occur the interrupt request).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.4 Timer status register mn (TSRmn)

The TSRmn register indicates the overflow status of the counter of channel n.

The TSRmn register is valid only in the capture mode (MDmn3 to MDmn1 = 010B) and capture & one-count mode (MDmn3 to MDmn1 = 110B). See **Table 6-4** for the operation of the OVF bit in each operation mode and set/clear conditions.

The TSRmn register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSRmn register can be set with an 8-bit memory manipulation instruction with TSRmnL.

Reset signal generation clears this register to 0000H.

**Figure 6-12. Format of Timer Status Register mn (TSRmn)**

Address: F01A0H, F01A1H (TSR00) to F01A6H, F01A7H (TSR03) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSRmn	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OVF

OVF	Counter overflow status of channel n
0	Overflow does not occur.
1	Overflow occurs.
When OVF = 1, this flag is cleared (OVF = 0) when the next value is captured without overflow.	

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**Table 6-4. OVF Bit Operation and Set/Clear Conditions in Each Operation Mode**

Timer Operation Mode	OVF Bit	Set/Clear Conditions
• Capture mode	clear	When no overflow has occurred upon capturing
• Capture & one-count mode	set	When an overflow has occurred upon capturing
• Interval timer mode	clear	– (Use prohibited)
• Event counter mode	set	
• One-count mode		

**Remark** The OVF bit does not change immediately after the counter has overflowed, but changes upon the subsequent capture.

### 6.3.5 Timer channel enable status register m (TE<sub>m</sub>)

The TE<sub>m</sub> register is used to enable or stop the timer operation of each channel.

Each bit of the TE<sub>m</sub> register corresponds to each bit of the timer channel start register m (T<sub>Sm</sub>) and the timer channel stop register m (T<sub>Tm</sub>). When a bit of the T<sub>Sm</sub> register is set to 1, the corresponding bit of this register is set to 1. When a bit of the T<sub>Tm</sub> register is set to 1, the corresponding bit of this register is cleared to 0.

The TE<sub>m</sub> register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the TE<sub>m</sub> register can be set with a 1-bit or 8-bit memory manipulation instruction with TE<sub>mL</sub>.

Reset signal generation clears this register to 0000H.

**Figure 6-13. Format of Timer Channel Enable Status register m (TE<sub>m</sub>)**

Address: F01B0H, F01B1H (TE0) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TE <sub>m</sub>	0	0	0	0	TEH <sub>m</sub> 3	0	TEH <sub>m</sub> 1	0	0	0	0	0	TE <sub>m</sub> 3	TE <sub>m</sub> 2	TE <sub>m</sub> 1	TE <sub>m</sub> 0

TEH 03	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 3 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TEH 01	Indication of whether operation of the higher 8-bit timer is enabled or stopped when channel 1 is in the 8-bit timer mode
0	Operation is stopped.
1	Operation is enabled.

TE <sub>m</sub> <sub>n</sub>	Indication of operation enable/stop status of channel n
0	Operation is stopped.
1	Operation is enabled.
This bit displays whether operation of the lower 8-bit timer for TE <sub>m</sub> 1 and TE <sub>m</sub> 3 is enabled or stopped when channel 1 or 3 is in the 8-bit timer mode.	

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.6 Timer channel start register m (TSm)

The TSm register is a trigger register that is used to initialize timer count register mn (TCRmn) and start the counting operation of each channel.

When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is set to 1. The TSmn, TSHm1, TSHm3 bits are immediately cleared when operation is enabled (TEmn, TEHm1, TEHm3 = 1), because they are trigger bits.

The TSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TSm register can be set with a 1-bit or 8-bit memory manipulation instruction with TSmL.

Reset signal generation clears this register to 0000H.

**Figure 6-14. Format of Timer Channel Start register m (TSm)**

Address: F01B2H, F01B3H (TS0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSm	0	0	0	0	TSHm 3	0	TSHm 1	0	0	0	0	0	TSm 3	TSm 2	TSm 1	TSm 0

TSHm3	Trigger to enable operation (start operation) of the higher 8-bit timer when channel 3 is in the 8-bit timer mode
0	No trigger operation
1	The TEHm3 bit is set to 1 and the count operation becomes enabled. The TCRm3 register count operation start in the interval timer mode in the count operation enabled state (see <b>Table 6-5</b> in <b>6.5.2 Start timing of counter</b> ).

TSHm1	Trigger to enable operation (start operation) of the higher 8-bit timer when channel 1 is in the 8-bit timer mode
0	No trigger operation
1	The TEHm1 bit is set to 1 and the count operation becomes enabled. The TCRm1 register count operation start in the interval timer mode in the count operation enabled state (see <b>Table 6-5</b> in <b>6.5.2 Start timing of counter</b> ).

TSmn	Operation enable (start) trigger of channel n
0	No trigger operation
1	The TEMn bit is set to 1 and the count operation becomes enabled. The TCRmn register count operation start in the count operation enabled state varies depending on each operation mode (see <b>Table 6-5</b> in <b>6.5.2 Start timing of counter</b> ). This bit is the trigger to enable operation (start operation) of the lower 8-bit timer for TSm1 and TSm3 when channel 1 or 3 is in the 8-bit timer mode.

**Cautions** 1. Be sure to clear bits 15 to 12, 10, 8 to 4 to "0"

2. When switching from a function that does not use TImn pin input to one that does, the following wait period is required from when timer mode register mn (TMRmn) is set until the TSmn (TSHm1, TSHm3) bit is set to 1.

When the TImn pin noise filter is enabled (TNFENm = 1): Four cycles of the operation clock (f<sub>mck</sub>)

When the TImn pin noise filter is disabled (TNFENm = 0): Two cycles of the operation clock (f<sub>mck</sub>)

**Remarks** 1. When the TSm register is read, 0 is always read.

2. m: Unit number (m = 0, n: Channel number (n = 0 to 3))



### 6.3.7 Timer channel stop register m (TTm)

The TTm register is a trigger register that is used to stop the counting operation of each channel.

When a bit of this register is set to 1, the corresponding bit of timer channel enable status register m (TEm) is cleared to 0. The TTmn, TTHm1, TTHm3 bits are immediately cleared when operation is stopped (TEmn, TTHm1, TTHm3 = 0), because they are trigger bits.

The TTm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TTm register can be set with a 1-bit or 8-bit memory manipulation instruction with TTmL.

Reset signal generation clears this register to 0000H.

**Figure 6-15. Format of Timer Channel Stop register m (TTm)**

Address: F01B4H, F01B5H (TT0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TTm	0	0	0	0	TTHm 3	0	TTHm 1	0	0	0	0	0	TTm 3	TTm 2	TTm 1	TTm 0

<R>

TTHm3	Trigger to stop operation of the higher 8-bit timer when channel 3 is in the 8-bit timer mode
0	No trigger operation
1	TEHm3 bit is cleared to 0 and the count operation is stopped.

<R>

TTHm1	Trigger to stop operation of the higher 8-bit timer when channel 1 is in the 8-bit timer mode
0	No trigger operation
1	TEHm1 bit is cleared to 0 and the count operation is stopped.

<R>

TTm n	Operation stop trigger of channel n
0	TEmn bit is cleared to 0 and the count operation is stopped.
1	TEmn bit clear to 0, to be count operation stop enable status. This bit is the trigger to stop operation of the lower 8-bit timer for TTm1 and TTm3 when channel 1 or 3 is in the 8-bit timer mode.

**Caution** Be sure to clear bits 15 to 12, 10, 8 to 4 to "0".

**Remarks 1.** When the TTm register is read, 0 is always read.

**2.** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.8 Timer input select register 0 (TIS0)

The TIS0 register is used to select the channel 1 of unit 0 timer input.

The TIS0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 6-16. Format of Timer Input Select register 0 (TIS0)**

Address: F0074H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
TIS0	0	0	0	0	0	TIS02	TIS01	TIS00

TIS02	TIS01	TIS00	Selection of timer input used with channel 1
0	0	0	Input signal of timer input pin (TI01)
0	0	1	
0	1	0	
0	1	1	
1	0	0	Low-speed on-chip oscillator clock (f <sub>IL</sub> )
Other than above			Setting prohibited

**Caution** High-level width, low-level width of timer input is selected, will require more than  $1/f_{MCK} + 10$  ns.

**6.3.9 Timer output enable register m (TOEm)**

The TOEm register is used to enable or disable timer output of each channel.

Channel n for which timer output has been enabled becomes unable to rewrite the value of the TOmn bit of timer output register m (TOM) described later by software, and the value reflecting the setting of the timer output function through the count operation is output from the timer output pin (TOMn).

The TOEm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with TOEmL. Reset signal generation clears this register to 0000H.

**Figure 6-17. Format of Timer Output Enable register m (TOEm)**

Address: F01BAH, F01BBH (TOE0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOEm	0	0	0	0	0	0	0	0	0	0	0	0	TOE m3	TOE m2	TOE m1	TOE m0

TOE mn	Timer output enable/disable of channel n
0	Disable output of timer. Without reflecting on TOMn bit timer operation, to fixed the output. Writing to the TOMn bit is enabled and the level set in the TOMn bit is output from the TOMn pin.
1	Enable output of timer. Reflected in the TOMn bit timer operation, to generate the output waveform. Writing to the TOMn bit is disabled (writing is ignored).

<R>

**Caution** Be sure to clear bits 15 to 4 to “0”.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.10 Timer output register m (TOM)

The TOM register is a buffer register of timer output of each channel.

The value of each bit in this register is output from the timer output pin (TOMn) of each channel.

The TOMn bit of this register can be rewritten by software only when timer output is disabled (TOEmn = 0). When timer output is enabled (TOEmn = 1), rewriting this register by software is ignored, and the value is changed only by the timer operation.

To use the P13/TI00/TO00, P16/TI01/TO01/INTP5, P33/TI02/TO02/SSI00, or P12/TI03/TO03/INTP4/PCLBUZ0 pin as a port function pin, set the corresponding TOMn bit to "0".

The TOM register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOM register can be set with an 8-bit memory manipulation instruction with TOML.

Reset signal generation clears this register to 0000H.

**Figure 6-18. Format of Timer Output register m (TOM)**

Address: F01B8H, F01B9H (TO0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOM	0	0	0	0	0	0	0	0	0	0	0	0	TOM <sub>3</sub>	TOM <sub>2</sub>	TOM <sub>1</sub>	TOM <sub>0</sub>

TOM <sub>n</sub>	Timer output of channel n
0	Timer output value is "0".
1	Timer output value is "1".

**Caution** Be sure to clear bits 15 to 4 to "0".

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.11 Timer output level register m (TOLm)

The TOLm register is a register that controls the timer output level of each channel.

The setting of the inverted output of channel n by this register is reflected at the timing of set or reset of the timer output signal while the timer output is enabled (TOEmn = 1) in the Slave channel output mode (TOMmn = 1). In the master channel output mode (TOMmn = 0), this register setting is invalid.

The TOLm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOLm register can be set with an 8-bit memory manipulation instruction with TOLmL.

Reset signal generation clears this register to 0000H.

**Figure 6-19. Format of Timer Output Level register m (TOLm)**

Address: F01BCH, F01BDH (TOL0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOLm	0	0	0	0	0	0	0	0	0	0	0	0	TOL m3	TOL m2	TOL m1	0

TOL mn	Control of timer output level of channel n														
0	Positive logic output (active-high)														
1	Negative logic output (active-low)														

**Caution** Be sure to clear bits 15 to 4, and 0 to “0”.

- Remarks 1.** If the value of this register is rewritten during timer operation, the timer output logic is inverted when the timer output signal changes next, instead of immediately after the register value is rewritten.
- 2.** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.3.12 Timer output mode register m (TOMm)

The TOMm register is used to control the timer output mode of each channel.

When a channel is used for the independent channel operation function, set the corresponding bit of the channel to be used to 0.

When a channel is used for the simultaneous channel operation function (PWM output, one-shot pulse output, or multiple PWM output), set the corresponding bit of the master channel to 0 and the corresponding bit of the slave channel to 1.

The setting of each channel n by this register is reflected at the timing when the timer output signal is set or reset while the timer output is enabled ( $TOEmn = 1$ ).

The TOMm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the TOMm register can be set with an 8-bit memory manipulation instruction with TOMmL.

Reset signal generation clears this register to 0000H.

**Figure 6-20. Format of Timer Output Mode register m (TOMm)**

Address: F01BEH, F01BFH (TOM0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOMm	0	0	0	0	0	0	0	0	0	0	0	0	TOM m3	TOM m2	TOM m1	0

TOM mn	Control of timer output mode of channel n														
0	Master channel output mode (to produce toggle output by timer interrupt request signal (INTTMmn))														
1	Slave channel output mode (output is set by the timer interrupt request signal (INTTMmn) of the master channel, and reset by the timer interrupt request signal (INTTM0p) of the slave channel)														

**Caution** Be sure to clear bits 15 to 4, and 0 to “0”.

**Remark** m: Unit number (m = 0)

n: Channel number

n = 0 to 3 (n = 0, 2 for master channel)

p: Slave channel number

$n < p \leq 3$

(For details of the relation between the master channel and slave channel, see **6.4.1 Basic rules of simultaneous channel operation function**.)

### 6.3.13 Noise filter enable register 1 (NFEN1)

The NFEN1 register is used to set whether the noise filter can be used for the timer input signal to each channel.

Enable the noise filter by setting the corresponding bits to 1 on the pins in need of noise removal.

When the noise filter is ON, match detection and synchronization of the 2 clocks is performed with the CPU/peripheral hardware clock ( $f_{MCK}$ ). When the noise filter is OFF, only synchronization is performed with the CPU/peripheral hardware clock ( $f_{MCK}$ ).<sup>Note</sup>

The NFEN1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Note** For details, see **6.5.1 (2) When valid edge of input signal input from the TImn pin is selected (CCSmn = 1)** and **6.5.2 Start timing of counter**.

**Figure 6-21. Format of Noise Filter Enable Register 1 (NFEN1)**

Address: F0071H		After reset: 00H		R/W				
Symbol	7	6	5	4	3	2	1	0
NFEN1	0	0	0	0	TNFEN03	TNFEN02	TNFEN01	TNFEN00
<R>	TNFEN03		Enable/disable using noise filter of TI03 pin input signal					
	0	Noise filter OFF						
	1	Noise filter ON						
<R>	TNFEN02		Enable/disable using noise filter of TI02 pin input signal					
	0	Noise filter OFF						
	1	Noise filter ON						
<R>	TNFEN01		Enable/disable using noise filter of TI01 pin input signal					
	0	Noise filter OFF						
	1	Noise filter ON						
<R>	TNFEN00		Enable/disable using noise filter of TI00 pin input signal					
	0	Noise filter OFF						
	1	Noise filter ON						

### 6.3.14 Port mode registers 1, 3 (PM1, PM3)

These registers set input/output of ports 1 and 3 in 1-bit units.

When using the ports (such as P13/TO00/TI00, and P33/TO02/TI02/ $\overline{\text{SSI00}}$ ) to be shared with the timer output pin for timer output, set the port mode register (PMxx) bit and port register (Pxx) bit corresponding to each port to 0.

Example: When using P33/TO02/TI02/ $\overline{\text{SSI00}}$  for timer output

Set the PM33 bit of port mode register 3 to "0".

Set the P33 bit of port register 3 to "0".

When using the ports (such as P13/TO00/TI00, P33/TO02/TI02/ $\overline{\text{SSI00}}$ ) to be shared with the timer input pin for timer input, set the port mode register (PMxx) bit corresponding to each port to 1. At this time, the port register (Pxx) bit may be 0 or 1.

Example: When using P33/TO02/TI02/ $\overline{\text{SSI00}}$  for timer input

Set the PM33 bit of port mode register 3 to "1".

Set the P33 bit of port register 3 to "0" or "1".

The PM1 and PM3 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Figure 6-22. Format of Port Mode Registers 1, 3 (PM1, PM3) (32-pin products)**

Address: FFF21H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM1	PM17	PM16	PM15	PM14	PM13	PM12	PM11	PM10

Address: FFF23H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM3	1	1	PM35	PM34	PM33	PM32	PM31	PM30

PMmn	Pmn pin I/O mode selection (m = 1, 3; n = 0 to 7)
0	Output mode (output buffer on)
1	Input mode (output buffer off)



## 6.4 Basic Rules of Timer Array Unit

### 6.4.1 Basic rules of simultaneous channel operation function

When simultaneously using multiple channels, namely, a combination of a master channel (a reference timer mainly counting the cycle) and slave channels (timers operating according to the master channel), the following rules apply.

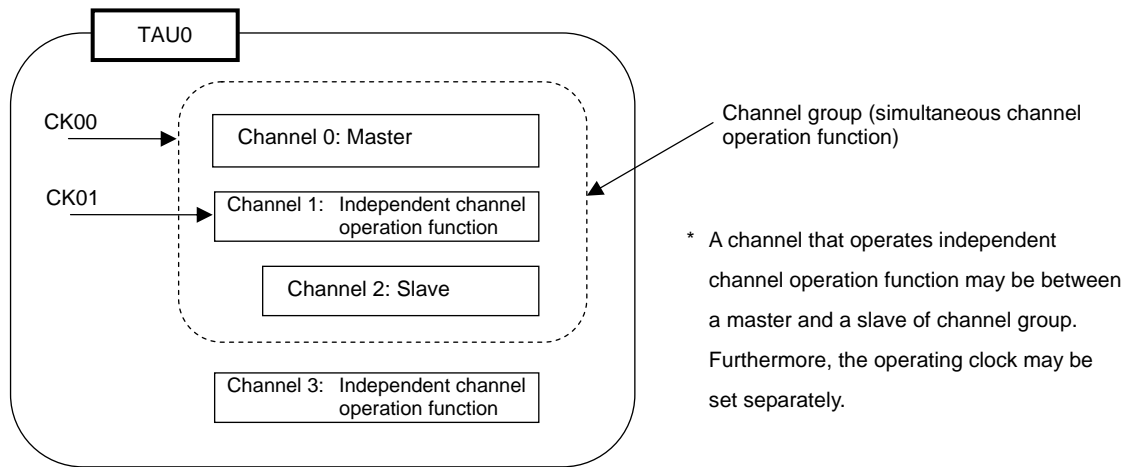
- (1) Only an even channel (channel 0, 2, etc.) can be set as a master channel.
- (2) Any channel, except channel 0, can be set as a slave channel.
- (3) The slave channel must be lower than the master channel.  
Example: If channel 2 is set as a master channel, channel 3 can be set as a slave channel.
- (4) Two or more slave channels can be set for one master channel.
- (5) When two or more master channels are to be used, slave channels with a master channel between them may not be set.  
Example: If channel 0 is set as a master channel, channels 1 to 3 can be set as the slave channels of master channel 0.
- (6) The operating clock for a slave channel in combination with a master channel must be the same as that of the master channel. The CKSmn0, CKSmn1 bits (bit 15, 14 of timer mode register mn (TMRmn)) of the slave channel that operates in combination with the master channel must be the same value as that of the master channel.
- (7) A master channel can transmit INTTMmn (interrupt), start software trigger, and count clock to the lower channels.
- (8) A slave channel can use INTTMmn (interrupt), a start software trigger, or the count clock of the master channel as a source clock, but cannot transmit its own INTTMmn (interrupt), start software trigger, or count clock to channels with lower channel numbers.
- (9) A master channel cannot use INTTMmn (interrupt), a start software trigger, or the count clock from the other higher master channel as a source clock.
- (10) To simultaneously start channels that operate in combination, the channel start trigger bit (TSmn) of the channels in combination must be set at the same time.
- (11) During the counting operation, a TSmn bit of a master channel or TSmn bits of all channels which are operating simultaneously can be set. It cannot be applied to TSmn bits of slave channels alone.
- (12) To stop the channels in combination simultaneously, the channel stop trigger bit (TTmn) of the channels in combination must be set at the same time.
- (13) CKm2/CKm3 cannot be selected while channels are operating simultaneously, because the operating clocks of master channels and slave channels have to be synchronized.
- (14) Timer mode register m0 (TMRm0) has no master bit (it is fixed as "0"). However, as channel 0 is the highest channel, it can be used as a master channel during simultaneous operation.

The rules of the simultaneous channel operation function are applied in a channel group (a master channel and slave channels forming one simultaneous channel operation function).

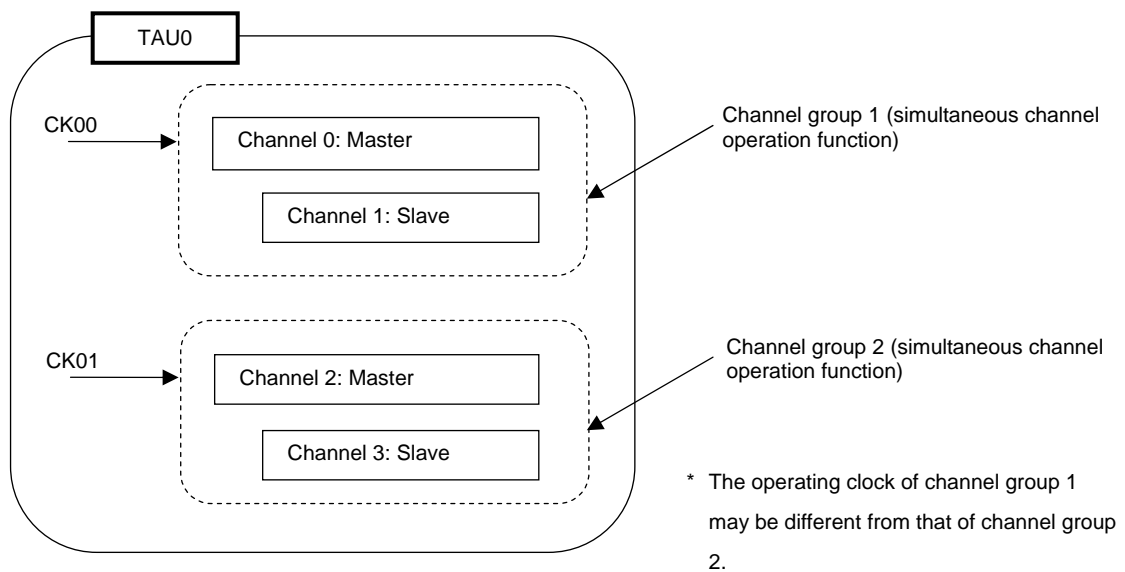
If two or more channel groups that do not operate in combination are specified, the basic rules of the simultaneous channel operation function in **6.4.1 Basic rules of simultaneous channel operation function** do not apply to the channel groups.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Example 1



Example 2



### 6.4.2 Basic rules of 8-bit timer operation function (channels 1 and 3 only)

The 8-bit timer operation function makes it possible to use a 16-bit timer channel in a configuration consisting of two 8-bit timer channels.

This function can only be used for channels 1 and 3, and there are several rules for using it.

The basic rules for this function are as follows:

- (1) The 8-bit timer operation function applies only to channels 1 and 3.
- (2) When using 8-bit timers, set the SPLIT bit of timer mode register mn (TMRmn) to 1.
- (3) The higher 8 bits can be operated as the interval timer function.
- (4) At the start of operation, the higher 8 bits output INTTm1H/INTTm3H (an interrupt) (which is the same operation performed when MDmn0 is set to 1).
- (5) The operation clock of the higher 8 bits is selected according to the CKSmn1 and CKSmn0 bits of the lower-bit TMRmn register.
- (6) For the higher 8 bits, the TSHm1/TSHm3 bit is manipulated to start channel operation and the TTHm1/TTHm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEHm1/TEHm3 bit.
- (7) The lower 8 bits operate according to the TMRmn register settings. The following three functions support operation of the lower 8 bits:
  - Interval timer function
  - External event counter function
  - Delay count function
- (8) For the lower 8 bits, the TSm1/TSm3 bit is manipulated to start channel operation and the TTm1/TTm3 bit is manipulated to stop channel operation. The channel status can be checked using the TEM1/TEm3 bit.
- (9) During 16-bit operation, manipulating the TSHm1, TSHm3, TTHm1, and TTHm3 bits is invalid. The TSm1, TSm3, TTm1, and TTm3 bits are manipulated to operate channels 1 and 3. The TEHm3 and TEHm1 bits are not changed.
- (10) For the 8-bit timer function, the simultaneous operation functions (one-shot pulse, PWM, and multiple PWM) cannot be used.

**Remark** m: Unit number (m = 0), n: Channel number (n = 1, 3)

## 6.5 Operation of Counter

### 6.5.1 Count clock ( $f_{TCLK}$ )

The count clock ( $f_{TCLK}$ ) of the timer array unit can be selected between following by CCSmn bit of timer mode register mn (TMRmn).

- Operation clock ( $f_{MCK}$ ) specified by the CKSmn0 and CKSmn1 bits
- Valid edge of input signal input from the TImn pin

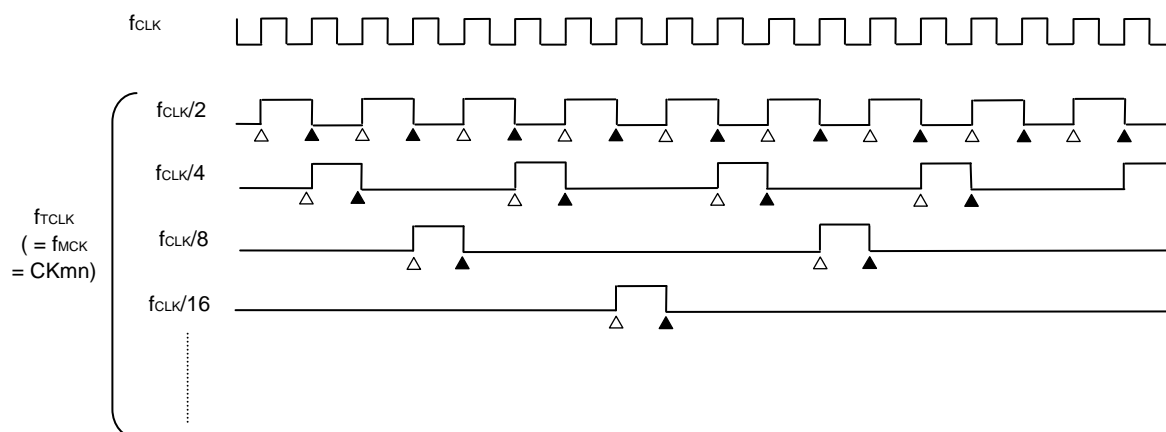
Because the timer array unit is designed to operate in synchronization with  $f_{CLK}$ , the timings of the count clock ( $f_{TCLK}$ ) are shown below.

#### (1) When operation clock ( $f_{MCK}$ ) specified by the CKSmn0 and CKSmn1 bits is selected (CCSmn = 0)

The count clock ( $f_{TCLK}$ ) is between  $f_{CLK}$  to  $f_{CLK}/2^{15}$  by setting of timer clock select register m (TPSm). When a divided  $f_{CLK}$  is selected, however, the clock selected in TPSmn register, but a signal which becomes high level for one period of  $f_{CLK}$  from its rising edge. When a  $f_{CLK}$  is selected, fixed to high level

Counting of timer count register mn (TCRmn) delayed by one period of  $f_{CLK}$  from rising edge of the count clock, because of synchronization with  $f_{CLK}$ . But, this is described as “counting at rising edge of the count clock”, as a matter of convenience.

Figure 6-23. Timing of  $f_{CLK}$  and Count Clock ( $f_{TCLK}$ ) (When Ccsmn = 0)



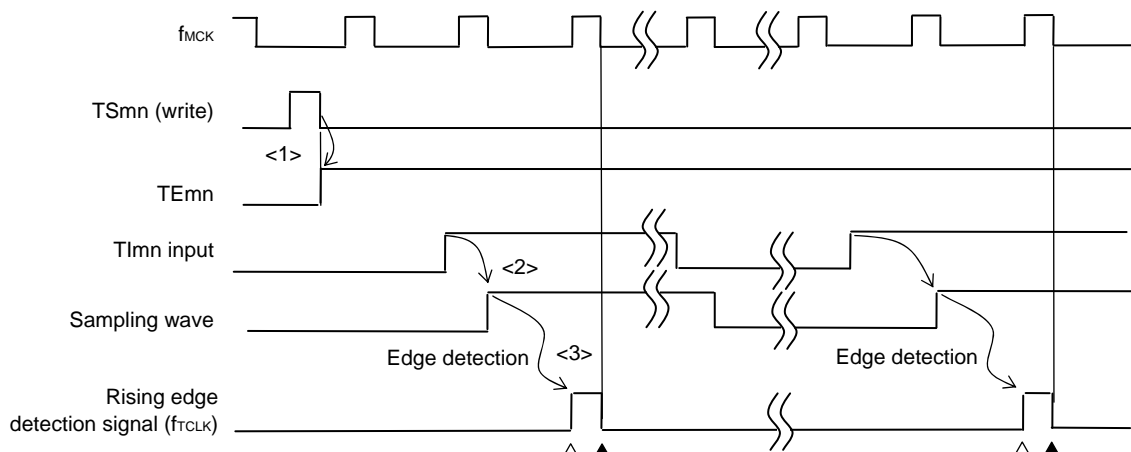
- Remarks 1.** Δ : Rising edge of the count clock  
 ▲ : Synchronization, increment/decrement of counter
- 2.**  $f_{CLK}$ : CPU/peripheral hardware clock

**(2) When valid edge of input signal via the TImn pin is selected (CCSmn = 1)**

The count clock ( $f_{TCLK}$ ) becomes the signal that detects valid edge of input signal via the TImn pin and synchronizes next rising  $f_{MCK}$ . The count clock ( $f_{TCLK}$ ) is delayed for 1 to 2 period of  $f_{MCK}$  from the input signal via the TImn pin (when a noise filter is used, the delay becomes 3 to 4 clock).

Counting of timer count register mn (TCRmn) delayed by one period of  $f_{CLK}$  from rising edge of the count clock, because of synchronization with  $f_{CLK}$ . But, this is described as “counting at valid edge of input signal via the TImn pin”, as a matter of convenience.

**Figure 6-24. Timing of  $f_{CLK}$  and Count Clock ( $f_{TCLK}$ ) (When CCSmn = 1, Noise Filter Unused)**



<1> Setting TSmn bit to 1 enables the timer to be started and to become wait state for valid edge of input signal via the TImn pin.

<2> The rise of input signal via the TImn pin is sampled by  $f_{MCK}$ .

<3> The edge is detected by the rising of the sampled signal and the detection signal (count clock) is output.

**Remarks 1.** Δ : Rising edge of the count clock

▲ : Synchronization, increment/decrement of counter

**2.**  $f_{CLK}$ : CPU/peripheral hardware clock

$f_{MCK}$ : Operation clock of channel n

**3.** The waveform of the input signal via TImn pin of the input pulse interval measurement, the measurement of high/low width of input signal, and the delay counter, the one-shot pulse output are the same as that shown in Figure 6-23.

### 6.5.2 Start timing of counter

Timer count register mn (TCRmn) becomes enabled to operation by setting of TSmn bit of timer channel start register m (TSm).

Operations from count operation enabled state to timer count Register mn (TCRmn) count start is shown in Table 6-5.

**Table 6-5. Operations from Count Operation Enabled State to Timer count Register mn (TCRmn) Count Start**

Timer Operation Mode	Operation When TSmn = 1 Is Set
<ul style="list-style-type: none"> <li>Interval timer mode</li> </ul>	<p>No operation is carried out from start trigger detection (TSmn=1) until count clock generation.</p> <p>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see <b>6.5.3 (1) Operation of interval timer mode</b>).</p>
<ul style="list-style-type: none"> <li>Event counter mode</li> </ul>	<p>Writing 1 to the TSmn bit loads the value of the TDRmn register to the TCRmn register.</p> <p>If detect edge of TImn input. The subsequent count clock performs count down operation (see <b>6.5.3 (2) Operation of event counter mode</b>).</p>
<ul style="list-style-type: none"> <li>Capture mode</li> </ul>	<p>No operation is carried out from start trigger detection (TSmn = 1) until count clock generation.</p> <p>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see <b>6.5.3 (3) Operation of capture mode (input pulse interval measurement)</b>).</p>
<ul style="list-style-type: none"> <li>One-count mode</li> </ul>	<p>The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0).</p> <p>No operation is carried out from start trigger detection until count clock generation.</p> <p>The first count clock loads the value of the TDRmn register to the TCRmn register and the subsequent count clock performs count down operation (see <b>6.5.3 (4) Operation of one-count mode</b>).</p>
<ul style="list-style-type: none"> <li>Capture &amp; one-count mode</li> </ul>	<p>The waiting-for-start-trigger state is entered by writing 1 to the TSmn bit while the timer is stopped (TEmn = 0).</p> <p>No operation is carried out from start trigger detection until count clock generation.</p> <p>The first count clock loads 0000H to the TCRmn register and the subsequent count clock performs count up operation (see <b>6.5.3 (5) Operation of capture &amp; one-count mode (high-level interval measurement)</b>).</p>

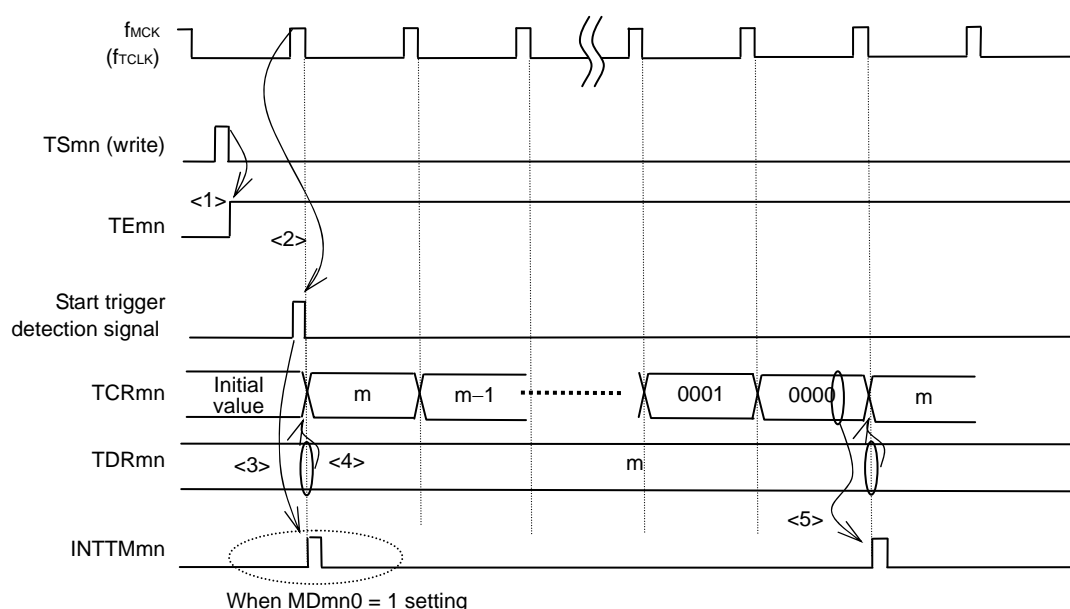
### 6.5.3 Operation of counter

Here, the counter operation in each mode is explained.

#### (1) Operation of interval timer mode

- <1> Operation is enabled ( $TE_{mn} = 1$ ) by writing 1 to the  $TS_{mn}$  bit. Timer count register  $mn$  ( $TCR_{mn}$ ) holds the initial value until count clock generation.
- <2> A start trigger is generated at the first count clock after operation is enabled.
- <3> When the  $MD_{mn0}$  bit is set to 1,  $INTTM_{mn}$  is generated by the start trigger.
- <4> By the first count clock after the operation enable, the value of timer data register  $mn$  ( $TDR_{mn}$ ) is loaded to the  $TCR_{mn}$  register and counting starts in the interval timer mode.
- <5> When the  $TCR_{mn}$  register counts down and its count value is 0000H,  $INTTM_{mn}$  is generated and the value of timer data register  $mn$  ( $TDR_{mn}$ ) is loaded to the  $TCR_{mn}$  register and counting keeps on.

Figure 6-25. Operation Timing (In Interval Timer Mode)



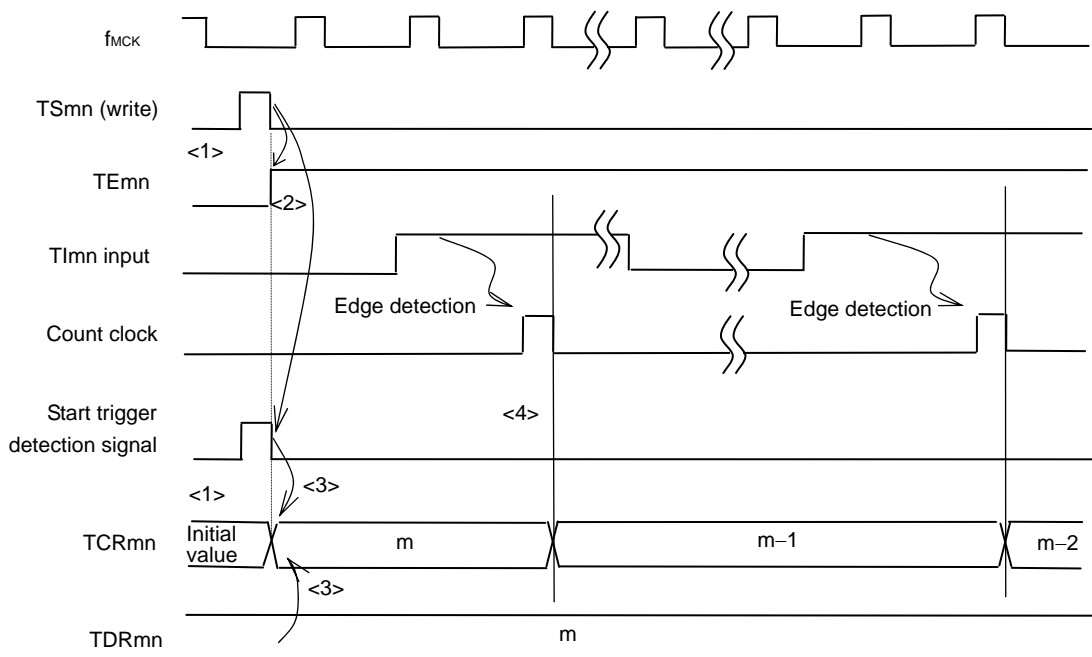
**Caution** In the first cycle operation of count clock after writing the  $TS_{mn}$  bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting  $MD_{mn0} = 1$ .

**Remark**  $f_{MCK}$ , the start trigger detection signal, and  $INTTM_{mn}$  become active between one clock in synchronization with  $f_{CLK}$ .

(2) Operation of event counter mode

- <1> Timer count register mn (TCRmn) holds its initial value while operation is stopped (TEmn = 0).
- <2> Operation is enabled (TEmn = 1) by writing 1 to the TSmn bit.
- <3> As soon as 1 has been written to the TSmn bit and 1 has been set to the TEmn bit, the value of timer data register mn (TDRmn) is loaded to the TCRmn register to start counting.
- <4> After that, the TCRmn register value is counted down according to the count clock of the valid edge of the TImn input.

Figure 6-26. Operation Timing (In Event Counter Mode)

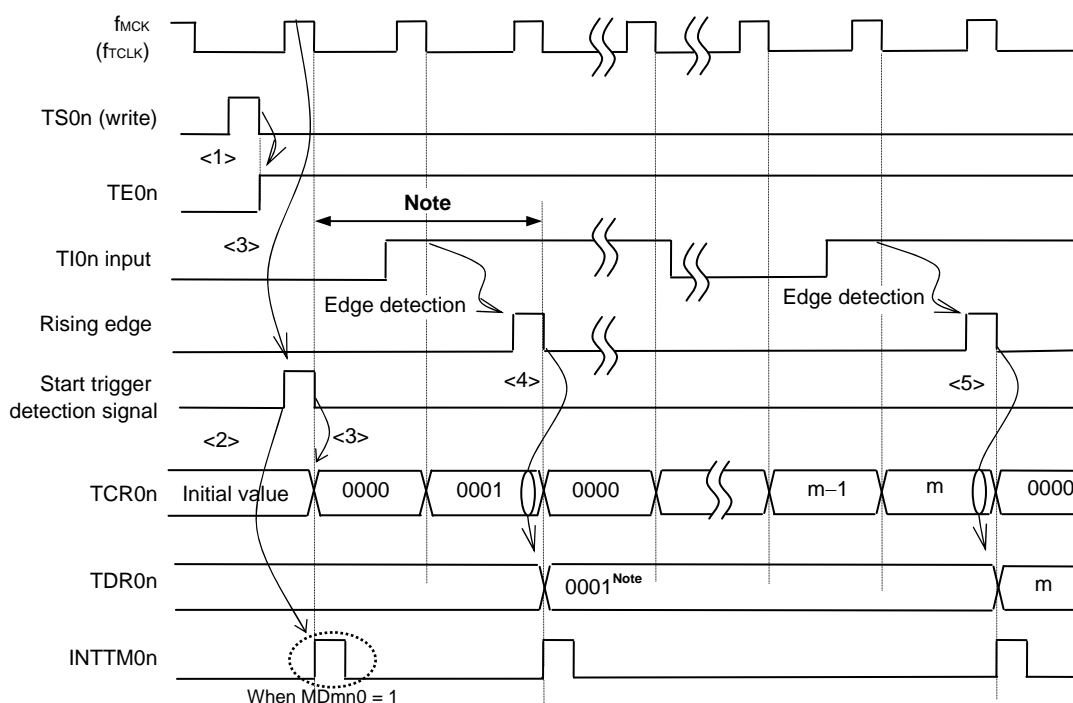


<R> **Remark** The timing is shown in Figure 6-25 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes 2 f<sub>MCK</sub> cycles (it sums up to 3 to 4 cycles) later than the normal cycle of TImn input. The error per one period occurs be the asynchronous between the period of the TImn input and that of the count clock (f<sub>MCK</sub>).



**(3) Operation of capture mode (input pulse interval measurement)**

- <1> Operation is enabled ( $TE_{mn} = 1$ ) by writing 1 to the  $TS_{mn}$  bit.
- <2> Timer count register  $mn$  ( $TCR_{mn}$ ) holds the initial value until count clock generation.
- <3> A start trigger is generated at the first count clock after operation is enabled. And the value of 0000H is loaded to the  $TCR_{mn}$  register and counting starts in the capture mode. (When the  $MD_{mn0}$  bit is set to 1,  $INTTM_{mn}$  is generated by the start trigger.)
- <4> On detection of the valid edge of the  $TI_{mn}$  input, the value of the  $TCR_{mn}$  register is captured to timer data register  $mn$  ( $TDR_{mn}$ ) and  $INTTM_{mn}$  is generated. However, this capture value is nomenaing. The  $TCR_{mn}$  register keeps on counting from 0000H.
- <5> On next detection of the valid edge of the  $TI_{mn}$  input, the value of the  $TCR_{mn}$  register is captured to timer data register  $mn$  ( $TDR_{mn}$ ) and  $INTTM_{mn}$  is generated.

**Figure 6-27. Operation Timing (In Capture Mode: Input Pulse Interval Measurement)**

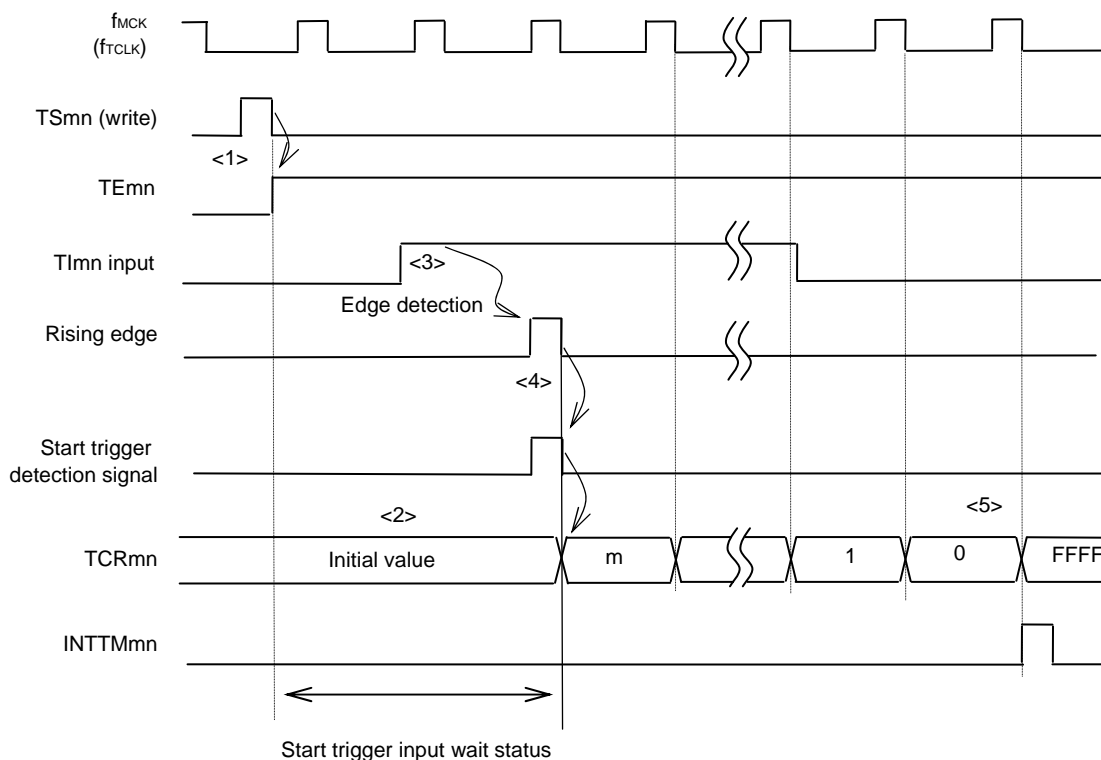
**Note** If a clock has been input to  $TI_{mn}$  (the trigger exists) when capturing starts, counting starts when a trigger is detected, even if no edge is detected. Therefore, the first captured value (<4>) does not determine a pulse interval (in the above figure, 0001 just indicates two clock cycles but does not determine the pulse interval) and so the user can ignore it.

**Caution** In the first cycle operation of count clock after writing the  $TS_{mn}$  bit, an error at a maximum of one clock is generated since count start delays until count clock has been generated. When the information on count start timing is necessary, an interrupt can be generated at count start by setting  $MD_{mn0} = 1$ .

**Remark** The timing is shown in Figure 6-26 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes  $2 f_{MCK}$  cycles (it sums up to 3 to 4 cycles) later than the normal cycle of  $TI_{mn}$  input. The error per one period occurs be the asynchronous between the period of the  $TI_{mn}$  input and that of the count clock ( $f_{MCK}$ ).

**(4) Operation of one-count mode**

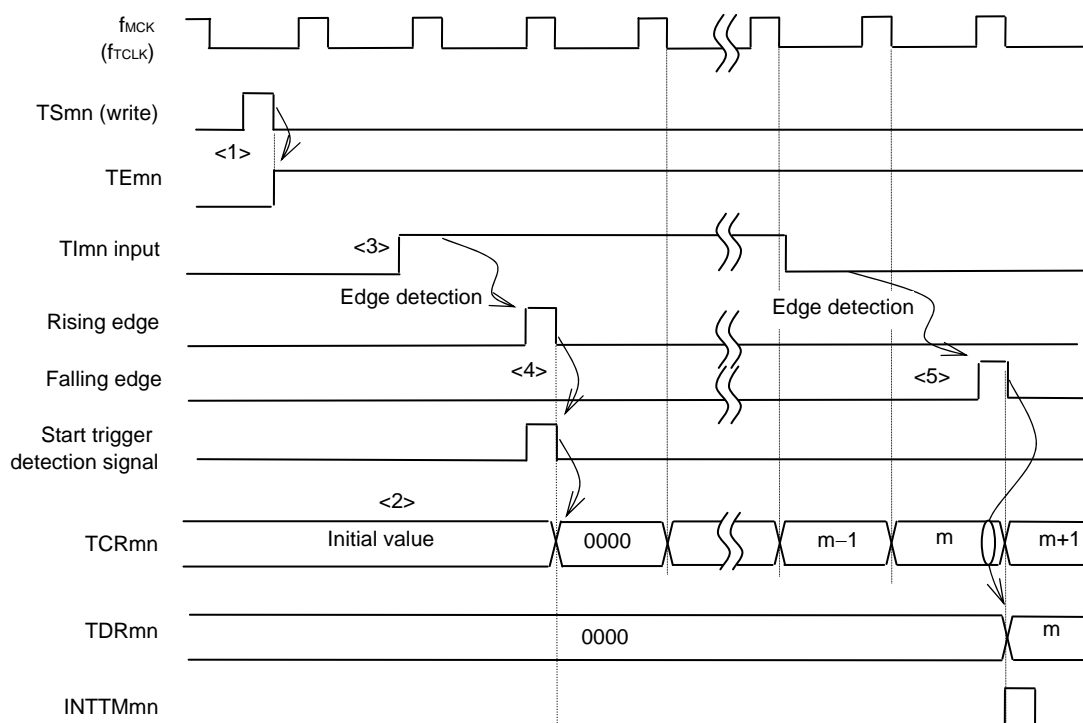
- <1> Operation is enabled ( $TE_{mn} = 1$ ) by writing 1 to the  $TS_{mn}$  bit.
- <2> Timer count register  $mn$  ( $TCR_{mn}$ ) holds the initial value until start trigger generation.
- <3> Rising edge of the  $TI_{mn}$  input is detected.
- <4> On start trigger detection, the value of timer data register  $mn$  ( $TDR_{mn}$ ) is loaded to the  $TCR_{mn}$  register and count starts.
- <5> When the  $TCR_{mn}$  register counts down and its count value is 0000H,  $INTTM_{mn}$  is generated and the value of the  $TCR_{mn}$  register becomes FFFFH and counting stops.

**Figure 6-28. Operation Timing (In One-count Mode)**

**Remark** The timing is shown in Figure 6-27 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes  $2 f_{MCK}$  cycles (it sums up to 3 to 4 cycles) later than the normal cycle of  $TI_{mn}$  input. The error per one period occurs be the asynchronous between the period of the  $TI_{mn}$  input and that of the count clock ( $f_{MCK}$ ).

**(5) Operation of capture & one-count mode (high-level width measurement)**

- <1> Operation is enabled ( $TE_{mn} = 1$ ) by writing 1 to the  $TS_{mn}$  bit of timer channel start register  $m$  ( $TS_{mn}$ ).
- <2> Timer count register  $m$  ( $TCR_{mn}$ ) holds the initial value until start trigger generation.
- <3> Rising edge of the  $TI_{mn}$  input is detected.
- <4> On start trigger detection, the value of 0000H is loaded to the  $TCR_{mn}$  register and count starts.
- <5> On detection of the falling edge of the  $TI_{mn}$  input, the value of the  $TCR_{mn}$  register is captured to timer data register  $m$  ( $TDR_{mn}$ ) and  $INTT_{mn}$  is generated.

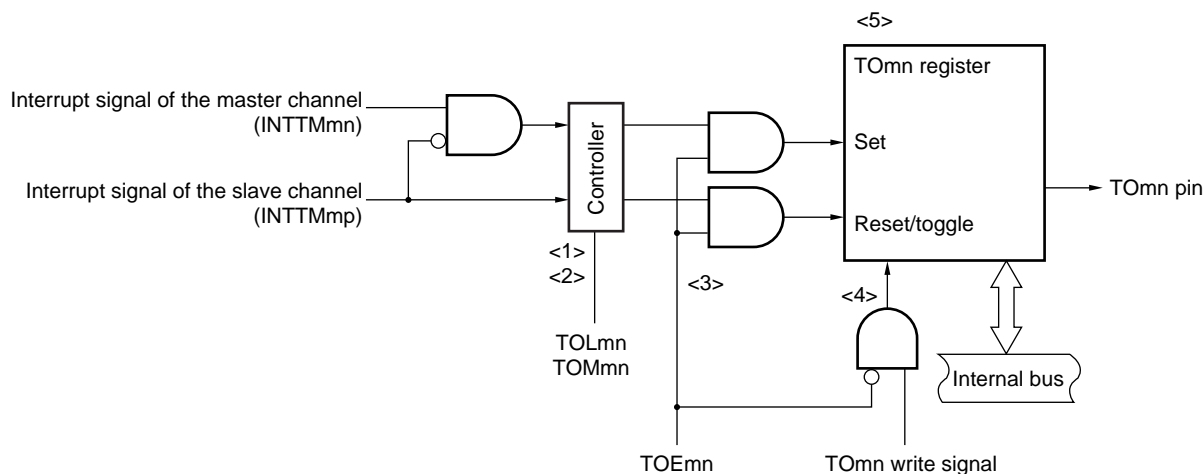
**Figure 6-29. Operation Timing (In Capture & One-count Mode : High-level Width Measurement)**

**Remark** The timing is shown in Figure 6-28 indicates while the noise filter is not used. By making the noise filter on-state, the edge detection becomes  $2 f_{MCK}$  cycles (it sums up to 3 to 4 cycles) later than the normal cycle of  $TI_{mn}$  input. The error per one period occurs because of the asynchronous between the period of the  $TI_{mn}$  input and that of the count clock ( $f_{MCK}$ ).

## 6.6 Channel Output (TOMn pin) Control

### 6.6.1 TOMn pin output circuit configuration

Figure 6-30. Output Circuit Configuration



The following describes the TOMn pin output circuit.

- <1> When  $TOMmn = 0$  (master channel output mode), the set value of timer output level register  $m$  ( $TOLm$ ) is ignored and only  $INTTM0p$  (slave channel timer interrupt) is transmitted to timer output register  $m$  ( $TOM$ ).
- <2> When  $TOMmn = 1$  (slave channel output mode), both  $INTTMmn$  (master channel timer interrupt) and  $INTTM0p$  (slave channel timer interrupt) are transmitted to the  $TOM$  register. At this time, the  $TOLm$  register becomes valid and the signals are controlled as follows:

When  $TOLmn = 0$ : Positive logic output ( $INTTMmn \rightarrow \text{set}$ ,  $INTTMmp \rightarrow \text{reset}$ )  
 When  $TOLmn = 1$ : Negative logic output ( $INTTMmn \rightarrow \text{reset}$ ,  $INTTMmp \rightarrow \text{set}$ )

When  $INTTMmn$  and  $INTTM0p$  are simultaneously generated, (0% output of PWM),  $INTTM0p$  (reset signal) takes priority, and  $INTTMmn$  (set signal) is masked.

- <3> While timer output is enabled ( $TOEmn = 1$ ),  $INTTMmn$  (master channel timer interrupt) and  $INTTM0p$  (slave channel timer interrupt) are transmitted to the  $TOM$  register. Writing to the  $TOM$  register ( $TOMn$  write signal) becomes invalid. When  $TOEmn = 1$ , the  $TOMn$  pin output never changes with signals other than interrupt signals. To initialize the  $TOMn$  pin output level, it is necessary to set timer operation is stopeed ( $TOEmn = 0$ ) and to write a value to the  $TOM$  register.
- <4> While timer output is disabled ( $TOEmn = 0$ ), writing to the  $TOMn$  bit to the target channel ( $TOMn$  write signal) becomes valid. When timer output is disabled ( $TOEmn = 0$ ), neither  $INTTMmn$  (master channel timer interrupt) nor  $INTTM0p$  (slave channel timer interrupt) is transmitted to the  $TOM$  register.
- <5> The  $TOM$  register can always be read, and the  $TOMn$  pin output level can be checked.

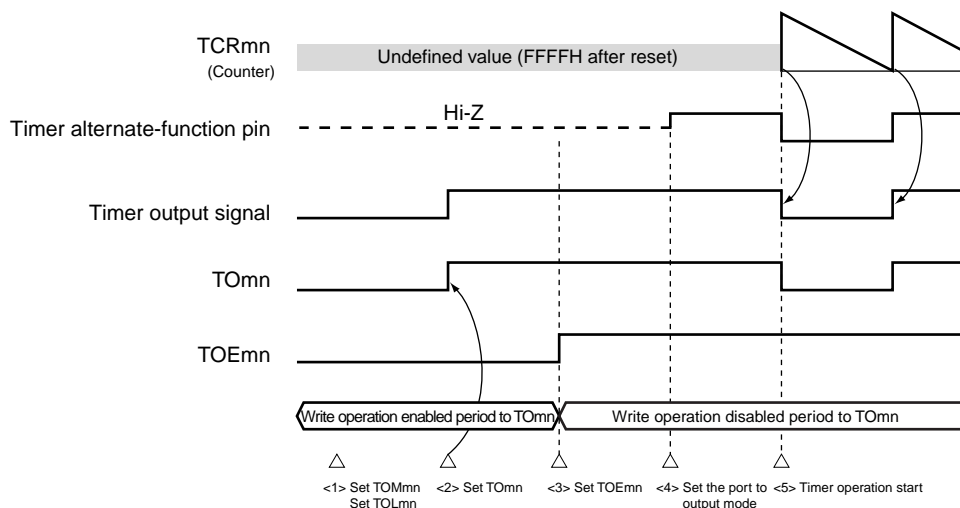
**Remark**  $m$ : Unit number ( $m = 0$ )  
 $n$ : Channel number  
 $n = 0$  to 3 ( $n = 0, 2$  for master channel)  
 $p$ : Slave channel number  
 $n = 0$ :  $p = 1, 2, 3$   
 $n = 2$ :  $p = 3$

<R>

### 6.6.2 TOmn pin output setting

The following figure shows the procedure and status transition of the TOmn output pin from initial setting to timer operation start.

**Figure 6-31. Status Transition from Timer Output Setting to Operation Start**



<1> The operation mode of timer output is set.

- TOMmn bit (0: Master channel output mode, 1: Slave channel output mode)
- TOLmn bit (0: Positive logic output, 1: Negative logic output)

<2> The timer output signal is set to the initial status by setting timer output register m (TOm).

<3> The timer output operation is enabled by writing 1 to the TOEmn bit (writing to the TOm register is disabled).

<4> The port I/O setting is set to output (see **6.3.14 Port mode registers 1, 3 (PM1, PM3)**).

<5> The timer operation is enabled (TSmn = 1).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.6.3 Cautions on channel output operation

#### (1) Changing values set in the registers TOM, TOEm, and TOLm during timer operation

Since the timer operations (operations of timer count register mn (TCRmn) and timer data register mn (TDRmn)) are independent of the TOMn output circuit and changing the values set in timer output register m (TOM), timer output enable register m (TOEm), and timer output level register m (TOLm) does not affect the timer operation, the values can be changed during timer operation. To output an expected waveform from the TOMn pin by timer operation, however, set the TOM, TOEm, TOLm, and TOMm registers to the values stated in the register setting example of each operation shown by 6.7 and 6.8.

When the values set to the TOEm, and TOMm registers (but not the TOM register) are changed close to the occurrence of the timer interrupt (INTTMmn) of each channel, the waveform output to the TOMn pin might differ, depending on whether the values are changed immediately before or immediately after the timer interrupt (INTTMmn) occurs.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

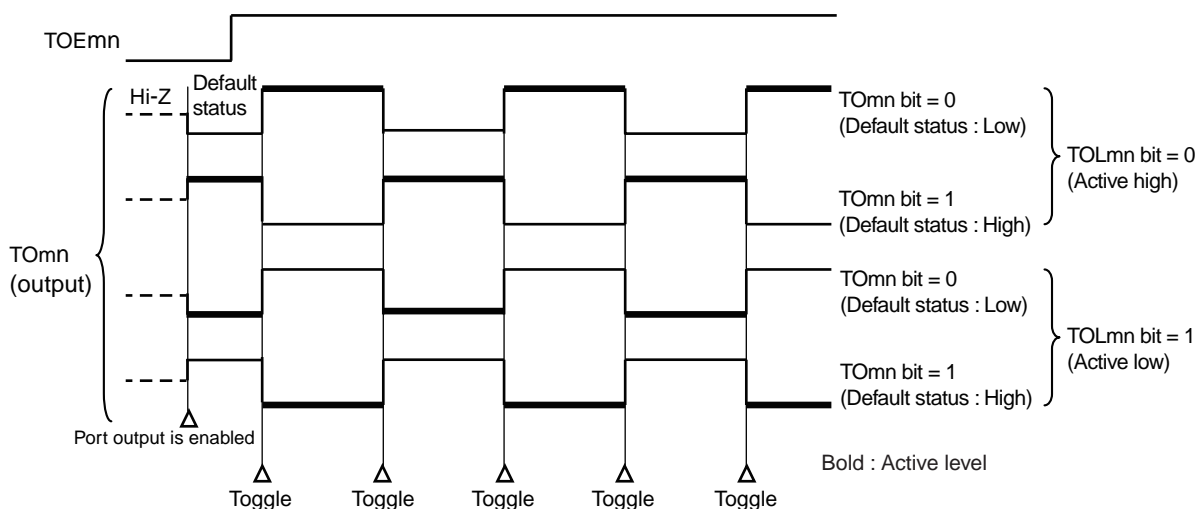
**(2) Default level of TOMn pin and output level after timer operation start**

The change in the output level of the TOMn pin when timer output register m (TOM) is written while timer output is disabled (TOEmn = 0), the initial level is changed, and then timer output is enabled (TOEmn = 1) before port output is enabled, is shown below.

**(a) When operation starts with master channel output mode (TOMmn = 0) setting**

The setting of timer output level register m (TOLm) is invalid when master channel output mode (TOMmn = 0). When the timer operation starts after setting the default level, the toggle signal is generated and the output level of the TOMn pin is reversed.

**Figure 6-32. TOMn Pin Output Status at Toggle Output (TOMmn = 0)**

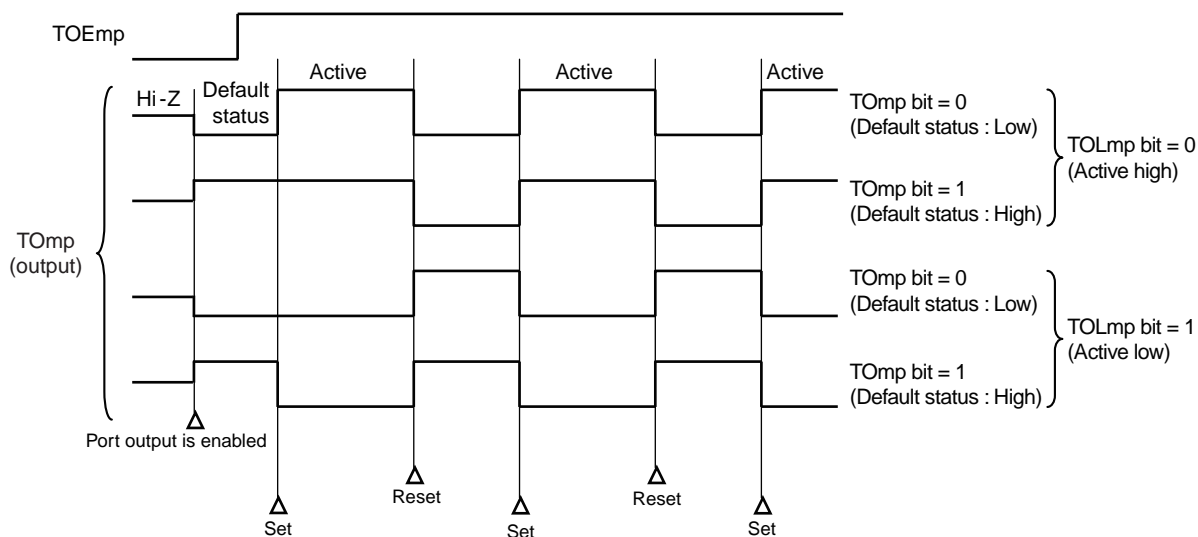


- Remarks**
1. Toggle: Reverse TOMn pin output status
  2. m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**(b) When operation starts with slave channel output mode (TOMmp = 1) setting (PWM output)**

When slave channel output mode (TOMmp = 1), the active level is determined by timer output level register m (TOLm) setting.

**Figure 6-33. TOmp Pin Output Status at PWM Output (TOMmp = 1)**



- Remarks**
1. Set: The output signal of the TOmp pin changes from inactive level to active level.  
Reset: The output signal of the TOmp pin changes from active level to inactive level.
  2. m: Unit number (m = 0), p: Channel number (p = 1 to 3)

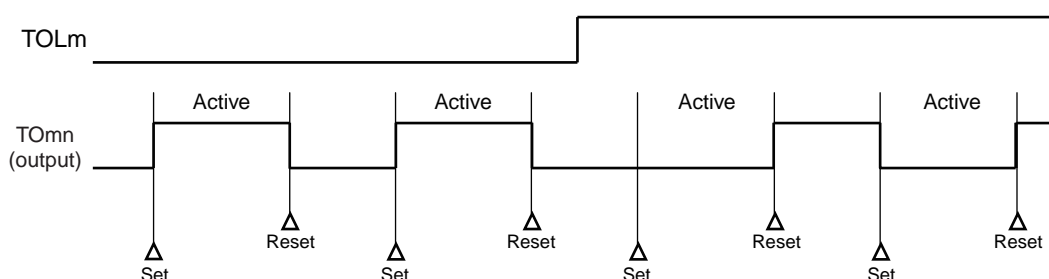


**(3) Operation of TOMn pin in slave channel output mode (TOMmn = 1)****(a) When timer output level register m (TOLm) setting has been changed during timer operation**

When the TOLm register setting has been changed during timer operation, the setting becomes valid at the generation timing of the TOMn pin change condition. Rewriting the TOLm register does not change the output level of the TOMn pin.

The operation when TOMmn is set to 1 and the value of the TOLm register is changed while the timer is operating (TEMn = 1) is shown below.

**Figure 6-34. Operation when TOLm Register Has Been Changed Contents during Timer Operation**



- Remarks**
1. Set: The output signal of the TOMn pin changes from inactive level to active level.
  - Reset: The output signal of the TOMn pin changes from active level to inactive level.
  2. m: Unit number ( $m = 0$ ), n: Channel number ( $n = 0$  to 3)

**(b) Set/reset timing**

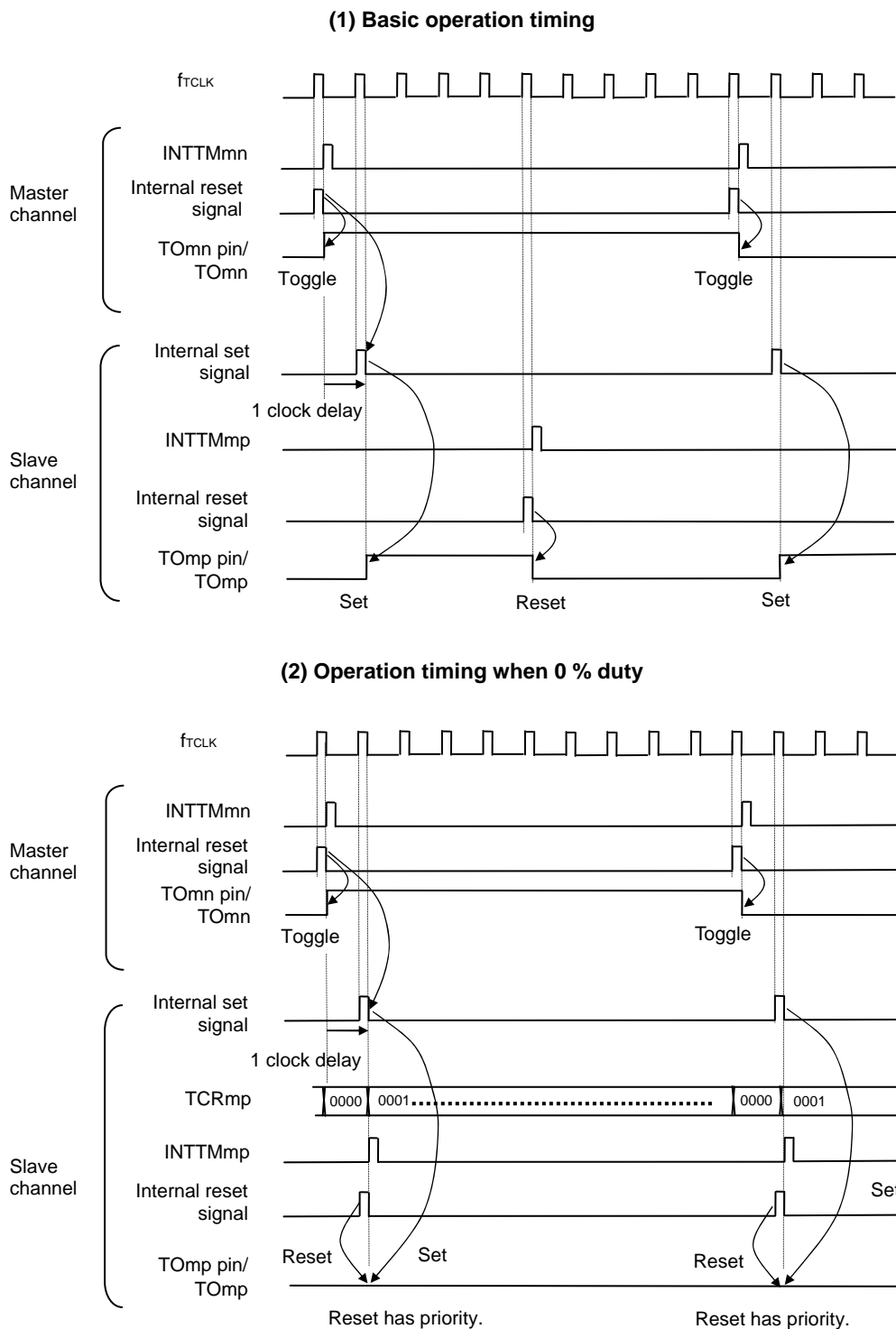
To realize 0%/100% output at PWM output, the TOMn pin/TOMn bit set timing at master channel timer interrupt (INTTMmn) generation is delayed by 1 count clock by the slave channel.

If the set condition and reset condition are generated at the same time, a higher priority is given to the latter.

Figure 6-34 shows the set/reset operating statuses where the master/slave channels are set as follows.

Master channel: TOEmn = 1, TOMmn = 0, TOLmn = 0  
 Slave channel: TOEmp = 1, TOMmp = 1, TOLmp = 0

Figure 6-35. Set/Reset Timing Operating Statuses



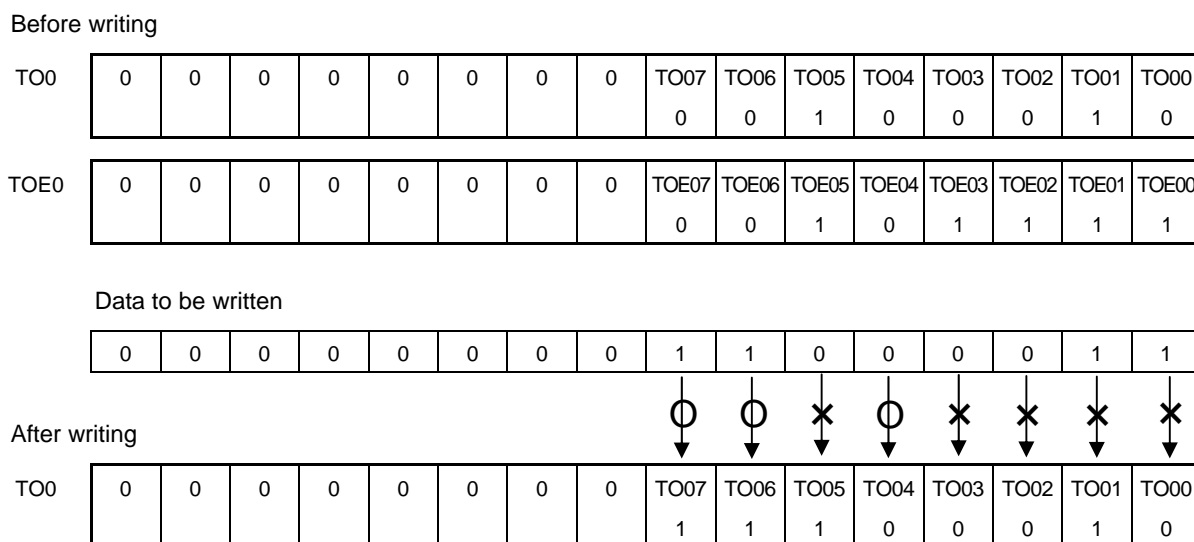
- Remarks 1.** Internal reset signal: TOmn pin reset/toggle signal  
 Internal set signal: TOmn pin set signal
- 2.** m: Unit number (m = 0)  
 n: Channel number  
 n = 0 to 3 (n = 0, 2 for master channel)  
 p: Slave channel number  
 n < p ≤ 3

### 6.6.4 Collective manipulation of TO<sub>m</sub>n bit

In timer output register m (TO<sub>m</sub>), the setting bits for all the channels are located in one register in the same way as timer channel start register m (TS<sub>m</sub>). Therefore, the TO<sub>m</sub>n bit of all the channels can be manipulated collectively.

Only the desired bits can also be manipulated by enabling writing only to the TO<sub>m</sub>n bits (TOE<sub>m</sub>n = 0) that correspond to the relevant bits of the channel used to perform output (TO<sub>m</sub>n).

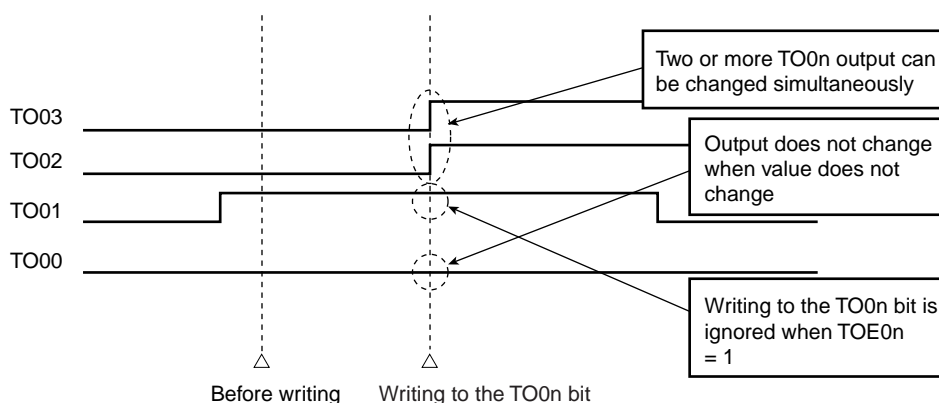
Figure 6-36. Example of TO<sub>0</sub>n Bit Collective Manipulation



Writing is done only to the TO<sub>m</sub>n bit with TOE<sub>m</sub>n = 0, and writing to the TO<sub>m</sub>n bit with TOE<sub>m</sub>n = 1 is ignored.

TO<sub>m</sub>n (channel output) to which TOE<sub>m</sub>n = 1 is set is not affected by the write operation. Even if the write operation is done to the TO<sub>m</sub>n bit, it is ignored and the output change by timer operation is normally done.

Figure 6-37. TO<sub>0</sub>n Pin Statuses by Collective Manipulation of TO<sub>0</sub>n Bit



**Caution** While timer output is enabled (TOE<sub>m</sub>n = 1), even if the output by timer interrupt of each timer (INTTM<sub>m</sub>n) contends with writing to the TO<sub>m</sub>n bit, output is normally done to the TO<sub>m</sub>n pin.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

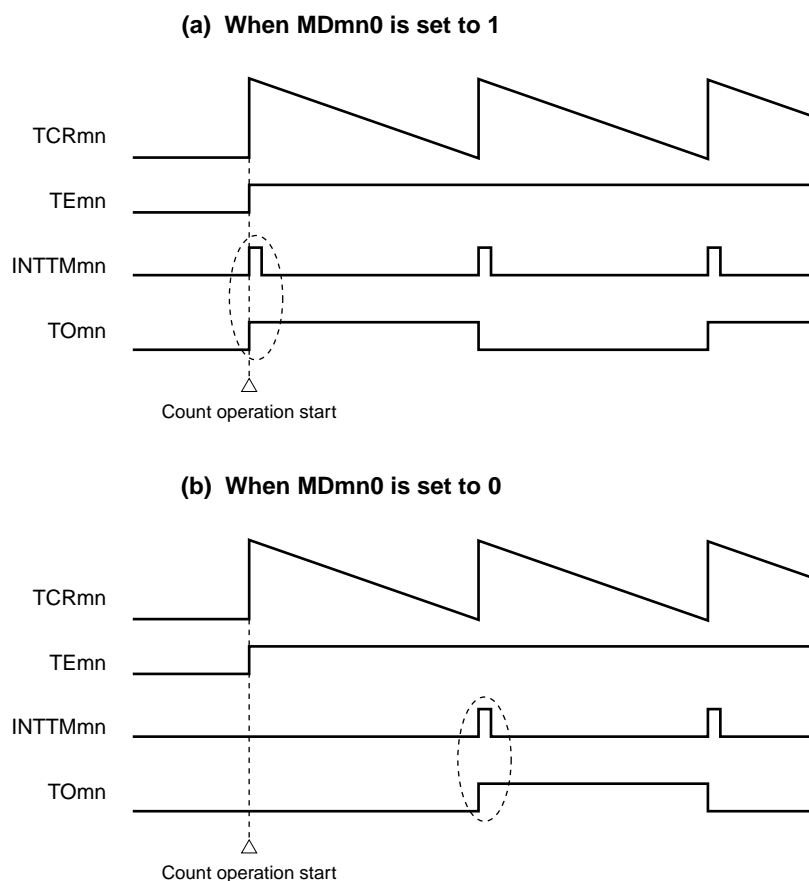
### 6.6.5 Timer interrupt and TOMn pin output at operation start

In the interval timer mode or capture mode, the MDmn0 bit in timer mode register mn (TMRmn) sets whether or not to generate a timer interrupt at count start.

When MDmn0 is set to 1, the count operation start timing can be known by the timer interrupt (INTTMmn) generation. In the other modes, neither timer interrupt at count operation start nor TOMn output is controlled.

Figure 6-37 shows operation examples when the interval timer mode (TOEmn = 1, TOMmn = 0) is set.

**Figure 6-38. Operation examples of timer interrupt at count operation start and TOMn output**



When MDmn0 is set to 1, a timer interrupt (INTTMmn) is output at count operation start, and TOMn performs a toggle operation.

When MDmn0 is set to 0, a timer interrupt (INTTMmn) is not output at count operation start, and TOMn does not change either. After counting one cycle, INTTMmn is output and TOMn performs a toggle operation.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

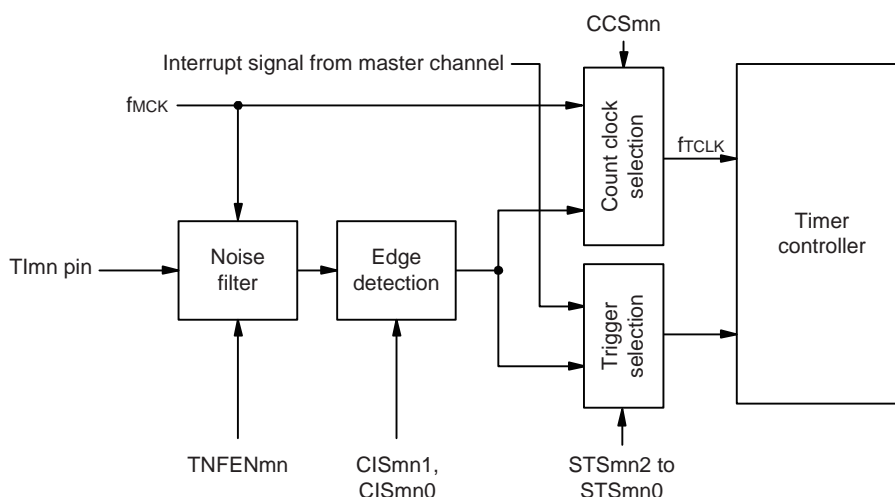
&lt;R&gt;

## 6.7 Timer Input (Tlmn) Control

### 6.7.1 Tlmn input circuit configuration

A signal is input from a timer input pin, goes through a noise filter and an edge detector, and is sent to a timer controller. Enable the noise filter for the pin in need of noise removal. The following shows the configuration of the input circuit.

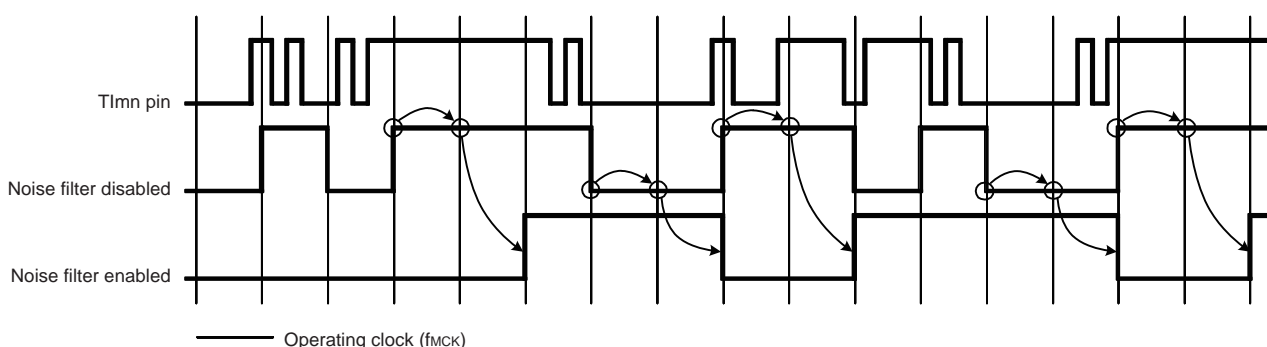
Figure 6 - 39 Input Circuit Configuration



### 6.7.2 Noise filter

When the noise filter is disabled, the input signal is only synchronized with the operating clock ( $f_{MCK}$ ) for channel n. When the noise filter is enabled, after synchronization with the operating clock ( $f_{MCK}$ ) for channel n, whether the signal keeps the same value for two clock cycles is detected. The following shows differences in waveforms output from the noise filter between when the noise filter is enabled and disabled.

Figure 6 - 40 Sampling Waveforms through Tlmn Input Pin with Noise Filter Enabled and Disabled



**Caution** The input waveforms to the Tlmn pin are shown to explain the operation when the noise filter is enabled or disabled. When actually inputting waveforms, input them according to the Tlmn input high-level and low-level widths listed in 27.4 AC Characteristics.

&lt;R&gt;

### 6.7.3 Cautions on channel input operation

When a timer input pin is set as unused, the operating clock is not supplied to the noise filter. Therefore, after settings are made to use the timer input pin, the following wait time is necessary before a trigger is specified to enable operation of the channel corresponding to the timer input pin.

(1) Noise filter is disabled

When bits 12 (CCSmn), 9 (STSmn1), and 8 (STSmn0) in the timer mode register mn (TMRmn) are 0 and then one of them is set to 1, wait for at least two cycles of the operating clock (fMCK), and then set the operation enable trigger bit in the timer channel start register (TSM).

(2) Noise filter is enabled

When bits 12 (CCSmn), 9 (STSmn1), and 8 (STSmn0) in the timer mode register mn (TMRmn) are all 0 and then one of them is set to 1, wait for at least four cycles of the operating clock (fMCK), and then set the operation enable trigger bit in the timer channel start register (TSM).

## 6.8 Independent Channel Operation Function of Timer Array Unit

### 6.8.1 Operation as interval timer/square wave output

#### (1) Interval timer

The timer array unit can be used as a reference timer that generates INTTMmn (timer interrupt) at fixed intervals. The interrupt generation period can be calculated by the following expression.

$$\text{Generation period of INTTMmn (timer interrupt)} = \text{Period of count clock} \times (\text{Set value of TDRmn} + 1)$$

#### (2) Operation as square wave output

TOMn performs a toggle operation as soon as INTTMmn has been generated, and outputs a square wave with a duty factor of 50%.

The period and frequency for outputting a square wave from TOMn can be calculated by the following expressions.

$$\bullet \text{ Period of square wave output from TOMn} = \text{Period of count clock} \times (\text{Set value of TDRmn} + 1) \times 2$$

$$\bullet \text{ Frequency of square wave output from TOMn} = \text{Frequency of count clock} / \{(\text{Set value of TDRmn} + 1) \times 2\}$$

Timer count register mn (TCRmn) operates as a down counter in the interval timer mode.

The TCRmn register loads the value of timer data register mn (TDRmn) at the first count clock after the channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSm) is set to 1. If the MDmn0 bit of timer mode register mn (TMRmn) is 0 at this time, INTTMmn is not output and TOMn is not toggled. If the MDmn0 bit of the TMRmn register is 1, INTTMmn is output and TOMn is toggled.

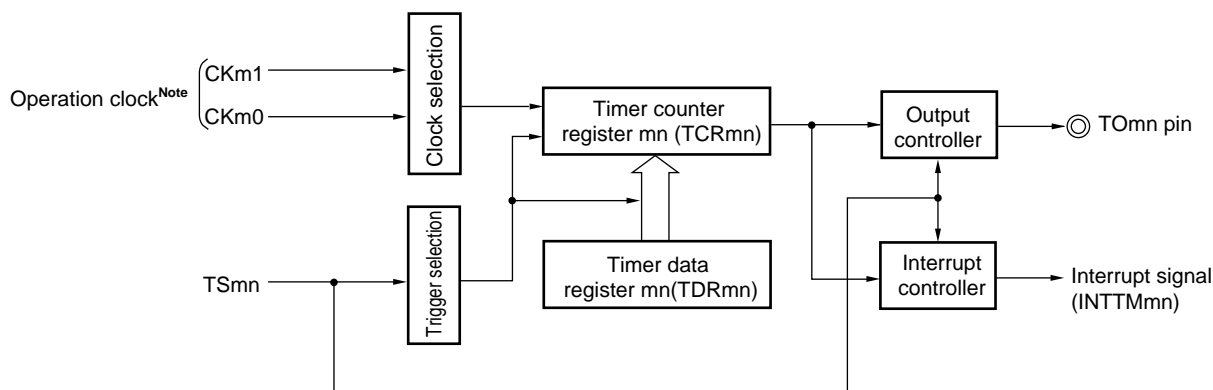
After that, the TCRmn register count down in synchronization with the count clock.

When TCRmn = 0000H, INTTMmn is output and TOMn is toggled at the next count clock. At the same time, the TCRmn register loads the value of the TDRmn register again. After that, the same operation is repeated.

The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period.

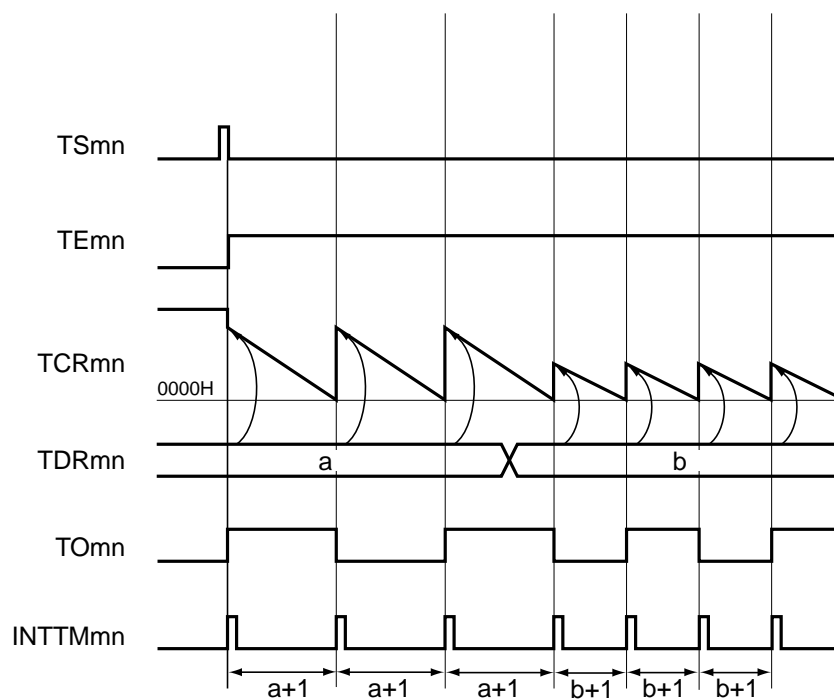
**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-41. Block Diagram of Operation as Interval Timer/Square Wave Output



**Note** When channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

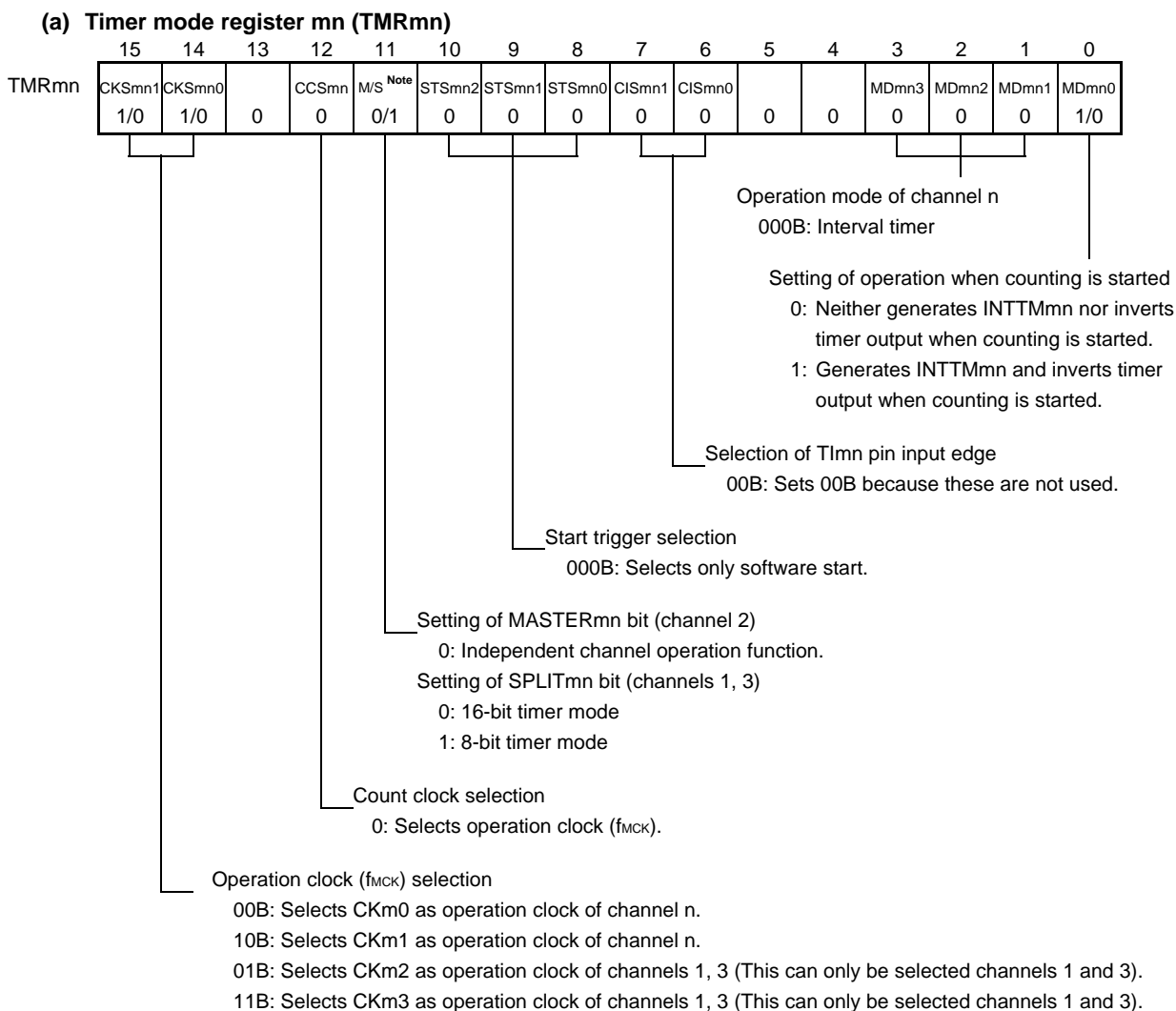
Figure 6-42. Example of Basic Timing of Operation as Interval Timer/Square Wave Output (MDmn0 = 1)



- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0 to 3)
  2. TSmn: Bit n of timer channel start register m (TSm)  
 TEMn: Bit n of timer channel enable status register m (TEm)  
 TCRmn: Timer count register mn (TCRmn)  
 TDRmn: Timer data register mn (TDRmn)  
 TOmn: TOmn pin output signal



Figure 6-43. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (1/2)



<R>

(b) Timer output register m (TOM)

TOM	Bit n	
	TOMn	0: Outputs 0 from TOMn. 1: Outputs 1 from TOMn.

(c) Timer output enable register m (TOEm)

TOEm	Bit n	
	TOEmn	0: Stops the TOMn output operation by counting operation. 1: Enables the TOMn output operation by counting operation.

**Note** TMRm2: MASTERmn bit  
 TMRm1, TMRm3: SPLITmn bit  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**Figure 6-43. Example of Set Contents of Registers During Operation as Interval Timer/Square Wave Output (2/2)****(d) Timer output level register m (TOLm)**

TOLm 

Bit n
TOLmn
0

 0: Cleared to 0 when TOMmn = 0 (master channel output mode)

**(e) Timer output mode register m (TOMm)**

TOMm 

Bit n
TOMmn
0

 0: Sets master channel output mode.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-44. Operation Procedure of Interval Timer/Square Wave Output Function (1/2)

	Software Operation	Hardware Status
<R>	TAU default setting	Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets timer mode register mn (TMRmn) (determines operation mode of channel). Sets interval (period) value to timer data register mn (TDRmn).	Channel stops operating. (Clock is supplied and some power is consumed.)
	To use the TOMn output Clears the TOMmn bit of timer output mode register m (TOMm) to 0 (master channel output mode). Clears the TOLmn bit to 0. Sets the TOMn bit and determines default level of the TOMn output.	The TOMn pin goes into Hi-Z output state.
	Sets the TOEmn bit to 1 and enables operation of TOMn. Clears the port register and port mode register to 0.	The TOMn default setting level is output when the port mode register is in the output mode and the port register is 0. TOMn does not change because channel stops operating. The TOMn pin outputs the TOMn set level.
Operation start	(Sets the TOEmn bit to 1 only if using TOMn output and resuming operation.) Sets the TSmn (TSHm1, TSHm3) bit to 1. The TSmn (TSHm1, TSHm3) bit automatically returns to 0 because it is a trigger bit.	TEmn (TEHm1, TEHm3) = 1, and count operation starts. Value of the TDRmn register is loaded to timer count register mn (TCRmn) at the count clock input. INTTMmn is generated and TOMn performs toggle operation if the MDmn0 bit of the TMRmn register is 1.
During operation	Set values of the TMRmn register, TOMmn, and TOLmn bits cannot be changed. Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used. Set values of the TOM and TOEm registers can be changed.	Counter (TCRmn) counts down. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again and the count operation is continued. By detecting TCRmn = 0000H, INTTMmn is generated and TOMn performs toggle operation. After that, the above operation is repeated.
Operation stop	The TTmn (TTHm1, TTHm3) bit is set to 1. The TTmn (TTHm1, TTHm3) bit automatically returns to 0 because it is a trigger bit.	TEmn (TEHm1, TEHm3), and count operation stops. The TCRmn register holds count value and stops. The TOMn output is not initialized but holds current status.
	The TOEmn bit is cleared to 0 and value is set to the TOMn bit.	The TOMn pin outputs the TOMn bit set level.

Operation is resumed.

(Remark is listed on the next page.)

Figure 6-44 Operation Procedure of Interval Timer/Square Wave Output Function (2/2)

	Software Operation	Hardware Status
TAU stop	To hold the TOMn pin output level Clears the TOMn bit to 0 after the value to be held is set to the port register. →	The TOMn pin output level is held by port function.
	When holding the TOMn pin output level is not necessary Setting not required. ----- The TAUmEN bit of the PER0 register is cleared to 0. →	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized. (The TOMn bit is cleared to 0 and the TOMn pin is set to port mode.)

&lt;R&gt;

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.8.2 Operation as external event counter

The timer array unit can be used as an external event counter that counts the number of times the valid input edge (external event) is detected in the TImn pin. When a specified count value is reached, the event counter generates an interrupt. The specified number of counts can be calculated by the following expression.

$$\text{Specified number of counts} = \text{Set value of TDRmn} + 1$$

Timer count register mn (TCRmn) operates as a down counter in the event counter mode.

The TCRmn register loads the value of timer data register mn (TDRmn) by setting any channel start trigger bit (TSMn, TSHm1, TSHm3) of timer channel start register m (TSM) to 1.

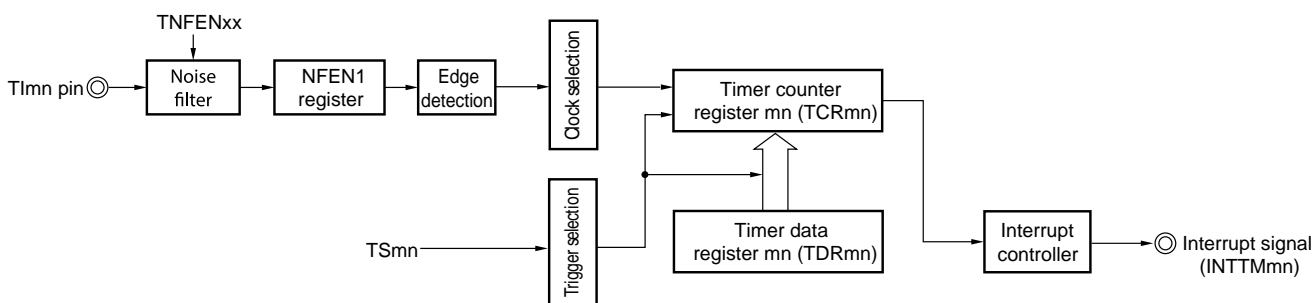
The TCRmn register counts down each time the valid input edge of the TImn pin has been detected. When TCRmn = 0000H, the TCRmn register loads the value of the TDRmn register again, and outputs INTTMmn.

After that, the above operation is repeated.

An irregular waveform that depends on external events is output from the TOmn pin. Stop the output by setting the TOEmn bit of timer output enable register m (TOEm) to 0.

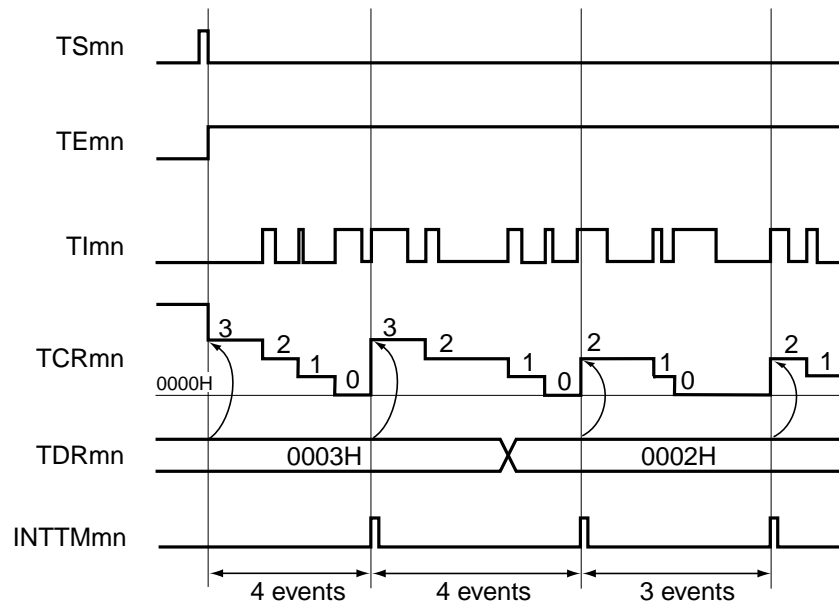
The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid during the next count period.

**Figure 6-45. Block Diagram of Operation as External Event Counter**



**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

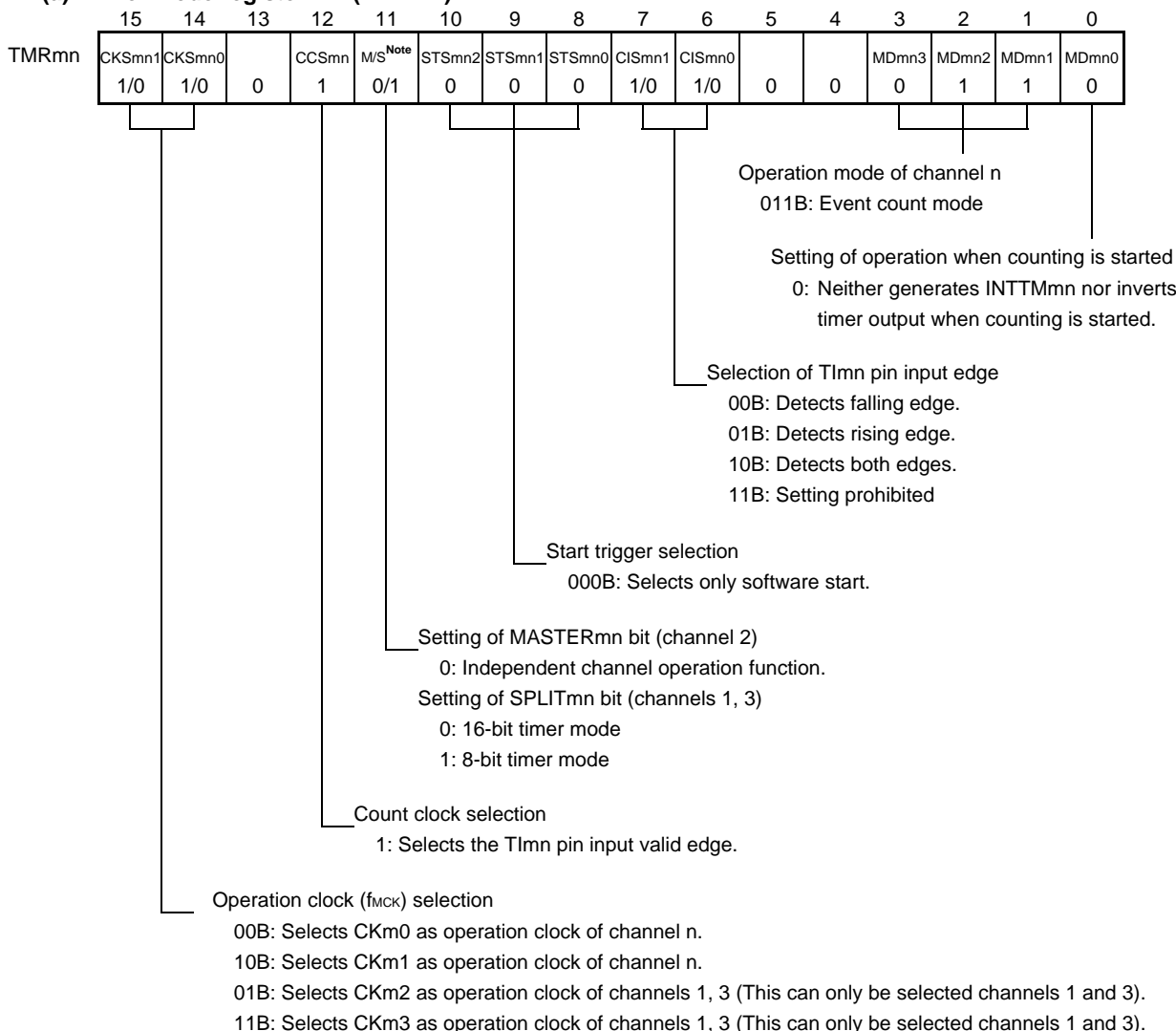
Figure 6-46. Example of Basic Timing of Operation as External Event Counter



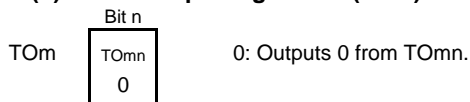
- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0 to 3)
  2. TSmn: Bit n of timer channel start register m (TSm)
  - TE<sub>mn</sub>: Bit n of timer channel enable status register m (TE<sub>m</sub>)
  - TI<sub>mn</sub>: TI<sub>mn</sub> pin input signal
  - TCR<sub>mn</sub>: Timer count register mn (TCR<sub>mn</sub>)
  - TDR<sub>mn</sub>: Timer data register mn (TDR<sub>mn</sub>)

Figure 6-47. Example of Set Contents of Registers in External Event Counter Mode (1/2)

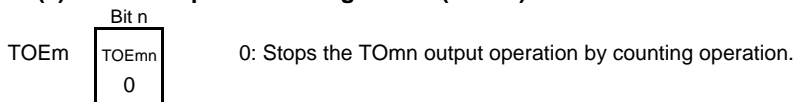
(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



**Note** TMRm2: MASTERmn bit  
 TMRm1, TMRm3: SPLITmn bit  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

**Figure 6-47. Example of Set Contents of Registers in External Event Counter Mode (2/2)****(d) Timer output level register m (TOLm)**

TOLm 

Bit n
TOLmn
0

 0: Cleared to 0 when TOMmn = 0 (master channel output mode).

**(e) Timer output mode register m (TOMm)**

TOMm 

Bit n
TOMmn
0

 0: Sets master channel output mode.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)



Figure 6-48. Operation Procedure When External Event Counter Function Is Used

	Software Operation	Hardware Status
<R>	TAU default setting	Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). Sets number of counts to timer data register mn (TDRmn). Clears the TOEmn bit of timer output enable register m (TOEm) to 0.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation is resumed.	Operation start Sets the TSmn bit to 1. The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and count operation starts. Value of the TDRmn register is loaded to timer count register mn (TCRmn) and detection of the TImn pin input edge is awaited.
	During operation Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used. Set values of the TMRmn register, TOMmn, TOLmn, TOMn, and TOEmn bits cannot be changed.	Counter (TCRmn) counts down each time input edge of the TImn pin has been detected. When count value reaches 0000H, the value of the TDRmn register is loaded to the TCRmn register again, and the count operation is continued. By detecting TCRmn = 0000H, the INTTMmn output is generated. After that, the above operation is repeated.
	Operation stop The TTmn bit is set to 1. The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.8.3 Operation as input pulse interval measurement

The count value can be captured at the Tl<sub>mn</sub> valid edge and the interval of the pulse input to Tl<sub>mn</sub> can be measured. In addition, the count value can be captured by using software operation (TS<sub>mn</sub> = 1) as a capture trigger while the TE<sub>mn</sub> bit is set to 1.

The pulse interval can be calculated by the following expression.

$$\text{Tl}_{mn} \text{ input pulse interval} = \text{Period of count clock} \times ((10000\text{H} \times \text{TSR}_{mn}:\text{OVF}) + (\text{Capture value of TDR}_{mn} + 1))$$

**Caution** The Tl<sub>mn</sub> pin input is sampled using the operating clock selected with the CKS<sub>mn</sub> bit of timer mode register mn (TMR<sub>mn</sub>), so an error of up to one operating clock cycle occurs.

Timer count register mn (TCR<sub>mn</sub>) operates as an up counter in the capture mode.

When the channel start trigger bit (TS<sub>mn</sub>) of timer channel start register m (TS<sub>m</sub>) is set to 1, the TCR<sub>mn</sub> register counts up from 0000H in synchronization with the count clock.

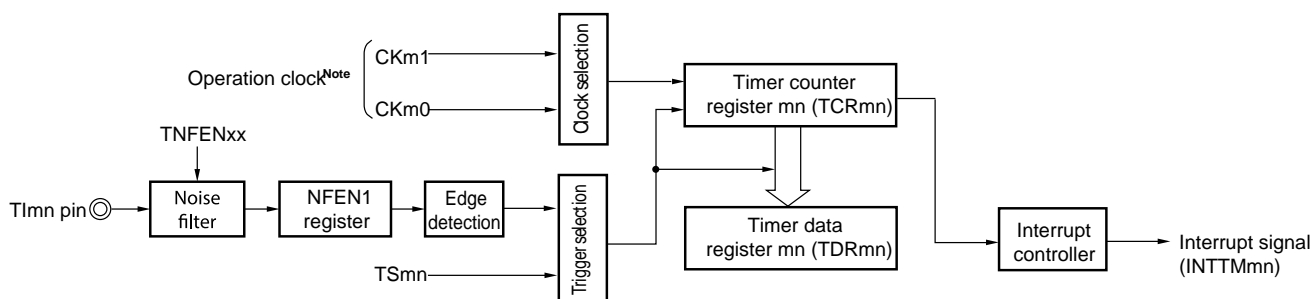
When the Tl<sub>mn</sub> pin input valid edge is detected, the count value of the TCR<sub>mn</sub> register is transferred (captured) to timer data register mn (TDR<sub>mn</sub>) and, at the same time, the TCR<sub>mn</sub> register is cleared to 0000H, and the INTT<sub>Mmn</sub> is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSR<sub>mn</sub>) is set to 1. If the counter does not overflow, the OVF bit is cleared. After that, the above operation is repeated.

As soon as the count value has been captured to the TDR<sub>mn</sub> register, the OVF bit of the TSR<sub>mn</sub> register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow status of the captured value can be checked.

If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSR<sub>mn</sub> register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur.

Set the STS<sub>mn2</sub> to STS<sub>mn0</sub> bits of the TMR<sub>mn</sub> register to 001B to use the valid edges of Tl<sub>mn</sub> as a start trigger and a capture trigger.

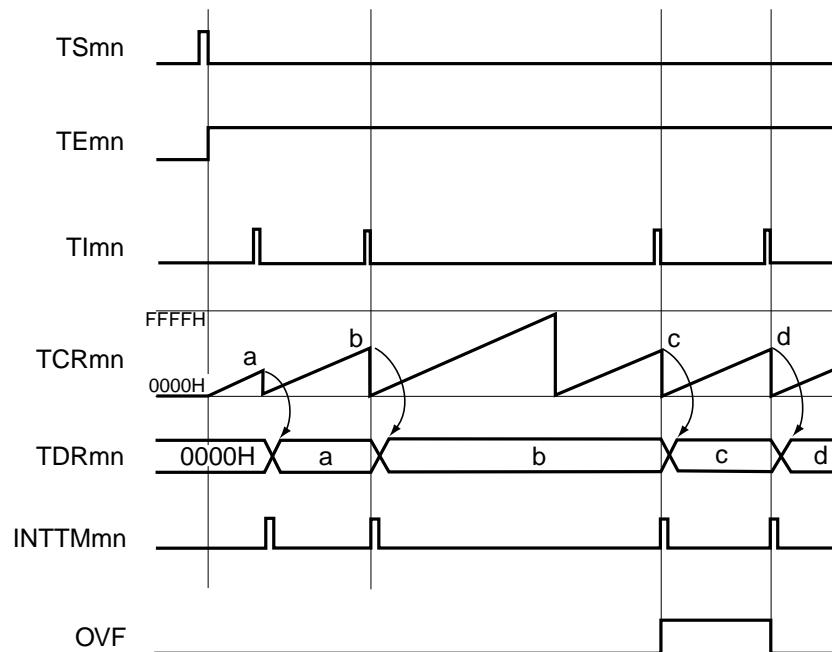
**Figure 6-49. Block Diagram of Operation as Input Pulse Interval Measurement**



**Note** When channels 1 and 3, the clock can be selected from CK<sub>m0</sub>, CK<sub>m1</sub>, CK<sub>m2</sub> and CK<sub>m3</sub>.

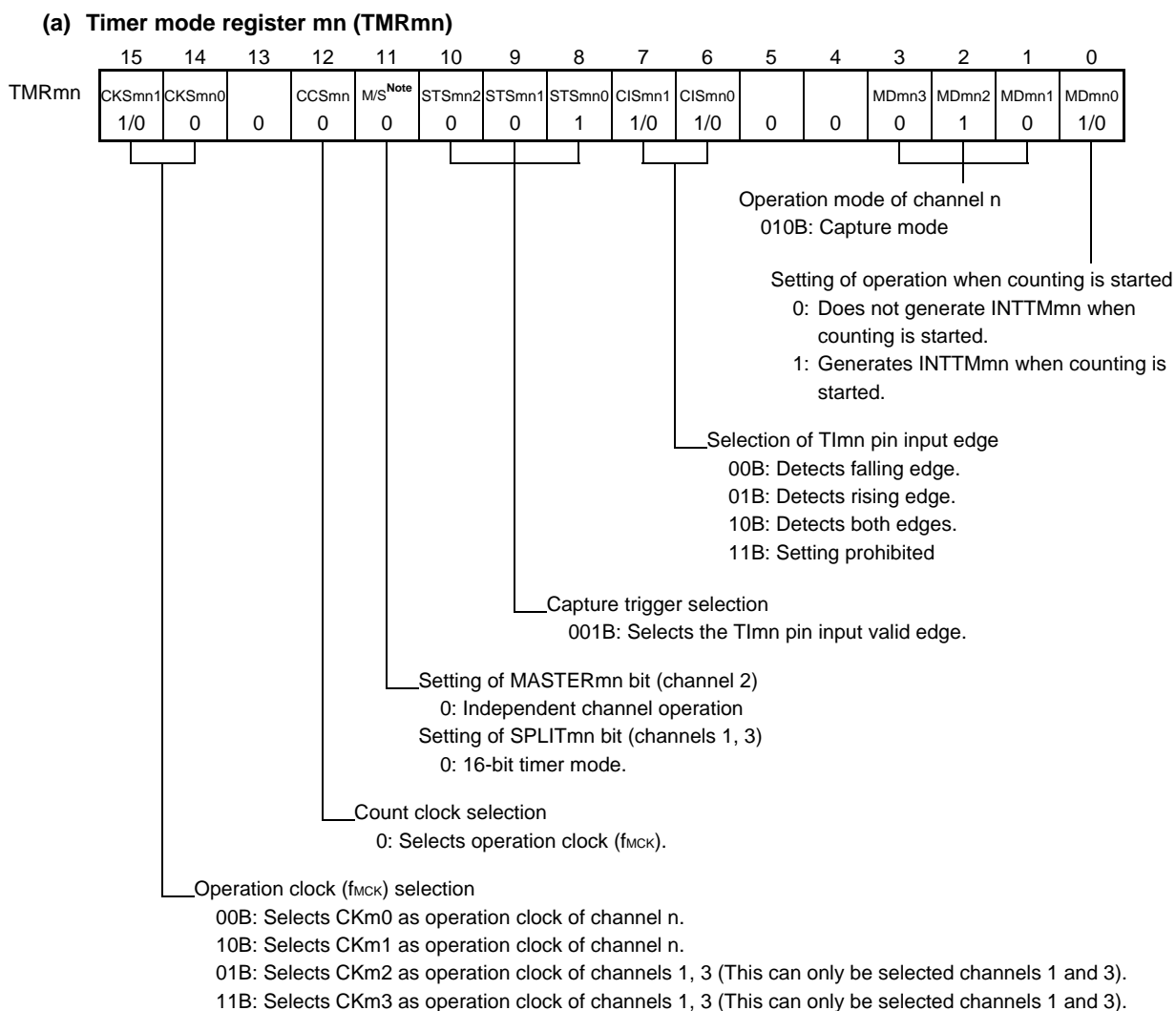
**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-50. Example of Basic Timing of Operation as Input Pulse Interval Measurement (MDmn0 = 0)

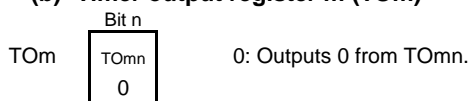


- Remarks**
1. m: Unit number (m = 0)n: Channel number (n = 0 to 3)
  2. TSmn: Bit n of timer channel start register m (TSm)
  - TEmn: Bit n of timer channel enable status register m (TEm)
  - TImn: TImn pin input signal
  - TCRmn: Timer count register mn (TCRmn)
  - TDRmn: Timer data register mn (TDRmn)
  - OVF: Bit 0 of timer status register mn (TSRmn)

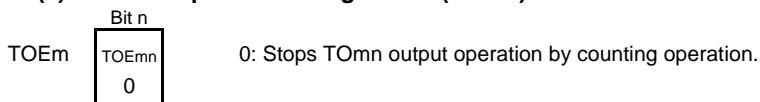
Figure 6-51. Example of Set Contents of Registers to Measure Input Pulse Interval



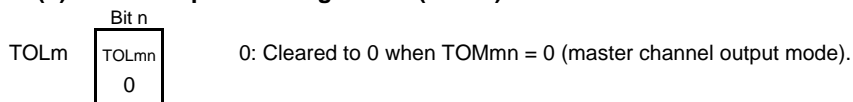
(b) Timer output register m (TOM)



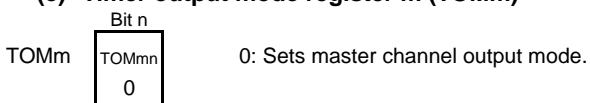
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



**Note** TMRm2: MASTERmn bit  
 TMRm1, TMRm3: SPLITmn bit  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-52. Operation Procedure When Input Pulse Interval Measurement Function Is Used

	Software Operation	Hardware Status
<R>	TAU default setting	Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets timer mode register mn (TMRmn) (determines operation mode of channel). Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on).	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets TSmn bit to 1. The TSmn bit automatically returns to 0 because it is a trigger bit. Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on).	TEmn = 1, and count operation starts. Timer count register mn (TCRmn) is cleared to 0000H at the count clock input. When the MDmn0 bit of the TMRmn register is 1, INTTMmn is generated.
During operation	Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed. The TDRmn register can always be read. The TCRmn register can always be read. The TSRmn register can always be read. Set values of the TOMmn, TOLmn, TOMn, and TOEmn bits cannot be changed.	Counter (TCRmn) counts up from 0000H. When the TImn pin input valid edge is detected, the count value is transferred (captured) to timer data register mn (TDRmn). At the same time, the TCRmn register is cleared to 0000H, and the INTTMmn signal is generated. If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared. After that, the above operation is repeated.
Operation stop	The TTmn bit is set to 1. The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops. The OVF bit of the TSRmn register is also held.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

#### 6.8.4 Operation as input signal high-/low-level width measurement

By starting counting at one edge of the TImn pin input and capturing the number of counts at another edge, the signal width (high-level width/low-level width) of TImn can be measured. The signal width of TImn can be calculated by the following expression.

$$\text{Signal width of TImn input} = \text{Period of count clock} \times ((10000\text{H} \times \text{TSRmn: OVF}) + (\text{Capture value of TDRmn} + 1))$$

**Caution** The TImn pin input is sampled using the operating clock selected with the CKSmn bit of timer mode register mn (TMRmn), so an error equivalent to one operation clock occurs.

Timer count register mn (TCRmn) operates as an up counter in the capture & one-count mode.

When the channel start trigger bit (TSmn) of timer channel start register m (TSm) is set to 1, the TEmn bit is set to 1 and the TImn pin start edge detection wait status is set.

When the TImn pin input start edge (rising edge of the TImn pin input when the high-level width is to be measured) is detected, the counter counts up from 0000H in synchronization with the count clock. When the valid capture edge (falling edge of the TImn pin input when the high-level width is to be measured) is detected later, the count value is transferred to timer data register mn (TDRmn) and, at the same time, INTTMmn is output. If the counter overflows at this time, the OVF bit of timer status register mn (TSRmn) is set to 1. If the counter does not overflow, the OVF bit is cleared. The TCRmn register stops at the value "value transferred to the TDRmn register + 1", and the TImn pin start edge detection wait status is set. After that, the above operation is repeated.

As soon as the count value has been captured to the TDRmn register, the OVF bit of the TSRmn register is updated depending on whether the counter overflows during the measurement period. Therefore, the overflow status of the captured value can be checked.

If the counter reaches a full count for two or more periods, it is judged to be an overflow occurrence, and the OVF bit of the TSRmn register is set to 1. However, a normal interval value cannot be measured for the OVF bit, if two or more overflows occur.

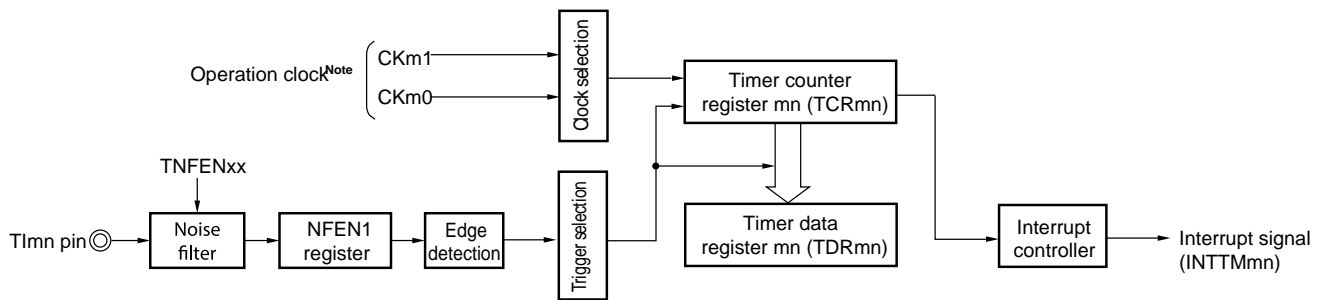
Whether the high-level width or low-level width of the TImn pin is to be measured can be selected by using the CISmn1 and CISmn0 bits of the TMRmn register.

Because this function is used to measure the signal width of the TImn pin input, the TSmn bit cannot be set to 1 while the TEmn bit is 1.

CISmn1, CISmn0 of TMRmn register = 10B: Low-level width is measured.

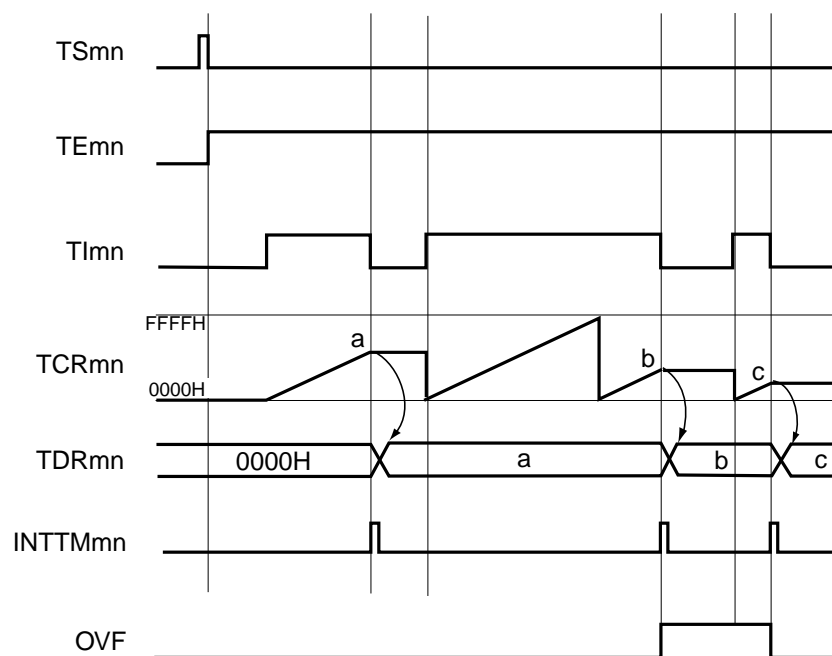
CISmn1, CISmn0 of TMRmn register = 11B: High-level width is measured.

Figure 6-53. Block Diagram of Operation as Input Signal High-/Low-Level Width Measurement



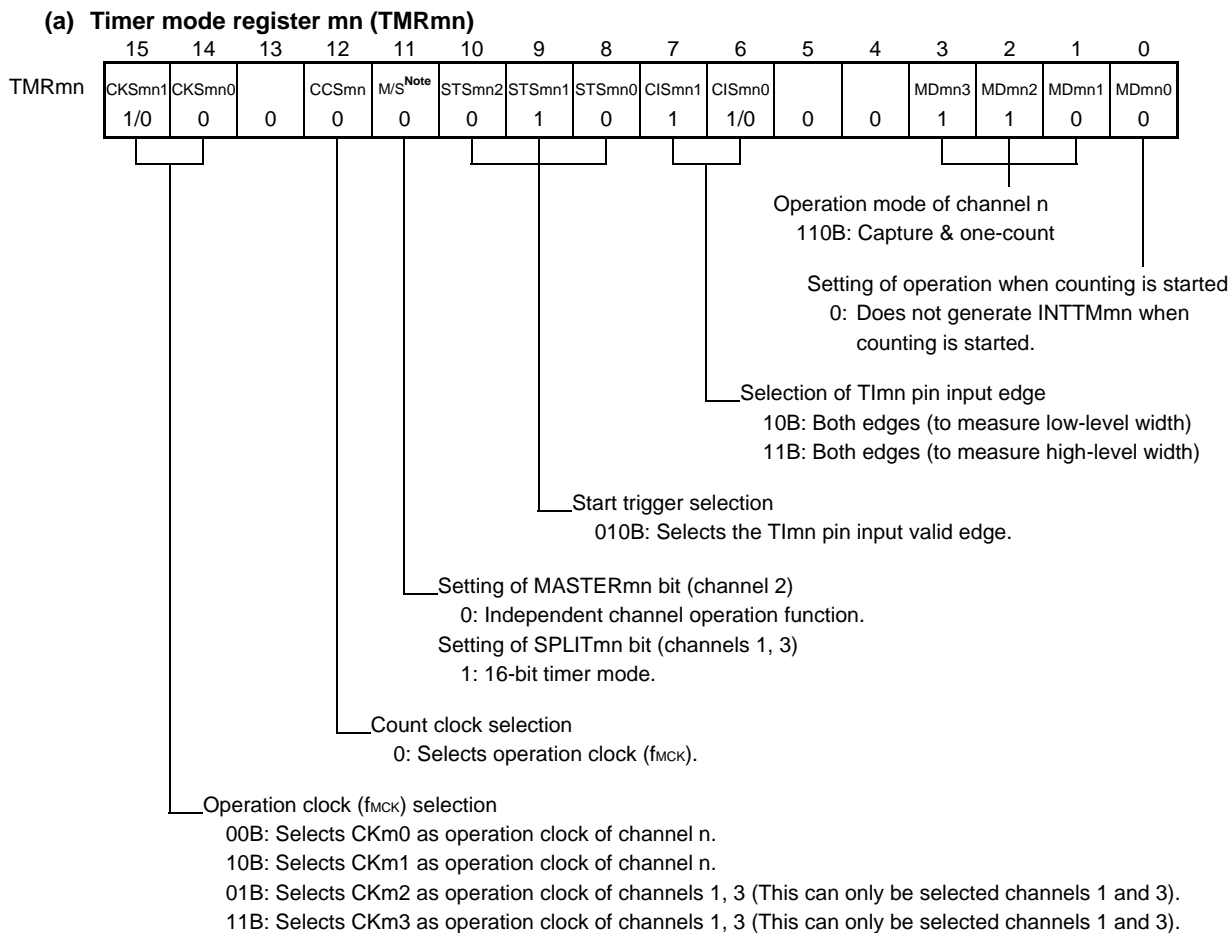
**Note** For channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

Figure 6-54. Example of Basic Timing of Operation as Input Signal High-/Low-Level Width Measurement

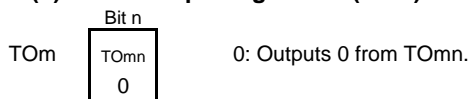


- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0 to 3)
  2. TSmn: Bit n of timer channel start register m (TSM)
  - TE mn: Bit n of timer channel enable status register m (TEM)
  - TImn: TImn pin input signal
  - TCRmn: Timer count register mn (TCRmn)
  - TDRmn: Timer data register mn (TDRmn)
  - OVF: Bit 0 of timer status register mn (TSRmn)

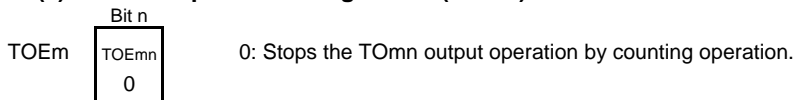
Figure 6-55. Example of Set Contents of Registers to Measure Input Signal High-/Low-Level Width



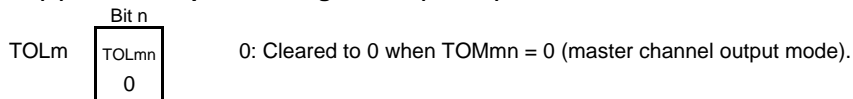
(b) Timer output register m (TOM)



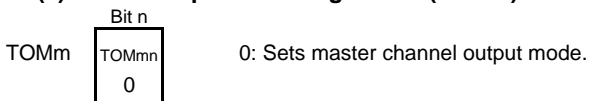
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



**Note** TMRm2: MASTERmn bit  
 TMRm1, TMRm3: SPLITmn bit  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)



Figure 6-56. Operation Procedure When Input Signal High-/Low-Level Width Measurement Function Is Used

	Software Operation	Hardware Status
<R>	TAU default setting	Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). Clears the TOEmn bit to 0 and stops operation of TOmn.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets the TSmn bit to 1. The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and the TImn pin start edge detection wait status is set.
	Detects the TImn pin input count start valid edge.	Clears timer count register mn (TCRmn) to 0000H and starts counting up.
During operation	Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used. Set values of the TMRmn register, TOMmn, TOLmn, TOmn, and TOEmn bits cannot be changed.	When the TImn pin start edge is detected, the counter (TCRmn) counts up from 0000H. If a capture edge of the TImn pin is detected, the count value is transferred to timer data register mn (TDRmn) and INTTMmn is generated. If an overflow occurs at this time, the OVF bit of timer status register mn (TSRmn) is set; if an overflow does not occur, the OVF bit is cleared. The TCRmn register stops the count operation until the next TImn pin start edge is detected.
Operation stop	The TTmn bit is set to 1. The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops. The OVF bit of the TSRmn register is also held.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

### 6.8.5 Operation as delay counter

It is possible to start counting down when the valid edge of the TImn pin input is detected (an external event), and then generate INTTMmn (a timer interrupt) after any specified interval.

It can also generate INTTMmn (timer interrupt) at any interval by making a software set TSmn = 1 and the count down start during the period of TEMn = 1.

The interrupt generation period can be calculated by the following expression.

$$\text{Generation period of INTTMmn (timer interrupt)} = \text{Period of count clock} \times (\text{Set value of TDRmn} + 1)$$

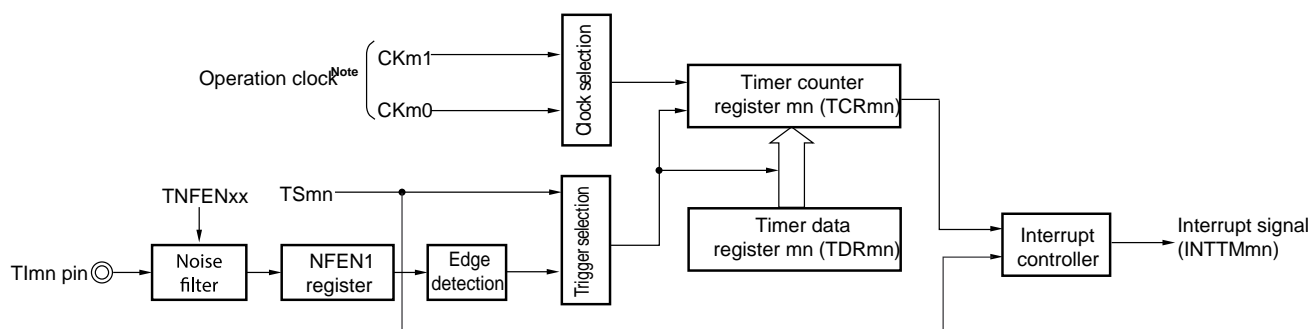
Timer count register mn (TCRmn) operates as a down counter in the one-count mode.

When the channel start trigger bit (TSmn, TSHm1, TSHm3) of timer channel start register m (TSM) is set to 1, the TEMn, TEHm1, TEHm3 bits are set to 1 and the TImn pin input valid edge detection wait status is set.

Timer count register mn (TCRmn) starts operating upon TImn pin input valid edge detection and loads the value of timer data register mn (TDRmn). The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the count clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next TImn pin input valid edge is detected.

The TDRmn register can be rewritten at any time. The new value of the TDRmn register becomes valid from the next period.

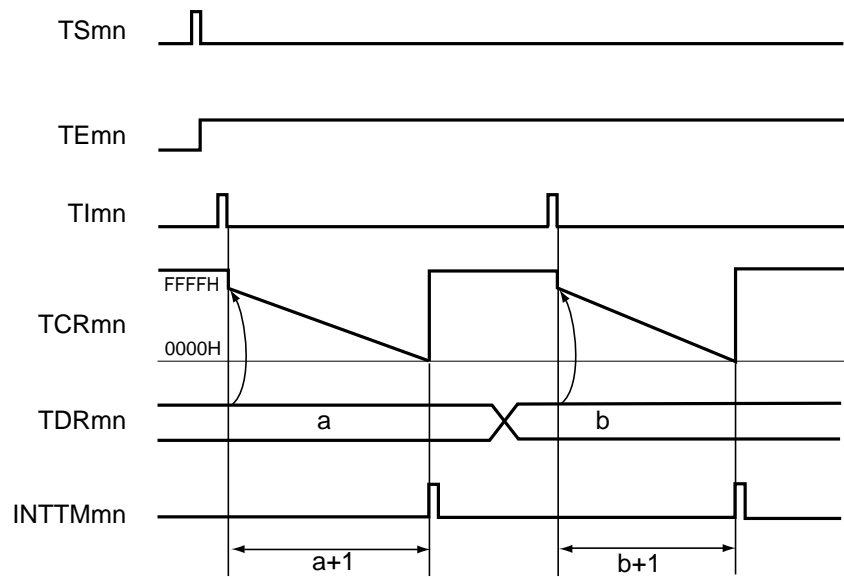
Figure 6-57. Block Diagram of Operation as Delay Counter



**Note** For using channels 1 and 3, the clock can be selected from CKm0, CKm1, CKm2 and CKm3.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

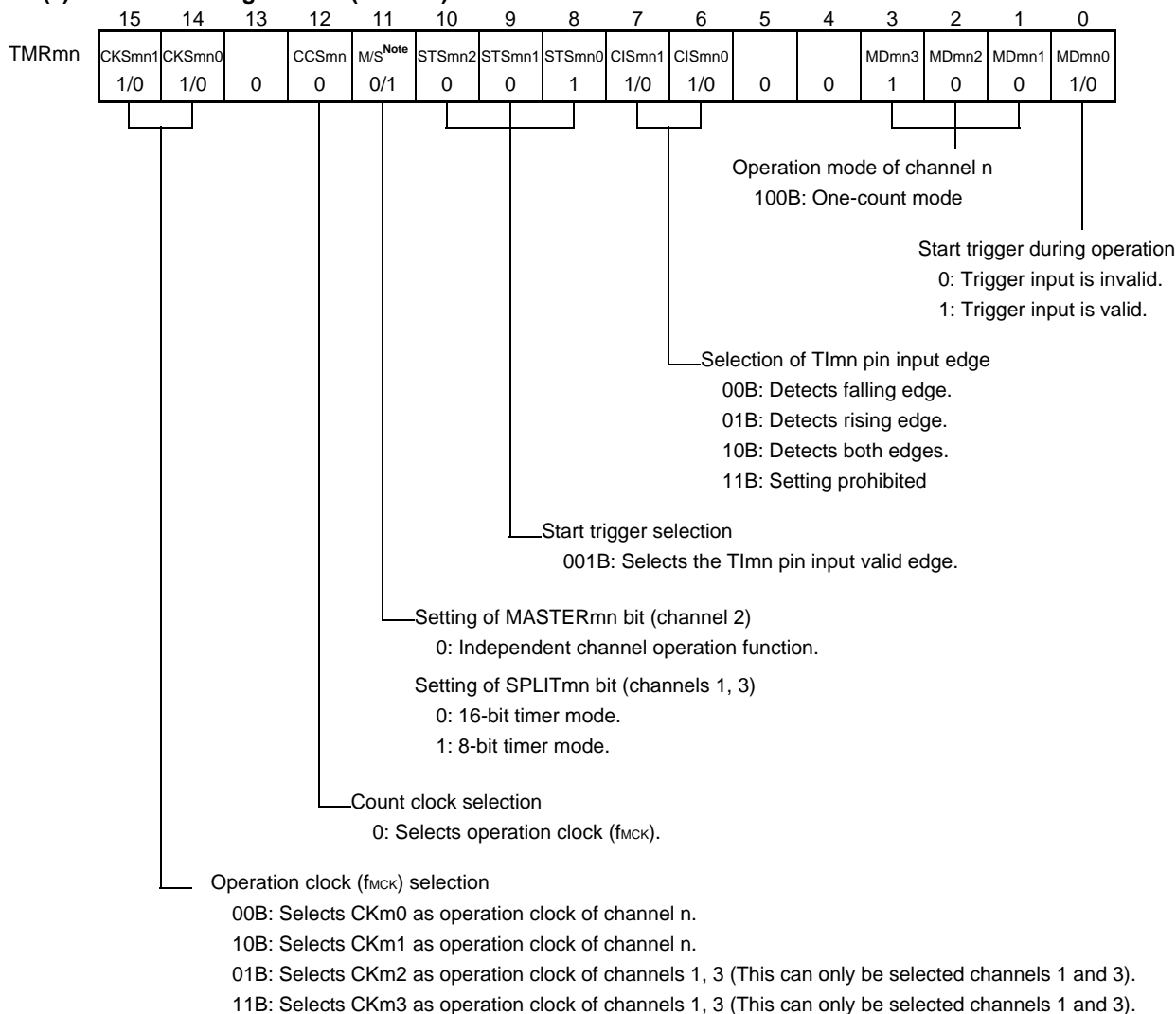
Figure 6-58. Example of Basic Timing of Operation as Delay Counter



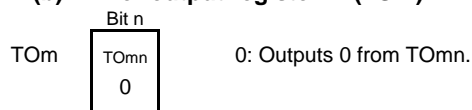
- Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0 to 3)
- 2.** TSmn: Bit n of timer channel start register m (TSm)
- TE<sub>mn</sub>: Bit n of timer channel enable status register m (TE<sub>m</sub>)
- TI<sub>mn</sub>: TI<sub>mn</sub> pin input signal
- TCR<sub>mn</sub>: Timer count register mn (TCR<sub>mn</sub>)
- TDR<sub>mn</sub>: Timer data register mn (TDR<sub>mn</sub>)

Figure 6-59. Example of Set Contents of Registers to Delay Counter (1/2)

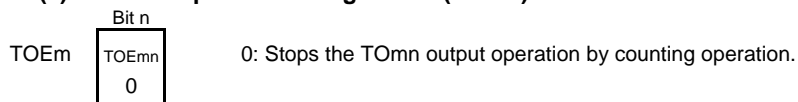
(a) Timer mode register mn (TMRmn)



(b) Timer output register m (TOM)



(c) Timer output enable register m (TOEm)



**Note** TMRm2: MASTERmn bit  
 TMRm1, TMRm3: SPLITmn bit  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

<R>

**Figure 6-59. Example of Set Contents of Registers to Delay Counter (2/2)****(d) Timer output level register m (TOLm)**

TOLm 

Bit n
TOLmn
0

 0: Cleared to 0 when TOMmn = 0 (master channel output mode).

**(e) Timer output mode register m (TOMm)**

TOMm 

Bit n
TOMmn
0

 0: Sets master channel output mode.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

Figure 6-60. Operation Procedure When Delay Counter Function Is Used

	Software Operation	Hardware Status
<R>	TAU default setting	Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 to CKm3.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 0 (off) or 1 (on). Sets timer mode register mn (TMRmn) (determines operation mode of channel). INTTMmn output delay is set to timer data register mn (TDRmn). Clears the TOEmn bit to 0 and stops operation of TOmn.	Channel stops operating. (Clock is supplied and some power is consumed.)
Operation start	Sets the TSmn bit to 1. The TSmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 1, and the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1) wait status is set.
	The counter starts counting down by the next start trigger detection. • Detects the TImn pin input valid edge. • Sets the TSmn bit to 1 by the software.	Value of the TDRmn register is loaded to the timer count register mn (TCRmn).
During operation	Set value of the TDRmn register can be changed. The TCRmn register can always be read. The TSRmn register is not used.	The counter (TCRmn) counts down. When TCRmn counts down to 0000H, INTTMmn is output, and counting stops until the next start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit is set to 1).
Operation stop	The TTmn bit is set to 1. The TTmn bit automatically returns to 0 because it is a trigger bit.	TEmn = 0, and count operation stops. The TCRmn register holds count value and stops.
TAU stop	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized.

Operation is resumed.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0 to 3)

## 6.9 Simultaneous Channel Operation Function of Timer Array Unit

### 6.9.1 Operation as one-shot pulse output function

By using two channels as a set, a one-shot pulse having any delay pulse width can be generated from the signal input to the TImn pin.

The delay time and pulse width can be calculated by the following expressions.

$\text{Delay time} = \{\text{Set value of TDRmn (master)} + 2\} \times \text{Count clock period}$ $\text{Pulse width} = \{\text{Set value of TDRmp (slave)}\} \times \text{Count clock period}$
---

The master channel operates in the one-count mode and counts the delays. Timer count register mn (TCRmn) of the master channel starts operating upon start trigger detection and loads the value of timer data register mn (TDRmn).

The TCRmn register counts down from the value of the TDRmn register it has loaded, in synchronization with the count clock. When TCRmn = 0000H, it outputs INTTMmn and stops counting until the next start trigger is detected.

The slave channel operates in the one-count mode and counts the pulse width. The TCRmp register of the slave channel starts operation using INTTMmn of the master channel as a start trigger, and loads the value of the TDRmp register. The TCRmp register counts down from the value of the TDRmp register it has loaded, in synchronization with the count value. When count value = 0000H, it outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) is detected. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

Instead of using the TImn pin input, a one-shot pulse can also be output using the software operation (TSmn = 1) as a start trigger.

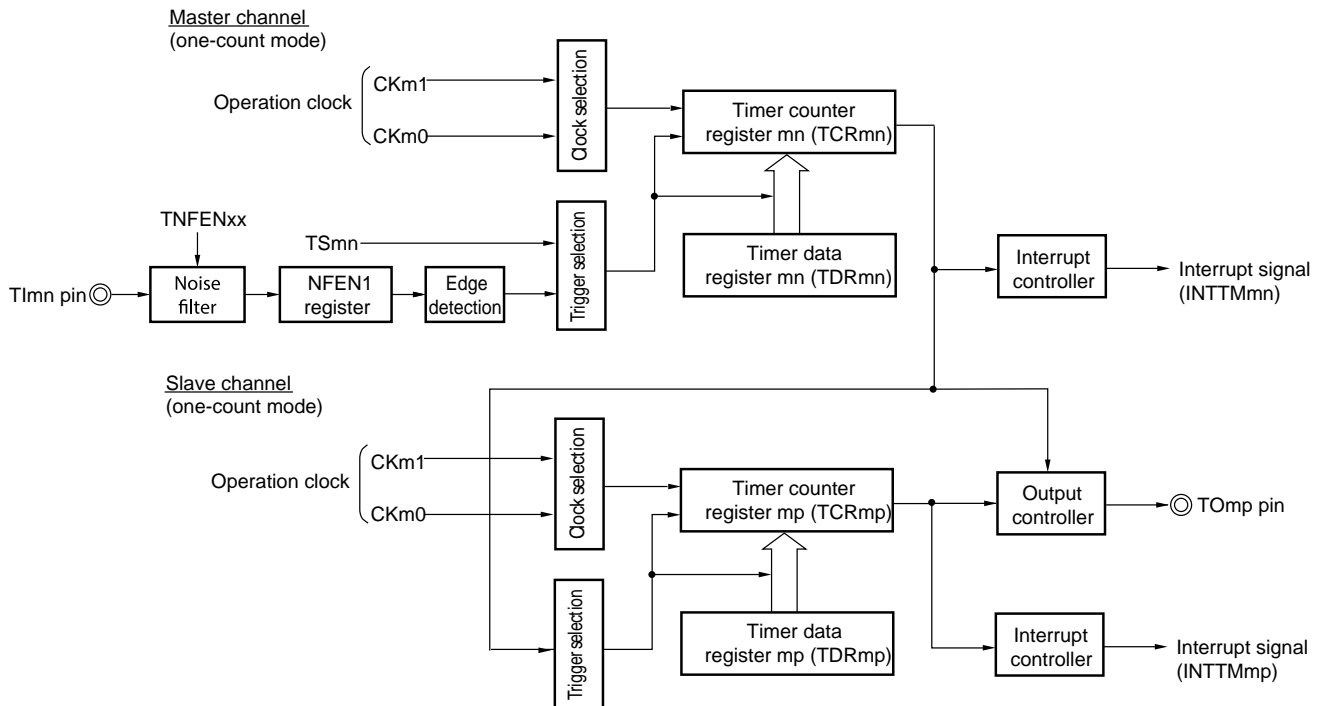
**Caution** The timing of loading of timer data register mn (TDRmn) of the master channel is different from that of the TDRmp register of the slave channel. If the TDRmn and TDRmp registers are rewritten during operation, therefore, an illegal waveform is output. Rewrite the TDRmn register after INTTMmn is generated and the TDRmp register after INTTMmp is generated.

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

<R>

Figure 6-61. Block Diagram of Operation as One-Shot Pulse Output Function

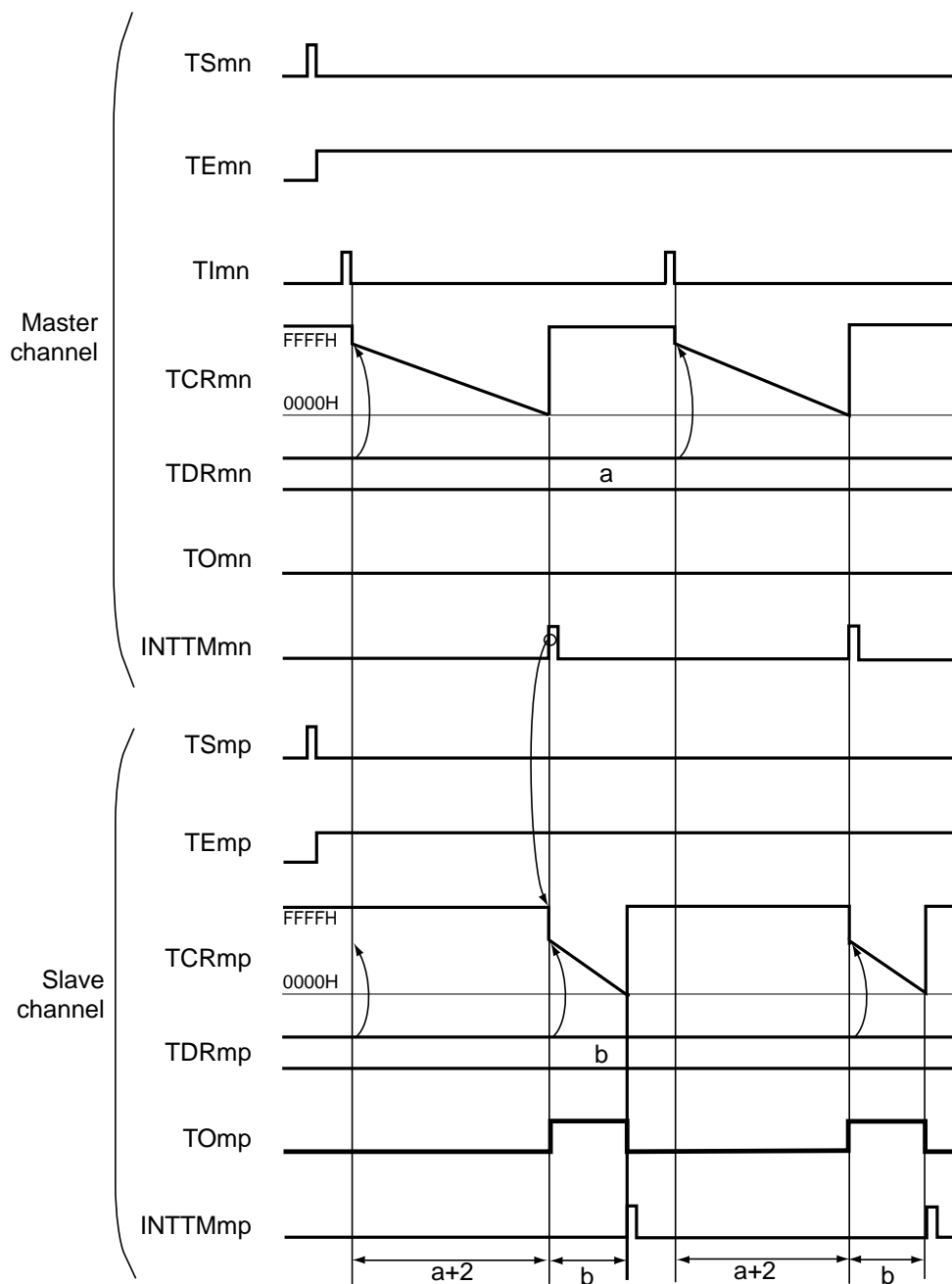
&lt;R&gt;



**Remark** m: Unit number ( $m = 0$ ), n: Master channel number ( $n = 0, 2$ )  
 p: Slave channel number ( $n = 0$ :  $p = 1, 2, 3$ ,  $n = 2$ :  $p = 3$ )



Figure 6-62. Example of Basic Timing of Operation as One-Shot Pulse Output Function

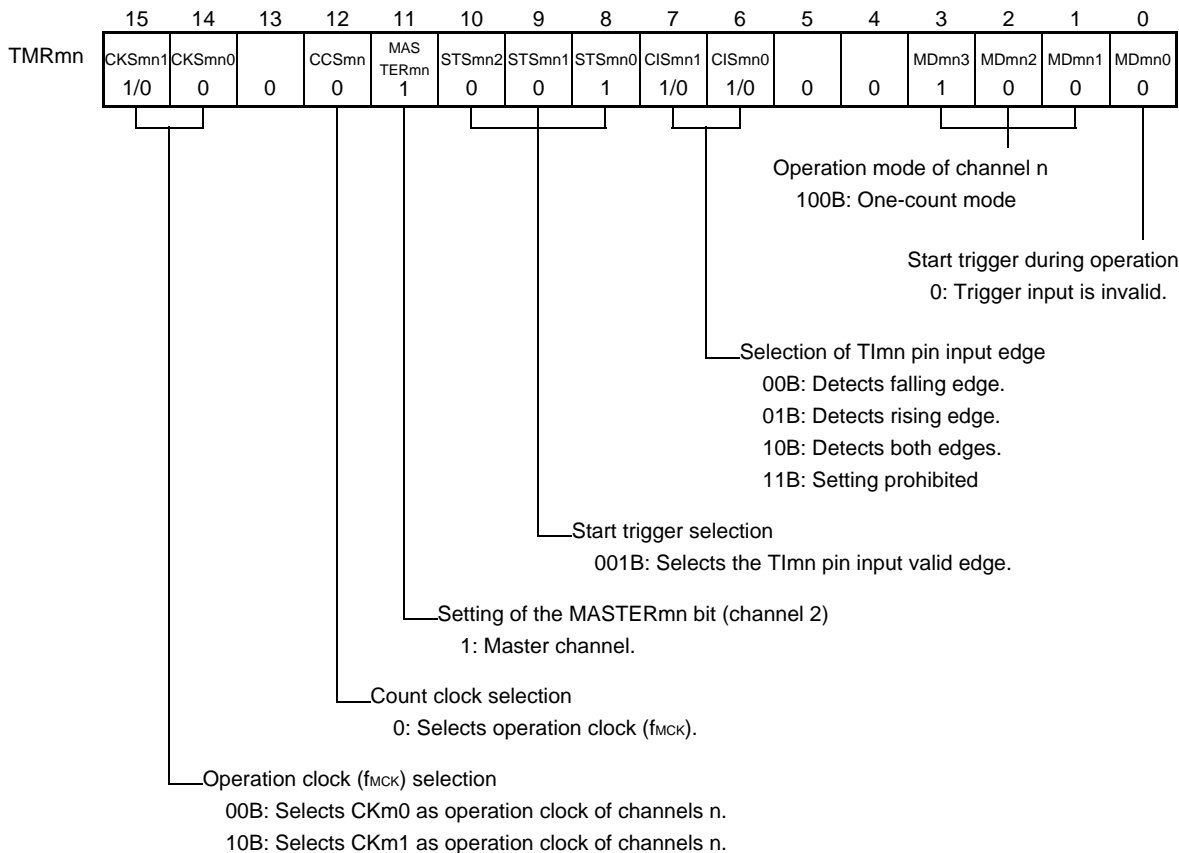


<R>

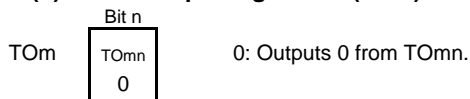
- Remarks**
- m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)
  - TS<sub>mn</sub>, TS<sub>mp</sub>: Bit n, p of timer channel start register m (TS<sub>m</sub>)  
TE<sub>mn</sub>, TE<sub>mp</sub>: Bit n, p of timer channel enable status register m (TE<sub>m</sub>)  
TI<sub>mn</sub>, TI<sub>mp</sub>: TI<sub>mn</sub> and TI<sub>mp</sub> pins input signal  
TCR<sub>mn</sub>, TCR<sub>mp</sub>: Timer count registers mn, mp (TCR<sub>mn</sub>, TCR<sub>mp</sub>)  
TDR<sub>mn</sub>, TDR<sub>mp</sub>: Timer data registers mn, mp (TDR<sub>mn</sub>, TDR<sub>mp</sub>)  
TO<sub>mn</sub>, TO<sub>mp</sub>: TO<sub>mn</sub> and TO<sub>mp</sub> pins output signal

Figure 6-63 Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)

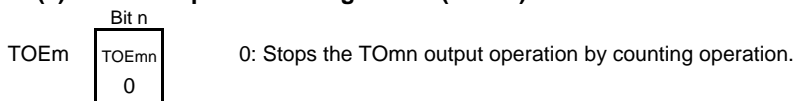
(a) Timer mode register mn (TMRmn)



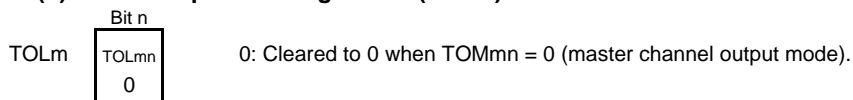
(b) Timer output register m (TOM)



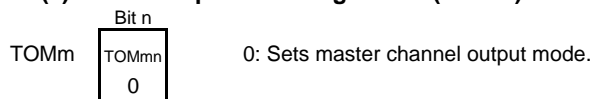
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)

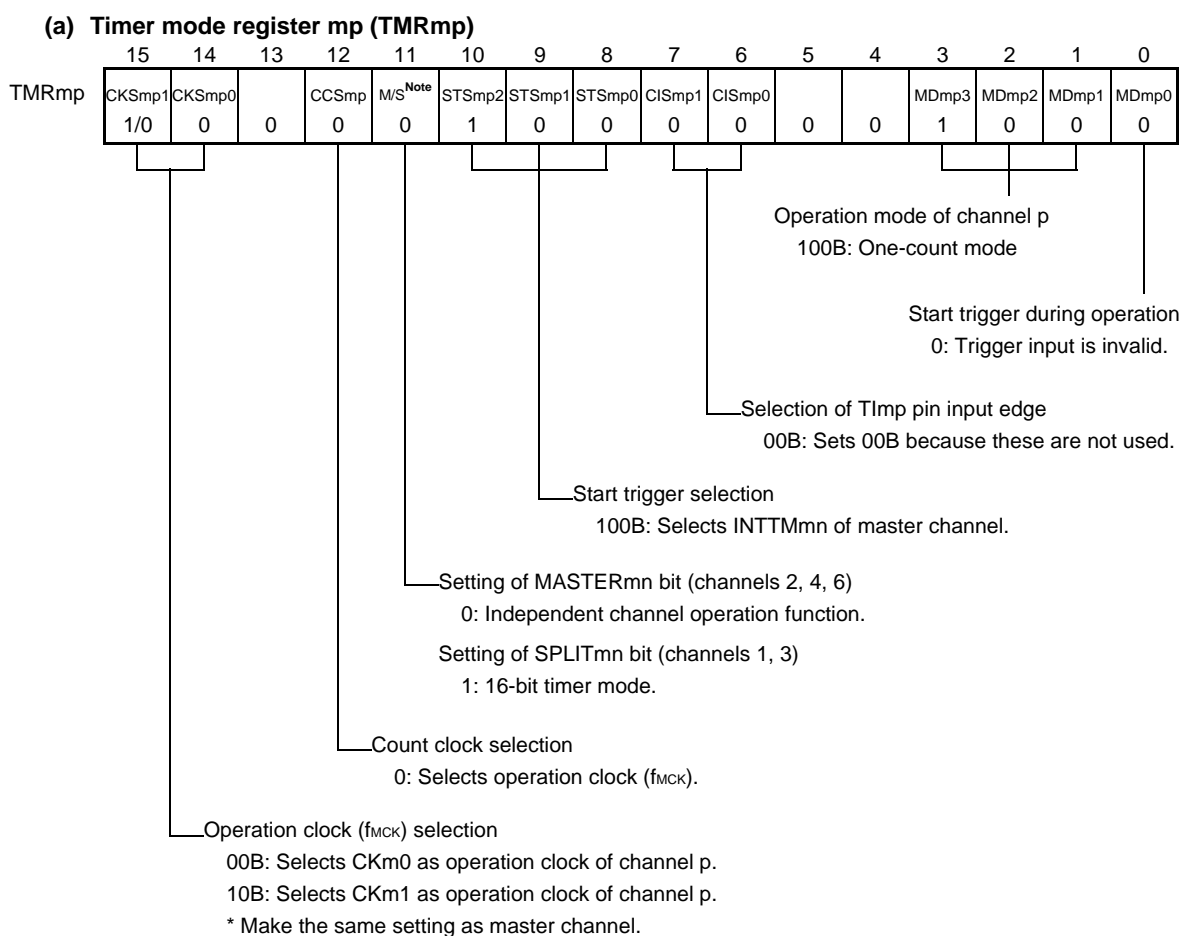


(e) Timer output mode register m (TOMm)



**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

Figure 6-64 Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Slave Channel)



## (b) Timer output register m (TOM)

	Bit p	
TOM	TOmp	0: Outputs 0 from TOmp. 1: Outputs 1 from TOmp.
	1/0	

## (c) Timer output enable register m (TOEm)

	Bit p	
TOEm	TOEmp	0: Stops the TOmp output operation by counting operation. 1: Enables the TOmp output operation by counting operation.
	1/0	

## (d) Timer output level register m (TOLm)

	Bit p	
TOLm	TOLmp	0: Positive logic output (active-high) 1: Negative logic output (active-low)
	1/0	

## (e) Timer output mode register m (TOMm)

	Bit p	
TOMm	TOMmp	1: Sets the slave channel output mode.
	1	

**Note** TMRm2, TMRm4, TMRm6: MASTERmn bit

TMRm1, TMRm3: SPLITmp bit

TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

&lt;R&gt;

Figure 6-65 Operation Procedure of One-Shot Pulse Output Function (1/2)

<R>

	Software Operation	Hardware Status
TAU default setting		Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable registers 0 (PER0) to 1.	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets the corresponding bit of the noise filter enable register 1 (NFEN1) to 1. Sets timer mode register mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels). An output delay is set to timer data register mn (TDRmn) of the master channel, and a pulse width is set to the TDRmp register of the slave channel.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channel. The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode). Sets the TOLmp bit. Sets the TOmp bit and determines default level of the TOmp output. Sets the TOEmp bit to 1 and enables operation of TOmp. Clears the port register and port mode register to 0.	The TOmp pin goes into Hi-Z output state.  The TOmp default setting level is output when the port mode register is in output mode and the port register is 0. TOmp does not change because channel stops operating. The TOmp pin outputs the TOmp set level.

(Note and Remark are listed on the next page.)

Figure 6-65 Operation Procedure of One-Shot Pulse Output Function (2/2)

	Software Operation	Hardware Status
<R> Operation start	Sets the TOEmp bit (slave) to 1 (only when operation is resumed). The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSM) are set to 1 at the same time. The TSmn and TSmp bits automatically return to 0 because they are trigger bits.	The TEMn and TEm bits are set to 1 and the master channel enters the start trigger detection (the valid edge of the TImn pin input is detected or the TSmn bit of the master channel is set to 1) wait status. Counter stops operating.
	Count operation of the master channel is started by start trigger detection of the master channel. <ul style="list-style-type: none"> <li>• Detects the TImn pin input valid edge.</li> <li>• Sets the TSmn bit of the master channel to 1 by software</li> </ul>	Master channel starts counting.
During operation	Set values of only the CISmn1 and CISmn0 bits of the TMRmn register can be changed. Set values of the TMRmp, TDRmn, TDRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed. The TCRmn and TCRmp registers can always be read. The TSRmn and TSRmp registers are not used. Set values of the TOm and TOEm registers by slave channel can be changed.	Master channel loads the value of the TDRmn register to timer count register mn (TCRmn) when the TImn pin valid input edge is detected, and the counter starts counting down. When the count value reaches TCRmn = 0000H, the INTTMmn output is generated, and the counter stops until the next valid edge is input to the TImn pin. The slave channel, triggered by INTTMmn of the master channel, loads the value of the TDRmp register to the TCRmp register, and the counter starts counting down. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Operation stop	The TTmn (master) and TTmp (slave) bits are set to 1 at the same time. The TTmn and TTmp bits automatically return to 0 because they are trigger bits.	TEMn, TEm = 0, and count operation stops. The TCRmn and TCRmp registers hold count value and stop. The TOmp output is not initialized but holds current status.
	The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit.	The TOmp pin outputs the TOmp set level.
TAU stop	To hold the TOmp pin output level Clears the TOmp bit to 0 after the value to be held is set to the port register. When holding the TOmp pin output level is not necessary Setting not required.	The TOmp pin output level is held by port function.
	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized. (The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.)

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

### 6.9.2 Operation as PWM function

Two channels can be used as a set to generate a pulse of any period and duty factor.

The period and duty factor of the output pulse can be calculated by the following expressions.

$$\begin{aligned} \text{Pulse period} &= \{\text{Set value of TDRmn (master)} + 1\} \times \text{Count clock period} \\ \text{Duty factor [\%]} &= \{\text{Set value of TDRmp (slave)}\} / \{\text{Set value of TDRmn (master)} + 1\} \times 100 \\ \text{0\% output:} & \quad \text{Set value of TDRmp (slave)} = 0000\text{H} \\ \text{100\% output:} & \quad \text{Set value of TDRmp (slave)} \geq \{\text{Set value of TDRmn (master)} + 1\} \end{aligned}$$

**Remark** The duty factor exceeds 100% if the set value of TDRmp (slave) > (set value of TDRmn (master) + 1), it summarizes to 100% output.

The master channel operates in the interval timer mode. If the channel start trigger bit (TSmn) of timer channel start register m (TSM) is set to 1, an interrupt (INTTMmn) is output, the value set to timer data register mn (TDRmn) is loaded to timer count register mn (TCRmn), and the counter counts down in synchronization with the count clock. When the counter reaches 0000H, INTTMmn is output, the value of the TDRmn register is loaded again to the TCRmn register, and the counter counts down. This operation is repeated until the channel stop trigger bit (TTmn) of timer channel stop register m (TTM) is set to 1.

If two channels are used to output a PWM waveform, the period until the master channel counts down to 0000H is the PWM output (TOmp) cycle.

The slave channel operates in one-count mode. By using INTTMmn from the master channel as a start trigger, the TCRmp register loads the value of the TDRmp register and the counter counts down to 0000H. When the counter reaches 0000H, it outputs INTTMmp and waits until the next start trigger (INTTMmn from the master channel) is generated.

If two channels are used to output a PWM waveform, the period until the slave channel counts down to 0000H is the PWM output (TOmp) duty.

PWM output (TOmp) goes to the active level one clock after the master channel generates INTTMmn and goes to the inactive level when the TCRmp register of the slave channel becomes 0000H.

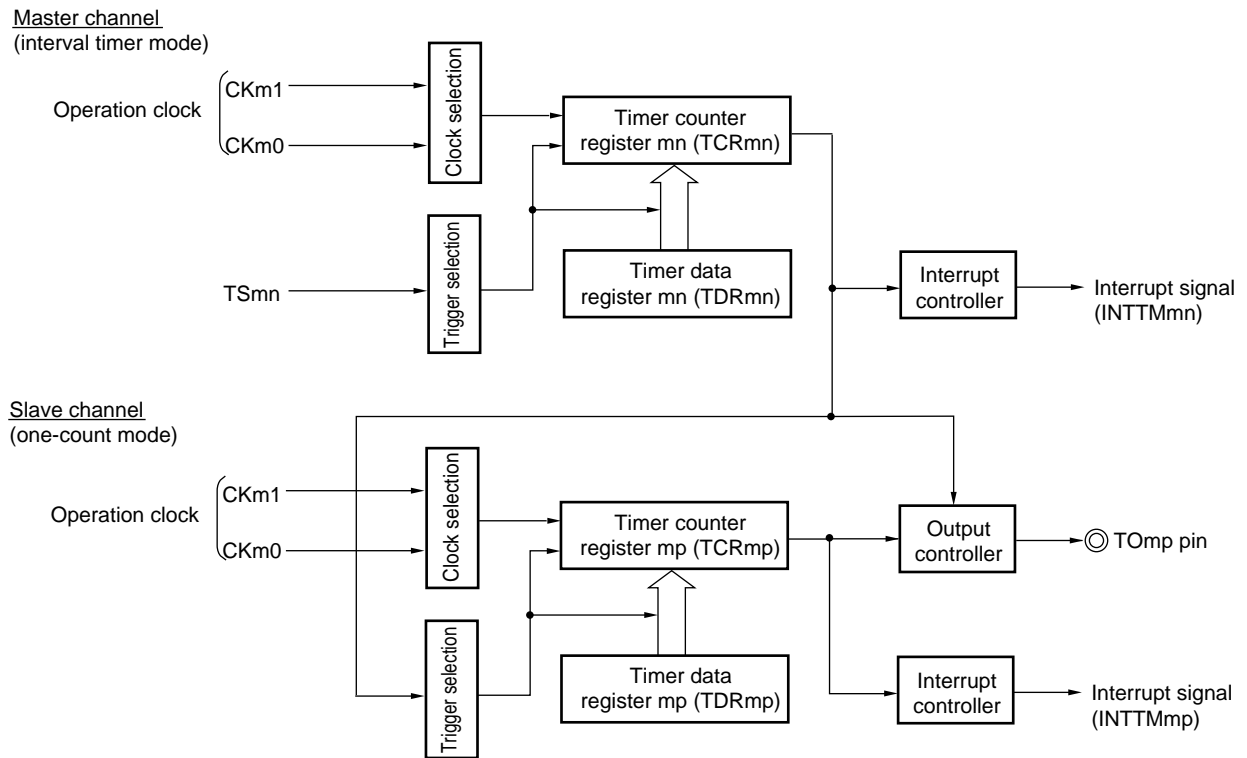
**Caution** To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel, a write access is necessary two times. The timing at which the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers is upon occurrence of INTTMmn of the master channel. Thus, when rewriting is performed split before and after occurrence of INTTMmn of the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, therefore, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel.

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

<R>

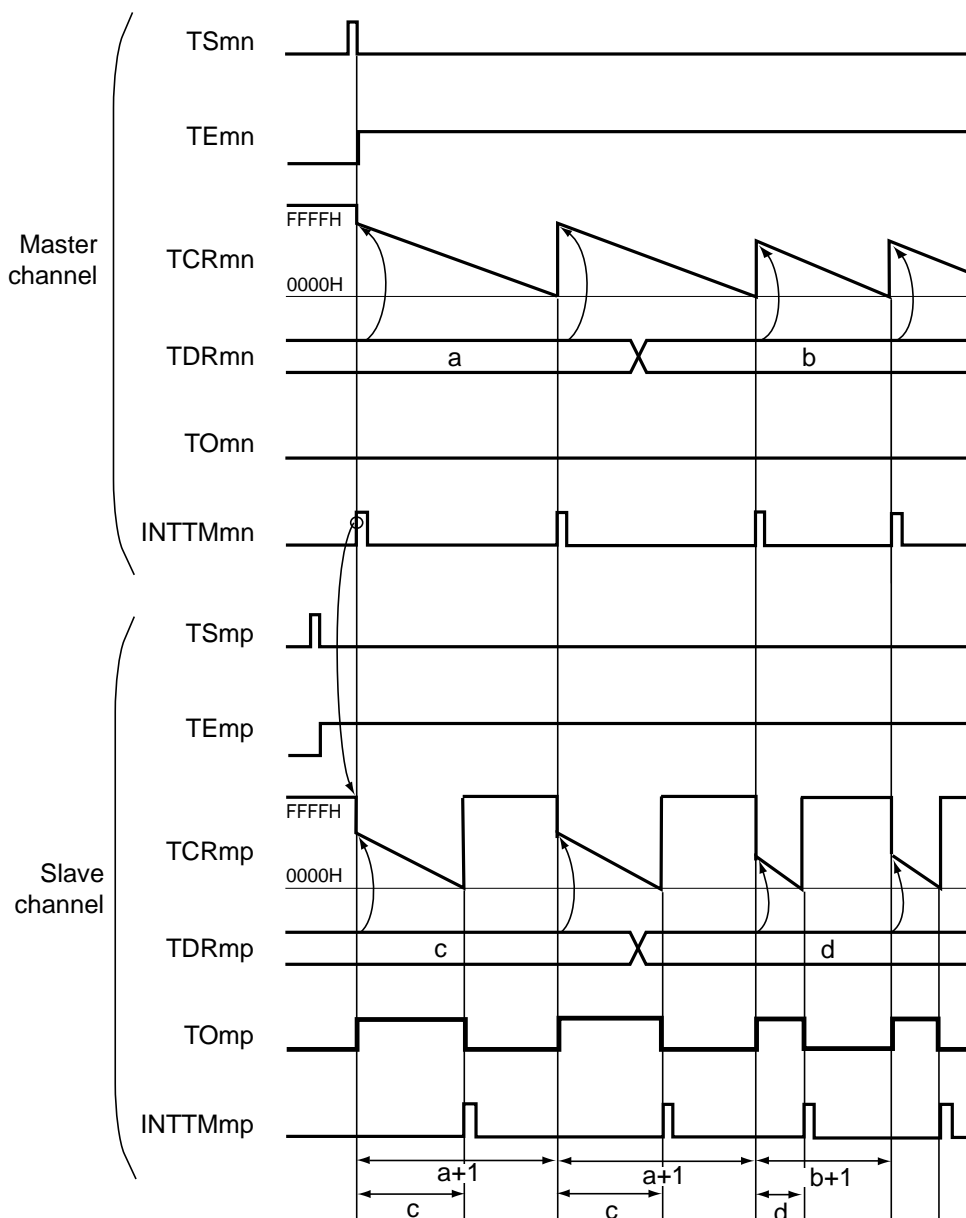
Figure 6-66 Block Diagram of Operation as PWM Function



&lt;R&gt;

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
 p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

Figure 6-67 Example of Basic Timing of Operation as PWM Function



**Remark 1.** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

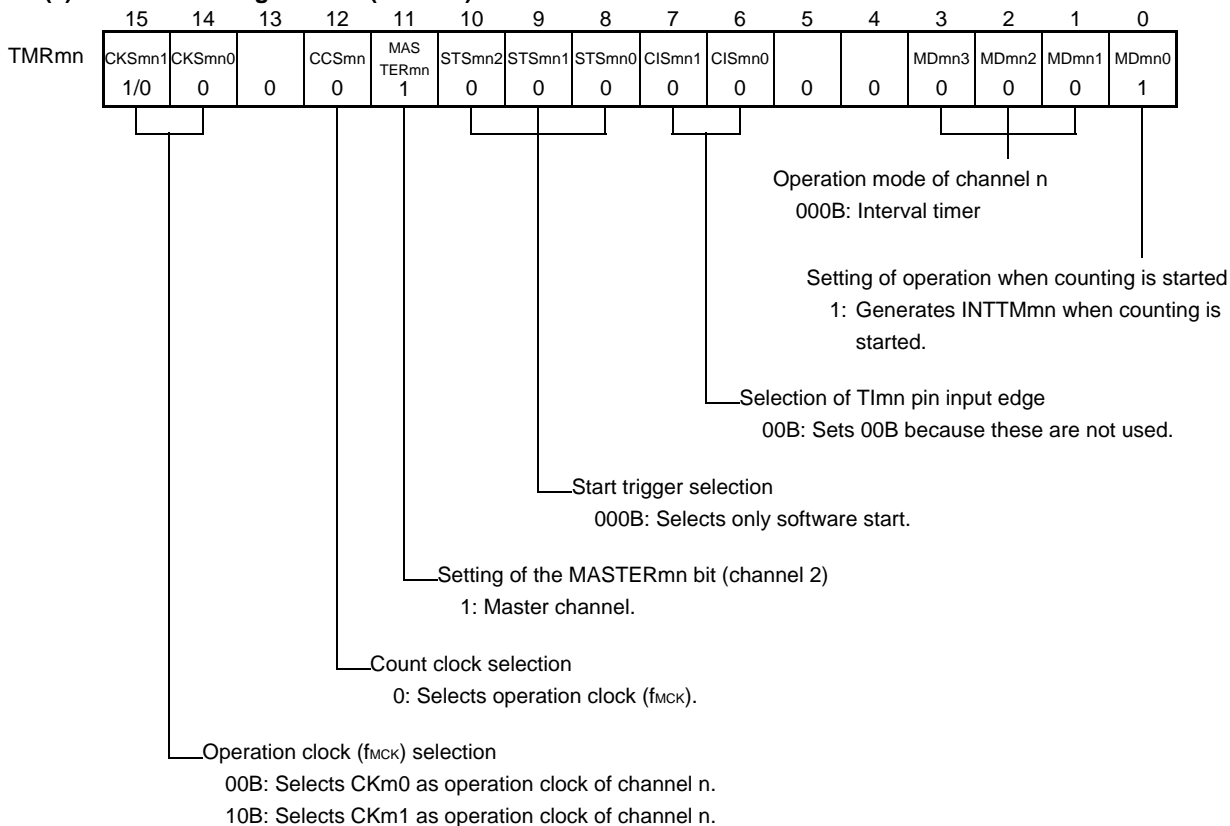
- 2. TSmn, TSmp: Bit n, p of timer channel start register m (TSm)
- TEmn, TEmp: Bit n, p of timer channel enable status register m (TEm)
- TCRmn, TCRmp: Timer count registers mn, mp (TCRmn, TCRmp)
- TDRmn, TDRmp: Timer data registers mn, mp (TDRmn, TDRmp)
- TOmn, TOmp: TOmn and TOmp pins output signal

<R>



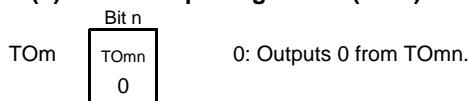
Figure 6-68 Example of Set Contents of Registers When PWM Function (Master Channel) Is Used

(a) Timer mode register mn (TMRmn)

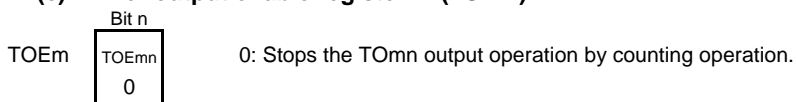


<R>

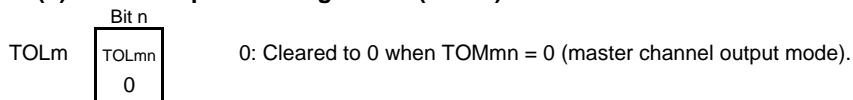
(b) Timer output register m (TOM)



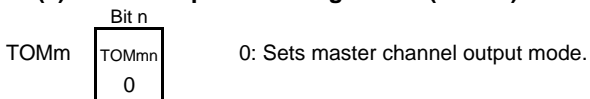
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



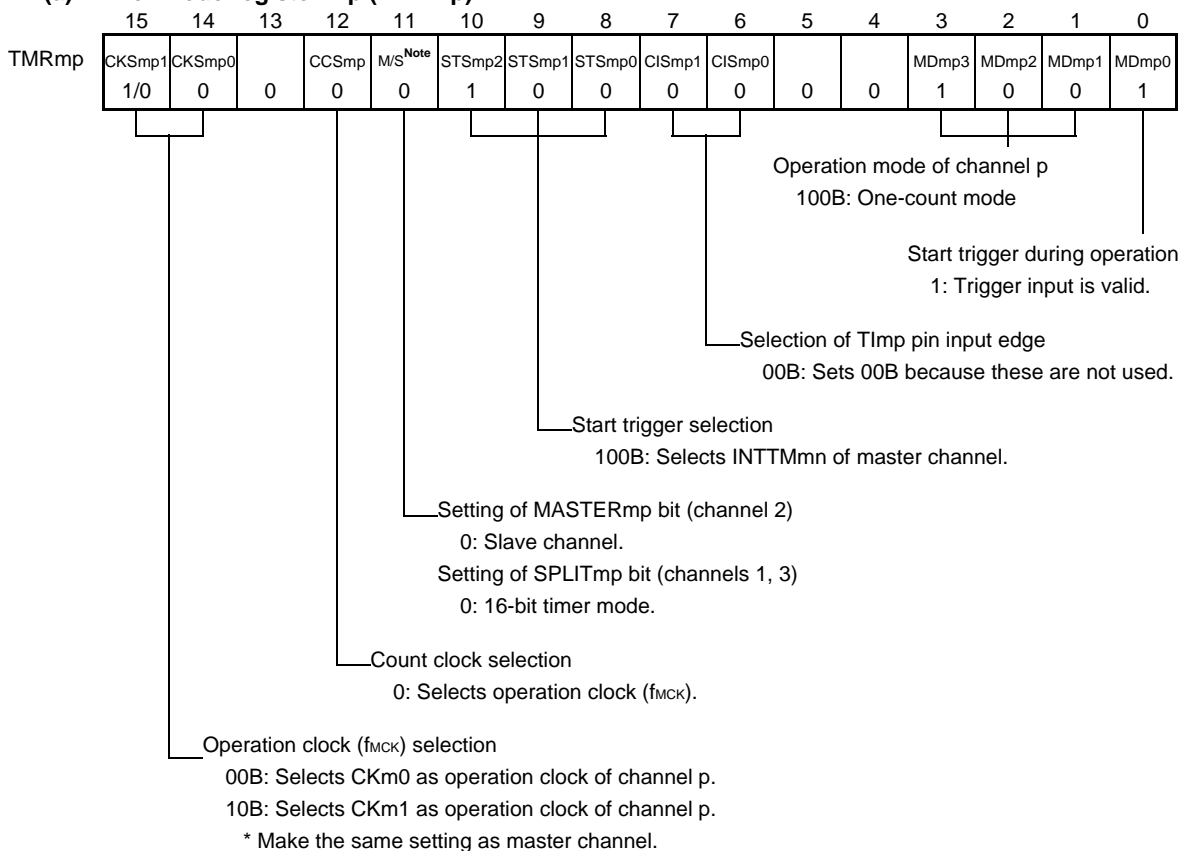
(e) Timer output mode register m (TOMm)



**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

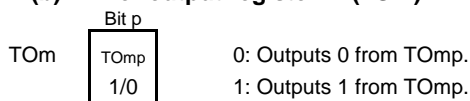
Figure 6-69 Example of Set Contents of Registers When PWM Function (Slave Channel) Is Used

(a) Timer mode register mp (TMRmp)

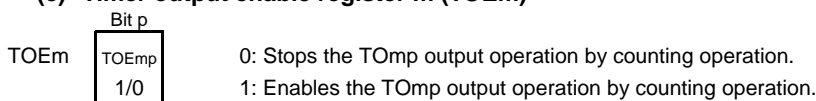


<R>

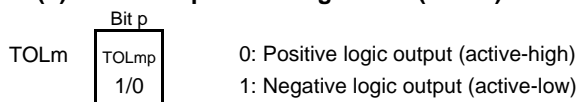
(b) Timer output register m (TOM)



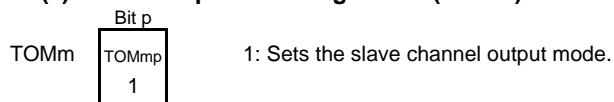
(c) Timer output enable register m (TOEm)



(d) Timer output level register m (TOLm)



(e) Timer output mode register m (TOMm)



**Note** TMRm5, TMRm7: Fixed to 0  
 TMRm1, TMRm3: SPLITmp bit

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
 p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

Figure 6-70 Operation Procedure When PWM Function Is Used (1/2)

<R>

	Software Operation	Hardware Status
TAU default setting		Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets timer mode registers mn, mp (TMRmn, TMRmp) of two channels to be used (determines operation mode of channels). An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp register of the slave channel.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channel. The TOMmp bit of timer output mode register m (TOMm) is set to 1 (slave channel output mode). Sets the TOLmp bit. Sets the TOmp bit and determines default level of the TOmp output. →	The TOmp pin goes into Hi-Z output state.
	Sets the TOEmp bit to 1 and enables operation of TOmp. → Clears the port register and port mode register to 0. →	The TOmp default setting level is output when the port mode register is in output mode and the port register is 0. TOmp does not change because channel stops operating. The TOmp pin outputs the TOmp set level.

(Remark is listed on the next page.)

Figure 6-70. Operation Procedure When PWM Function Is Used (2/2)

	Software Operation	Hardware Status
Operation start	<p>Sets the TOEmp bit (slave) to 1 (only when operation is resumed).</p> <p>The TSmn (master) and TSmp (slave) bits of timer channel start register m (TSM) are set to 1 at the same time.</p> <p>The TSmn and TSmp bits automatically return to 0 because they are trigger bits.</p>	<p>TEmn = 1, TEmP = 1</p> <p>▶ When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.</p>
During operation	<p>Set values of the TMRmn and TMRmp registers, TOMmn, TOMmp, TOLmn, and TOLmp bits cannot be changed.</p> <p>Set values of the TDRmn and TDRmp registers can be changed after INTTMmn of the master channel is generated.</p> <p>The TCRmn and TCRmp registers can always be read.</p> <p>The TSRmn and TSRmp registers are not used.</p>	<p>The counter of the master channel loads the TDRmn register value to timer count register mn (TCRmn), and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again.</p> <p>At the slave channel, the value of the TDRmp register is loaded to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output level of TOmp becomes active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped.</p> <p>After that, the above operation is repeated.</p>
Operation stop	<p>The TTmn (master) and TTmp (slave) bits are set to 1 at the same time.</p> <p>The TTmn and TTmp bits automatically return to 0 because they are trigger bits.</p>	<p>TEmn, TEmP = 0, and count operation stops.</p> <p>The TCRmn and TCRmp registers hold count value and stop.</p> <p>The TOmp output is not initialized but holds current status.</p>
	<p>The TOEmp bit of slave channel is cleared to 0 and value is set to the TOmp bit.</p>	<p>▶ The TOmp pin outputs the TOmp set level.</p>
TAU stop	<p>To hold the TOmp pin output level</p> <p>Clears the TOmp bit to 0 after the value to be held is set to the port register.</p> <p>When holding the TOmp pin output level is not necessary</p> <p>Setting not required.</p>	<p>▶ The TOmp pin output level is held by port function.</p>
	<p>The TAUmEN bit of the PER0 register is cleared to 0.</p>	<p>▶ Input clock supply for timer array unit 0 is stopped.</p> <p>All circuits are initialized and SFR of each channel is also initialized.</p> <p>(The TOmp bit is cleared to 0 and the TOmp pin is set to port mode.)</p>

Operation is resumed.

<R>

<R>

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
 p: Slave channel number (n = 0: p = 1, 2, 3, n = 2: p = 3)

### 6.9.3 Operation as multiple PWM output function

By extending the PWM function and using multiple slave channels, many PWM waveforms with different duty values can be output.

For example, when using two slave channels, the period and duty factor of an output pulse can be calculated by the following expressions.

$$\begin{aligned} \text{Pulse period} &= \{\text{Set value of TDRmn (master)} + 1\} \times \text{Count clock period} \\ \text{Duty factor 1 [\%]} &= \{\text{Set value of TDRmp (slave 1)}\} / \{\text{Set value of TDRmn (master)} + 1\} \times 100 \\ \text{Duty factor 2 [\%]} &= \{\text{Set value of TDRmq (slave 2)}\} / \{\text{Set value of TDRmn (master)} + 1\} \times 100 \end{aligned}$$

**Remark** Although the duty factor exceeds 100% if the set value of TDRmp (slave 1) > {set value of TDRmn (master) + 1} or if the {set value of TDRmq (slave 2)} > {set value of TDRmn (master) + 1}, it is summarized into 100% output.

Timer count register mn (TCRmn) of the master channel operates in the interval timer mode and counts the periods.

The TCRmp register of the slave channel 1 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmp pin. The TCRmp register loads the value of timer data register mp (TDRmp), using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmp = 0000H, TCRmp outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

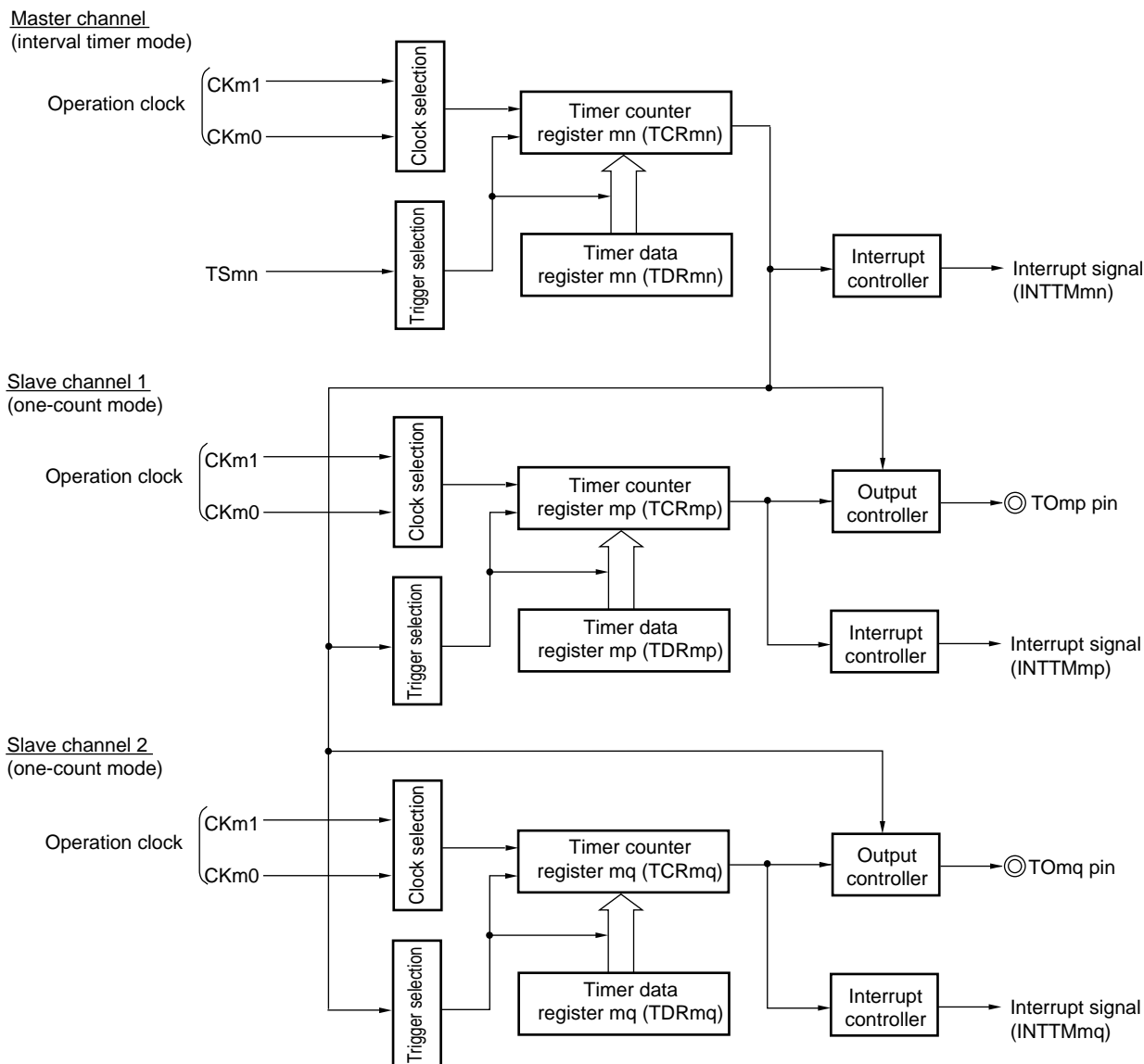
In the same way as the TCRmp register of the slave channel 1, the TCRmq register of the slave channel 2 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOMq pin. The TCRmq register loads the value of the TDRmq register, using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmq = 0000H, the TCRmq register outputs INTTMmq and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOMq becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmq = 0000H.

When channel 0 is used as the master channel as above, up to seven types of PWM signals can be output at the same time.

**Caution** To rewrite both timer data register mn (TDRmn) of the master channel and the TDRmp register of the slave channel 1, write access is necessary at least twice. Since the values of the TDRmn and TDRmp registers are loaded to the TCRmn and TCRmp registers after INTTMmn is generated from the master channel, if rewriting is performed separately before and after generation of INTTMmn from the master channel, the TOmp pin cannot output the expected waveform. To rewrite both the TDRmn register of the master and the TDRmp register of the slave, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel (This applies also to the TDRmq register of the slave channel 2).

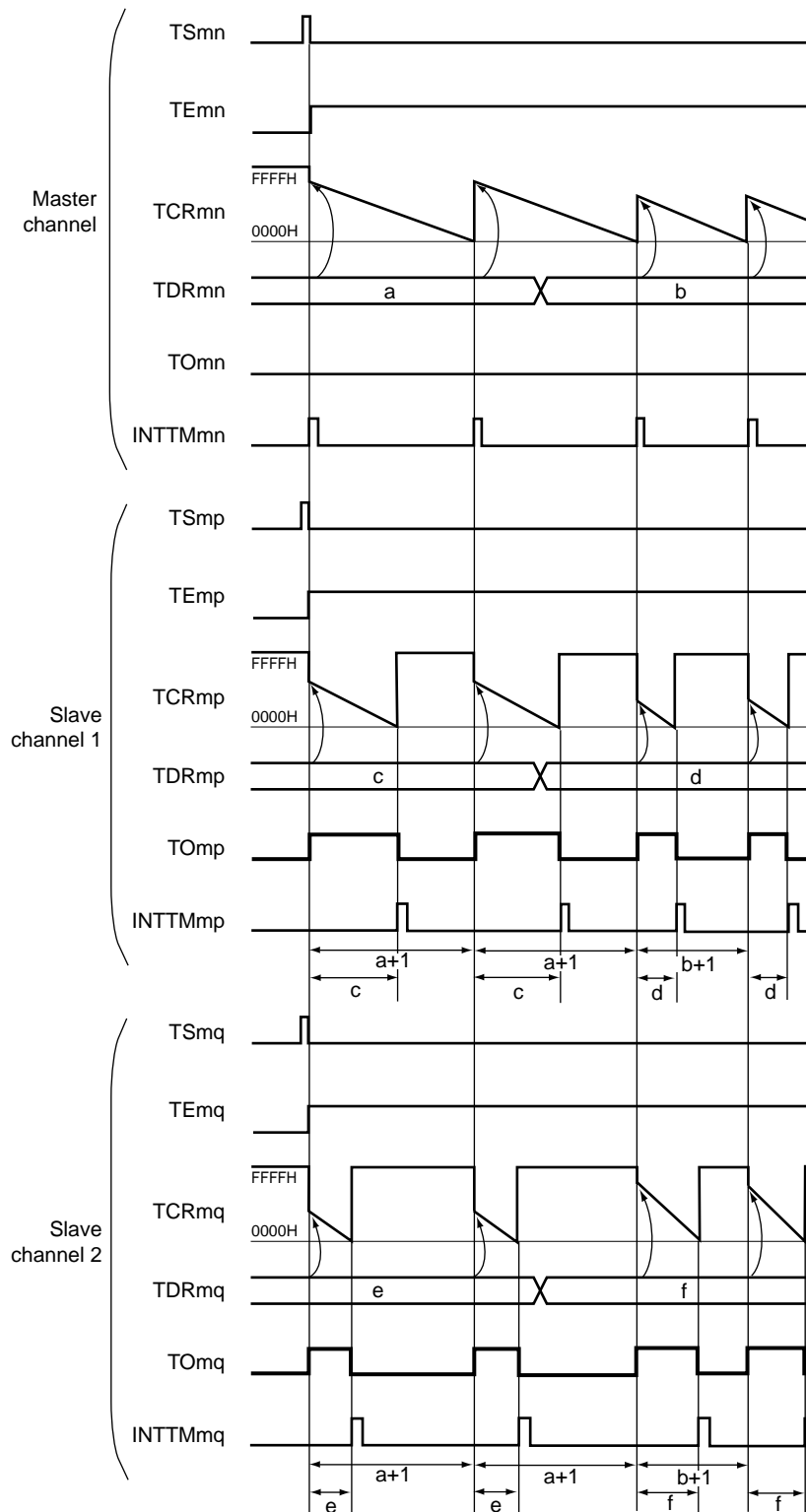
**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
 p: Slave channel number 1, q: Slave channel number 2  
 n < p < q ≤ 3 (Where p and q are integers greater than n)

Figure 6-71. Block Diagram of Operation as Multiple PWM Output Function (output two types of PWMs)



**Remark** m: Unit number ( $m = 0$ ), n: Master channel number ( $n = 0, 2$ )  
 p: Slave channel number 1, q: Slave channel number 2  
 $n < p < q \leq 3$  (Where p and q are integers greater than n)

**Figure 6-72. Example of Basic Timing of Operation as Multiple PWM Output Function (Output two types of PWMs)**

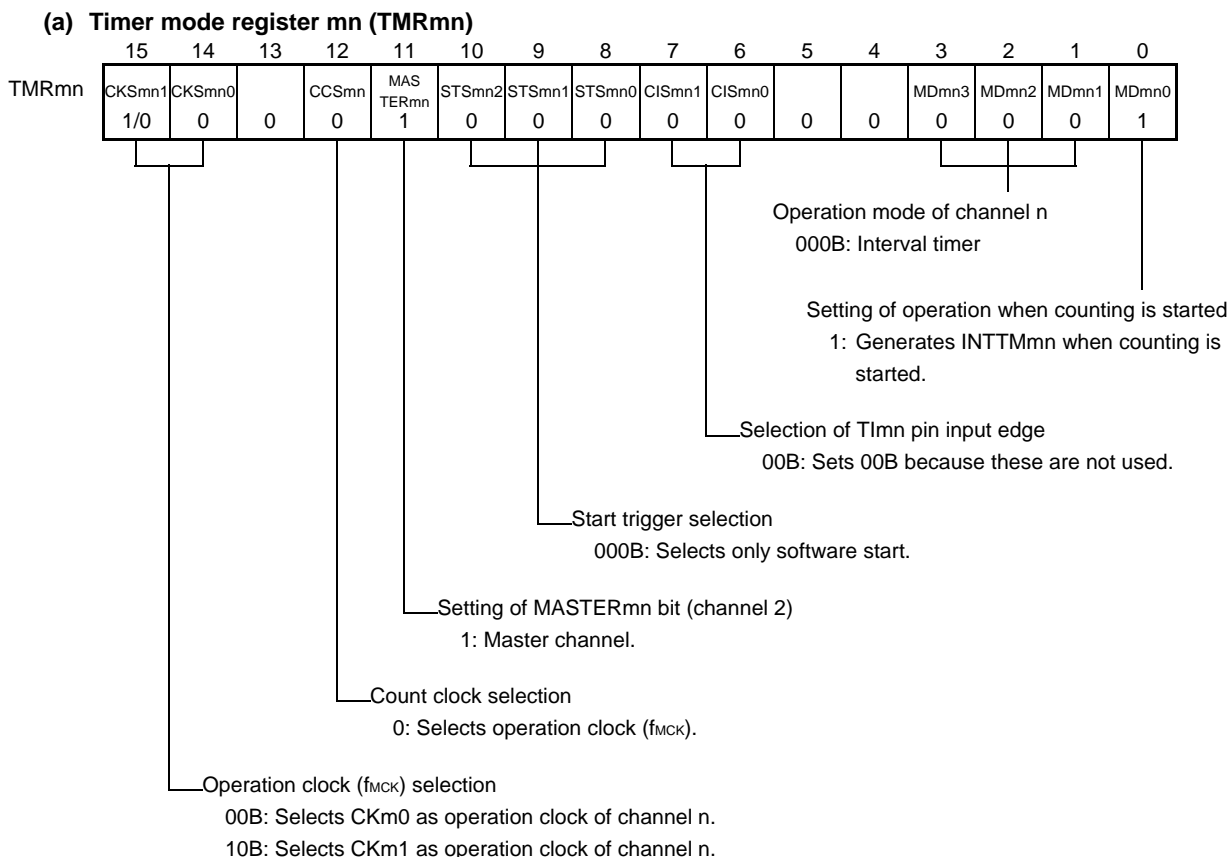


(Remarks are listed on the next page.)

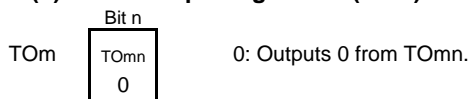
- Remarks**
1. m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
p: Slave channel number 1, q: Slave channel number 2  
n < p < q ≤ 3 (Where p and q are integers greater than n)
  2. TSmn, TSmp, TSmq: Bit n, p, q of timer channel start register m (TSm)  
TEmn, TEmp, TEMq: Bit n, p, q of timer channel enable status register m (TEm)  
TCRmn, TCRmp, TCRmq: Timer count registers mn, mp, mq (TCRmn, TCRmp, TCRmq)  
TDRmn, TDRmp, TDRmq: Timer data registers mn, mp, mq (TDRmn, TDRmp, TDRmq)  
TOmn, TOmp, TOmq: TOmn, TOmp, and TOmq pins output signal



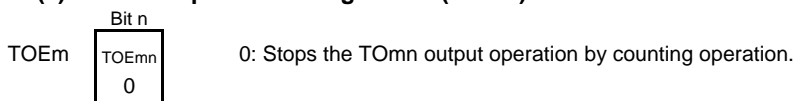
**Figure 6-73. Example of Set Contents of Registers  
When Multiple PWM Output Function (Master Channel) Is Used**



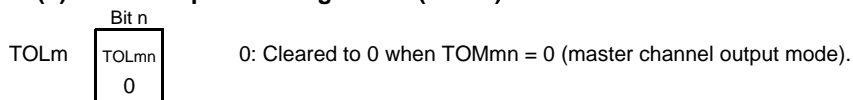
(b) **Timer output register m (TOM)**



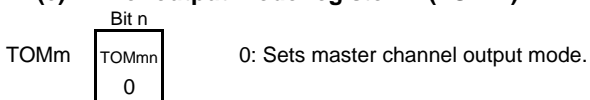
(c) **Timer output enable register m (TOEm)**



(d) **Timer output level register m (TOLm)**



(e) **Timer output mode register m (TOMm)**

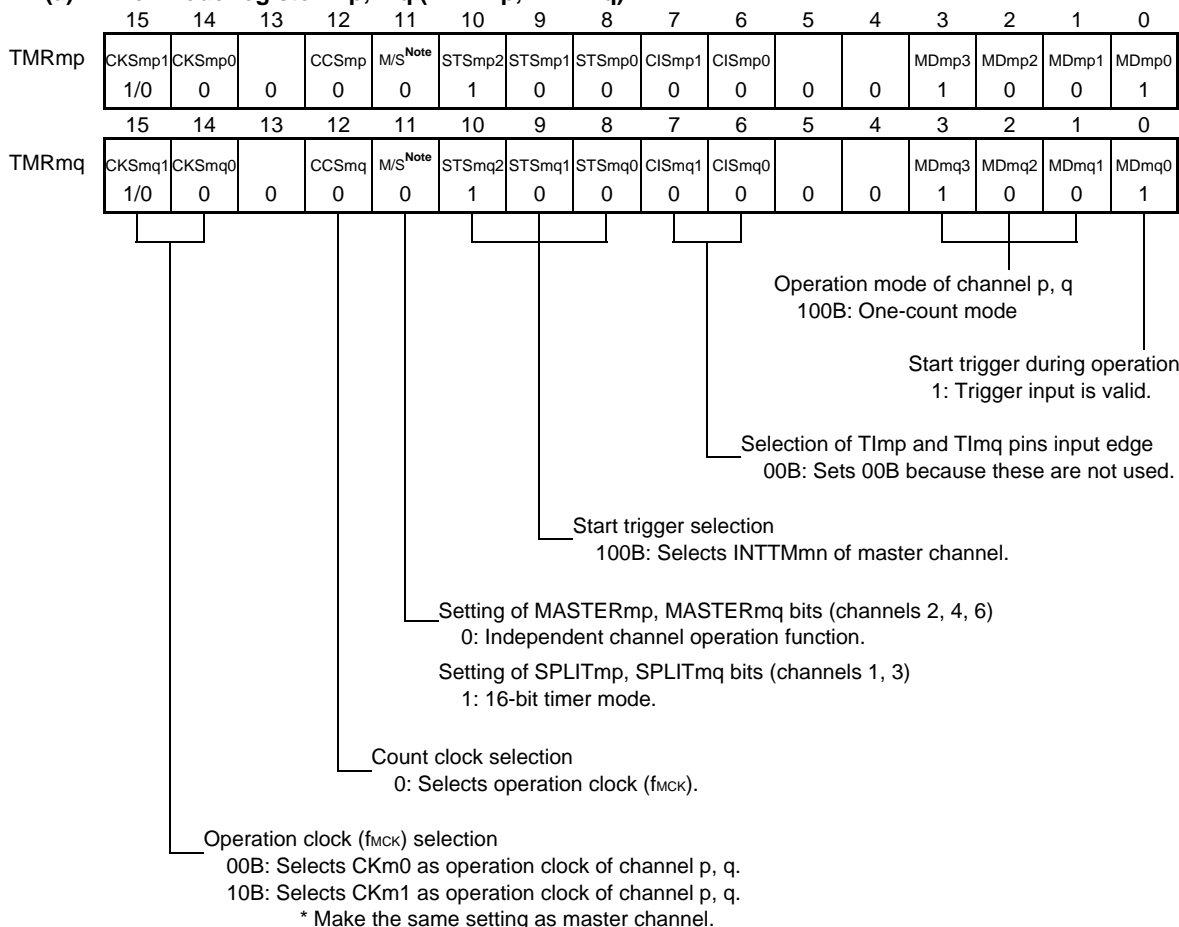


**Note** MRm2: MASTERmn = 1  
 TMRm0: Fixed to 0

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)

**Figure 6-74. Example of Set Contents of Registers**  
**When Multiple PWM Output Function (Slave Channel) Is Used (output two types of PWMs)**

**(a) Timer mode register mp, mq (TMRmp, TMRmq)**



**(b) Timer output register m (TOM)**

	Bit q	Bit p	
TOM	TOMq	TOMP	
	1/0	1/0	0: Outputs 0 from TOMP or TOMq. 1: Outputs 1 from TOMP or TOMq.

**(c) Timer output enable register m (TOEm)**

	Bit q	Bit p	
TOEm	TOEmq	TOEmp	
	1/0	1/0	0: Stops the TOMP or TOMq output operation by counting operation. 1: Enables the TOMP or TOMq output operation by counting operation.

**(d) Timer output level register m (TOLm)**

	Bit q	Bit p	
TOLm	TOLmq	TOLmp	
	1/0	1/0	0: Positive logic output (active-high) 1: Negative logic output (active-low)

**(e) Timer output mode register m (TOMm)**

	Bit q	Bit p	
TOMm	TOMmq	TOMmp	
	1	1	1: Sets the slave channel output mode.

**Note** TMRm2, TMRm4, TMRm6: MASTERmp, MASTERmq bit  
 TMRm1, TMRm3: SPLITmp, SPLIT0q bit  
 TMRm5, TMRm7: Fixed to 0

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
 p: Slave channel number 1, q: Slave channel number 2  
 n < p < q ≤ 3 (Where p and q are integers greater than n)

Figure 6-75. Operation Procedure When Multiple PWM Output Function Is Used (1/2)

	Software Operation	Hardware Status
<R> TAU default setting		Input clock supply for timer array unit 0 is stopped. (Clock supply is stopped and writing to each register is disabled.)
	Sets the TAUmEN bit of peripheral enable register 0 (PER0) to 1. →	Input clock supply for timer array unit 0 is supplied. Each channel stops operating. (Clock supply is started and writing to each register is enabled.)
	Sets timer clock select register m (TPSm). Determines clock frequencies of CKm0 and CKm1.	
Channel default setting	Sets timer mode registers mn, mp, 0q (TMRmn, TMRmp, TMRmq) of each channel to be used (determines operation mode of channels). An interval (period) value is set to timer data register mn (TDRmn) of the master channel, and a duty factor is set to the TDRmp and TDRmq registers of the slave channels.	Channel stops operating. (Clock is supplied and some power is consumed.)
	Sets slave channels. The TOMmp and TOMmq bits of timer output mode register m (TOMm) are set to 1 (slave channel output mode). Sets the TOLmp and TOLmq bits. Sets the TOmp and TOMq bits and determines default level of the TOmp and TOMq outputs. →	The TOmp and TOMq pins go into Hi-Z output state.  The TOmp and TOMq default setting levels are output when the port mode register is in output mode and the port register is 0.
	Sets the TOEmp and TOEmq bits to 1 and enables operation of TOmp and TOMq. →	TOmp and TOMq do not change because channels stop operating.
	Clears the port register and port mode register to 0. →	The TOmp and TOMq pins output the TOmp and TOMq set levels.

(Remark is listed on the next page.)

Figure 6-75. Operation Procedure When Multiple PWM Output Function Is Used (2/2)

	Software Operation	Hardware Status
Operation start	(Sets the TOEmp and TOEmq (slave) bits to 1 only when resuming operation.) The TSmn bit (master), and TSmp and TSmq (slave) bits of timer channel start register m (TSm) are set to 1 at the same time. The TSmn, TSmp, and TSmq bits automatically return to 0 because they are trigger bits.	TEmn = 1, TEmq = 1 When the master channel starts counting, INTTMmn is generated. Triggered by this interrupt, the slave channel also starts counting.
During operation	Set values of the TMRmn, TMRmp, TMRmq registers, TOMmn, TOMmp, TOMmq, TOLmn, TOLmp, and TOLmq bits cannot be changed. Set values of the TDRmn, TDRmp, and TDRmq registers can be changed after INTTMmn of the master channel is generated. The TCRmn, TCRmp, and TCRmq registers can always be read. The TSRmn, TSRmp, and TSR0q registers are not used.	The counter of the master channel loads the TDRmn register value to timer count register mn (TCRmn) and counts down. When the count value reaches TCRmn = 0000H, INTTMmn output is generated. At the same time, the value of the TDRmn register is loaded to the TCRmn register, and the counter starts counting down again. At the slave channel 1, the values of the TDRmp register are transferred to the TCRmp register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOmp become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmp = 0000H, and the counting operation is stopped. At the slave channel 2, the values of the TDRmq register are transferred to TCRmq register, triggered by INTTMmn of the master channel, and the counter starts counting down. The output levels of TOMq become active one count clock after generation of the INTTMmn output from the master channel. It becomes inactive when TCRmq = 0000H, and the counting operation is stopped. After that, the above operation is repeated.
Operation stop	The TTmn bit (master), TTmp, and TTmq (slave) bits are set to 1 at the same time. The TTmn, TTmp, and TTmq bits automatically return to 0 because they are trigger bits.	TEmn, TEmq = 0, and count operation stops. The TCRmn, TCRmp, and TCRmq registers hold count value and stop. The TOmp and TOMq output are not initialized but hold current status.
	The TOEmp and TOEmq bits of slave channels are cleared to 0 and value is set to the TOmp and TOMq bits.	The TOmp and TOMq pins output the TOmp and TOMq set levels.
TAU stop	To hold the TOmp and TOMq pin output levels Clears the TOmp and TOMq bits to 0 after the value to be held is set to the port register. When holding the TOmp and TOMq pin output levels are not necessary Setting not required	The TOmp and TOMq pin output levels are held by port function.
	The TAUmEN bit of the PER0 register is cleared to 0.	Input clock supply for timer array unit 0 is stopped. All circuits are initialized and SFR of each channel is also initialized. (The TOmp and TOMq bits are cleared to 0 and the TOmp and TOMq pins are set to port mode.)

Operation is resumed.

<R>

**Remark** m: Unit number (m = 0), n: Master channel number (n = 0, 2)  
p: Slave channel number 1, q: Slave channel number 2  
n < p < q ≤ 3 (Where p and q are a consecutive integer greater than n)

## 6.10 Cautions When Using Timer Array Unit

### 6.10.1 Cautions when using timer output

Depends on products, a pin is assigned as a timer output and other alternate functions. In this case, outputs of the other alternate functions must be set in initial status.

- Using TO03 output assigned to the P12

So that the alternated PCLBUZ0 output becomes 0, not only set the port mode register (the PM12 bit) and the port register (the P12 bit) to 0, but also use the bit 7 of the clock output select register 0 (CKS0) with the same setting as the initial status.

## CHAPTER 7 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER

### 7.1 Functions of Clock Output/Buzzer Output Controller

The clock output controller is intended for carrier output during remote controlled transmission and clock output for supply to peripheral ICs.

Buzzer output is a function to output a square wave of buzzer frequency.

One pin can be used to output a clock or buzzer sound.

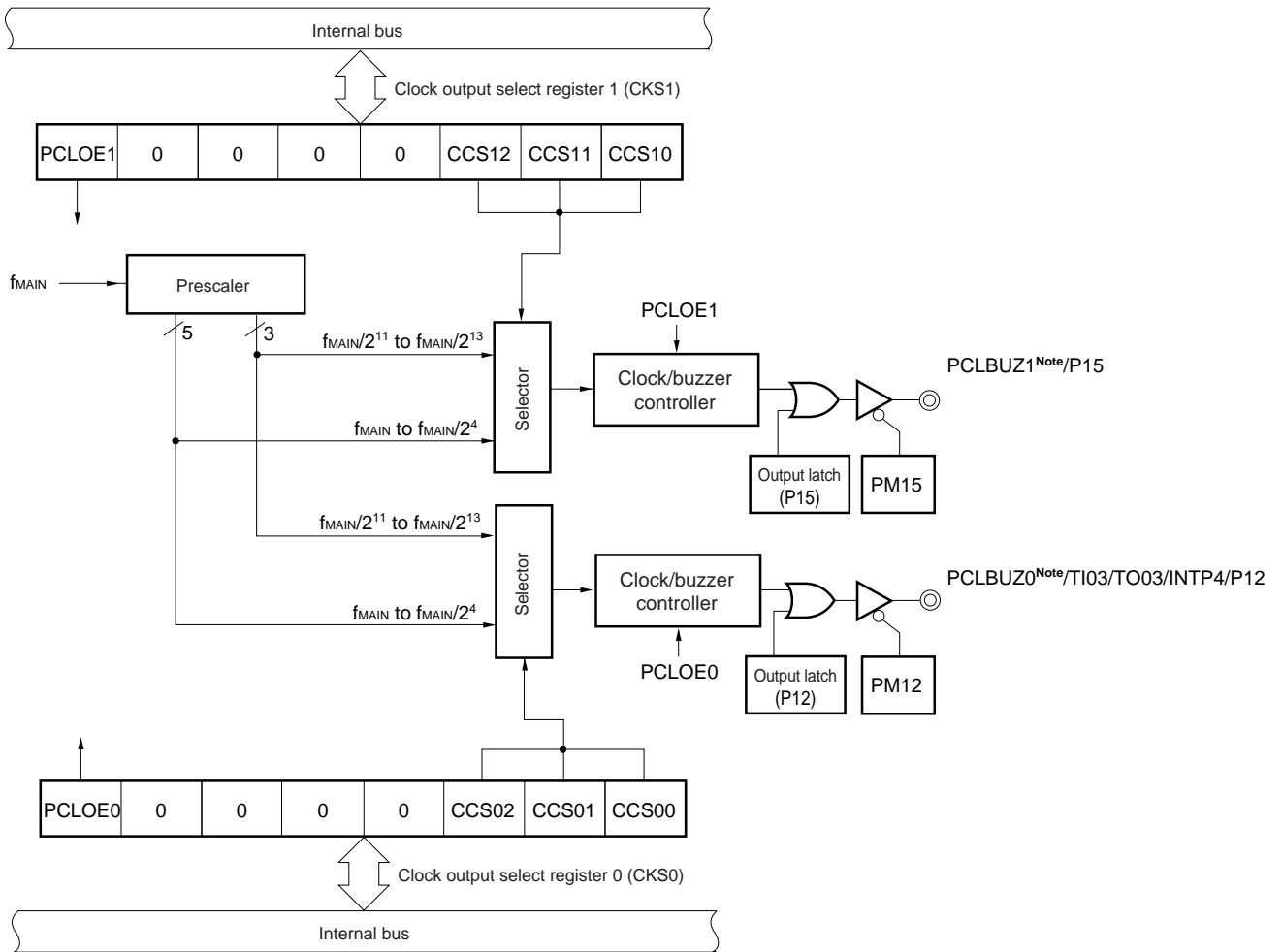
Two output pins, PCLBUZ0 and PCLBUZ1, are available.

The PCLBUZn pin outputs a clock selected by clock output select register n (CKSn).

Figure 7-1 shows the block diagram of clock output/buzzer output controller.

**Remark** n = 0, 1

Figure 7-1. Block Diagram of Clock Output/Buzzer Output Controller



**Note** For output frequencies available from PCLBUZ0 and PCLBUZ1, see 27.4 AC Characteristics.

## 7.2 Configuration of Clock Output/Buzzer Output Controller

The clock output/buzzer output controller includes the following hardware.

**Table 7-1. Configuration of Clock Output/Buzzer Output Controller**

Item	Configuration
Control registers	Clock output select registers n (CKSn) Port mode register 1 (PM1) Port register 1 (P1)

## 7.3 Registers Controlling Clock Output/Buzzer Output Controller

<R>

### 7.3.1 Clock output select registers n (CKSn)

These registers set output enable/disable for clock output or for the buzzer frequency output pin (PCLBUZn), and set the output clock.

Select the clock to be output from the PCLBUZn pin by using the CKSn register.

The CKSn register are set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.



**Figure 7-2. Format of Clock Output Select Register n (CKSn)**

Address: FFFA5H (CKS0), FFFA6H (CKS1) After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	0
CKSn	PCLOEn	0	0	0	0	CCSn2	CCSn1	CCSn0

PCLOEn	PCLBUZn pin output enable/disable specification
0	Output disable (default)
1	Output enable

CCSn2	CCSn1	CCSn0	PCLBUZn pin output clock selection				
				$f_{\text{MAIN}} =$ 5 MHz	$f_{\text{MAIN}} =$ 10 MHz	$f_{\text{MAIN}} =$ 20 MHz	$f_{\text{MAIN}} =$ 32 MHz
0	0	0	$f_{\text{MAIN}}$	5 MHz	Setting prohibited <sup>Note</sup>	Setting prohibited <sup>Note</sup>	Setting prohibited <sup>Note</sup>
0	0	1	$f_{\text{MAIN}}/2$	2.5 MHz	5 MHz	Setting prohibited <sup>Note</sup>	Setting prohibited <sup>Note</sup>
0	1	0	$f_{\text{MAIN}}/2^2$	1.25 MHz	2.5 MHz	5 MHz	8 MHz <sup>Note</sup>
0	1	1	$f_{\text{MAIN}}/2^3$	625 kHz	1.25 MHz	2.5 MHz	4 MHz
1	0	0	$f_{\text{MAIN}}/2^4$	312.5 kHz	625 kHz	1.25 MHz	2 MHz
1	0	1	$f_{\text{MAIN}}/2^{11}$	2.44 kHz	4.88 kHz	9.76 kHz	15.63 kHz
1	1	0	$f_{\text{MAIN}}/2^{12}$	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	1	$f_{\text{MAIN}}/2^{13}$	610 Hz	1.22 kHz	2.44 kHz	3.91 kHz

**Note** Use the output clock within a range of 8 MHz. See **27.4 AC Characteristics** for details.

- Cautions**
1. Change the output clock after disabling clock output (PCLOEn = 0).
  2. To shift to STOP mode when the main system clock is selected (CSELn = 0), set PCLOEn = 0 before executing the STOP instruction.

- Remarks**
1. n = 0, 1
  2.  $f_{\text{MAIN}}$ : Main system clock frequency

**<R> 7.3.2 Registers controlling port functions of pins to be used for clock or buzzer output**

Using a port pin for clock or buzzer output requires setting of the registers that control the port functions multiplexed on the target pin (port mode register (PMxx), port register (Pxx)). For details, see 4.3.1 Port mode registers (PMxx) and 4.3.2 Port registers (Pxx).

Specifically, using a port pin with a multiplexed clock or buzzer output function (e.g. P12/TI03/TO03/INTP4/PCLBUZ0, P15/PCLBUZ1) for clock or buzzer output, requires setting the corresponding bits in the port mode register (PMxx) and port register (Pxx) to 0.

**<R>** Example: When P12/TI03/TO03/INTP4/PCLBUZ0 is to be used for clock or buzzer output

Set the PM12 bit of port mode register 1 to 0.

Set the P12 bit of port register 1 to 0.

## 7.4 Operations of Clock Output/Buzzer Output Controller

One pin can be used to output a clock or buzzer sound.

The PCLBUZ0 pin outputs a clock/buzzer selected by the clock output select register 0 (CKS0).

The PCLBUZ1 pin outputs a clock/buzzer selected by the clock output select register 1 (CKS1).

### 7.4.1 Operation as output pin

The PCLBUZn pin is output as the following procedure.

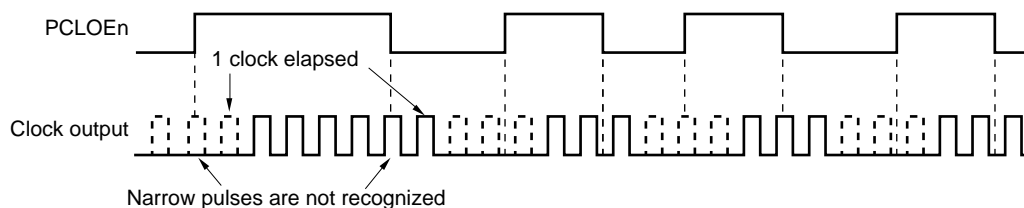
- <R>
- <1> Set 0 in the bit of the port mode register (PMxx) and port register (Pxx) which correspond to the port which has a pin used as the PCLBUZ0 pin.
  - <2> Select the output frequency with bits 0 to 3 (CCSn0 to CCSn2) of the clock output select register (CKSn) of the PCLBUZn pin (output in disabled status).
  - <3> Set bit 7 (PCLOEn) of the CKSn register to 1 to enable clock/buzzer output.

**Remarks 1.** The controller used for outputting the clock starts or stops outputting the clock one clock after enabling or disabling clock output (PCLOEn bit) is switched. At this time, pulses with a narrow width are not output.

Figure 7-3 shows enabling or stopping output using the PCLOEn bit and the timing of outputting the clock.

2. n = 0, 1

**Figure 7-3. Timing of Outputting Clock from PCLBUZn Pin**



## 7.5 Cautions of Clock Output/Buzzer Output Controller

- <R>
- When the main system clock is selected for the PCLBUZn output (CSEL = 0), if STOP mode is entered within 1.5 clock cycles output from the PCLBUZn pin after the output is disabled (PCLOEn = 0), the PCLBUZn output width becomes shorter.

## CHAPTER 8 WATCHDOG TIMER

### 8.1 Functions of Watchdog Timer

<R> The counting operation of the watchdog timer is set by the option byte (000C0H).

The watchdog timer operates on the low-speed on-chip oscillator clock.

The watchdog timer is used to detect an inadvertent program loop. If a program loop is detected, an internal reset signal is generated.

Program loop is detected in the following cases.

- If the watchdog timer counter overflows
- If a 1-bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
- If data other than "ACH" is written to the WDTE register
- If data is written to the WDTE register during a window close period

When a reset occurs due to the watchdog timer, bit 4 (WDTRF) of the reset control flag register (RESF) is set to 1. For details of the RESF register, see **CHAPTER 17 RESET FUNCTION**.

When  $75\% + 1/2/f_{IL}$  of the overflow time is reached, an interval interrupt can be generated.

### 8.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

**Table 8-1. Configuration of Watchdog Timer**

Item	Configuration
Counter	Internal counter (17 bits)
Control register	Watchdog timer enable register (WDTE)

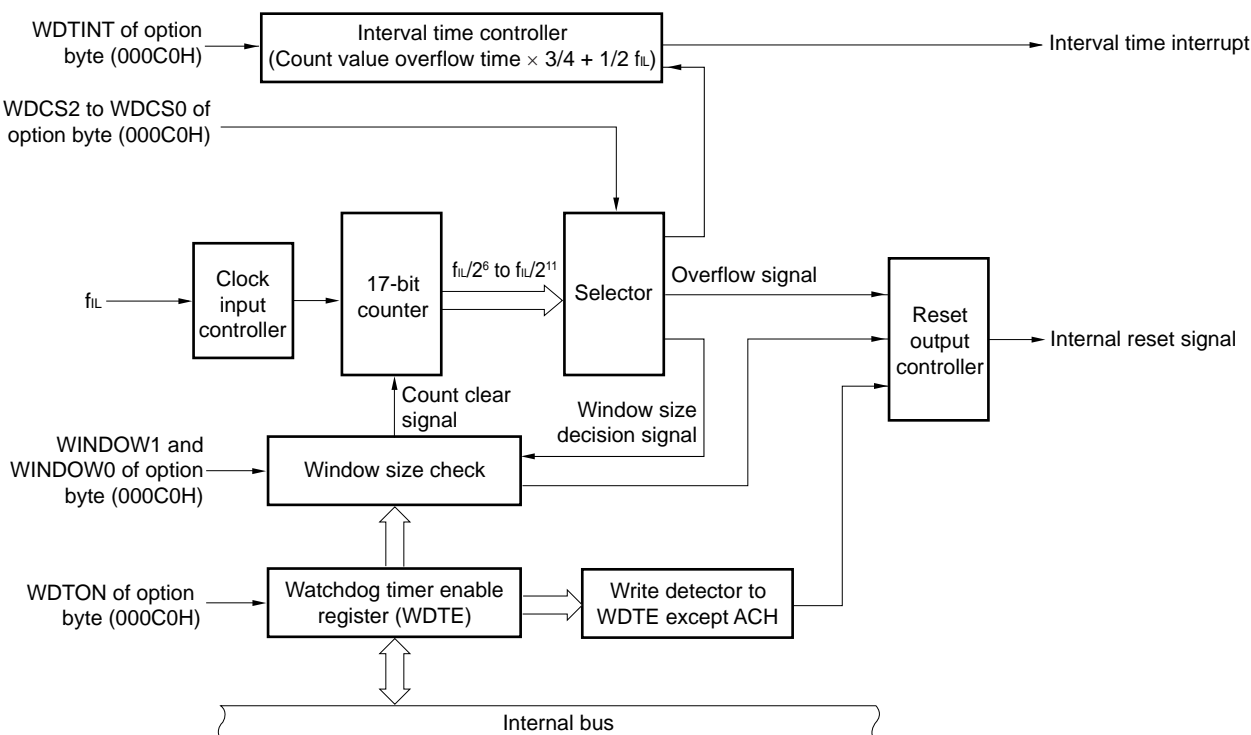
How the counter operation is controlled, overflow time, window open period, and interval interrupt are set by the option byte.

**Table 8-2. Setting of Option Bytes and Watchdog Timer**

Setting of Watchdog Timer	Option Byte (000C0H)
Watchdog timer interval interrupt	Bit 7 (WDTINT)
Window open period	Bits 6 and 5 (WINDOW1, WINDOW0)
Controlling counter operation of watchdog timer	Bit 4 (WDTON)
Overflow time of watchdog timer	Bits 3 to 1 (WDCS2 to WDCS0)
Controlling counter operation of watchdog timer (in HALT/STOP mode)	Bit 0 (WDSTBYON)

**Remark** For the option byte, see **CHAPTER 22 OPTION BYTE**.

**Figure 8-1. Block Diagram of Watchdog Timer**



### 8.3 Register Controlling Watchdog Timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

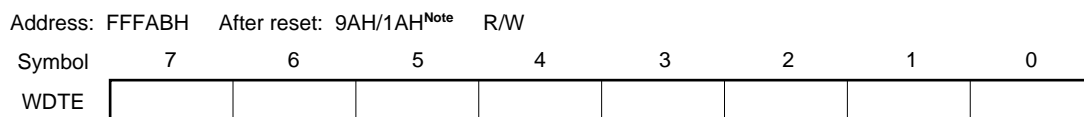
#### 8.3.1 Watchdog timer enable register (WDTE)

Writing "ACH" to the WDTE register clears the watchdog timer counter and starts counting again.

This register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 9AH or 1AH<sup>Note</sup>.

**Figure 8-2. Format of Watchdog Timer Enable Register (WDTE)**



**Note** The WDTE register reset value differs depending on the WDTON bit setting value of the option byte (000C0H). To operate watchdog timer, set the WDTON bit to 1.

WDTON Bit Setting Value	WDTE Register Reset Value
0 (watchdog timer count operation disabled)	1AH
1 (watchdog timer count operation enabled)	9AH

- Cautions**
1. If a value other than "ACH" is written to the WDTE register, an internal reset signal is generated.
  2. If a 1-bit memory manipulation instruction is executed for the WDTE register, an internal reset signal is generated.
  3. The value read from the WDTE register is 9AH/1AH (this differs from the written value (ACH)).

## 8.4 Operation of Watchdog Timer

### 8.4.1 Controlling operation of watchdog timer

- When the watchdog timer is used, its operation is specified by the option byte (000C0H).
  - Enable counting operation of the watchdog timer by setting bit 4 (WDTON) of the option byte (000C0H) to 1 (the counter starts operating after a reset release) (for details, see **CHAPTER 22**).

WDTON	Watchdog Timer Counter
0	Counter operation disabled (counting stopped after reset)
1	Counter operation enabled (counting started after reset)

- Set an overflow time by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H) (for details, see **8.4.2** and **CHAPTER 22**).
  - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the option byte (000C0H) (for details, see **8.4.3** and **CHAPTER 22**).
- After a reset release, the watchdog timer starts counting.
  - By writing "ACH" to the watchdog timer enable register (WDTE) after the watchdog timer starts counting and before the overflow time set by the option byte, the watchdog timer is cleared and starts counting again.
  - After that, write the WDTE register the second time or later after a reset release during the window open period. If the WDTE register is written during a window close period, an internal reset signal is generated.
  - If the overflow time expires without "ACH" written to the WDTE register, an internal reset signal is generated. An internal reset signal is generated in the following cases.
    - If a 1-bit manipulation instruction is executed on the WDTE register
    - If data other than "ACH" is written to the WDTE register

- Cautions**
- When data is written to the watchdog timer enable register (WDTE) for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.
  - After "ACH" is written to the WDTE register, an error of up to 2 clocks (fIL) may occur before the watchdog timer is cleared.
  - The watchdog timer can be cleared immediately before the count value overflows.

&lt;R&gt;

**Cautions 4.** The operation of the watchdog timer in the HALT and STOP and SNOOZE modes differs as follows depending on the set value of bit 0 (WDSTBYON) of the option byte (000C0H).

	WDSTBYON = 0	WDSTBYON = 1
In HALT mode	Watchdog timer operation stops.	Watchdog timer operation continues.
In STOP mode		
In SNOOZE mode		

If **WDSTBYON = 0**, the watchdog timer resumes counting after the HALT or STOP mode is released. At this time, the counter is cleared to 0 and counting starts.

When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed.

Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.

Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.

#### 8.4.2 Setting overflow time of watchdog timer

Set the overflow time of the watchdog timer by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (000C0H).

If an overflow occurs, an internal reset signal is generated. The present count is cleared and the watchdog timer starts counting again by writing "ACH" to the watchdog timer enable register (WDTE) during the window open period before the overflow time.

The following overflow times can be set.

**Table 8-3. Setting of Overflow Time of Watchdog Timer**

WDCS2	WDCS1	WDCS0	Overflow Time of Watchdog Timer ( $f_{IL} = 17.25 \text{ kHz (MAX.)}$ )
0	0	0	$2^6/f_{IL}$ (3.71 ms)
0	0	1	$2^7/f_{IL}$ (7.42 ms)
0	1	0	$2^8/f_{IL}$ (14.84 ms)
0	1	1	$2^9/f_{IL}$ (29.68 ms)
1	0	0	$2^{11}/f_{IL}$ (118.72 ms)
Other than above			Setting prohibited

<R>

**Remark**  $f_{IL}$ : Low-speed on-chip oscillator clock frequency

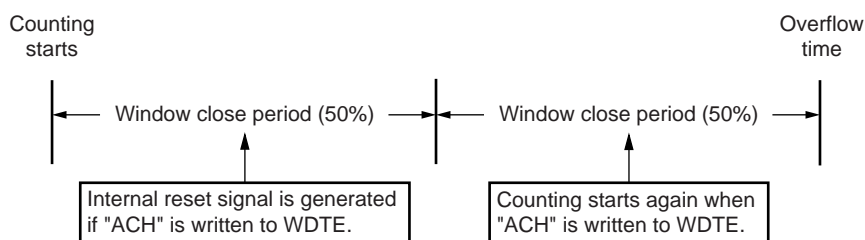


### 8.4.3 Setting window open period of watchdog timer

Set the window open period of the watchdog timer by using bits 6 and 5 (WINDOW1, WINDOW0) of the option byte (000C0H). The outline of the window is as follows.

- If "ACH" is written to the watchdog timer enable register (WDTE) during the window open period, the watchdog timer is cleared and starts counting again.
- Even if "ACH" is written to the WDTE register during the window close period, an abnormality is detected and an internal reset signal is generated.

**Example:** If the window open period is 50%



**Caution** When data is written to the WDTE register for the first time after reset release, the watchdog timer is cleared in any timing regardless of the window open time, as long as the register is written before the overflow time, and the watchdog timer starts counting again.

The window open period can be set as follows.

**Table 8-4. Setting Window Open Period of Watchdog Timer**

WINDOW1	WINDOW0	Window Open Period of Watchdog Timer
0	0	Setting prohibited
0	1	50%
1	0	75%
1	1	100%

**Caution** When bit 0 (WDSTBYON) of the option byte (000C0H) = 0, the window open period is 100% regardless of the values of the WINDOW1 and WINDOW0 bits.

**Remark** If the overflow time is set to  $2^9/f_{IL}$ , the window close time and open time are as follows.

	Setting of Window Open Period		
	50%	75%	100%
Window close time	0 to 20.08 ms	0 to 10.04 ms	None
Window open time	20.08 to 29.68 ms	10.04 to 29.68 ms	0 to 29.68 ms

<When window open period is 50%>

- Overflow time:  
 $2^9/f_{IL} \text{ (MAX.)} = 2^9/17.25 \text{ kHz} = 29.68 \text{ ms}$
- Window close time:  
 $0 \text{ to } 2^9/f_{IL} \text{ (MIN.)} \times (1 - 0.5) = 0 \text{ to } 2^9/12.75 \text{ kHz} \times 0.5 = 0 \text{ to } 20.08 \text{ ms}$
- Window open time:  
 $2^9/f_{IL} \text{ (MIN.)} \times (1 - 0.5) \text{ to } 2^9/f_{IL} \text{ (MAX.)} = 2^9/12.75 \text{ kHz} \times 0.5 \text{ to } 2^9/17.25 \text{ kHz} = 20.08 \text{ to } 29.68 \text{ ms}$

#### 8.4.4 Setting watchdog timer interval interrupt

Depending on the setting of bit 7 (WDTINT) of an option byte (000C0H), an interval interrupt (INTWDTI) can be generated when  $75\% + 1/2f_{IL}$  of the overflow time is reached.

**Table 8-5. Setting of Watchdog Timer Interval Interrupt**

WDTINT	Use of Watchdog Timer Interval Interrupt
0	Interval interrupt is used.
1	Interval interrupt is generated when $75\% + 1/2f_{IL}$ of overflow time is reached.

**Caution** When operating with the X1 oscillation clock after releasing the STOP mode, the CPU starts operating after the oscillation stabilization time has elapsed.

Therefore, if the period between the STOP mode release and the watchdog timer overflow is short, an overflow occurs during the oscillation stabilization time, causing a reset.

Consequently, set the overflow time in consideration of the oscillation stabilization time when operating with the X1 oscillation clock and when the watchdog timer is to be cleared after the STOP mode release by an interval interrupt.

**Remark** The watchdog timer continues counting even after INTWDTI is generated (until ACH is written to the watchdog timer enable register (WDTE)). If ACH is not written to the WDTE register before the overflow time, an internal reset signal is generated.

## CHAPTER 9 A/D CONVERTER

The number of analog input channels of the A/D converter differs, depending on the product.

	24-pin	32-pin
Analog input channels	6 ch (ANI0 to ANI3, ANI7, ANI16)	8 ch (ANI0 to ANI7)

**Remark** Most of the following descriptions in this chapter use the 32-pin products as an example.

## 9.1 Function of A/D Converter

The A/D converter converts analog input signals into digital values, and is configured to control analog inputs, including up to 8 channels of A/D converter analog inputs (ANI0 to ANI17).

<R> 12-bit or 8-bit resolution can be selected by the ADTYP bit of the A/D converter mode register 2 (ADM2).  
The A/D converter has the following function.

<R> • **12-bit or 8-bit resolution A/D conversion**

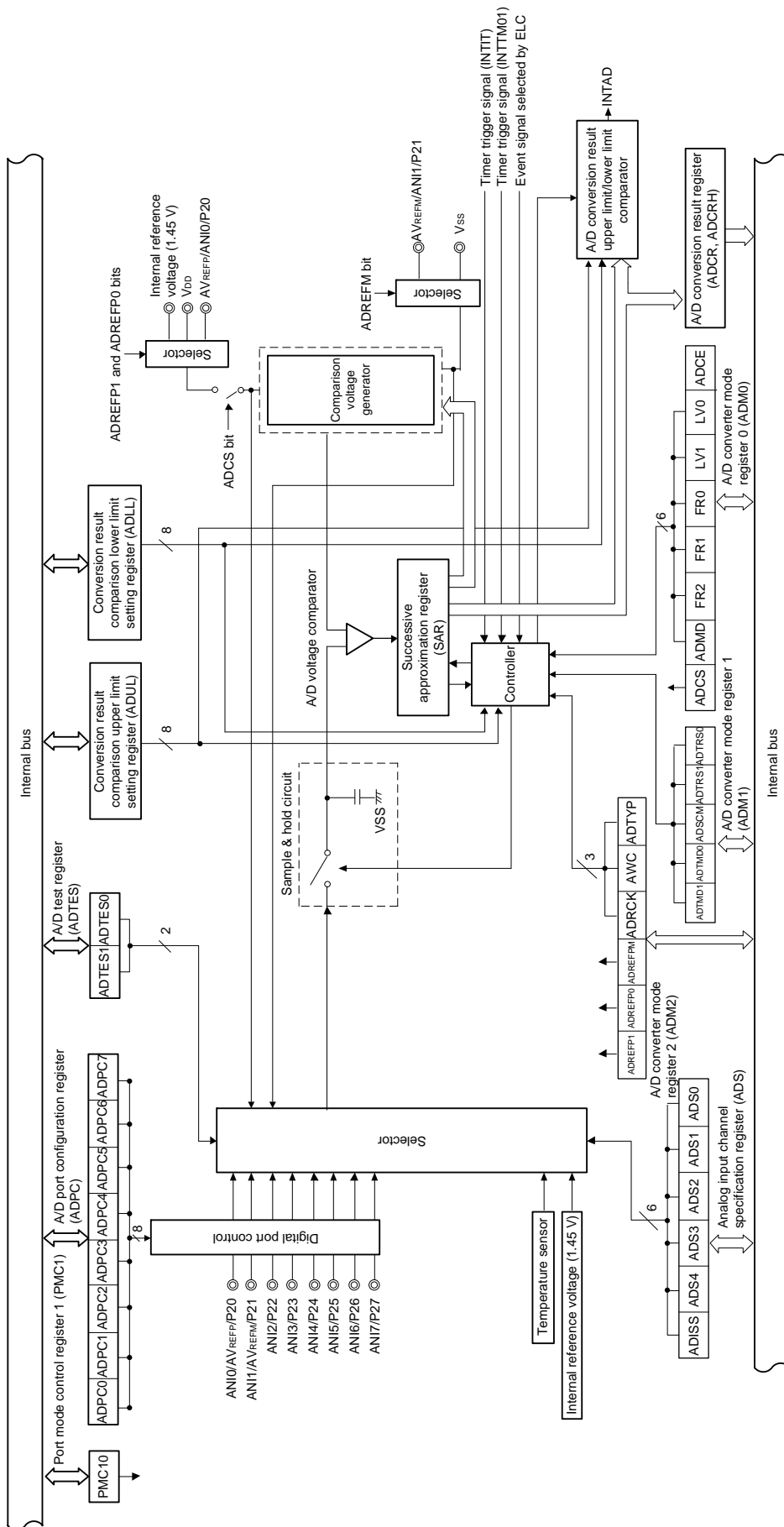
<R> 12-bit or 8-bit resolution A/D conversion is carried out repeatedly for one analog input channel selected from ANI0 to ANI7 and ANI16. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated (when in the select mode).

Various A/D conversion modes can be specified by using the mode combinations below.

<R> Trigger Mode	Software trigger	Conversion is started by software.
	Hardware trigger no-wait mode	Conversion is started by detecting a hardware trigger.
	Hardware trigger wait mode	The power is turned on by detecting a hardware trigger while the system is off and in the conversion standby state, and conversion is then started automatically after the stabilization wait time passes. When using the SNOOZE mode function, specify the hardware trigger wait mode.
Channel selection mode	Select mode	A/D conversion is performed on the analog input of one selected channel.
	Scan mode	A/D conversion is performed on the analog input of four channels in order. Four consecutive channels can be selected from ANI0 to ANI7 as analog input channels.
Conversion operation mode	One-shot conversion mode	A/D conversion is performed on the selected channel once.
	Sequential conversion mode	A/D conversion is sequentially performed on the selected channels until it is stopped by software.
Operation voltage mode	Standard 1 or standard 2 mode	Conversion is done in the operation voltage range of $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ .
Sampling time selection	Sampling clock cycles: $7 f_{AD}$	The sampling time in standard 1 mode is seven cycles of the conversion clock ( $f_{AD}$ ). Select this mode when the output impedance of the analog input source is high and the sampling time should be long.
	Sampling clock cycles: $5 f_{AD}$	The sampling time in standard 2 mode is five cycles of the conversion clock ( $f_{AD}$ ). Select this mode when enough sampling time is ensured (for example, when the output impedance of the analog input source is low).

<R>

Figure 9-1. Block Diagram of A/D Converter



**Remark** Analog input pins in figure 9-1 when a 32-pin product is used.

## 9.2 Configuration of A/D Converter

The A/D converter includes the following hardware.

<R>

### (1) ANI0 to ANI17 and ANI16 pins

These are the analog input pins of the 9 channels of the A/D converter. They input analog signals to be converted into digital signals. Pins other than the one selected as the analog input pin can be used as I/O port pins.

### (2) Sample & hold circuit

The sample & hold circuit samples each of the analog input voltages sequentially sent from the input circuit, and sends them to the A/D voltage comparator. This circuit also holds the sampled analog input voltage during A/D conversion.

### (3) A/D voltage comparator

This A/D voltage comparator compares output from the voltage tap of the comparison voltage generator with the sampled voltage value.

If the analog input voltage is found to be greater than the reference voltage ( $1/2 AV_{REF}$ ) as a result of the comparison, the most significant bit (MSB) of the successive approximation register (SAR) is set. If the analog input voltage is less than the reference voltage ( $1/2 AV_{REF}$ ), the MSB bit of the SAR is reset.

After that, bit 10 of the SAR register is automatically set, and the next comparison is made. The voltage tap of the comparison voltage generator is selected by the value of bit 11, to which the result has been already set.

Bit 11 = 0: ( $1/4 AV_{REF}$ )

Bit 11 = 1: ( $3/4 AV_{REF}$ )

The voltage tap of the comparison voltage generator and the analog input voltage are compared and bit 10 of the SAR register is manipulated according to the result of the comparison.

Analog input voltage  $\geq$  Voltage tap of comparison voltage generator: Bit 10 = 1

Analog input voltage  $\leq$  Voltage tap of comparison voltage generator: Bit 10 = 0

Comparison is continued like this to bit 0 of the SAR register.

When performing A/D conversion at a resolution of 8 bits, the comparison continues until bit 4 of the SAR register.

**Remark**  $AV_{REF}$ : The + side reference voltage of the A/D converter.

(This can be selected from  $AV_{REFP}$ , the internal reference voltage (1.45 V), and  $AV_{DD}$ .)

### (4) Comparison voltage generator

The comparison voltage generator generates the comparison voltage input from an analog input pin.

**(5) Successive approximation register (SAR)**

The SAR register is a register that sets voltage tap data whose values from the comparison voltage generator match the voltage values of the analog input pins, 1 bit at a time starting from the most significant bit (MSB).

If data is set in the SAR register all the way to the least significant bit (LSB) (end of A/D conversion), the contents of the SAR register (conversion results) are held in the A/D conversion result register (ADCR). When all the specified A/D conversion operations have ended, an A/D conversion end interrupt request signal (INTAD) is generated.

**(6) 12-bit A/D conversion result register (ADCR)**

The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCR register holds the A/D conversion result in its lower 12 bits (the higher 4 bits are fixed to 0).

**(7) 8-bit A/D conversion result register (ADCRH)**

The A/D conversion result is loaded from the successive approximation register to this register each time A/D conversion is completed, and the ADCRH register stores the higher 8 bits of the A/D conversion result.

**(8) Controller**

This circuit controls the conversion time of an input analog signal that is to be converted into a digital signal, as well as starting and stopping of the conversion operation. When A/D conversion has been completed, this controller generates INTAD through the A/D conversion result upper limit/lower limit comparator.

**(9) AV<sub>REFP</sub> pin**

This pin inputs an external reference voltage (AV<sub>REFP</sub>).

If using AV<sub>REFP</sub> as the + side reference voltage of the A/D converter, set the ADREFP1 and ADREFP0 bits of A/D converter mode register 2 (ADM2) to 1.

The analog signals input to ANI0 to ANI17 are converted to digital signals based on the voltage applied between AV<sub>REFP</sub> and the – side reference voltage (AV<sub>REFM</sub>/V<sub>SS</sub>).

In addition to AV<sub>REFP</sub>, it is possible to select V<sub>DD</sub>, or the internal reference voltage (1.45 V) as the + side reference voltage of the A/D converter.

**(10) AV<sub>REFM</sub> pin**

This pin inputs an external reference voltage (AV<sub>REFM</sub>). If using AV<sub>REFM</sub> as the – side reference voltage of the A/D converter, set the ADREFM bit of the ADM2 register to 1.

In addition to AV<sub>REFM</sub>, it is possible to select V<sub>SS</sub> as the – side reference voltage of the A/D converter.

&lt;R&gt;

### 9.3 Registers Controlling A/D Converter

The A/D converter is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- A/D converter mode register 0 (ADM0)
- A/D converter mode register 1 (ADM1)
- A/D converter mode register 2 (ADM2)
- 12-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- Analog input channel specification register (ADS)
- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)
- A/D test register (ADTES)
- A/D port configuration register (ADPC)
- Port mode control register 1 (PMC1)<sup>Note</sup>
- Port mode registers 1, 2 (PM1<sup>Note</sup>, PM2)

**Note** 24-pin products only

### 9.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the A/D converter is used, be sure to set bit 5 (ADCEN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-2. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

ADCEN	Control of A/D converter input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the A/D converter cannot be written.</li> <li>• The A/D converter is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by the A/D converter can be read/written.</li> </ul>

**Cautions 1.** When setting the A/D converter, be sure to set the following registers first while the ADCEN bit is set to 1. If ADCEN = 0, the values of the A/D converter control registers are cleared to their initial values and writing to them is ignored (except for port mode registers 0, 2, 12 and 14 (PM0, PM2, PM12, PM14), port mode control registers 0, 12, and 14 (PMC0, PMC12, PMC14), and A/D port configuration register (ADPC)).

- A/D converter mode register 0 (ADM0)
- A/D converter mode register 1 (ADM1)
- A/D converter mode register 2 (ADM2)
- 12-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)
- Analog input channel specification register (ADS)
- Conversion result comparison upper limit setting register (ADUL)
- Conversion result comparison lower limit setting register (ADLL)
- A/D test register (ADTES)

**2.** Be sure to clear bits 1, 3, and 7 to “0”.

**Note** 24-pin products only



### 9.3.2 A/D converter mode register 0 (ADM0)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion.

The ADM0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Address: FFF30H After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	<0>
ADM1	ADCS	ADMD	FR2 <sup>Note 1</sup>	FR1 <sup>Note 1</sup>	FR0 <sup>Note 1</sup>	0	LV0 <sup>Note 1</sup>	ADCE

ADCS	A/D conversion operation control
0	Stops conversion operation [When read] Conversion stopped/standby status
1	Enables conversion operation [When read] While in the software trigger mode: Conversion operation status While in the hardware trigger wait mode: A/D power supply stabilization wait status + conversion operation status

ADMD	Specification of the A/D conversion channel selection mode
0	Select mode
1	Scan mode

ADCE	A/D voltage comparator operation control <sup>Note 2</sup>
0	Stops A/D voltage comparator operation
1	Enables A/D voltage comparator operation

**Notes 1.** For details of the FR2 to FR0, LV0 bits, and A/D conversion, see **Table 9-3 A/D Conversion Time Selection**.

**2.** While in the software trigger mode or hardware trigger no-wait mode, the operation of the A/D voltage comparator is controlled by the ADCS and ADCE bits, and it takes stabilization wait status from the start of operation for the operation to stabilize. Therefore, when the ADCS bit is set to 1 after stabilization wait status or more has elapsed from the time ADCE bit is set to 1, the conversion result at that time has priority over the first conversion result. If the ADCS bit was set to 1 before the stabilization time elapsed, ignore the first conversion data.

[Stabilization wait status]

If a high-accuracy channel is selected as the analog input channel: 0.5  $\mu$ s

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5  $\mu$ s

If a standard channel is selected as the analog input channel: 2  $\mu$ s

If a temperature sensor output/internal reference voltage output are selected as the analog input channel:  
(ADISS bit of ADS register = 1): 2  $\mu$ s

**Cautions 1.** Change the ADMD, FR2 to FR0, and LV0 bits while in the conversion stopped status (ADCS = 0, ADCE = 0).

**2.** Setting ADCS = 1, ADCE = 0 is prohibited.

**3.** Do not change the ADCE and ADCS bits from 0 to 1 at the same time by using an 8-bit manipulation instruction. Be sure to set these bits in the order described in **9.7 A/D Converter Setup Flowchart**.

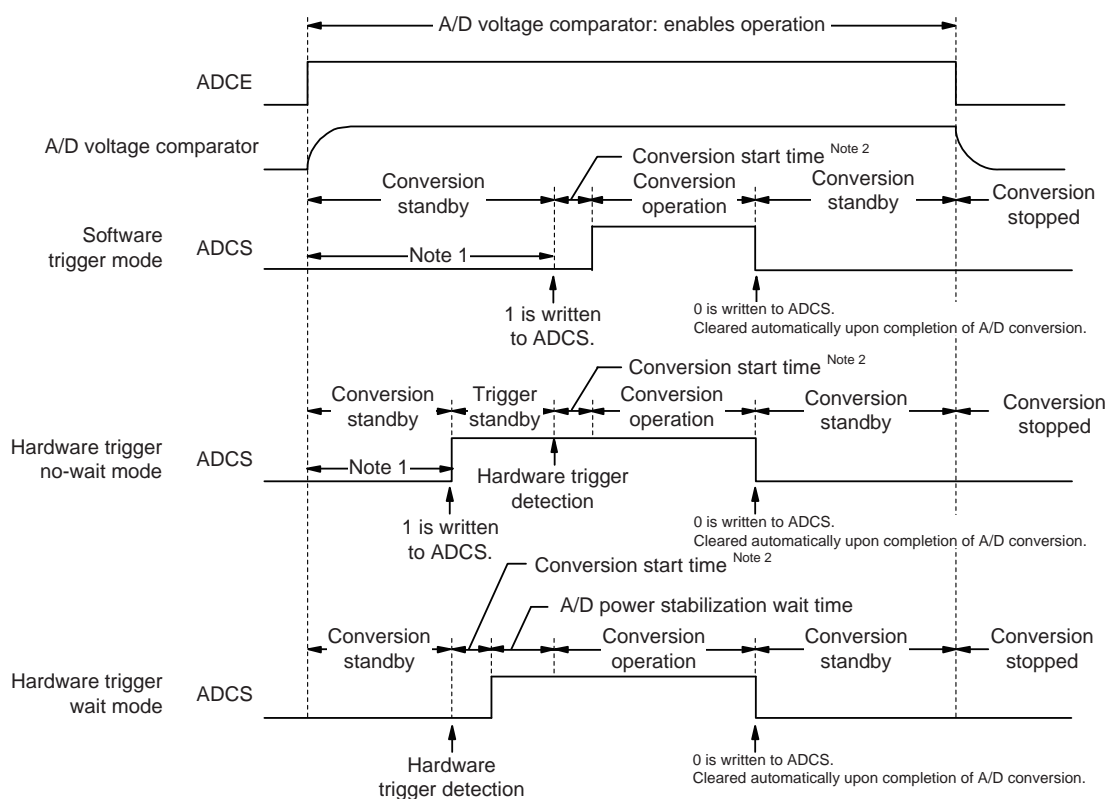
Table 9-1. Settings of ADCS and ADCE Bits

ADCS	ADCE	A/D Conversion Operation
0	0	Conversion stopped state
0	1	Conversion standby state
1	0	Setting prohibited
1	1	Conversion-in-progress state

Table 9-2. Setting and Clearing Conditions for ADCS Bit

A/D Conversion Mode			Set Conditions	Clear Conditions
Software trigger	Select mode	Sequential conversion mode	When 1 is written to ADCE and ADCS	When 0 is written to ADCS
		One-shot conversion mode		<ul style="list-style-type: none"> <li>When 0 is written to ADCS</li> <li>The bit is automatically cleared to 0 when A/D conversion ends.</li> </ul>
	Scan mode	Sequential conversion mode		When 0 is written to ADCS
		One-shot conversion mode		<ul style="list-style-type: none"> <li>When 0 is written to ADCS</li> <li>The bit is automatically cleared to 0 when conversion ends on the specified four channels.</li> </ul>
Hardware trigger no-wait mode	Select mode	Sequential conversion mode	When 1 is written to ADCE and a hardware trigger is input	When 0 is written to ADCS
		One-shot conversion mode		When 0 is written to ADCS
	Scan mode	Sequential conversion mode		When 0 is written to ADCS
		One-shot conversion mode		When 0 is written to ADCS
Hardware trigger wait mode	Select mode	Sequential conversion mode	When 1 is written to ADCE and a hardware trigger is input	When 0 is written to ADCS
		One-shot conversion mode		<ul style="list-style-type: none"> <li>When 0 is written to ADCS</li> <li>The bit is automatically cleared to 0 when A/D conversion ends.</li> </ul>
	Scan mode	Sequential conversion mode		When 0 is written to ADCS
		One-shot conversion mode		<ul style="list-style-type: none"> <li>When 0 is written to ADCS</li> <li>The bit is automatically cleared to 0 when conversion ends on the specified four channels.</li> </ul>

Figure 9-4. Timing Chart When A/D Voltage Comparator Is Used



**Notes 1.** While in the software trigger mode or hardware trigger no-wait mode, the time from the rising of the ADCE bit to the falling of the ADCS bit must be following time or longer to stabilize the internal circuit.

[Stabilization wait status]

If a high-accuracy channel is selected as the analog input channel: 0.5  $\mu\text{s}$

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5  $\mu\text{s}$

If a standard channel is selected as the analog input channel: 2  $\mu\text{s}$

If a temperature sensor output/internal reference voltage output are selected as the analog input channel: (ADISS bit of ADS register = 1): 2  $\mu\text{s}$

- 2.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the A/D power supply stabilization wait time do not occur after a hardware trigger is detected.

**Cautions 1.** If using the hardware trigger wait mode, setting the ADCS bit to 1 is prohibited (but the bit is automatically switched to 1 when the hardware trigger signal is detected). However, it is possible to clear the ADCS bit to 0 to specify the A/D conversion standby status.

- While in the one-shot conversion mode of the hardware trigger no-wait mode, the ADCS flag is not automatically cleared to 0 when A/D conversion ends. Instead, 1 is retained.
- Only rewrite the value of the ADCE bit when ADCS = 0 (while in the conversion stopped/conversion standby status).
- To complete A/D conversion, specify at least the following time as the hardware trigger interval:  
 Hardware trigger no wait mode:  $2 f_{\text{CLK}} \text{ clock} + \text{conversion start time} + \text{A/D conversion time}$   
 Hardware trigger wait mode:  $2 f_{\text{CLK}} \text{ clock} + \text{Conversion start time} + \text{A/D power supply stabilization wait time} + \text{A/D conversion time}$

**Remark**  $f_{\text{CLK}}$ : CPU/peripheral hardware clock frequency

**Table 9-3. A/D Conversion Time Selection (1/4)**

(1) When there is no A/D power supply stabilization wait time Normal mode 1, 2  
(software trigger mode/hardware trigger no-wait mode)

A/D Converter Mode Register 0 (ADM0)				Mode	Conversion Clock ( $f_{AD}$ )	Number of Conversion Clock <sup>Note</sup>	Conversion Time	Conversion Time at 10-Bit Resolution				
FR2	FR1	FR0	LV0					$V_{DD} = 2.7$ to $3.6$ V				
								$f_{CLK} = 1$ MHz	$f_{CLK} = 4$ MHz	$f_{CLK} = 8$ MHz	$f_{CLK} = 16$ MHz	$f_{CLK} = 32$ MHz
0	0	0	0	Normal 1	$f_{CLK}/32$	54 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$ )	$1728/f_{CLK}$	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	54 $\mu s$
0	0	1	$f_{CLK}/16$		$864/f_{CLK}$				54 $\mu s$	27 $\mu s$		
0	1	0	$f_{CLK}/8$		$432/f_{CLK}$			54 $\mu s$	27 $\mu s$	13.5 $\mu s$		
0	1	1	$f_{CLK}/6$		$324/f_{CLK}$			40.5 $\mu s$	20.25 $\mu s$	10.125 $\mu s$		
1	0	0	$f_{CLK}/5$		$270/f_{CLK}$			33.75 $\mu s$	16.875 $\mu s$	8.4375 $\mu s$		
1	0	1	$f_{CLK}/4$		$216/f_{CLK}$			54 $\mu s$	27 $\mu s$	13.5 $\mu s$	6.75 $\mu s$	
1	1	0	$f_{CLK}/2$		$108/f_{CLK}$			27 $\mu s$	13.5 $\mu s$	6.75 $\mu s$	3.375 $\mu s$	
1	1	1	$f_{CLK}/1$		$54/f_{CLK}$			54 $\mu s$	13.5 $\mu s$	6.75 $\mu s$	3.375 $\mu s$	Setting prohibited
0	0	0	1	Normal 2	$f_{CLK}/32$	66 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$ )	$2112/f_{CLK}$	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	66 $\mu s$
0	0	1	$f_{CLK}/16$		$1056/f_{CLK}$				66 $\mu s$	33 $\mu s$		
0	1	0	$f_{CLK}/8$		$528/f_{CLK}$			66 $\mu s$	33 $\mu s$	16.5 $\mu s$		
0	1	1	$f_{CLK}/6$		$396/f_{CLK}$			49.5 $\mu s$	24.75 $\mu s$	12.375 $\mu s$		
1	0	0	$f_{CLK}/5$		$330/f_{CLK}$			41.25 $\mu s$	20.625 $\mu s$	10.3125 $\mu s$		
1	0	1	$f_{CLK}/4$		$264/f_{CLK}$			66 $\mu s$	33 $\mu s$	16.5 $\mu s$	8.25 $\mu s$	
1	1	0	$f_{CLK}/2$		$132/f_{CLK}$			33 $\mu s$	16.5 $\mu s$	8.25 $\mu s$	4.125 $\mu s$	
1	1	1	$f_{CLK}/1$		$66/f_{CLK}$			66 $\mu s$	16.5 $\mu s$	8.25 $\mu s$	4.125 $\mu s$	Setting prohibited

**Note** These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock ( $f_{AD}$ ).

- Cautions**
1. The A/D conversion time must also be within the relevant range of conversion times ( $t_{CONV}$ ) described in 27.6.1 A/D converter characteristics.
  2. When rewriting the FR2 to FR0, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).
  3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.

**Remark**  $f_{CLK}$ : CPU/peripheral hardware clock frequency

**Table 9-3. A/D Conversion Time Selection (2/4)**

(2) When there is no A/D power supply stabilization wait time Low-voltage mode 1, 2

(software trigger mode/hardware trigger no-wait mode)

A/D Converter Mode Register 0 (ADM0)				Mode	Conversion Clock (f <sub>AD</sub> )	Number of Stabilization Wait Clock	Number of Conversion Clock (Number of Sampling Clock) <sup>Note</sup>	Stabilization Wait Time + Conversion Time	Conversion Time at 10-Bit Resolution							
FR2	FR1	FR0	LV0						AV <sub>DD</sub> = 2.7 to 3.6 V							
									f <sub>CLK</sub> = 1 MHz	f <sub>CLK</sub> = 4 MHz	f <sub>CLK</sub> = 8 MHz	f <sub>CLK</sub> = 16 MHz	f <sub>CLK</sub> = 32 MHz			
0	0	0	0	Normal 1	f <sub>CLK</sub> /32	4 f <sub>CLK</sub>	54 f <sub>AD</sub> (number of sampling clock: 11 f <sub>AD</sub> )	1732/f <sub>CLK</sub>	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	54.125 μs			
0	0	1	f <sub>CLK</sub> /16		54.25 μs									27.125 μs		
0	1	0	f <sub>CLK</sub> /8		54.5 μs									27.25 μs	13.625 μs	
0	1	1	f <sub>CLK</sub> /6		41 μs									20.5 μs	10.25 μs	
1	0	0	f <sub>CLK</sub> /5		34.25 μs									17.125 μs	8.5625 μs	
1	0	1	f <sub>CLK</sub> /4		55 μs									27.5 μs	13.75 μs	6.875 μs
1	1	0	f <sub>CLK</sub> /2		28 μs									14 μs	7 μs	3.5 μs
1	1	1	f <sub>CLK</sub> /1		2 f <sub>CLK</sub>	56 μs	14 μs	7 μs	3.5 μs	Setting prohibited						
0	0	0	1	Normal 2	f <sub>CLK</sub> /32	58 f <sub>CLK</sub>	66 f <sub>AD</sub> (number of sampling clock: 23 f <sub>AD</sub> )	2170/f <sub>CLK</sub>	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	67.8125 μs			
0	0	1	f <sub>CLK</sub> /16		69.625 μs									34.8125 μs		
0	1	0	f <sub>CLK</sub> /8		73.25 μs									36.625 μs	18.3125 μs	
0	1	1	f <sub>CLK</sub> /6		56.75 μs									28.375 μs	14.1875 μs	
1	0	0	f <sub>CLK</sub> /5		48.5 μs									24.25 μs	12.125 μs	
1	0	1	f <sub>CLK</sub> /4		80.5 μs									40.25 μs	20.125 μs	10.0625 μs
1	1	0	f <sub>CLK</sub> /2		47.5 μs									23.75 μs	11.875 μs	5.9375 μs
1	1	1	f <sub>CLK</sub> /1		29 f <sub>CLK</sub>	95 μs	23.75 μs	11.875 μs	5.9375 μs	Setting prohibited						

&lt;R&gt;

**Note** These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f<sub>AD</sub>).

**Cautions 1.** The A/D conversion time must also be within the relevant range of conversion times (t<sub>CONV</sub>) described in 27.6.1 A/D converter characteristics.

2. When rewriting the FR2 to FR0, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).
3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.

**Remark** f<sub>CLK</sub>: CPU/peripheral hardware clock frequency

**Table 9-3. A/D Conversion Time Selection (3/4)**

(3) When there is A/D power supply stabilization wait time Normal mode 1, 2

(hardware trigger wait mode <sup>Note 1</sup>)

A/D Converter Mode Register 0 (ADM0)				Mode	Conversion Clock (f <sub>AD</sub> )	Number of Conversion Clock <sup>Note 2</sup>	A/D Power Supply Stabilization Wait Time + Conversion Time	Conversion Time Selection								
FR2	FR1	FR0	LV0					V <sub>DD</sub> = 2.7 to 3.6 V								
								f <sub>CLK</sub> = 1 MHz	f <sub>CLK</sub> = 4 MHz	f <sub>CLK</sub> = 8 MHz	f <sub>CLK</sub> = 16 MHz	f <sub>CLK</sub> = 32 MHz				
0	0	0	0	Normal 1	f <sub>CLK</sub> /32	41 f <sub>AD</sub> (number of sampling clock: 11 f <sub>AD</sub> )	1312/f <sub>CLK</sub>	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	41 μs				
0	0	1	f <sub>CLK</sub> /16		656/f <sub>CLK</sub>								41 μs	20.5 μs		
0	1	0	f <sub>CLK</sub> /8		328/f <sub>CLK</sub>								41 μs	20.5 μs	10.25 μs	
0	1	1	f <sub>CLK</sub> /6		246/f <sub>CLK</sub>								30.75 μs	15.375 μs	7.6875 μs	
1	0	0	f <sub>CLK</sub> /5		205/f <sub>CLK</sub>								25.625 μs	12.8125 μs	6.40625 μs	
1	0	1	f <sub>CLK</sub> /4		164/f <sub>CLK</sub>								41 μs	20.5 μs	10.25 μs	5.125 μs
1	1	0	f <sub>CLK</sub> /2		82/f <sub>CLK</sub>								20.5 μs	10.25 μs	5.125 μs	2.5625 μs
1	1	1	f <sub>CLK</sub> /1		41/f <sub>CLK</sub>								41 μs	10.25 μs	5.125 μs	2.5625 μs
0	0	0	1	Normal 2	f <sub>CLK</sub> /32	53 f <sub>AD</sub> (number of sampling clock: 23 f <sub>AD</sub> )	1696/f <sub>CLK</sub>	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	53 μs				
0	0	1	f <sub>CLK</sub> /16		848/f <sub>CLK</sub>								53 μs	26.5 μs	13.25 μs	
0	1	0	f <sub>CLK</sub> /8		424/f <sub>CLK</sub>								53 μs	26.5 μs	13.25 μs	
0	1	1	f <sub>CLK</sub> /6		318/f <sub>CLK</sub>								39.75 μs	19.875 μs	9.9375 μs	
1	0	0	f <sub>CLK</sub> /5		265/f <sub>CLK</sub>								33.125 μs	16.5625 μs	8.28125 μs	
1	0	1	f <sub>CLK</sub> /4		212/f <sub>CLK</sub>								53 μs	26.5 μs	13.25 μs	6.625 μs
1	1	0	f <sub>CLK</sub> /2		106/f <sub>CLK</sub>								26.5 μs	13.25 μs	6.625 μs	3.3125 μs
1	1	1	f <sub>CLK</sub> /1		53/f <sub>CLK</sub>								53 μs	13.25 μs	6.625 μs	3.3125 μs

**Notes 1.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see Table 9 - 3).

**2.** These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock (f<sub>AD</sub>).

**Cautions 1.** The A/D conversion time must also be within the relevant range of conversion times (t<sub>CONV</sub>) described in 27.6.1 A/D converter characteristics.

**2.** When rewriting the FR2 to FR0, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).

**3.** The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration. Select conversion time, taking clock frequency errors into consideration.

**4.** When software trigger mode/hardware trigger no-wait mode, specify the conversion time so that the following conditions are satisfied:

- f<sub>AD</sub> is used within a range of 1 to 16 MHz.

- If temperature sensor output or internal reference voltage output (ADISS bit of ADS register = 1) is specified for the analog input channel, the A/D converter is used in the following conditions:  
When LV0 = 0: Setting prohibit  
When LV0 = 1: Settable

**Remark**  $f_{CLK}$ : CPU/peripheral hardware clock frequency

**Table 9-3. A/D Conversion Time Selection (4/4)**(4) When there is A/D power supply stabilization wait time Low-voltage mode 1, 2 <sup>Note 1</sup>(hardware trigger wait mode <sup>Note 2</sup>)

A/D Converter Mode Register 0 (ADM0)				Mode	Conversion Clock ( $f_{AD}$ )	Number of A/D Power Supply Stabilization Wait Clock	Number of Conversion Clock (Number of Sampling Clock) <sup>Note 3</sup>	A/D Power Supply Stabilization Wait Time + Conversion Time	A/D Power Supply Stabilization Wait Time + Conversion Time at 10-Bit Resolution				
FR2	FR1	FR0	LV0						$AV_{DD} = 2.7 \text{ to } 3.6 \text{ V}$				
									$f_{CLK} = 1 \text{ MHz}$	$f_{CLK} = 4 \text{ MHz}$	$f_{CLK} = 8 \text{ MHz}$	$f_{CLK} = 16 \text{ MHz}$	$f_{CLK} = 32 \text{ MHz}$
0	0	0	0	Normal 1	$f_{CLK}/32$	4 $f_{CLK}$	41 $f_{AD}$ (number of sampling clock: 11 $f_{AD}$ )	$1316/f_{CLK}$	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	41.125 $\mu\text{s}$
0	0	1	$f_{CLK}/16$		$660/f_{CLK}$					41.25 $\mu\text{s}$	20.625 $\mu\text{s}$		
0	1	0	$f_{CLK}/8$		$332/f_{CLK}$				41.5 $\mu\text{s}$	20.75 $\mu\text{s}$	10.375 $\mu\text{s}$		
0	1	1	$f_{CLK}/6$		$250/f_{CLK}$				31.25 $\mu\text{s}$	15.625 $\mu\text{s}$	7.8125 $\mu\text{s}$		
1	0	0	$f_{CLK}/5$		$209/f_{CLK}$				26.125 $\mu\text{s}$	13.0625 $\mu\text{s}$	6.53125 $\mu\text{s}$		
1	0	1	$f_{CLK}/4$		$168/f_{CLK}$				42 $\mu\text{s}$	21 $\mu\text{s}$	10.5 $\mu\text{s}$	5.25 $\mu\text{s}$	
1	1	0	$f_{CLK}/2$		$86/f_{CLK}$				21.5 $\mu\text{s}$	10.75 $\mu\text{s}$	5.375 $\mu\text{s}$	2.6875 $\mu\text{s}$	
1	1	1	$f_{CLK}/1$		2 $f_{CLK}$				$43/f_{CLK}$	43 $\mu\text{s}$	10.75 $\mu\text{s}$	5.375 $\mu\text{s}$	2.6875 $\mu\text{s}$
0	0	0	1	Normal 2	$f_{CLK}/32$	58 $f_{CLK}$	53 $f_{AD}$ (number of sampling clock: 23 $f_{AD}$ )	$1754/f_{CLK}$	Setting prohibited	Setting prohibited	Setting prohibited	Setting prohibited	54.8125 $\mu\text{s}$
0	0	1	$f_{CLK}/16$		$906/f_{CLK}$						56.625 $\mu\text{s}$	28.3125 $\mu\text{s}$	
0	1	0	$f_{CLK}/8$		$482/f_{CLK}$				60.25 $\mu\text{s}$	30.125 $\mu\text{s}$	15.0625 $\mu\text{s}$		
0	1	1	$f_{CLK}/6$		$376/f_{CLK}$				47 $\mu\text{s}$	23.5 $\mu\text{s}$	11.75 $\mu\text{s}$		
1	0	0	$f_{CLK}/5$		$323/f_{CLK}$				40.375 $\mu\text{s}$	20.1875 $\mu\text{s}$	10.09375 $\mu\text{s}$		
1	0	1	$f_{CLK}/4$		$270/f_{CLK}$				67.5 $\mu\text{s}$	33.75 $\mu\text{s}$	16.875 $\mu\text{s}$	8.4375 $\mu\text{s}$	
1	1	0	$f_{CLK}/2$		$164/f_{CLK}$				41 $\mu\text{s}$	20.5 $\mu\text{s}$	10.25 $\mu\text{s}$	5.125 $\mu\text{s}$	
1	1	1	$f_{CLK}/1$		29 $f_{CLK}$				$82/f_{CLK}$	82 $\mu\text{s}$	20.5 $\mu\text{s}$	10.25 $\mu\text{s}$	5.125 $\mu\text{s}$

**Notes 1.** This mode is prohibited when using the temperature sensor.**2.** For the second and subsequent conversion in sequential conversion mode and for conversion of the channel specified by scan 1, 2, and 3 in scan mode, the conversion start time and stabilization wait time for A/D power supply do not occur after a hardware trigger is detected (see Table 12 - 4).**3.** These are the numbers of clock cycles when conversion is with 10-bit resolution. When eight-bit resolution is selected, the values are shorter by two cycles of the conversion clock ( $f_{AD}$ ).**Cautions 1.** The A/D conversion time must also be within the relevant range of conversion times ( $t_{CONV}$ ) described in 27.6.1 A/D converter characteristics.**Note** that the conversion time ( $t_{CONV}$ ) does not include the A/D power supply stabilization wait time.**2.** When rewriting the FR2 to FR0, and LV0 bits to other than the same data, while in the conversion stopped (ADCS = 0, ADCE = 0).**3.** The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration. Select conversion time, taking clock frequency errors into consideration. Select conversion time, taking clock frequency errors into consideration.**4.** When hardware trigger wait mode, specify the conversion time so that the following conditions are satisfied:

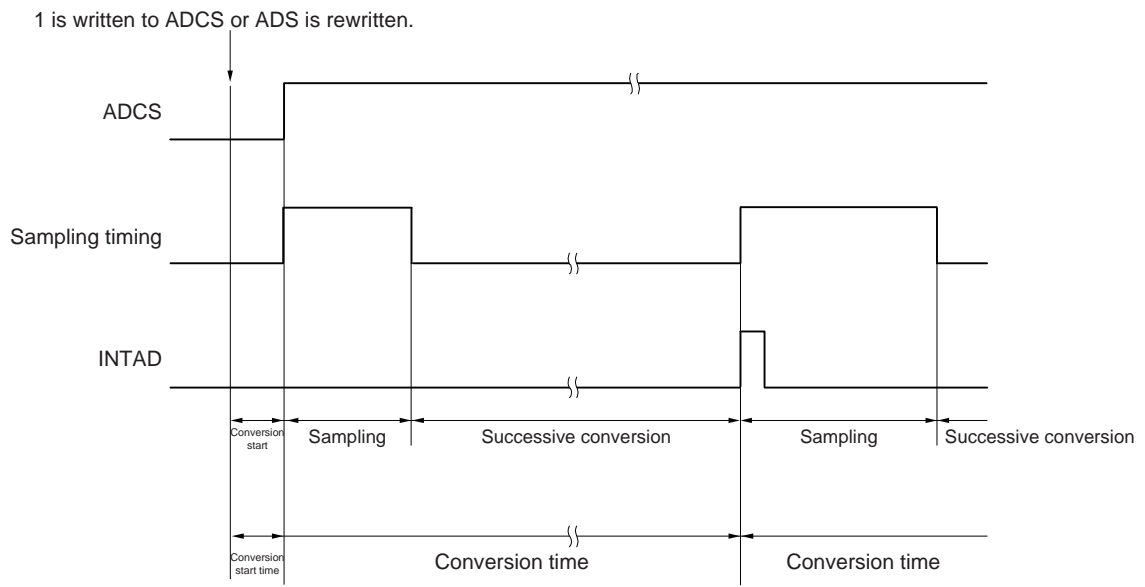


- $f_{AD}$  is used within a range of 1 to 16 MHz.
- If temperature sensor output or internal reference voltage output (ADISS bit of ADS register = 1) is specified for the analog input channel, the A/D converter is used in the following conditions:
  - When LV0 = 0: Setting prohibit
  - When LV0 = 1: Settable

**Remark**  $f_{CLK}$ : CPU/peripheral hardware clock frequency

Figure 9-5. A/D Converter Sampling and A/D Conversion Timing (Example for Software Trigger Mode)

<R>



### 9.3.3 A/D converter mode register 1 (ADM1)

This register is used to specify the A/D conversion trigger, conversion mode, and hardware trigger signal.

The ADM1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-6. Format of A/D Converter Mode Register 1 (ADM1)**

Address: FFF32H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADM1	ADTMD1	ADTMD0	ADSCM	0	0	0	ADTRS1	ADTRS0

ADTMD1	ADTMD0	Selection of the A/D conversion trigger mode
0	×	Software trigger mode
1	0	Hardware trigger no-wait mode
1	1	Hardware trigger wait mode

ADSCM	Specification of the A/D conversion mode
0	Sequential conversion mode
1	One-shot conversion mode

ADTRS1	ADTRS0	Selection of the hardware trigger signal
0	0	End of timer channel 01 count or capture interrupt signal (INTTM01)
0	1	Event signal selected by ELC
Other than above		Setting prohibited

<R>

- Cautions**
1. Only rewrite the value of the ADM1 register while conversion is stopped (ADCS = 0, ADCE = 0).
  2. To complete A/D conversion, specify at least the following time as the hardware trigger interval:  
**Hardware trigger no wait mode:** 2  $f_{CLK}$  clock + conversion start time + A/D conversion time  
**Hardware trigger wait mode:** 2  $f_{CLK}$  clock + conversion start time + A/D power supply stabilization wait time + A/D conversion time

- Remarks**
1. ×: don't care
  2.  $f_{CLK}$ : CPU/peripheral hardware clock frequency

### 9.3.4 A/D converter mode register 2 (ADM2)

This register is used to select the + side or - side reference voltage of the A/D converter, check the upper limit and lower limit A/D conversion result values, select the resolution, and specify whether to use the SNOOZE mode.

The ADM2 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-7. Format of A/D Converter Mode Register 2 (ADM2) (1/2)**

Address: F0010H After reset: 00H R/W

Symbol	7	6	5	4	<3>	<2>	1	<0>
ADM2	ADREFP1	ADREFP0	ADREFM	0	ADRCK	AWC	0	ADTYP

ADREFP1	ADREFP0	Selection of the + side reference voltage source of the A/D converter
0	0	Supplied from V <sub>DD</sub>
0	1	Supplied from P20/AV <sub>REFP</sub> /ANI0
1	0	Supplied from the internal reference voltage (1.45 V) <sup>Note</sup>
1	1	Setting prohibited

- When ADREFP1 or ADREFP0 bit is rewritten, this must be configured in accordance with the following procedures.
  - Set ADCE = 0
  - Change the values of ADREFP1 and ADREFP0
  - Reference voltage stabilization wait time (A)
  - Set ADCE = 1
  - Reference voltage stabilization wait time (B)

The stabilization wait time indicated by (3) is required when the value of the ADREFP1 and ADREFP0 bits is changed.

When ADREFP1 and ADREFP0 are changed to 1 and 0: A = 10  $\mu$ s

When ADREFP1 and ADREFP0 are changed to 0 and 0 or 0 and 1: A = 1  $\mu$ s

The stabilization wait time indicated by (5) is required when the value of the ADCE bit is changed to 1.

If a high-accuracy channel is selected as the analog input channel: 0.5  $\mu$ s

If a test mode setting (ADTES1 bit of ADTES register = 1) is selected: 0.5  $\mu$ s

If a standard channel is selected as the analog input channel: 2  $\mu$ s

If a temperature sensor output/internal reference voltage output are selected as the analog input channel: (ADISS bit of ADS register = 1): 2  $\mu$ s

After (5) stabilization time, start the A/D conversion.

- When ADREFP1 and ADREFP0 are set to 1 and 0, respectively, A/D conversion cannot be performed on the temperature sensor output and internal reference voltage output.  
Be sure to perform A/D conversion while ADISS = 0.

**Note** This setting can be used only in HS (high-speed main) mode. For detail, refer to **Figure 22-3 Format of User Option Byte (000C2H)**.

- Cautions**
- Only rewrite the value of the ADM2 register while conversion is stopped (ADCS = 0, ADCE = 0).
  - Do not set the ADREFP1 bit to 1 when shifting to STOP mode. Also, if the ADREFP1 bit is set to 1, the A/D converter reference voltage current (I<sub>ADREF</sub>) indicated in 27.3.2 Supply current characteristics will be added to the current consumption when shifting to HALT mode while the CPU is operating on the main system clock.
  - When using AV<sub>REFP</sub> and AV<sub>REFM</sub>, specify ANI0 and ANI1 as the analog input channels and specify input mode by using the port mode register.

Figure 9-7. Format of A/D Converter Mode Register 2 (ADM2) (2/2)

Address: F0010H After reset: 00H R/W

Symbol	7	6	5	4	<3>	<2>	1	<0>
ADM2	ADREFP1	ADREFP0	ADREFM	0	ADRCK	AWC	0	ADTYP

ADREFM	Selection of the – side reference voltage source of the A/D converter
0	Supplied from V <sub>SS</sub>
1	Supplied from P21/AV <sub>REFM</sub> /ANI1

ADRCK	Checking the upper limit and lower limit conversion result values
0	The interrupt signal (INTAD) is output when the ADLL register ≤ the ADCR register ≤ the ADUL register (Area 1).
1	The interrupt signal (INTAD) is output when the ADCR register < the ADLL register (Area 2) or the ADUL register < the ADCR register (Area 3).

Figure 9-8 shows the generation range of the interrupt signal (INTAD) for Area 1 to Area 3.

AWC	Specification of the SNOOZE mode
0	Do not use the SNOOZE mode function.
1	Use the SNOOZE mode function.

When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and A/D conversion is performed without operating the CPU (the SNOOZE mode).

- The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock (f<sub>CLK</sub>). If any other clock is selected, specifying this mode is prohibited.
- Using the SNOOZE mode function in the software trigger mode or hardware trigger no-wait mode is prohibited.
- Using the SNOOZE mode function in the sequential conversion mode is prohibited.
- When using the SNOOZE mode function, specify a hardware trigger interval of at least “shift time to SNOOZE mode<sup>Note</sup> + conversion start time + A/D power supply stabilization wait time + A/D conversion time + 2 f<sub>CLK</sub> clock”
- Even when using the SNOOZE function, set AWC bit to 0 in normal operation mode and change it to 1 just before shifting to STOP mode.

Also, be sure to change the AWC bit to 0 after returning from STOP mode to normal operation mode.

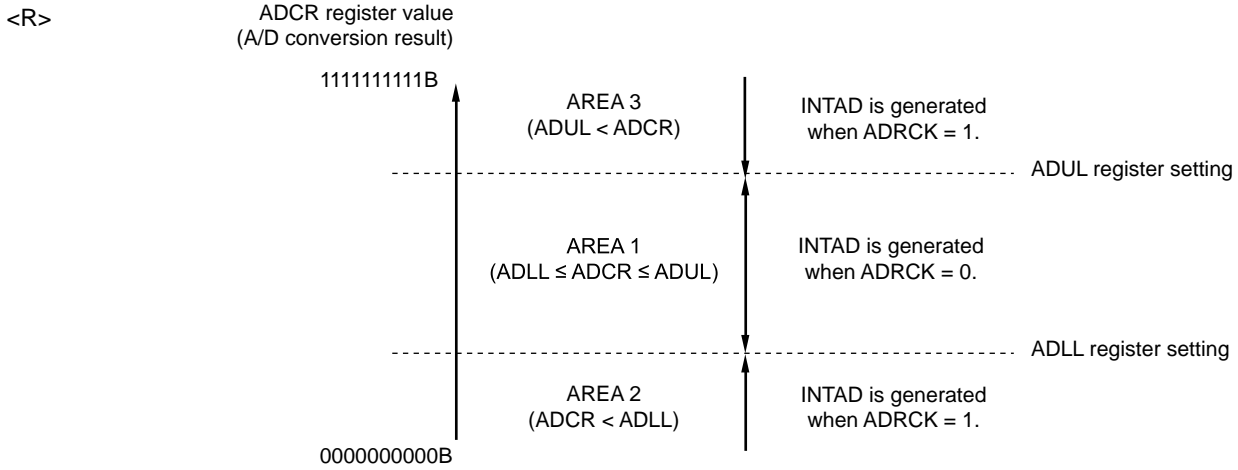
If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE or normal operation mode.

ADTYP	Selection of the A/D conversion resolution
0	12-bit resolution
1	8-bit resolution

**Note** Refer to “Transition time from STOP mode to SNOOZE mode” in **16.3.3 SNOOZE mode**

**Caution** Only rewrite the value of the ADM2 register while conversion operation is stopped (ADCS = 0, ADCE = 0).

**Figure 9-8. ADRCK Bit Interrupt Signal Generation Range**



**Remark** If INTAD does not occur, the A/D conversion result is not stored in the ADCR or ADCRH register.

### 9.3.5 12-bit A/D conversion result register (ADCR)

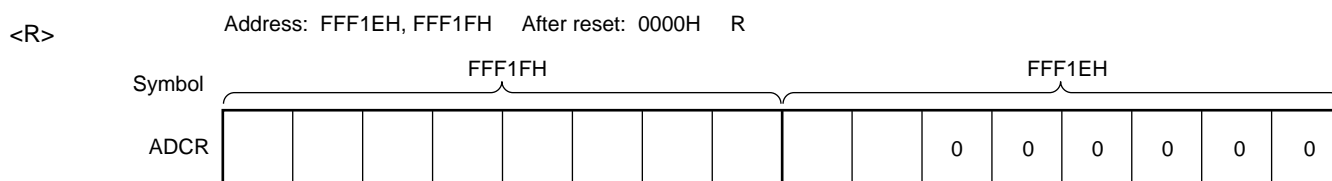
<R> The higher 4 bits are fixed to 0. Each time A/D conversion ends, the value of ADSAR [11:0] is stored in the A/D conversion result register (note that whether to store this value is determined by the setting of the ADRCK bit of the ADM2 register and by the settings of the ADUL and ADLL registers). The higher 4 bits of the conversion result are stored in FFF1FH and the lower 8 bits are stored in the lower 4 bits of FFF1EH<sup>Note</sup>.

The ADCR register can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see **Figure 9-8**), the result is not stored.

**Figure 9-9. Format of 12-bit A/D Conversion Result Register (ADCR)**



- <R> **Cautions 1.** When 8-bit resolution A/D conversion is selected (when the ADTYP bit of A/D converter mode register 2 (ADM2) is 1) and the ADCR register is read, 0 is read from the lower two bits (bits 7 and 6 of the ADCR register).
- 2.** When the ADCR register is accessed in 16-bit units, the higher 10 bits of the conversion result are read in order starting at bit 15.

### 9.3.6 8-bit A/D conversion result register (ADCRH)

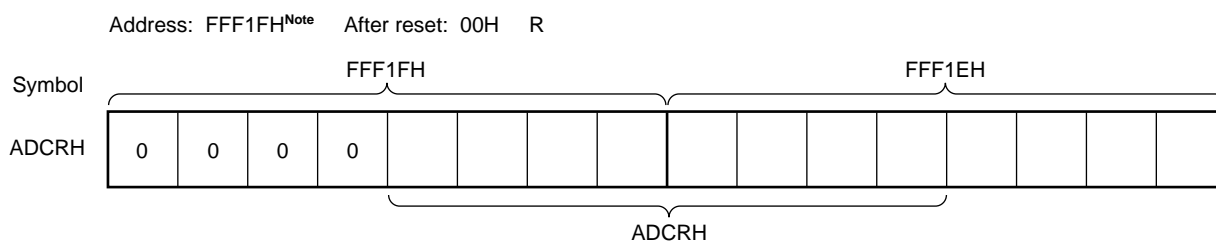
This register is an 8-bit register that indicate [11:4] bits of ADCR register. The higher 8 bits of 12-bit resolution are stored<sup>Note</sup>.

The ADCRH register can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Note** If the A/D conversion result is outside the range specified by using the A/D conversion comparison function (the value specified by the ADRCK bit of the ADM2 register and ADUL/ADLL registers; see **Figure 9-8**), the result is not stored.

**Figure 9-10. Format of 8-bit A/D Conversion Result Register (ADCRH)**



**Note** The ADCRH data (the lower 4 bits of FFF1FH + the higher 4 bits of FFF1EH) is to be read as a FFF1FH address.

- Cautions 1.** When writing to the A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of the ADCRH register may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, and ADPC registers. Using timing other than the above may cause an incorrect conversion result to be read.
- 2.** If INTAD does not occur, the A/D conversion result is not stored in the ADCRH register.



### 9.3.7 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-11. Format of Analog Input Channel Specification Register (ADS) (1/2)**

Address: FFF31H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	ADS4	0	ADS2	ADS1	ADS0

○ Select mode (ADMD = 0)

ADISS	ADS4	ADS2	ADS1	ADS0	Analog input channel	Input source
0	0	0	0	0	ANI0	P20/ANI0/AV <sub>REFP</sub> pin
0	0	0	0	1	ANI1	P21/ANI1/AV <sub>REFM</sub> pin
0	0	0	1	0	ANI2	P22/ANI2 pin
0	0	0	1	1	ANI3	P23/ANI3 pin
0	0	1	0	0	ANI4	P24/ANI4 pin <sup>Note 1</sup>
0	0	1	0	1	ANI5	P25/ANI5 pin <sup>Note 1</sup>
0	0	1	1	0	ANI6	P26/ANI6 pin <sup>Note 1</sup>
0	0	1	1	1	ANI7	P27/ANI7 pin
0	1	0	0	0	ANI16	P10/ANI16 pin <sup>Note 2</sup>
1	0	0	0	0	–	Temperature sensor output voltage <sup>Note 3</sup>
1	0	0	0	1	–	Internal reference voltage output (1.45 V) <sup>Note 3</sup>
Other than above					Setting prohibited	

<R>

- Notes**
1. ANI4 to ANI6: 32-pin products only
  2. ANI16: 24-pin products only
  3. This setting can be used only in HS (high-speed main) mode. For detail, refer to **Figure 22-3 Format of User Option Byte (000C2H)**.

Figure 9-11. Format of Analog Input Channel Specification Register (ADS) (2/2)

Address: FFF31H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	ADS4	0	ADS2	ADS1	ADS0

○ Scan mode (ADMD = 1)

ADISS	ADS4	ADS2	ADS1	ADS0	Analog input channel			
					Scan 0	Scan 1	Scan 2	Scan 3
0	0	0	0	0	ANI0	ANI1	ANI2	ANI3
0	0	0	0	1	ANI1 <sup>Note</sup>	ANI2 <sup>Note</sup>	ANI3 <sup>Note</sup>	ANI4 <sup>Note</sup>
0	0	0	1	0	ANI2 <sup>Note</sup>	ANI3 <sup>Note</sup>	ANI4 <sup>Note</sup>	ANI5 <sup>Note</sup>
0	0	0	1	1	ANI3 <sup>Note</sup>	ANI4 <sup>Note</sup>	ANI5 <sup>Note</sup>	ANI6 <sup>Note</sup>
0	0	1	0	0	ANI4 <sup>Note</sup>	ANI5 <sup>Note</sup>	ANI6 <sup>Note</sup>	ANI7 <sup>Note</sup>
Other than above					Setting prohibited			

**Note** 32-pin products only**Cautions** 1. Be sure to clear bits 3, 5, and 6 to "0".

- <R> 2. Set a channel to be set the analog input by ADPC and PMC1 registers in the input mode by using port mode register 1 (PM1).
3. Do not set the pin that is set by the A/D port configuration register (ADPC) as digital I/O by the ADS register.
4. Rewrite the value of the ADISS bit while conversion is stopped (ADCS = 0, ADCE = 0).
- <R> 5. If using AV<sub>REFP</sub> as the + side reference voltage of the A/D converter, do not select ANI0 as an A/D conversion channel.
- <R> 6. If using AV<sub>REFM</sub> as the – side reference voltage of the A/D converter, do not select ANI1 as an A/D conversion channel.
- <R> 7. If ADISS is set to 1, the internal reference voltage (1.45 V) cannot be used for the + side reference voltage. After the ADISS bit is set to 1, the initial conversion result cannot be used. For the setting flow, see 9.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected.
- <R> 8. Do not set the ADISS bit to 1 when shifting to STOP mode. When the ADISS bit is set to 1, the A/D converter reference voltage current (I<sub>ADREF</sub>) indicated in 27.3.2 Supply current characteristics will be added to the current consumption when shifting to HALT mode while the CPU is operating on the main system clock.
9. Ignore the conversion result if the corresponding ANI pin does not exist in the product used.

### 9.3.8 Conversion result comparison upper limit setting register (ADUL)

This register is used to specify the setting for checking the upper limit of the A/D conversion results.

The A/D conversion results and ADUL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 9-8**).

The ADUL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

- <R>
- Cautions 1.** When 12-bit resolution A/D conversion is selected, the higher eight bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL and ADLL registers.
  - 2.** Only write new values to the ADUL and ADLL registers while conversion is stopped (ADCS = 0, ADCE = 0).
  - 3.** The setting of the ADUL and ADLL registers must be greater than that of the ADLL register.

**Figure 9-12. Format of Conversion Result Comparison Upper Limit Setting Register (ADUL)**

Address: F0011H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
ADUL	ADUL7	ADUL6	ADUL5	ADUL4	ADUL3	ADUL2	ADUL1	ADUL0

### 9.3.9 Conversion result comparison lower limit setting register (ADLL)

This register is used to specify the setting for checking the lower limit of the A/D conversion results.

The A/D conversion results and ADLL register value are compared, and interrupt signal (INTAD) generation is controlled in the range specified for the ADRCK bit of A/D converter mode register 2 (ADM2) (shown in **Figure 9-8**).

The ADLL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-13. Format of Conversion Result Comparison Lower Limit Setting Register (ADLL)**

Address: F0012H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADLL	ADLL7	ADLL6	ADLL5	ADLL4	ADLL3	ADLL2	ADLL1	ADLL0

- <R>
- Cautions 1.** When 12-bit resolution A/D conversion is selected, the higher eight bits of the 12-bit A/D conversion result register (ADCR) are compared with the ADUL and ADLL registers.
  - 2.** Only write new values to the ADUL and ADLL registers while conversion is stopped (ADCS = 0, ADCE = 0).
  - 3.** The setting of the ADUL and ADLL registers must be greater than that of the ADLL register.

### 9.3.10 A/D test register (ADTES)

This register is used to select the + side reference voltage or – side reference voltage of the A/D converter, or the analog input channel (ANLxx), the temperature sensor output voltage, or the internal reference voltage (1.45 V) as the A/D conversion target for the A/D test function.

The ADTES register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 9-14. Format of A/D Test Register (ADTES)**

Address: F0013H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADTES	0	0	0	0	0	0	ADTES1	ADTES0

ADTES1	ADTES0	A/D conversion target
0	0	ANLxx/temperature sensor output voltage <sup>Note</sup> /internal reference voltage output (1.45 V) <sup>Note</sup> (This is specified using the analog input channel specification register (ADS).)
1	0	The - side reference voltage (selected by the ADREFM bit of the ADM2 register)
1	1	The + side reference voltage (selected by the ADREFP1 or ADREFP0 bit of the ADM2 register)
Other than above		Setting prohibited

**Note** The temperature sensor output voltage and internal reference voltage (1.45 V) can be selected only in the HS (high-speed main) mode.

**<R> 9.3.11 Registers controlling port function of analog input pins**

Set up the registers for controlling the functions of the ports shared with the analog input pins of the A/D converter (port mode registers (PMxx), port mode control registers 1 (PMC1), and A/D port configuration register (ADPC)). For details, see **4.3.1 Port mode registers (PMxx)**, **4.3.4 Port mode control registers 1 (PMC1)** (Only 24-pin products), and **4.3.5 A/D port configuration register (ADPC)**.

When using the ANI0 to ANI7 pins for analog input of the A/D converter, set the port mode register (PMxx) bit corresponding to each port to 1 and select analog input through the A/D port configuration register (ADPC).

When using the ANI16 pin for analog input of the A/D converter, set the port mode register (PMxx) bit and port mode control register 1 (PMC1) bit corresponding to each port to 1.

## 9.4 A/D Converter Conversion Operations

The A/D converter conversion operations are described below.

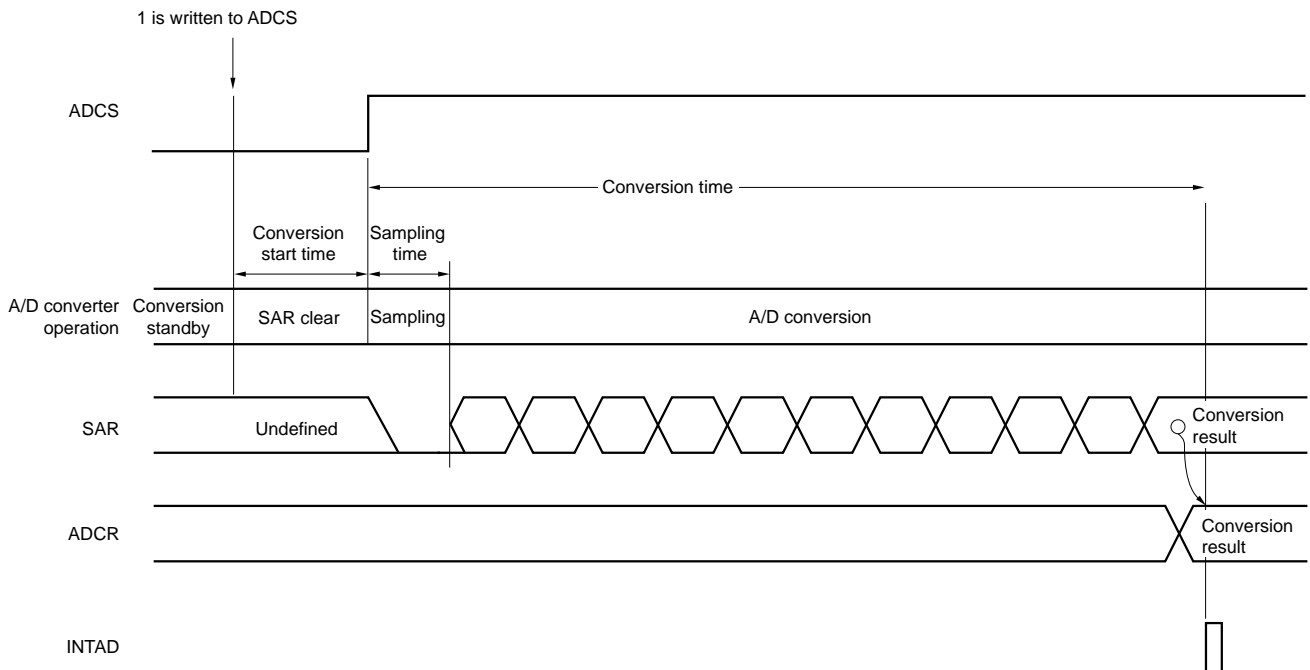
- <1> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.
- <2> When sampling has been done for a certain time, the sample & hold circuit is placed in the hold state and the sampled voltage is held until the A/D conversion operation has ended.
- <3> Bit 11 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to  $(1/2) AV_{REF}$  by the tap selector.
- <4> The voltage difference between the series resistor string voltage tap and sampled voltage is compared by the voltage comparator. If the analog input is greater than  $(1/2) AV_{REF}$ , the MSB bit of the SAR register remains set to 1. If the analog input is smaller than  $(1/2) AV_{REF}$ , the MSB bit is reset to 0.
- <5> Next, bit 10 of the SAR register is automatically set to 1, and the operation proceeds to the next comparison. The series resistor string voltage tap is selected according to the preset value of bit 11, as described below.
  - Bit 11 = 1:  $(3/4) AV_{REF}$
  - Bit 11 = 0:  $(1/4) AV_{REF}$
 The voltage tap and sampled voltage are compared and bit 10 of the SAR register is manipulated as follows.
  - Sampled voltage  $\geq$  Voltage tap: Bit 10 = 1
  - Sampled voltage  $<$  Voltage tap: Bit 10 = 0
- <6> Comparison is continued in this way up to bit 0 of the SAR register.
- <7> Upon completion of the comparison of 12 bits, an effective digital result value remains in the SAR register, and the result value is transferred to the A/D conversion result register (ADCR, ADCRH) and then latched<sup>Note 1</sup>. At the same time, the A/D conversion end interrupt request (INTAD) can also be generated<sup>Note 1</sup>.
- <8> Repeat steps <1> to <7>, until the ADCS bit is cleared to 0<sup>Note 2</sup>.  
To stop the A/D converter, clear the ADCS bit to 0.

- Notes**
1. If the A/D conversion result is outside the A/D conversion result range specified by the ADRCK bit and the ADUL and ADLL registers (see **Figure 9-8**), the A/D conversion result interrupt request signal is not generated and no A/D conversion results are stored in the ADCR and ADCRH registers.
  2. While in the sequential conversion mode, the ADCS flag is not automatically cleared to 0. This flag is not automatically cleared to 0 while in the one-shot conversion mode of the hardware trigger no-wait mode, either. Instead, 1 is retained.

- Remarks**
1. Two types of the A/D conversion result registers are available.
    - ADCR register (16 bits): Store 12-bit A/D conversion value
    - ADCRH register (8 bits): Store 8-bit A/D conversion value
  2.  $AV_{REF}$ : The + side reference voltage of the A/D converter. This can be selected from  $AV_{REFP}$ , the internal reference voltage (1.45 V), and  $V_{DD}$ .

Figure 9-15. Conversion Operation of A/D Converter (Software Trigger Mode)

&lt;R&gt;



&lt;R&gt;

In one-shot conversion mode, the ADCS bit is automatically cleared to 0 after completion of A/D conversion.

In sequential conversion mode, A/D conversion operations proceed continuously until the software clears bit 7 (ADCS) of the A/D converter mode register 0 (ADM0) to 0.

Writing to the analog input channel specification register (ADS) during A/D conversion interrupts the current conversion after which A/D conversion of the analog input specified by the ADS register proceeds. Data from the A/D conversion that was in progress are discarded.

Reset signal generation clears the A/D conversion result register (ADCR, ADCRH) to 0000H or 00H.

### 9.5 Input Voltage and Conversion Results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI7, ANI16) and the theoretical A/D conversion result (stored in the 12-bit A/D conversion result register (ADCR)) is shown by the following expression.

$$ADCR = \text{INT} \left( \frac{V_{AIN}}{AV_{REF}} \times 4096 + 0.5 \right)$$

or

$$(ADCR - 0.5) \times \frac{AV_{REF}}{4096} \leq V_{AIN} < (ADCR + 0.5) \times \frac{AV_{REF}}{4096}$$

where, INT( ): Function which returns integer part of value in parentheses

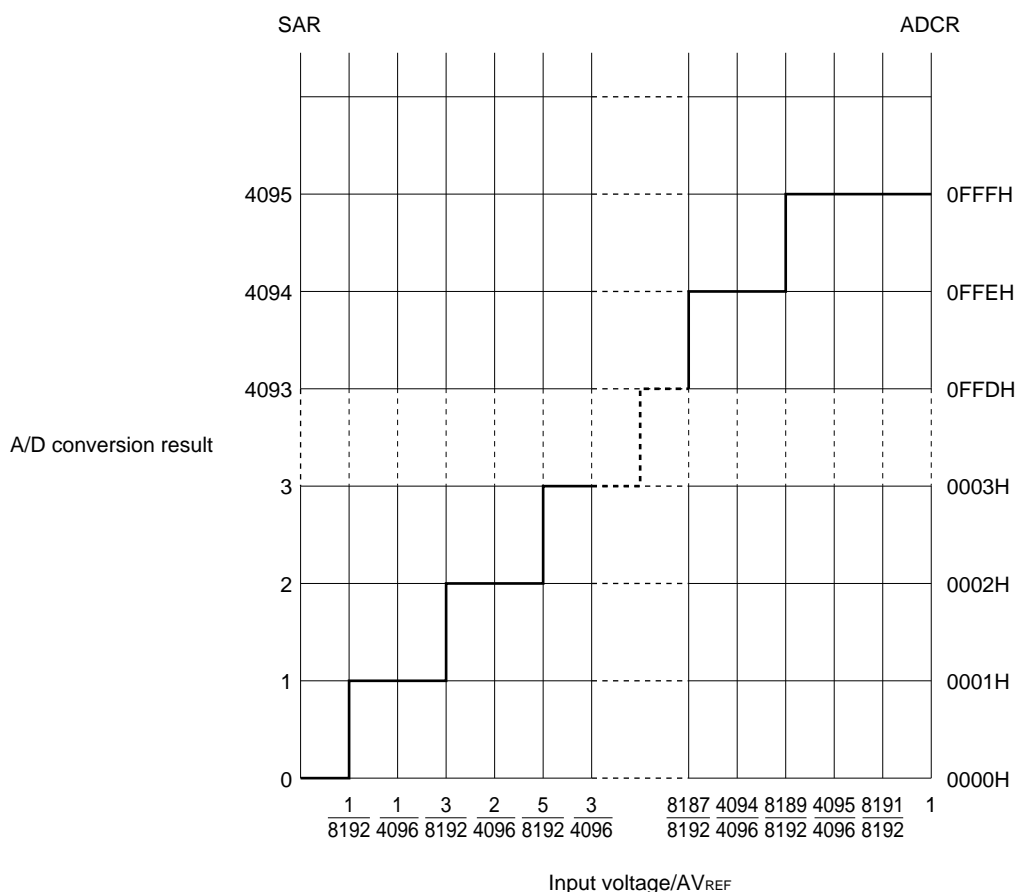
$V_{AIN}$ : Analog input voltage

$AV_{REF}$ :  $AV_{REF}$  pin voltage

ADCR: A/D conversion result register (ADCR) value

Figure 9-19 shows the relationship between the analog input voltage and the A/D conversion result.

**Figure 9-16. Relationship Between Analog Input Voltage and A/D Conversion Result**



**Remark**  $AV_{REF}$ : The + side reference voltage of the A/D converter. This can be selected from  $AV_{REFP}$ , the internal reference voltage (1.45 V), and  $V_{DD}$ .



## 9.6 A/D Converter Operation Modes

The operation of each A/D converter mode is described below. In addition, the procedure for specifying each mode is described in **9.7 A/D Converter Setup Flowchart**.

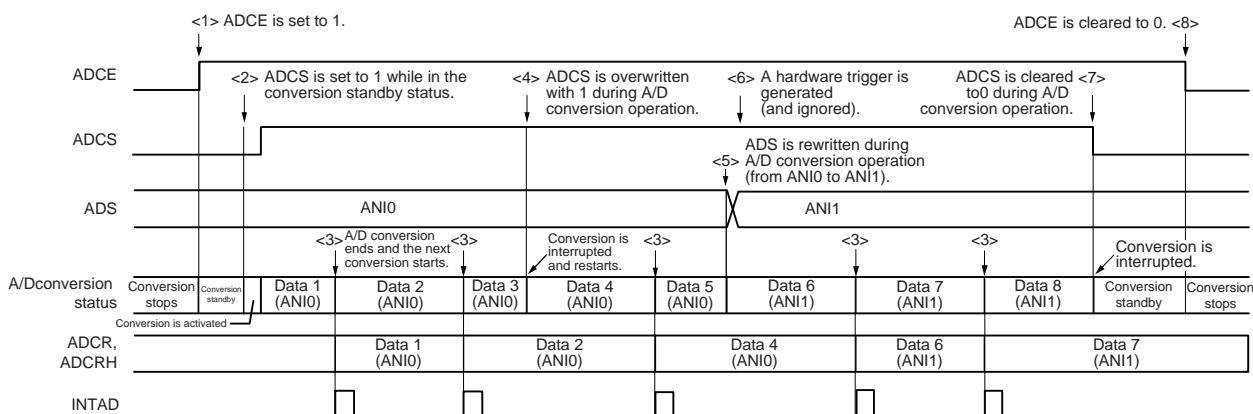
### 9.6.1 Software trigger mode (select mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).
- <3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.
- <4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.
- <8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

**Figure 9-17. Example of Software Trigger Mode (Select Mode, Sequential Conversion Mode) Operation Timing**

<R>

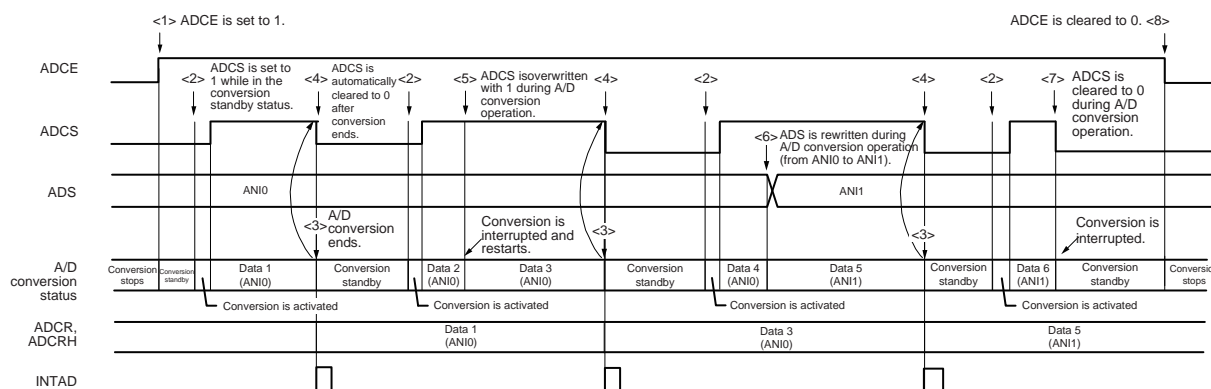


### 9.6.2 Software trigger mode (select mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to perform the A/D conversion of the analog input specified by the analog input channel specification register (ADS).
- <3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.
- <4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.
- <5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.
- <8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

**Figure 9-18. Example of Software Trigger Mode (Select Mode, One-Shot Conversion Mode) Operation Timing**



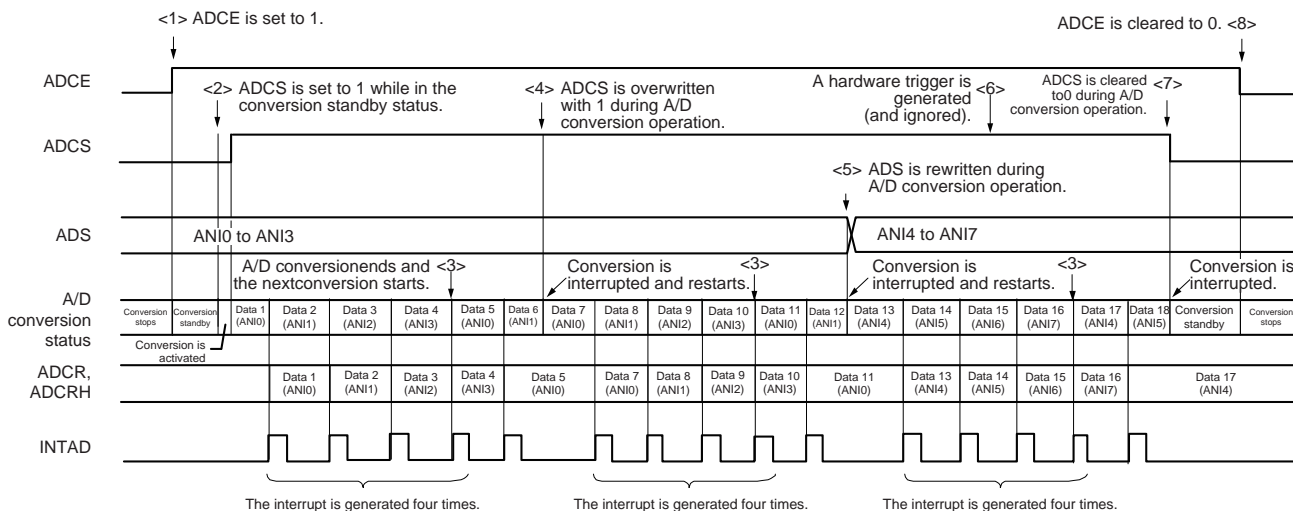
9.6.3 Software trigger mode (scan mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time <sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts (until all four channels are finished).
- <4> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <6> Even if a hardware trigger is input during conversion operation, A/D conversion does not start.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.
- <8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

Figure 9-19. Example of Software Trigger Mode (Scan Mode, Sequential Conversion Mode) Operation Timing

<R>



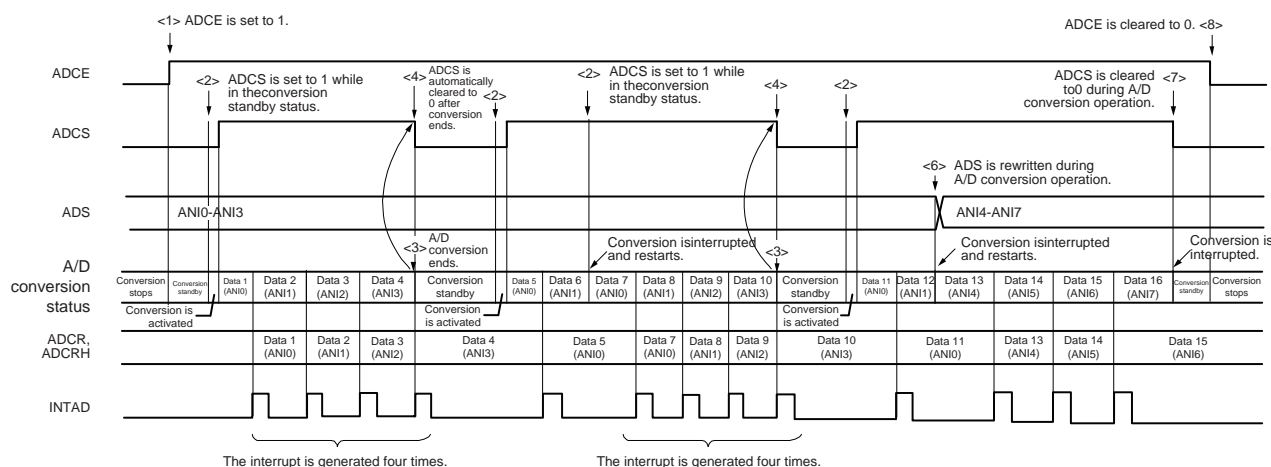
9.6.4 Software trigger mode (scan mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to perform A/D conversion on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.
- <4> After A/D conversion of the four channels ends, the ADCS bit is automatically cleared to 0, and the system enters the A/D conversion standby status.
- <5> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status.
- <8> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start. In addition, A/D conversion does not start even if a hardware trigger is input while in the A/D conversion standby status.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

Figure 9-20. Example of Software Trigger Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing

<R>



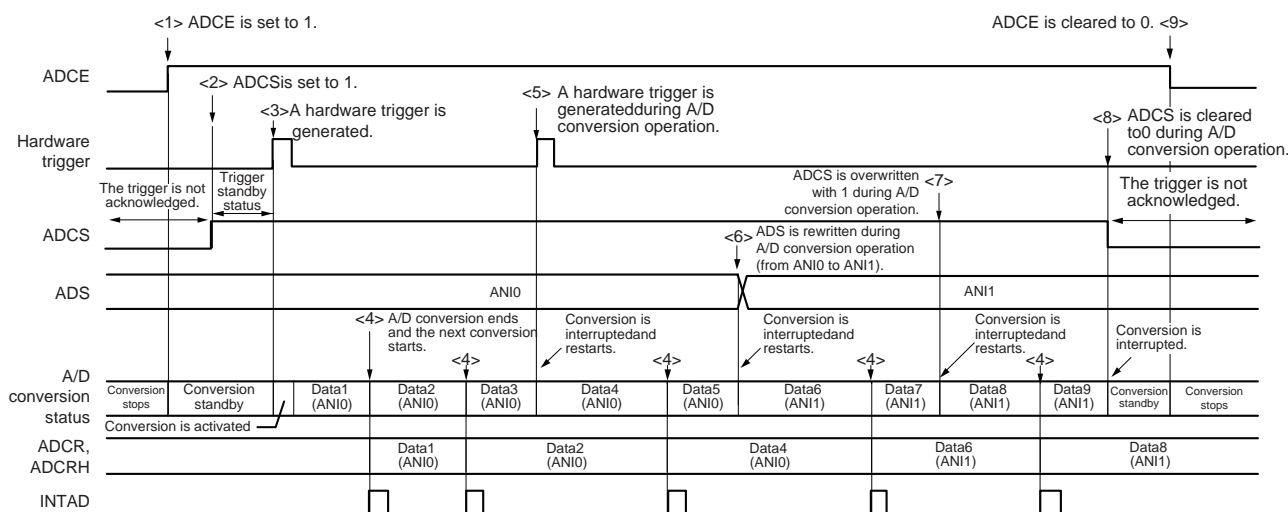
9.6.5 Hardware trigger no-wait mode (select mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.
- <3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).
- <4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts.
- <5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.
- <9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

Figure 9-21. Example of Hardware Trigger No-Wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing

<R>

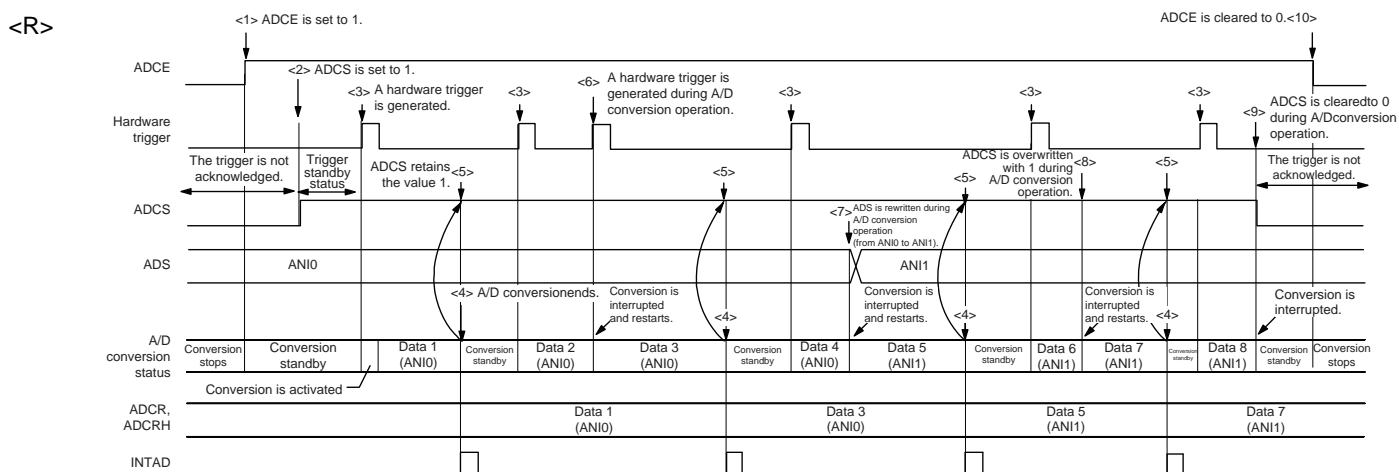


### 9.6.6 Hardware trigger no-wait mode (select mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.
- <3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS).
- <4> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.
- <5> After A/D conversion ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.
- <6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.
- <10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

**Figure 9-22. Example of Hardware Trigger No-Wait Mode (Select Mode, One-Shot Conversion Mode) Operation Timing**

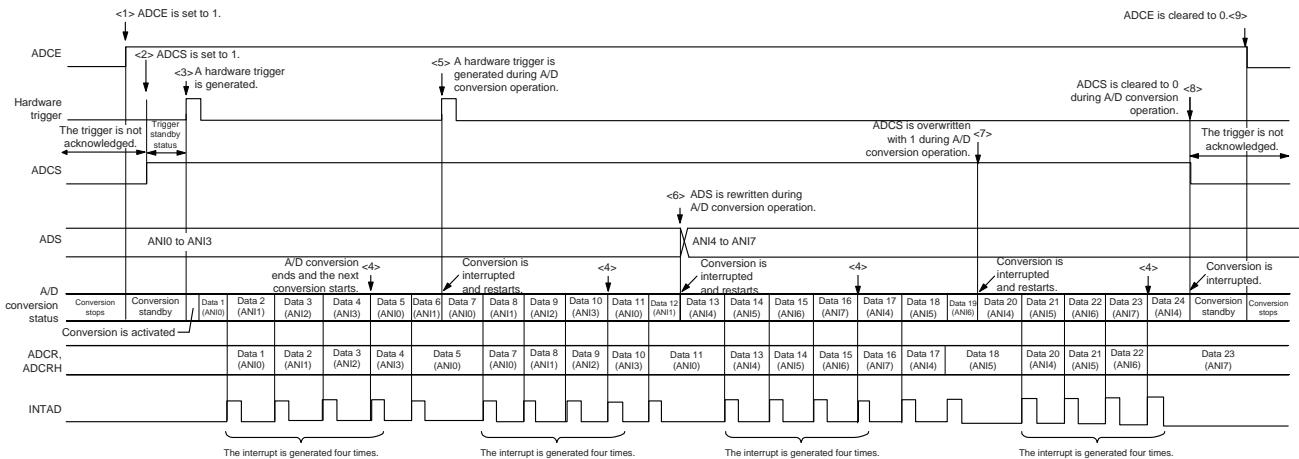


9.6.7 Hardware trigger no-wait mode (scan mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.
- <3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.
- <5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.
- <9> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCE = 0, specifying 1 for ADCS is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

Figure 9-23. Example of Hardware Trigger No-Wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing



<R>

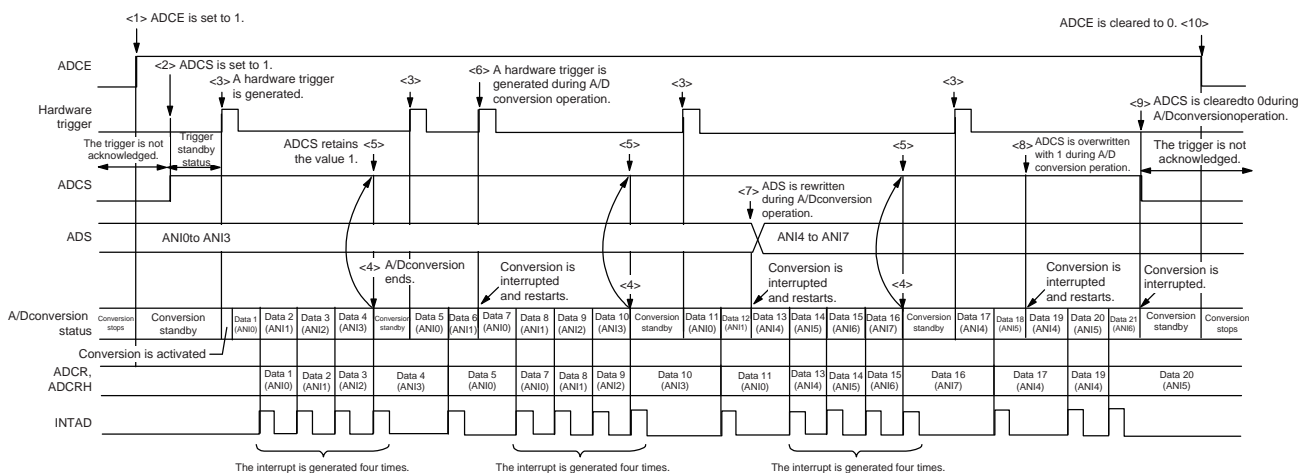
9.6.8 Hardware trigger no-wait mode (scan mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> After the software counts up to the stabilization wait time<sup>Note</sup>, the ADCS bit of the ADM0 register is set to 1 to place the system in the hardware trigger standby status (and conversion does not start at this stage). Note that, while in this status, A/D conversion does not start even if ADCS is set to 1.
- <3> If a hardware trigger is input while ADCS = 1, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <4> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.
- <5> After A/D conversion of the four channels ends, the ADCS bit remains set to 1, and the system enters the A/D conversion standby status.
- <6> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <7> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <8> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <9> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, and the system enters the A/D conversion standby status. However, the A/D converter does not stop in this status.
- <10> When ADCE is cleared to 0 while in the A/D conversion standby status, the A/D converter enters the stop status. When ADCS = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Note** If a high-accuracy channel is selected as the analog input channel: Stabilization wait time = 0.5  $\mu$ s  
 If a standard channel is selected as the analog input channel: Stabilization wait time = 2  $\mu$ s

Figure 9-24. Example of Hardware Trigger No-Wait Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing

<R>

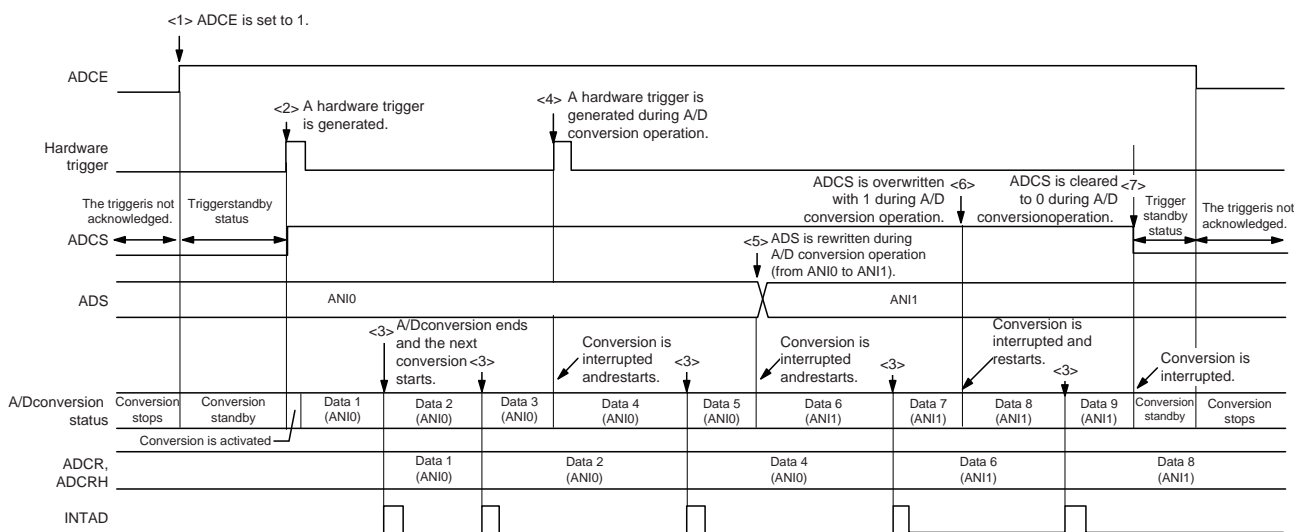




### 9.6.9 Hardware trigger wait mode (select mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.
- <2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.
- <3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion ends, the next A/D conversion immediately starts. (At this time, no hardware trigger is necessary.)
- <4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

**Figure 9-25. Example of Hardware Trigger Wait Mode (Select Mode, Sequential Conversion Mode) Operation Timing**

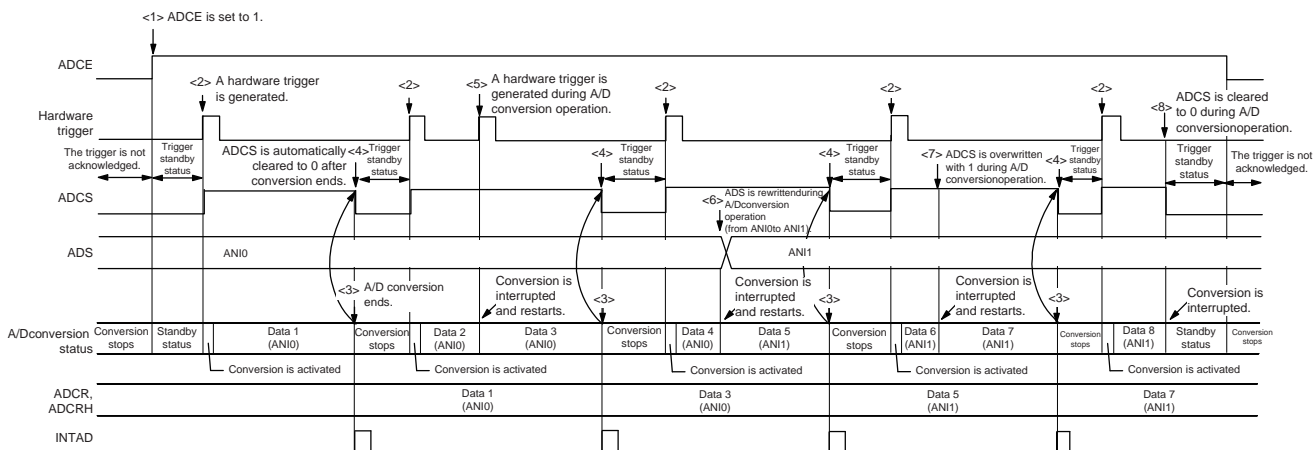


9.6.10 Hardware trigger wait mode (select mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the hardware trigger standby status.
- <2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the analog input specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input.
- <3> When A/D conversion ends, the conversion result is stored in the A/D conversion result register (ADCR, ADCRH), and the A/D conversion end interrupt request signal (INTAD) is generated.
- <4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.
- <5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the analog input respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts. The partially converted data is initialized.
- <8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 9-26. Example of Hardware Trigger Wait Mode (Select Mode, One-Shot Conversion Mode) Operation Timing

<R>

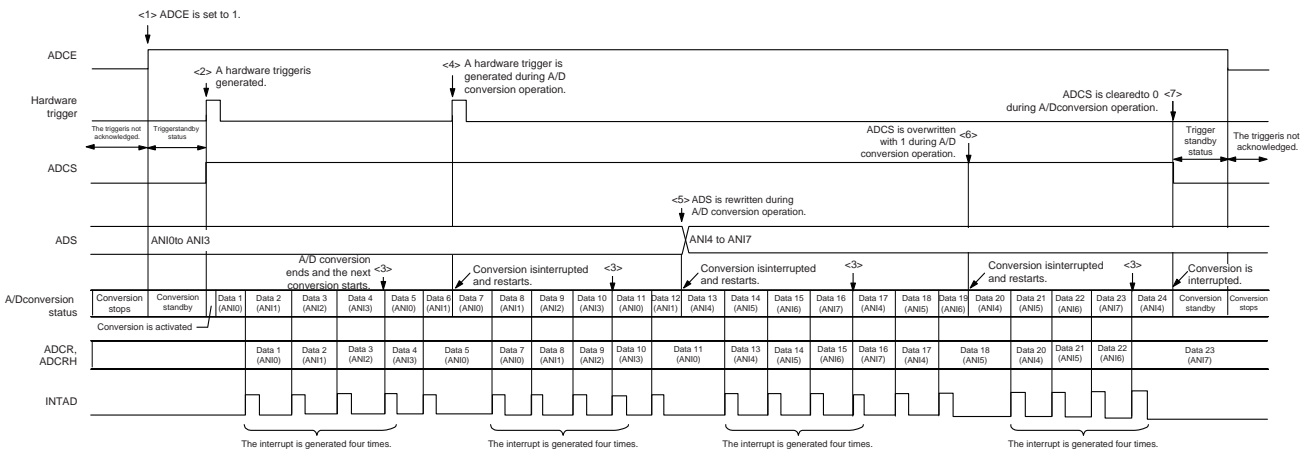


9.6.11 Hardware trigger wait mode (scan mode, sequential conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated. After A/D conversion of the four channels ends, the A/D conversion of the channel following the specified channel automatically starts.
- <4> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <5> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <6> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <7> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 9-27. Example of Hardware Trigger Wait Mode (Scan Mode, Sequential Conversion Mode) Operation Timing

<R>

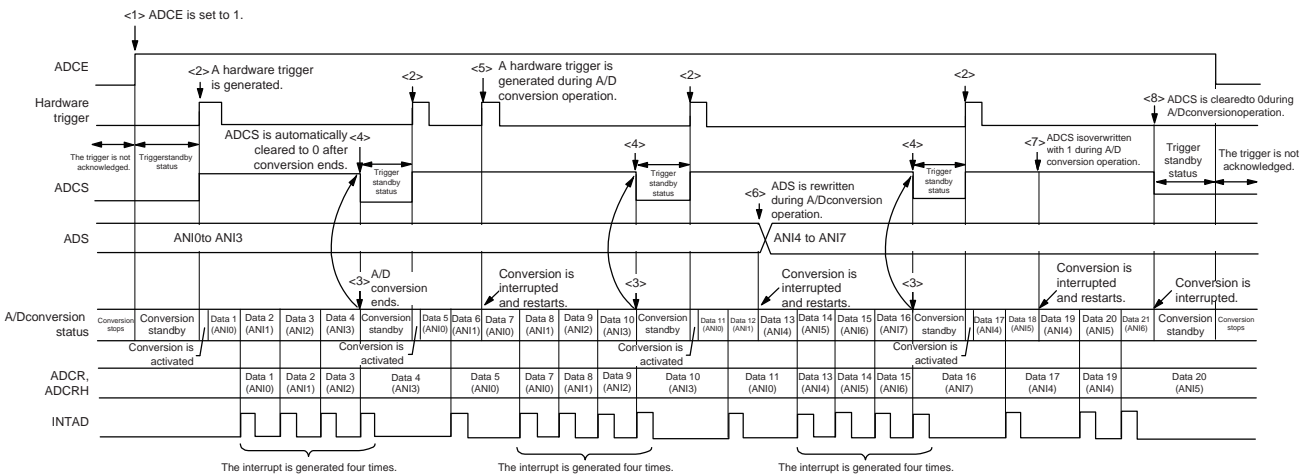


9.6.12 Hardware trigger wait mode (scan mode, one-shot conversion mode)

- <1> In the stop status, the ADCE bit of A/D converter mode register 0 (ADM0) is set to 1, and the system enters the A/D conversion standby status.
- <2> If a hardware trigger is input while in the hardware trigger standby status, A/D conversion is performed on the four analog input channels specified by scan 0 to scan 3, which are specified by the analog input channel specification register (ADS). The ADCS bit of the ADM0 register is automatically set to 1 according to the hardware trigger input. A/D conversion is performed on the analog input channels in order, starting with that specified by scan 0.
- <3> A/D conversion is sequentially performed on the four analog input channels, the conversion results are stored in the A/D conversion result register (ADCR, ADCRH) each time conversion ends, and the A/D conversion end interrupt request signal (INTAD) is generated.
- <4> After A/D conversion ends, the ADCS bit is automatically cleared to 0, and the A/D converter enters the stop status.
- <5> If a hardware trigger is input during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <6> When the value of the ADS register is rewritten or overwritten during conversion operation, the current A/D conversion is interrupted, and A/D conversion is performed on the first channel respecified by the ADS register. The partially converted data is discarded.
- <7> When ADCS is overwritten with 1 during conversion operation, the current A/D conversion is interrupted, and conversion restarts at the first channel. The partially converted data is discarded.
- <8> When ADCS is cleared to 0 during conversion operation, the current A/D conversion is interrupted, the system enters the hardware trigger standby status, and the A/D converter enters the stop status. When ADCE = 0, inputting a hardware trigger is ignored and A/D conversion does not start.

Figure 9-28. Example of Hardware Trigger Wait Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing

<R>

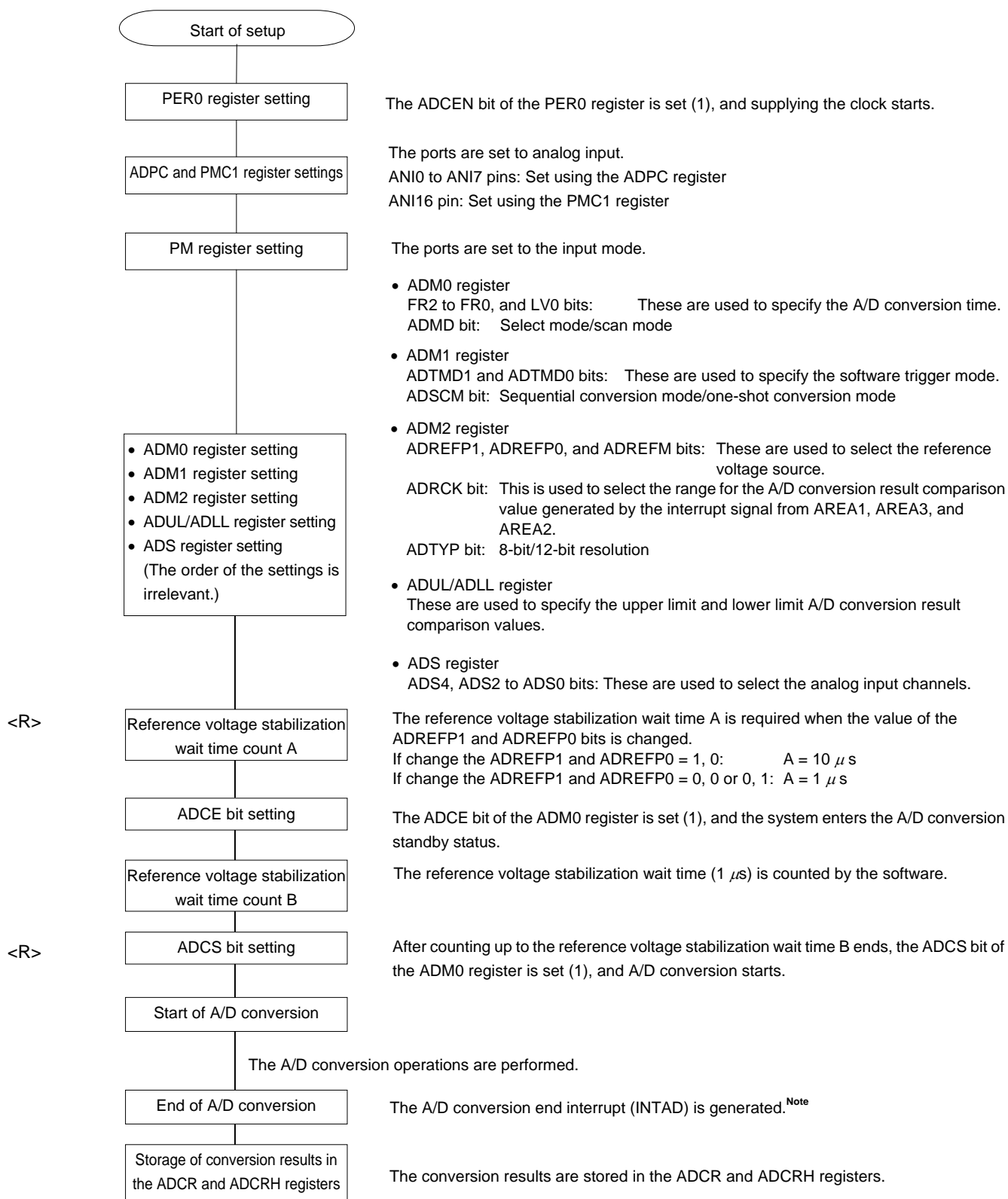


## 9.7 A/D Converter Setup Flowchart

The A/D converter setup flowchart in each operation mode is described below.

## 9.7.1 Setting up software trigger mode

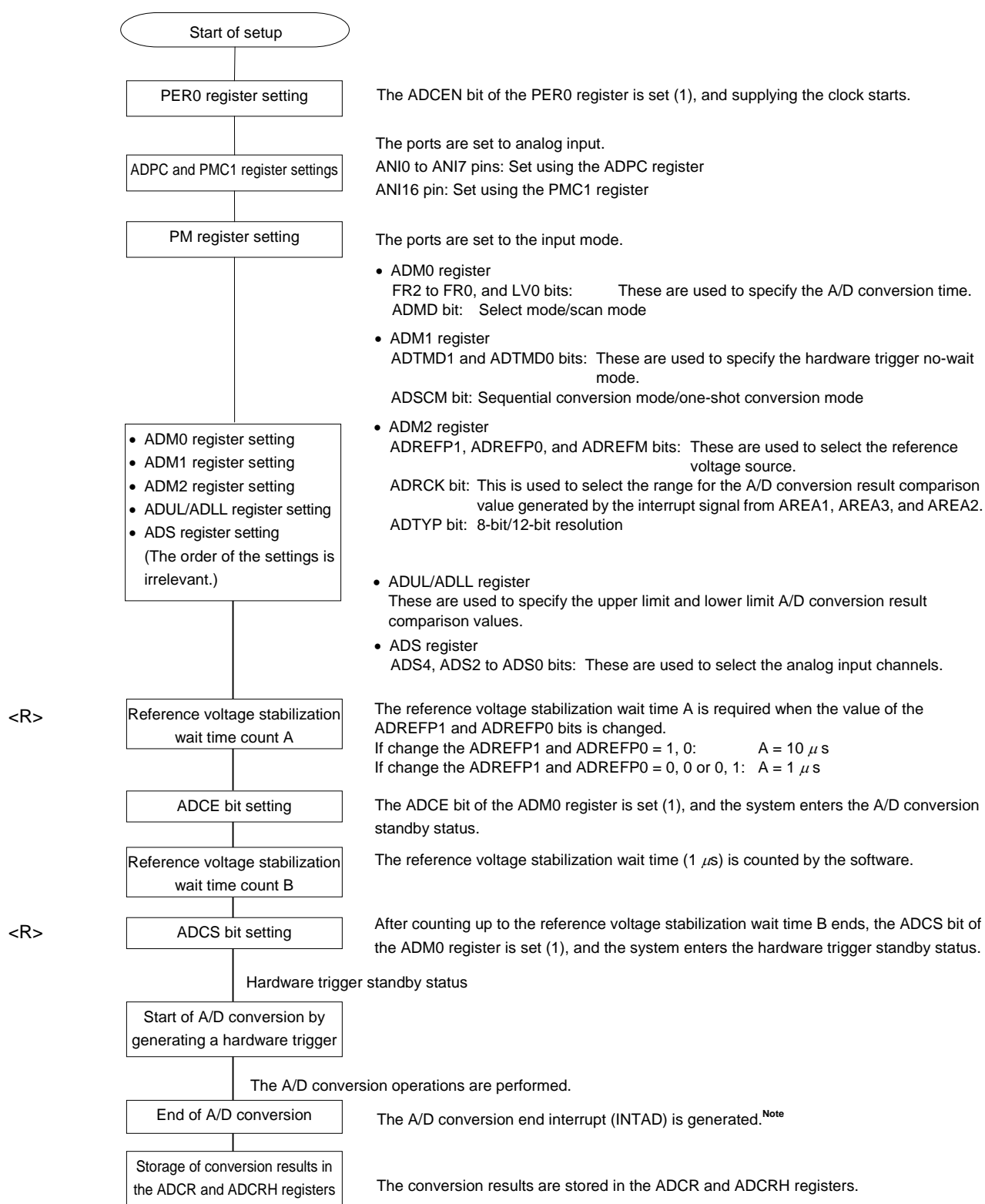
Figure 9-29. Setting up Software Trigger Mode



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

## 9.7.2 Setting up hardware trigger no-wait mode

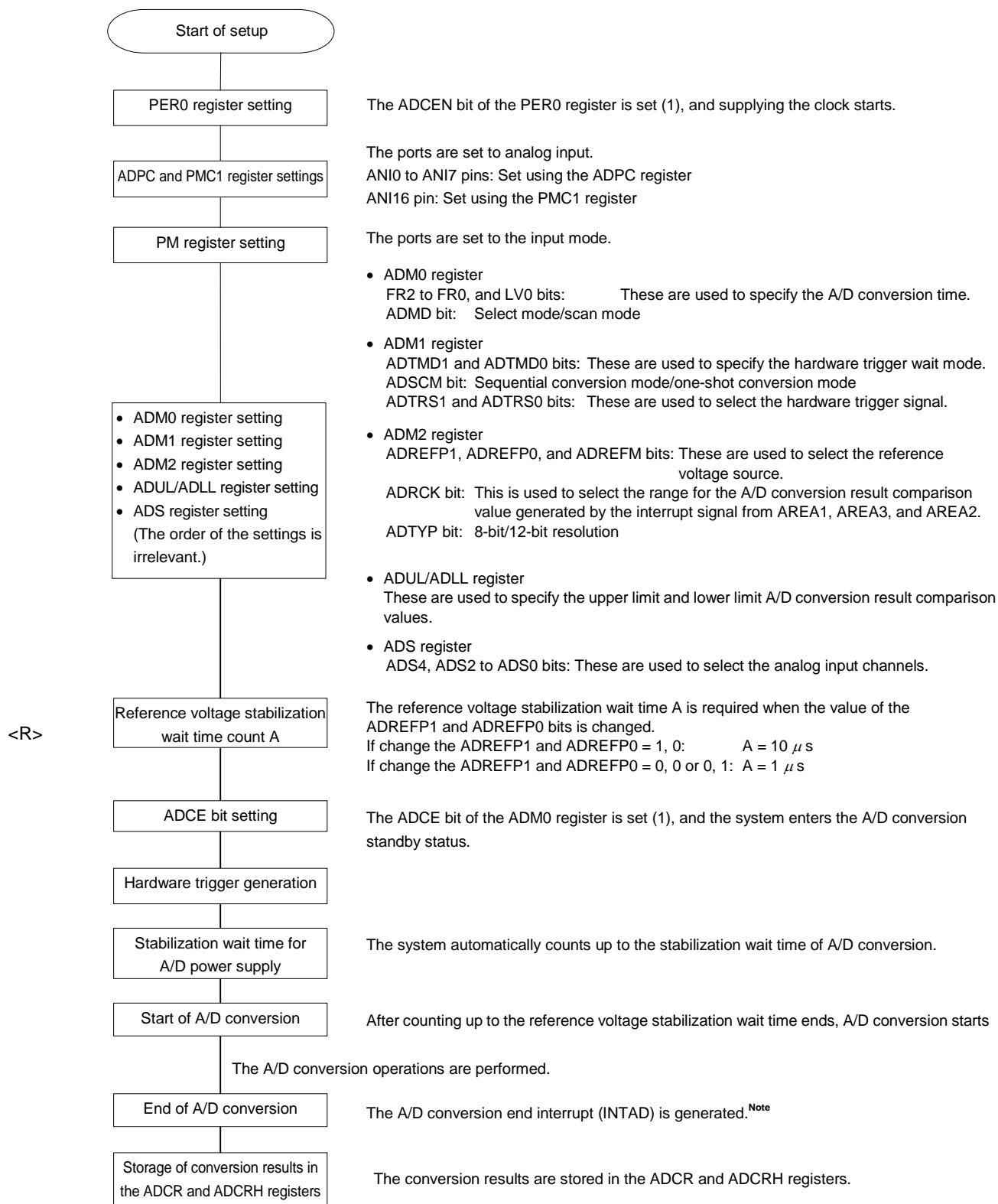
Figure 9-30. Setting up Hardware Trigger No-Wait Mode



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

## 9.7.3 Setting up hardware trigger wait mode

Figure 9-31. Setting up Hardware Trigger Wait Mode

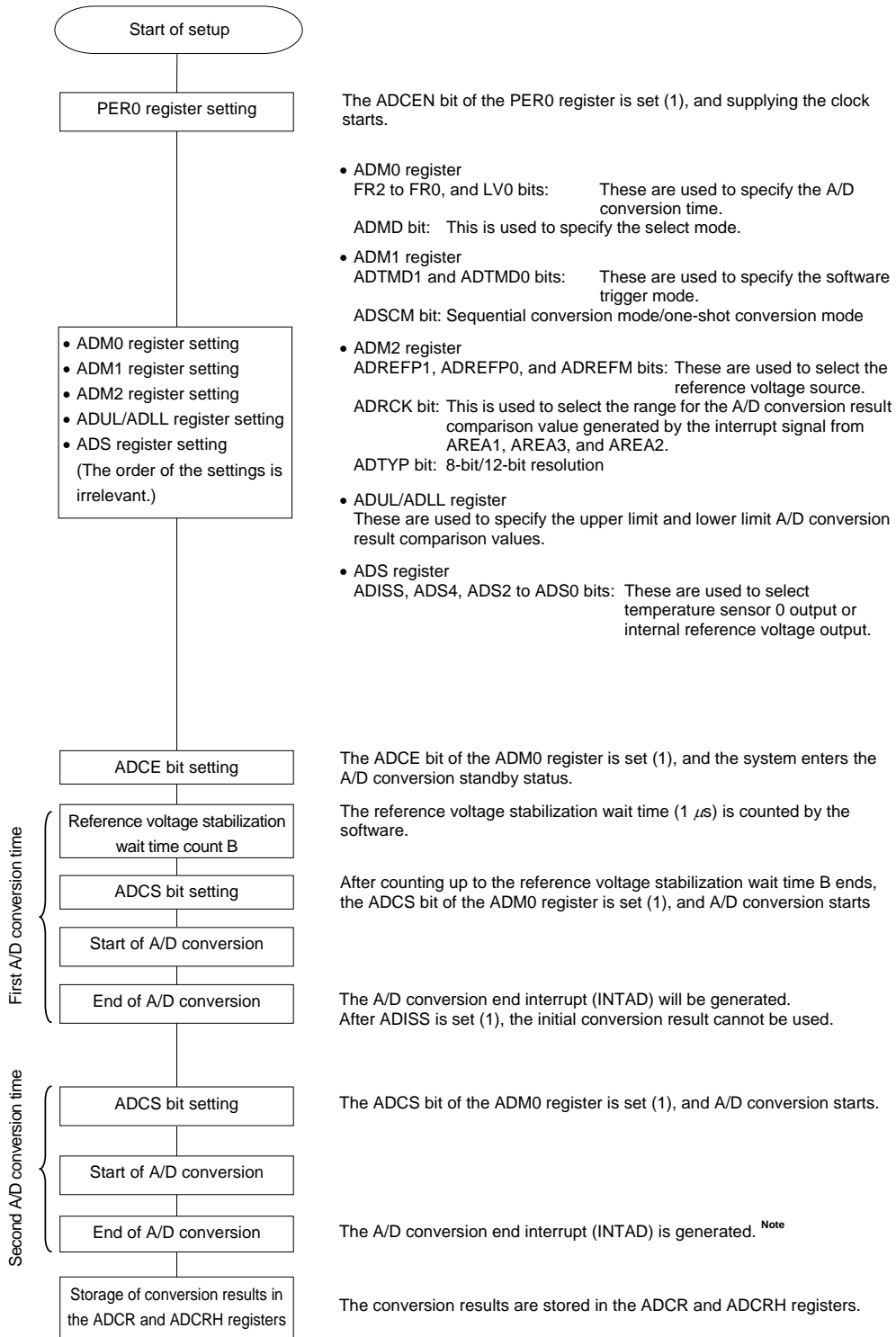


**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.



9.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode)

Figure 9-32. Setup when temperature sensor output voltage/internal reference voltage is selected (ADISS = 1)

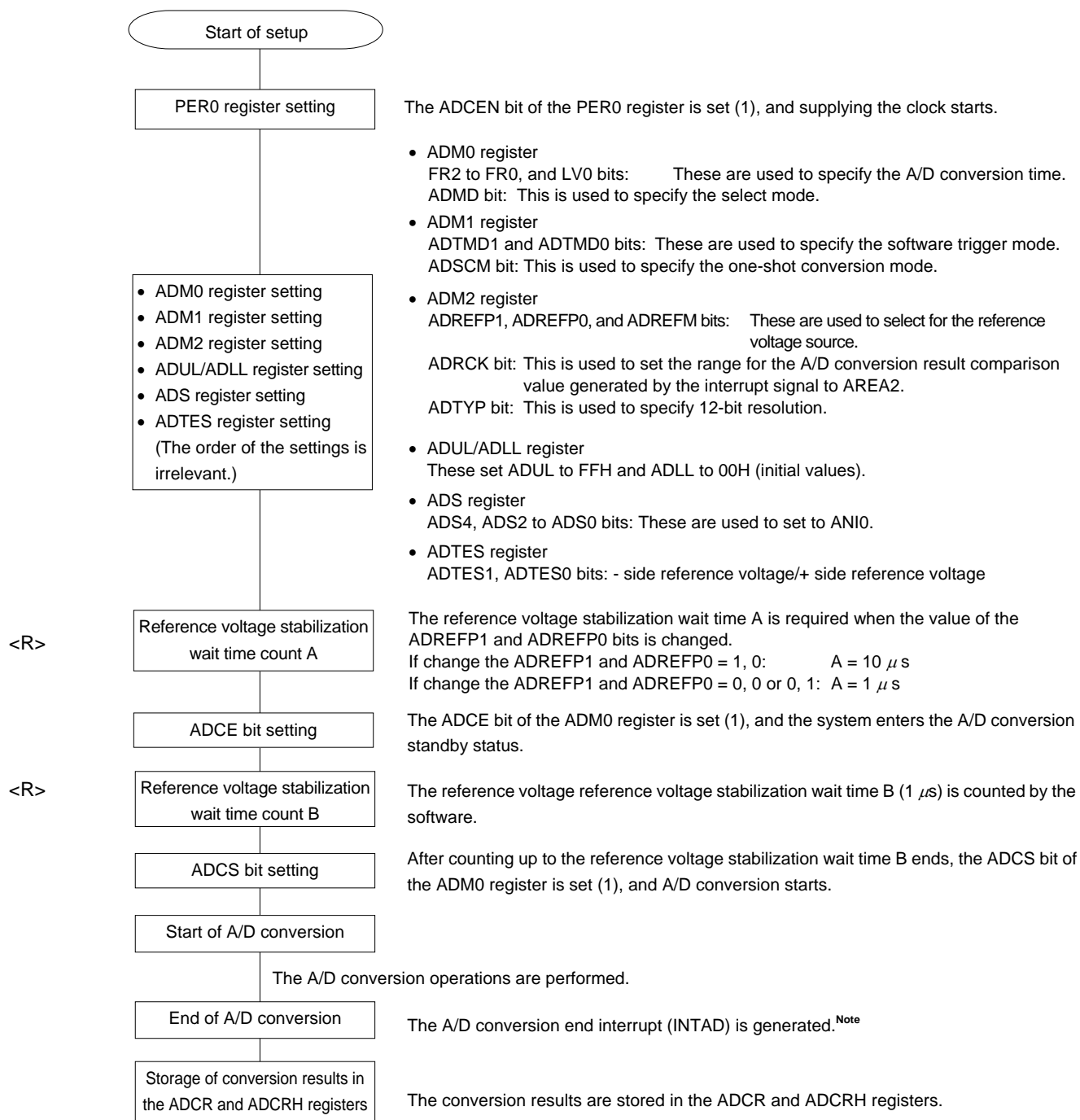


**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

**Caution** This setting can be used only in HS (high-speed main) mode. For detail, refer to Figure 22-3 Format of User Option Byte (000C2H).

## 9.7.5 Setting up test mode

Figure 9-33. Setting up Test Mode



**Note** Depending on the settings of the ADRCK bit and ADUL/ADLL register, there is a possibility of no interrupt signal being generated. In this case, the results are not stored in the ADCR, ADCRH registers.

**Caution** For the procedure for testing the A/D converter, see 20.10 A/D Test Function.

## 9.8 SNOOZE Mode Function

In the SNOOZE mode, A/D conversion is triggered by inputting a hardware trigger in the STOP mode. Normally, A/D conversion is stopped while in the STOP mode, but, by using the SNOOZE mode, A/D conversion can be performed without operating the CPU by inputting a hardware trigger. This is effective for reducing the operation current.

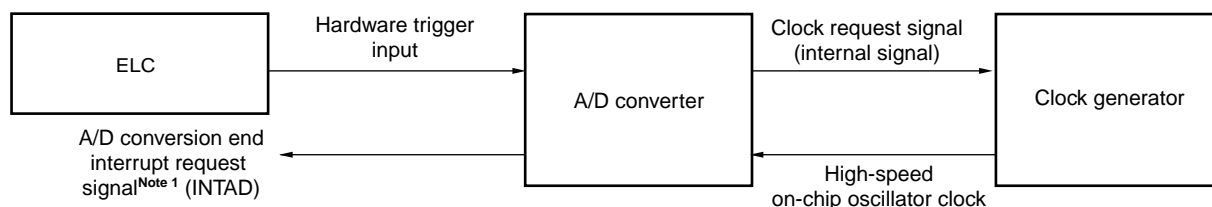
If the A/D conversion result range is specified using the ADUL and ADLL registers, A/D conversion results can be judged at a certain interval of time in SNOOZE mode. Using this function enables power supply voltage monitoring and input key judgment based on A/D inputs.

In the SNOOZE mode, only the following two conversion modes can be used:

- Hardware trigger wait mode (select mode, one-shot conversion mode)
- Hardware trigger wait mode (scan mode, one-shot conversion mode)

**Caution** The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for  $f_{CLK}$ .

Figure 9-34. Block Diagram When Using SNOOZE Mode Function



When using the SNOOZE mode function, the initial setting of each register is specified before switching to the STOP mode. (For details about these settings, see 9.7.3 **Setting up hardware trigger wait mode**<sup>Note 2</sup>.) Just before move to STOP mode, bit 2 (AWC) of A/D converter mode register 2 (ADM2) is set to 1. After the initial settings are specified, bit 0 (ADCE) of A/D converter mode register 0 (ADM0) is set to 1.

If a hardware trigger is input after switching to the STOP mode, the high-speed on-chip oscillator clock is supplied to the A/D converter. After supplying this clock, the system automatically counts up to the A/D power supply stabilization wait time, and then A/D conversion starts.

The SNOOZE mode operation after A/D conversion ends differs depending on whether an interrupt signal is generated<sup>Note 1</sup>.

**Notes** 1. Depending on the setting of the A/D conversion result comparison function (ADRCK bit, ADUL/ADLL register), there is a possibility of no interrupt signal being generated.

2. Be sure to set the ADM1 register to E1H.

**Remark** The hardware trigger is event signal (any of INTP0 to INTP5) selected by ELC. Specify the hardware trigger by using the A/D converter mode register 1 (ADM1).

**(1) If an interrupt is generated after A/D conversion ends**

If the A/D conversion result value is inside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL/ADLL register), the A/D conversion end interrupt request signal (INTAD) is generated.

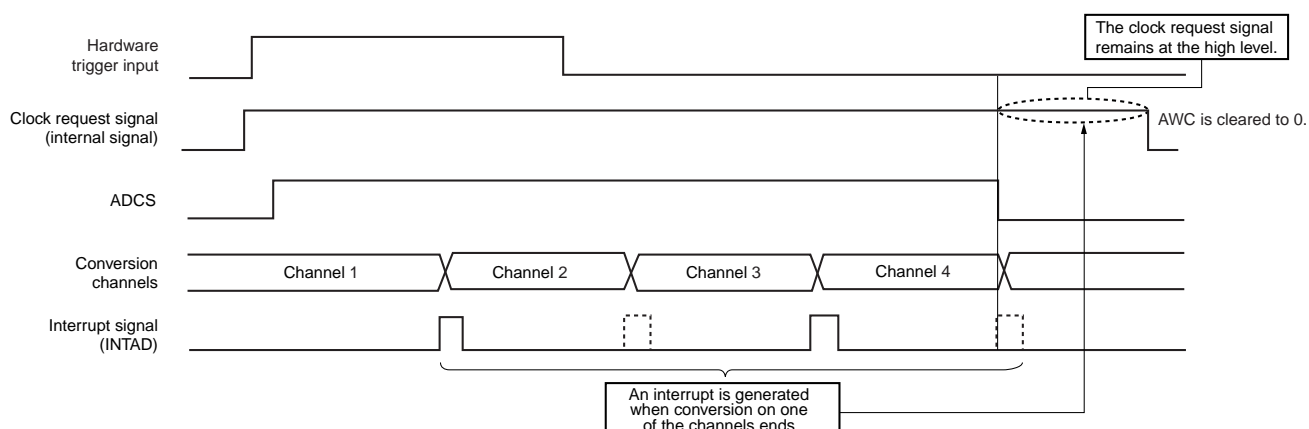
- While in the select mode

When A/D conversion ends and an A/D conversion end interrupt request signal (INTAD) is generated, the A/D converter returns to normal operation mode from SNOOZE mode. At this time, be sure to clear bit 2 (AWC = 0: SNOOZE mode release) of the A/D converter mode register 2 (ADM2). If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

- While in the scan mode

If even one A/D conversion end interrupt request signal (INTAD) is generated during A/D conversion of the four channels, the clock request signal remains at the high level, and the A/D converter switches from the SNOOZE mode to the normal operation mode. At this time, be sure to clear bit 2 (AWC = 0: SNOOZE mode release) of A/D converter mode register 2 (ADM2) to 0. If the AWC bit is left set to 1, A/D conversion will not start normally in the subsequent SNOOZE or normal operation mode.

**Figure 9-35. Operation Example When Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



**(2) If no interrupt is generated after A/D conversion ends**

If the A/D conversion result value is outside the range of values specified by the A/D conversion result comparison function (which is set up by using the ADRCK bit and ADUL/ADLL register), the A/D conversion end interrupt request signal (INTAD) is not generated.

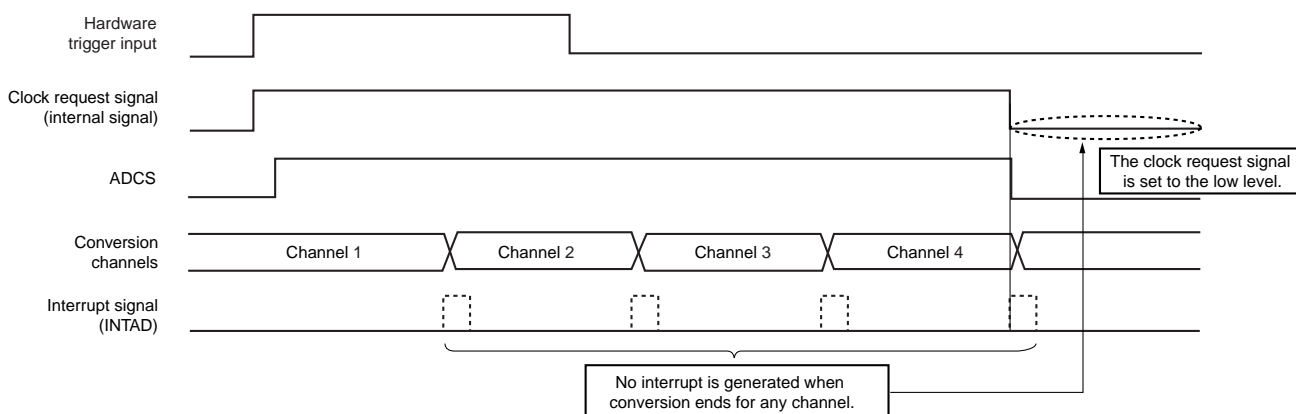
- While in the select mode

If the A/D conversion end interrupt request signal (INTAD) is not generated after A/D conversion ends, the clock request signal (an internal signal) is automatically set to the low level, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

- While in the scan mode

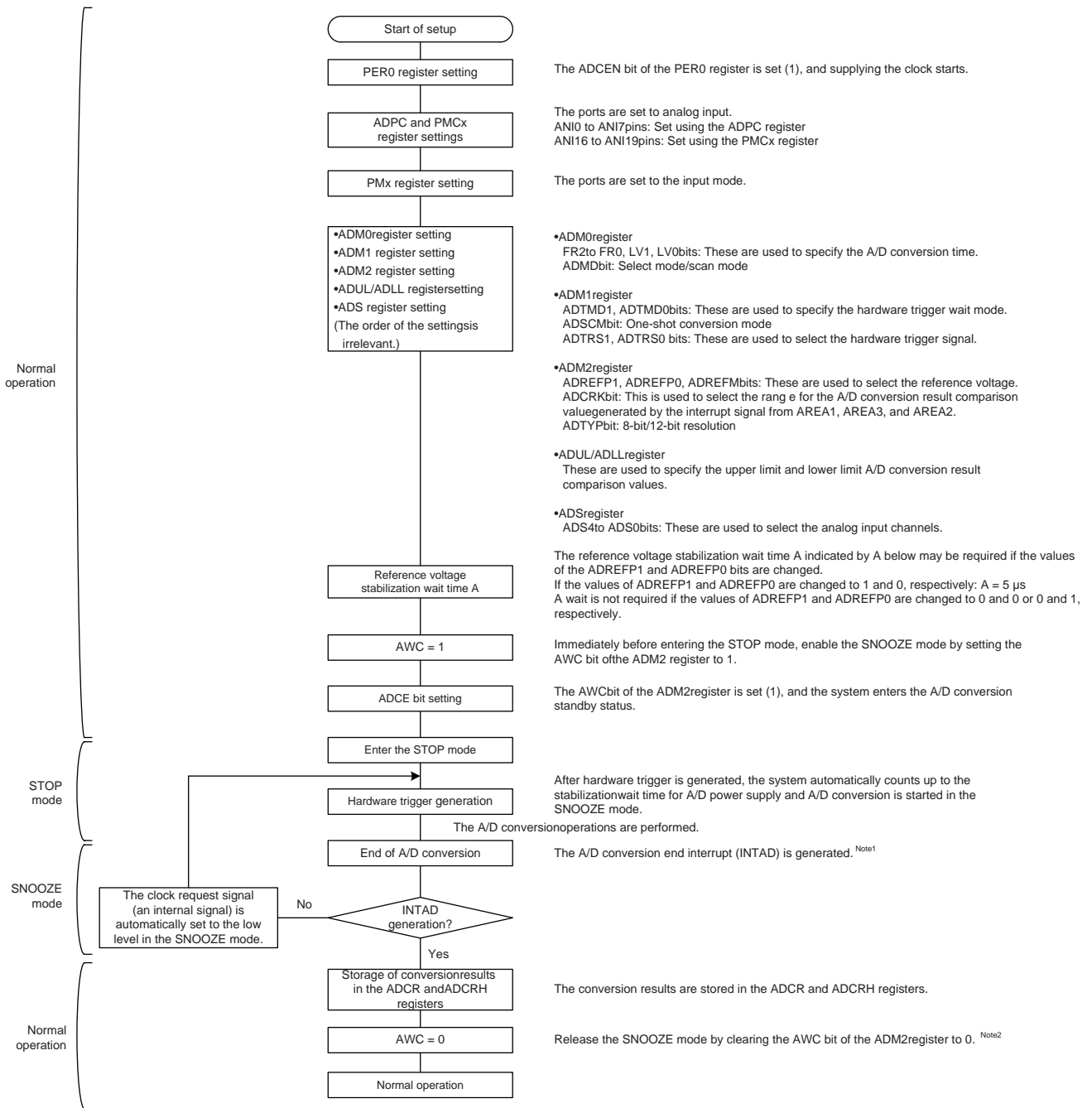
If the A/D conversion end interrupt request signal (INTAD) is not generated even once during A/D conversion of the four channels, the clock request signal (an internal signal) is automatically set to the low level after A/D conversion of the four channels ends, and supplying the high-speed on-chip oscillator clock stops. If a hardware trigger is input later, A/D conversion work is again performed in the SNOOZE mode.

**Figure 9-36. Operation Example When No Interrupt Is Generated After A/D Conversion Ends (While in Scan Mode)**



<R>

Figure 9-37. Flowchart for Setting up SNOOZE Mode



- Notes**
1. If the A/D conversion end interrupt request signal (INTAD) is not generated by setting ADRCK bit and ADUL/ADLL register, the result is not stored in the ADCR and ADCRH registers. The system enters the STOP mode again. If a hardware trigger is input later, A/D conversion operation is again performed in the SNOOZE mode..
  2. If the AWC bit is left set to 1, A/D conversion will not start normally in spite of the subsequent SNOOZE or normal operation mode. Be sure to clear the AWC bit to 0.

## 9.9 How to Read A/D Converter Characteristics Table

Here, special terms unique to the A/D converter are explained.

### (1) Resolution

This is the minimum analog input voltage that can be identified. That is, the percentage of the analog input voltage per bit of digital output is called 1LSB (Least Significant Bit). The percentage of 1LSB with respect to the full scale is expressed by %FSR (Full Scale Range).

1LSB is as follows when the resolution is 12 bits.

$$\begin{aligned} 1\text{LSB} &= 1/2^{12} = 1/4096 \\ &\approx 0.024\%\text{FSR} \end{aligned}$$

Accuracy has no relation to resolution, but is determined by overall error.

### (2) Overall error

This shows the maximum error value between the actual measured value and the theoretical value.

Zero-scale error, full-scale error, integral linearity error, and differential linearity errors that are combinations of these express the overall error.

Note that the quantization error is not included in the overall error in the characteristics table.

### (3) Quantization error

When analog values are converted to digital values, a  $\pm 1/2\text{LSB}$  error naturally occurs. In an A/D converter, an analog input voltage in a range of  $\pm 1/2\text{LSB}$  is converted to the same digital code, so a quantization error cannot be avoided.

Note that the quantization error is not included in the overall error, zero-scale error, full-scale error, integral linearity error, and differential linearity error in the characteristics table.

Figure 9-38. Overall Error

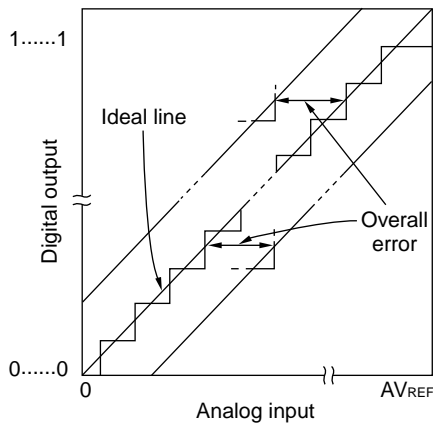
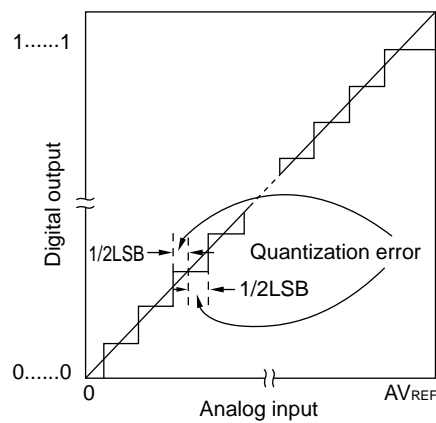


Figure 9-39. Quantization Error



### (4) Zero-scale error

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value ( $1/2\text{LSB}$ ) when the digital output changes from  $0.....000$  to  $0.....001$ .

If the actual measurement value is greater than the theoretical value, it shows the difference between the actual measurement value of the analog input voltage and the theoretical value ( $3/2\text{LSB}$ ) when the digital output changes from  $0.....001$  to  $0.....010$ .

**(5) Full-scale error**

This shows the difference between the actual measurement value of the analog input voltage and the theoretical value (Full-scale – 3/2LSB) when the digital output changes from 1.....110 to 1.....111.

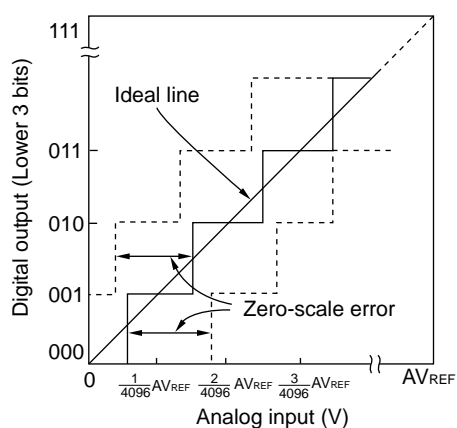
**(6) Integral linearity error**

This shows the degree to which the conversion characteristics deviate from the ideal linear relationship. It expresses the maximum value of the difference between the actual measurement value and the ideal straight line when the zero-scale error and full-scale error are 0.

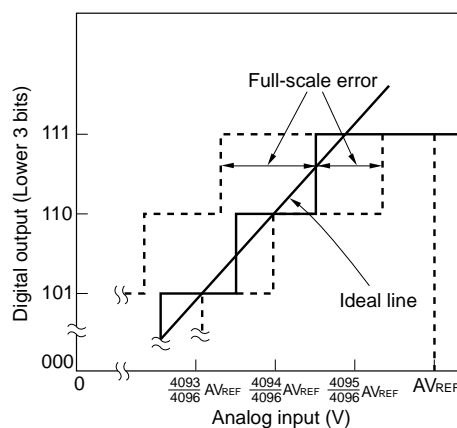
**(7) Differential linearity error**

While the ideal width of code output is 1LSB, this indicates the difference between the actual measurement value and the ideal value.

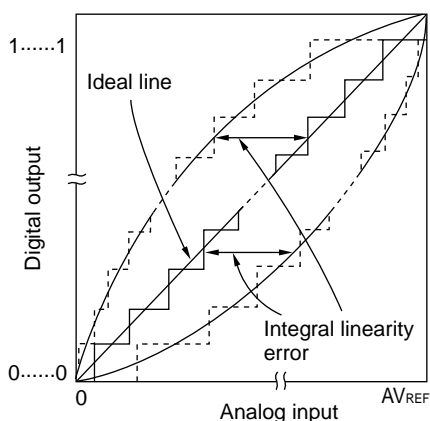
**Figure 9-40. Zero-Scale Error**



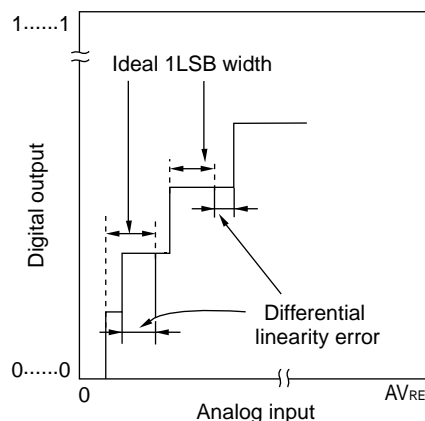
**Figure 9-41. Full-Scale Error**



**Figure 9-42. Integral Linearity Error**



**Figure 9-43. Differential Linearity Error**

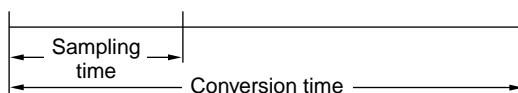


**(8) Conversion time**

This expresses the time from the start of sampling to when the digital output is obtained. The sampling time is included in the conversion time in the characteristics table.

**(9) Sampling time**

This is the time the analog switch is turned on for the analog voltage to be sampled by the sample & hold circuit.





## 9.10 Cautions for A/D Converter

### (1) Operating current in STOP mode

Shift to STOP mode after stopping the A/D converter (by setting bit 7 (ADCS) of A/D converter mode register 0 (ADM0) to 0). The operating current can be reduced by setting bit 0 (ADCE) of the ADM0 register to 0 at the same time.

To restart from the standby status, clear bit 0 (ADIF) of interrupt request flag register 1H (IF1H) to 0 and start operation.

### (2) Input range of ANI0 to ANI7 and ANI16 pins

Observe the rated range of the ANI0 to ANI7 and ANI16 pins input voltage. If a voltage of  $V_{DD}$ , and  $AV_{REFP}$  or higher and  $V_{SS}$ , and  $AV_{REFM}$  or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

When internal reference voltage (1.45 V) is selected reference voltage source for the + side of the A/D converter, do not input internal reference voltage or higher voltage to a pin selected by the ADS register. However, it is no problem that a pin not selected by the ADS register is input voltage greater than the internal reference voltage.

**Caution** Internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode. For detail, refer to Figure 22-3 Format of User Option Byte (000C2H).

### (3) Conflicting operations

<1> Conflict between the A/D conversion result register (ADCR, ADCRH) write and the ADCR or ADCRH register read by instruction upon the end of conversion

The ADCR or ADCRH register read has priority. After the read operation, the new conversion result is written to the ADCR or ADCRH registers.

<2> Conflict between the ADCR or ADCRH register write and the A/D converter mode register 0 (ADM0) write, the analog input channel specification register (ADS), or A/D port configuration register (ADPC) write upon the end of conversion

The ADM0, ADS, or ADPC registers write has priority. The ADCR or ADCRH register write is not performed, nor is the conversion end interrupt signal (INTAD) generated.

### (4) Noise countermeasures

To maintain the 12-bit resolution, attention must be paid to noise input to the  $AV_{REFP}$ ,  $V_{DD}$ , ANI0 to ANI7, and ANI16 pins.

<1> Be sure to separate  $V_{DD}$  and  $V_{SS}$  from other power supplies and connect a capacitor with low equivalent resistance and good frequency response characteristics (a capacitance of about 0.01  $\mu F$  is recommended) between  $V_{DD}$  and  $V_{SS}$ .

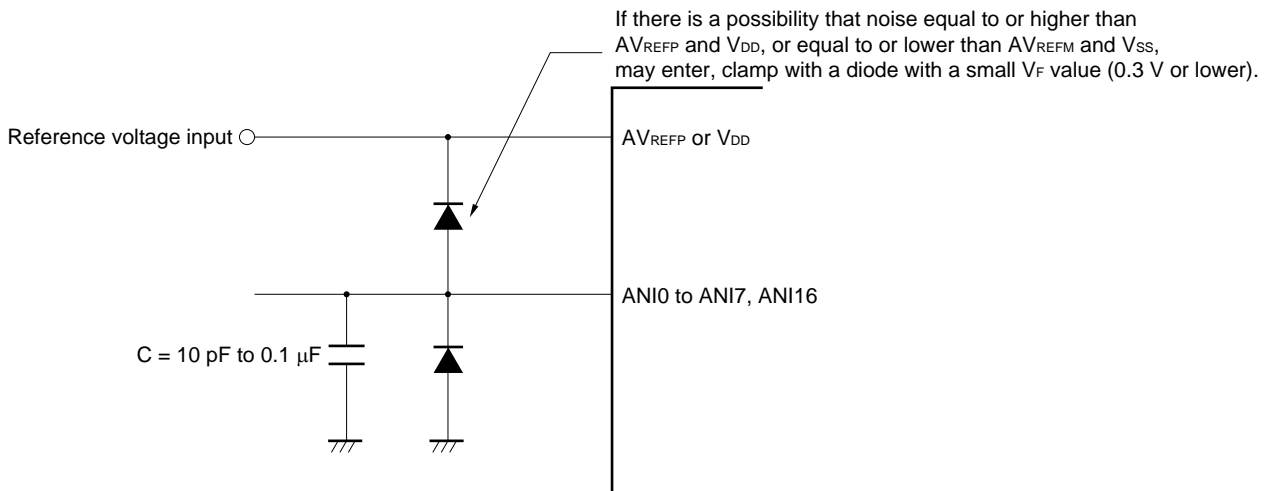
<2> The higher the output impedance of the analog input source, the greater the influence. To reduce the noise, connecting external C as shown in Figure 9-44 is recommended.

<3> Do not switch these pins with other pins during conversion.

<4> The accuracy is improved if the HALT mode is set immediately after the start of conversion.

<5> Separate digital and analog signals so that they do not cross or approach each other.

Figure 9-44. Analog Input Pin Connection



### (5) Analog input (ANIn) pins

<1> ANI0 to ANI7 pins are also used as P20 to P27 pins.

When A/D conversion is performed with any of the ANI0 to ANI17 pins selected, do not change the output value P20 to P27 while conversion is in progress; otherwise the conversion accuracy may be degraded.

<2> If a pin adjacent to the pin whose value is being A/D converted is used as a digital I/O port pin, the A/D conversion value might differ from the expected value due to coupling noise. To prevent coupling noise, make sure that pulses whose voltage suddenly change, such as digital pulses, are not input or output to a pin adjacent to the pin whose value is being A/D converted.

### (6) Input impedance of analog input (ANIn) pins

This A/D converter charges a sampling capacitor for sampling during sampling time.

Therefore, only a leakage current flows when sampling is not in progress, and a current that charges the capacitor flows during sampling. Consequently, the input impedance fluctuates depending on whether sampling is in progress, and on the other states.

To make sure that sampling is effective, however, we recommend using the converter with analog input sources that have output impedances no greater than  $1k\Omega$ . If a source has a higher output impedance, lengthen the sampling time or connect a larger capacitor (with a value of about  $0.1\ \mu F$ ) to the pin from among ANI0 to ANI7 and ANI16 to which the source is connected (see Figure 9-44). The sampling capacitor may be being charged while the setting of the ADCS bit is 0 and immediately after sampling is restarted and so is not defined at these times. Accordingly, the state of conversion is undefined after charging starts in the next round of conversion after the value of the ADCS bit has been 1 or when conversion is repeated. Thus, to secure full charging regardless of the size of fluctuations in the analog signal, ensure that the output impedances of the sources of analog inputs are low or secure sufficient time for the completion of conversion.

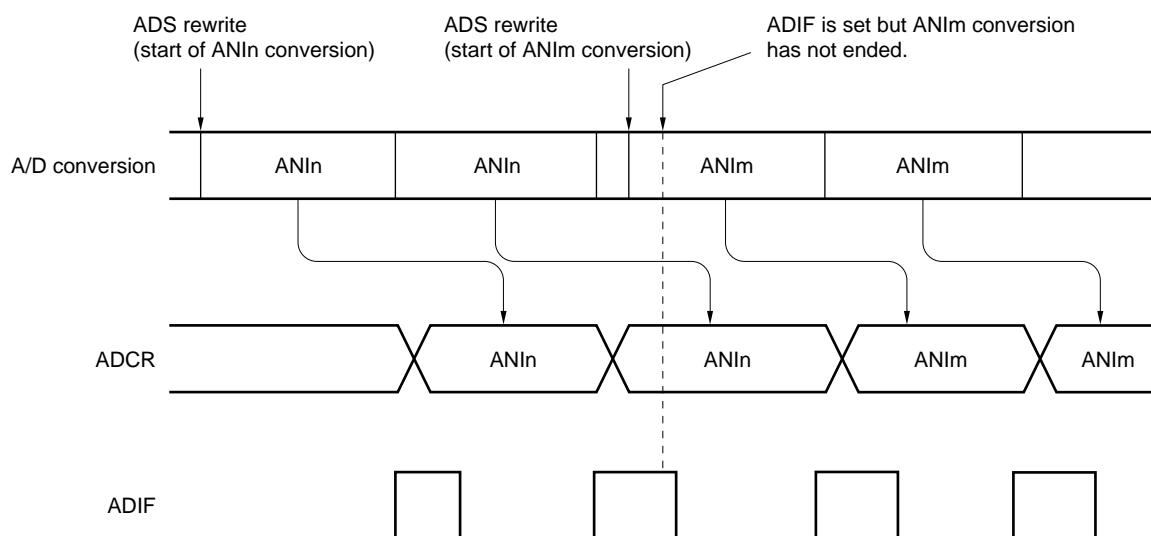
**(7) Interrupt request flag (ADIF)**

The interrupt request flag (ADIF) is not cleared even if the analog input channel specification register (ADS) is changed.

Therefore, if an analog input pin is changed during A/D conversion, the A/D conversion result and ADIF flag for the pre-change analog input may be set just before the ADS register rewrite. Caution is therefore required since, at this time, when ADIF flag is read immediately after the ADS register rewrite, ADIF flag is set despite the fact A/D conversion for the post-change analog input has not ended.

When A/D conversion is stopped and then resumed, clear ADIF flag before the A/D conversion operation is resumed.

**Figure 9-45. Timing of A/D Conversion End Interrupt Request Generation**

**(8) Conversion results just after A/D conversion start**

While in the software trigger mode or hardware trigger no-wait mode, the first A/D conversion value immediately after A/D conversion starts may not fall within the rating range if the ADCS bit is set to 1 within 1  $\mu$ s after the ADCE bit was set to 1. Take measures such as polling the A/D conversion end interrupt request (INTAD) and removing the first conversion result.

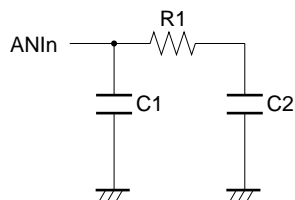
**(9) A/D conversion result register (ADCR, ADCRH) read operation**

When a write operation is performed to A/D converter mode register 0 (ADM0), analog input channel specification register (ADS), A/D port configuration register (ADPC), and port mode control register (PMC), the contents of the ADCR and ADCRH registers may become undefined. Read the conversion result following conversion completion before writing to the ADM0, ADS, ADPC, or PMC register. Using a timing other than the above may cause an incorrect conversion result to be read.

**(10) Internal equivalent circuit**

The equivalent circuit of the analog input block is shown below.

**Figure 9-46. Internal Equivalent Circuit of ANIn Pin**



**Table 9-4. Resistance and Capacitance Values of Equivalent Circuit (Reference Values)**

$V_{DD}$	ANIn Pin	R1[k $\Omega$ ]	C1[pF]	C2[pF]
$2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	ANI0 to ANI7	7.4	8	6.3
	ANI16	12.3	8	7.4

**Remark** The resistance and capacitance values shown in Table 9-4 are not guaranteed values.

**(11) Starting the A/D converter**

Start the A/D converter after the  $AV_{REFP}$  and  $V_{DD}$  voltages stabilize.

## CHAPTER 10 D/A CONVERTER

The D/A converter is a 10-bit resolution R-2R type unit that converts digital inputs into analog signals. It is used to control analog outputs for two independent channels.

### 10.1 Function of D/A Converter

The D/A converter has the following features.

- 10-bit resolution × 2 channels
- R-2R ladder method
- Output analog voltage
  - 10-bit resolution:  $V_{DD} \times m10/1024$  (m10: Value set to DACSi register)
- Operation mode
  - Normal mode
  - Real-time output mode

**Remark**  $i = 0, 1$

### 10.2 Configuration of D/A Converter

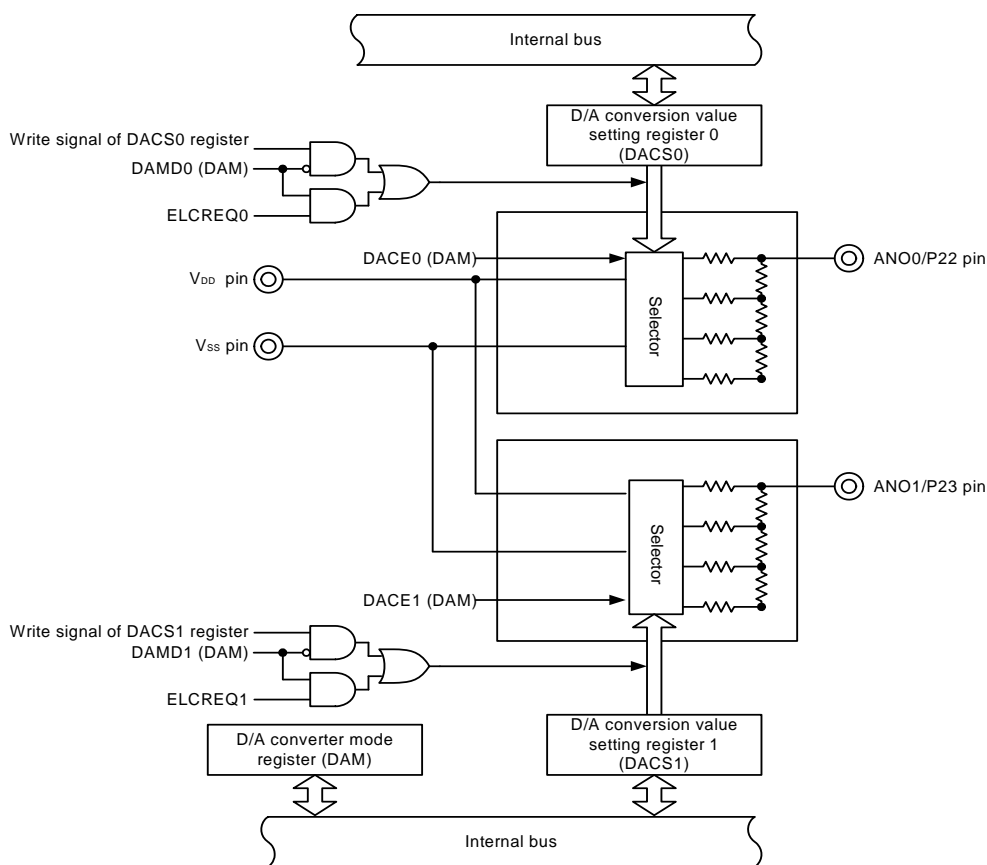
The D/A converter includes the following hardware.

**Table 10-1. Configuration of D/A Converter**

Item	Configuration
Control registers	A/D port configuration register (ADPC) Peripheral enable register 1 (PER1) D/A converter mode register (DAM) D/A conversion value setting registers 0, 1 (DACS0, DACS1) Port mode register 2 (PM2)

Figure 10-1 shows the block diagram of D/A converter.

**Figure 10-1. Block Diagram of D/A Converter**



**Remark** ELCREQ0 and ELCREQ1 are trigger signals (request signals from the ELC) that are used in the real-time output mode.

### 10.3 Configuration of A/D Converter

The D/A converter uses the following registers.

- A/D port configuration register (ADPC)
- Peripheral enable register 1 (PER1)
- D/A converter mode register (DAM)
- D/A conversion value setting registers 0, 1 (DACS0, DACS1)
- Port mode register 2 (PM2)

#### 10.3.1 A/D port configuration register (ADPC)

This register switches the ANI0/P20 to ANI7/P27 pins to either analog input or port digital I/O.

The ADPC register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-2. Format of A/D Port Configuration Register (ADPC)**

Address: F0076H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADPC	ADPC7 <sup>Note</sup>	ADPC6 <sup>Note</sup>	ADPC5 <sup>Note</sup>	ADPC4 <sup>Note</sup>	ADPC3	ADPC2	ADPC1	ADPC0
ADPCn	Analog I/O (A)/digital I/O (D) selection of P2n/ANI2n							
0	Analog I/O (A) (default)							
1	Digital I/O (D)							

**Note** 32-pin products only

- Cautions**
1. Set a channel to be used for D/A conversion in the input mode by using port mode register 2 (PM2).
  2. Do not set the pin that is set by the ADPC register as digital I/O to D/A conversion operation enable by using the D/A converter mode register (DAM).

### 10.3.2 Peripheral enable register 1 (PER1)

The PER1 register is used to enable or disable use of each peripheral hardware macro. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the D/A converter is used, be sure to set bit 7 (DACEN) of this register to 1.

The PER1 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-3. Format of Peripheral Enable Register 1 (PER1)**

Address: F007AH After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	0
PER1	DACEN	0	0	0	0	0	0	0

DACEN	Control of D/A converter input clock
0	Stops input clock supply <ul style="list-style-type: none"> <li>• SFR used by the D/A converter cannot be written.</li> <li>• The D/A converter is in the reset status.</li> </ul>
1	Supplies input clock. <ul style="list-style-type: none"> <li>• SFR used by the D/A converter can be read/written.</li> </ul>

- Cautions**
1. When setting the D/A converter, be sure to set the DACEN bit to 1 first. If DACEN = 0, writing to a control register of the D/A converter is ignored, and all read values are default values (except for port mode register 2 (PM2), and port register 2 (P2)).
  2. Be sure to clear bits 0 to 6 to "0".



### 10.3.3 D/A converter mode register (DAM)

This register controls the operation of the D/A converter.

The DAM register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-4. Format of D/A Converter Mode Register (DAM)**

Address: FFF5CH After reset: 00H R/W

Symbol	7	6	<5>	<4>	3	2	1	0
DAM	–	–	DACE1	DACE0	–	–	DAMD1	DAMD0

DACEi	D/A conversion operation control
0	Stops D/A conversion operation
1	Enables D/A conversion operation

DAMDi	D/A converter operation mode selection
0	Normal mode
1	Real-time output mode

**Remark** i = 0, 1

### 10.3.4 D/A conversion value setting register i (DACS<sub>i</sub>) (i = 0, 1)

This register is used to set the analog voltage value to be output to the ANO0 and ANO1 pins when the D/A converter is used.

The DACSi register can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 10-5. Format of D/A Conversion Value Setting Register i (DACS<sub>i</sub>) (i = 0, 1)**

Address: FFF58H, FFF59H (DACS<sub>0</sub>), FFF5AH, FFF5BH (DACS<sub>1</sub>) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DACS <sub>i</sub>							DAC Si9	DAC Si8	DAC Si7	DAC Si6	DAC Si5	DAC Si4	DAC Si3	DAC Si2	DAC Si1	DAC Si0

**Remark** The relation between the resolution and analog output voltage (VANO<sub>i</sub>) of the D/A converter are as follows.  
 $VANO_i = \text{Reference voltage for D/A converter} \times (\text{DACS}_i) / 1024$

When the D/A converter is not used, set the DACE<sub>i</sub> bit to 0 (output disable) and set the DACSi register to 00H to prevent current from flowing into the R-2R resistor ladder to reduce unnecessary current consumption.

### 10.3.5 Port mode register 2 (PM2)

When using ANO0/ANI2/P22 and ANO1/ANI3/P23 pins as analog input ports, set bits PM22 and PM23 to 1.

If bits PM22 and PM23 are set to 0, these pins cannot be used as analog input ports.

The PM2 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Caution** If a pin is set as an analog input port, not the pin level but 0 is always read.

**Figure 10-6. Format of Port Mode Register 2 (PM2)**

Address: FFF22H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM2	PM27	PM26 <sup>Note</sup>	PM25 <sup>Note</sup>	PM24 <sup>Note</sup>	PM23	PM22	PM21	PM20
PM2n	P2n pin I/O mode selection (n = 0 to 7)							
0	Output mode (output buffer on)							
1	Input mode (output buffer off)							

**Note** These are not provided in 24-pin products.

The function of the ANO0/ANI2/P22 to ANO1/ANI3/P23 pins can be selected by using the A/D port configuration register (ADPC), the D/A converter mode register (DAM), the analog input channel specification register (ADS), and the PM2 register.

**Table 10-2. Setting Functions of ANO0/ANI2/P22 to ANO1/ANI3/P23 Pins**

ADPC	PM2	DAM	ADS	Functions of ANO0/ANI2/P22 to ANO1/ANI3/P23 Pins
Digital I/O selection	Input mode	–	–	Digital input
	Output mode	–	–	Digital output
Analog I/O selection	Input mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	Analog output
		Stops D/A conversion operation	Selects ANI.	Analog input (to be converted)
			Does not select ANI.	Analog input (not to be converted)
	Output mode	Enables D/A conversion operation	Selects ANI.	Setting prohibited
			Does not select ANI.	
Stops D/A conversion operation	Stops D/A conversion operation	Selects ANI.	Setting prohibited	
		Does not select ANI.		

## 10.4 Operations of D/A Converter

### 10.4.1 Operation in normal mode

D/A conversion is performed using write operation to the DACSi register as the trigger.

The setting method is described below.

- <1> Set the DACEN bit of the PER1 register (peripheral enable register 1) to 1 to start the supply of the input clock to the D/A converter.
- <2> Use the ADPC register (port configuration register) to set the ports to analog pins.
- <3> Set the DAMDi bit of the DAM register (D/A converter mode register) to 0 (normal mode).
- <4> Set the analog voltage value to be output to the ANOi pin to the DACSi register (D/A conversion value setting register i).

Steps <1> and <4> above constitute the initial settings.

- <5> Set the DACEi bit of the DAM register to 1 (D/A conversion enable).  
D/A conversion starts, and then, after the settling time elapses, the analog voltage set in step <4> is output to the ANOi pin.
- <6> To perform subsequent D/A conversions, write to the DACSi register.

The previous D/A conversion result is held until the next D/A conversion is performed.

When the DACEi bit of the DAM register is set to 0 (D/A conversion operation stop), D/A conversion stops.

If the ports are set to digital pins using the ADPC register, the ANOi pin goes into a high-impedance state when the PM2i bit of the PM2 register for the port = 1 (input mode), and the ANOi pin outputs the set value of the PM2 register when the PM2i bit = 0 (output mode).

- Cautions**
1. Even if 1, 0, and then 1 is set to the DACEi bit, there is a wait after 1 is set for the last time.
  2. If the DACSi register is rewritten during the settling time, D/A conversion is aborted and reconversion by using the rewritten values starts.

**Remark** i = 0, 1

### 10.4.2 Operation in real-time output mode

D/A conversion is performed on each channel using the individual interrupt request signals from the ELC as triggers. The setting method is described below.

- <1> Set the DACEN bit of the PER1 register (peripheral enable register 1) to 1 to start the supply of the input clock to the D/A converter.
- <2> Use the ADPC register (port configuration register) to set the ports to analog pins.
- <3> Set the DAMDi bit of the DAM register (D/A converter mode register) to 0 (normal mode).
- <4> Set the analog voltage value to be output to the ANOi pin to the DACSi register (D/A conversion value setting register i).
- <5> Set the DACEi bit of the DAM register to 1 (D/A conversion enable).  
D/A conversion starts, and then, after the settling time elapses, the analog voltage set in step <3> is output to the ANOi pin.
- <6> Use the ELSELR register (ELC control register) to set the real-time trigger signal.
- <7> Set the DAMDi bit of the DAM register to 1 (real-time output mode).
- <8> Start the operation of the ECL request source.

Steps <1> to <8> above constitute the initial settings.

- <9> Generation of the real-time output triggers starts D/A conversion and the analog voltage set in step <4> will be output to the ANOi pin after a settling time has elapsed.  
Set the analog voltage value to be output to the ANOi pin, to the DACSi register before performing the next D/A conversion (real-time output trigger is generated).

Set the analog voltage value to be output to the ANOi pin, to the DACSi register before performing the next D/A conversion (ELC trigger signal is generated).

When the DACEi bit of the DAM register is set to 0 (D/A conversion operation stop), D/A conversion stops.

If the ports are set to digital pins by using the ADPC register, the ANOi pin goes into a high-impedance state when the PM2i bit of the PM2 register for the port = 1 (input mode), and the ANOi pin outputs the set value of the PM2 register when the PM2i bit = 0 (output mode).

- Cautions**
1. Even if 1, 0, and then 1 is set to the DACEi bit, there is a wait after 1 is set for the last time.
  2. Make the interval between each generation of the ELC event request trigger signal longer than the settling time. If an ELC event request trigger signal is generated during the settling time, D/A conversion is aborted and reconversion starts.
  3. Even if the generation of the ELC event request trigger signal and rewriting of the DACSi register conflict, the D/A conversion result is output.

## 10.5 Cautions for D/A Converter

Observe the following cautions when using the D/A converter.

- (1) The digital port I/O function, which is the alternate function of the ANO0 and ANO1 pins, does not operate if the ports are set to analog pins by using the ADPC register (port configuration register). When the P2 register is read while the ports are set to analog pins by using the ADPC register, 0 is read in the input mode and the set value of the P2 register is read in the output mode. If the digital output mode is set, no output data is output to pins.
- (2) The operation of the D/A converter continues in the HALT and STOP modes. To lower the power consumption, therefore, clear the DACE<sub>i</sub> bit to 0, and execute the HALT or STOP instruction after stopping the operation of the D/A converter.

**Remark**  $i = 0, 1$

- (3) To stop the real-time output mode (including when changing to normal mode), one of the following procedures must be used:
  - Wait for at least three clocks after stopping the trigger output source and then set bits DACE<sub>i</sub> and DAMD<sub>i</sub> to 0.
  - After setting bits DACE<sub>i</sub> and DAMD<sub>i</sub>, set the DACEN bit of the PER1 register to 0 (DAC stop).
  - When the DACEN bit is set to 0, all the registers in the DAC are cleared, so the settings of the SFRs are required to start the operation again.
- (4) When D/A conversion operation is enabled, do not perform A/D conversions from the analog input pins multiplexed with the ANO0 and ANO1 pins.
- (5) In the real-time output mode, set the value of the DACS<sub>i</sub> register before a timer trigger is generated. Do not change the set value of the DACS<sub>i</sub> register while the trigger signal is output.

**CHAPTER 11 SERIAL ARRAY UNIT**

Serial array unit 0 has a serial channel that can achieve 3-wire serial (CSI) and UART communication. Function assignment of each channel supported by the R7F0C010 is as shown below.

Unit	Channel	Used as CSI	Used as UART
0	0	CSI00 (supporting slave select input function)	UART0
	1	–	

<R> When UART0 is used, CSI00 cannot be used.

## 11.1 Functions of Serial Array Unit

Each serial interface supported by the R7F0C010 has the following features.

### 11.1.1 3-wire serial I/O (CSI00)

Data is transmitted or received in synchronization with the serial clock (SCK) output from the master channel.

3-wire serial communication is clocked communication performed by using three communication lines: one for the serial clock (SCK), one for transmitting serial data (SO), one for receiving serial data (SI).

For details about the settings, see **11.5 Operation of 3-Wire Serial I/O (CSI00) Communication**.

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable
- Level setting of transmit/receive data

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate <sup>Note</sup>

<R>

During master communication: Max.  $f_{MCK}/2$

During slave communication: Max.  $f_{MCK}/6$

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

CSI00 (channel 0) supports the slave select function.

[Slave select function]

- Slave select input (SSI00 pin is used)

<R>

**Note** Use the clocks within a range satisfying the SCK cycle time ( $t_{KCY}$ ) characteristics (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

### 11.1.2 UART (UART0)

This is a start-stop synchronization function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel).

#### [Data transmission/reception]

- Data length of 7, 8, or 9 bits <sup>Note</sup>
- Select the MSB/LSB first
- Level setting of transmit/receive data and select of reverse
- Parity bit appending and parity check functions
- Stop bit appending

#### [Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

#### [Error detection flag]

- Framing error, parity error, or overrun error

In addition, UART0 supports the SNOOZE mode. When RxD input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

<R>



## 11.2 Configuration of Serial Array Unit

The serial array unit includes the following hardware.

**Table 11-1. Configuration of Serial Array Unit**

Item	Configuration
Shift register	9 bits
Buffer register	Lower 9 bits of serial data register mn (SDRmn) <sup>Note</sup>
Serial clock I/O	SCK00 pin (for 3-wire serial I/O)
Serial data input	SI00 pin (for 3-wire serial I/O), RxD0 pin (for UART)
Serial data output	SO00 pin (for 3-wire serial I/O), TxD0 pin (for UART), output controller
Slave select input	SSI00 pin (for slave select input function)
Control registers	<Registers of unit setting block> <ul style="list-style-type: none"> <li>• Peripheral enable register 0 (PER0)</li> <li>• Serial clock select register m (SPSm)</li> <li>• Serial channel enable status register m (SEm)</li> <li>• Serial channel start register m (SSm)</li> <li>• Serial channel stop register m (STm)</li> <li>• Serial output enable register m (SOEm)</li> <li>• Serial output register m (SOM)</li> <li>• Serial output level register m (SOLm)</li> <li>• Serial standby control register m (SSCm)</li> <li>• Input switch control register (ISC)</li> <li>• Noise filter enable register 0 (NFEN0)</li> </ul>
	<Registers of each channel> <ul style="list-style-type: none"> <li>• Serial data register mn (SDRmn)</li> <li>• Serial mode register mn (SMRmn)</li> <li>• Serial communication operation setting register mn (SCRmn)</li> <li>• Serial status register mn (SSRmn)</li> <li>• Serial flag clear trigger register mn (SIRmn)</li> </ul>
	<ul style="list-style-type: none"> <li>• Port mode register 3 (PM3)</li> <li>• Port register 3 (P3)</li> </ul>

**Note** The lower 8 bits of serial data register mn (SDRmn) can be read or written as the following SFR, depending on the communication mode.

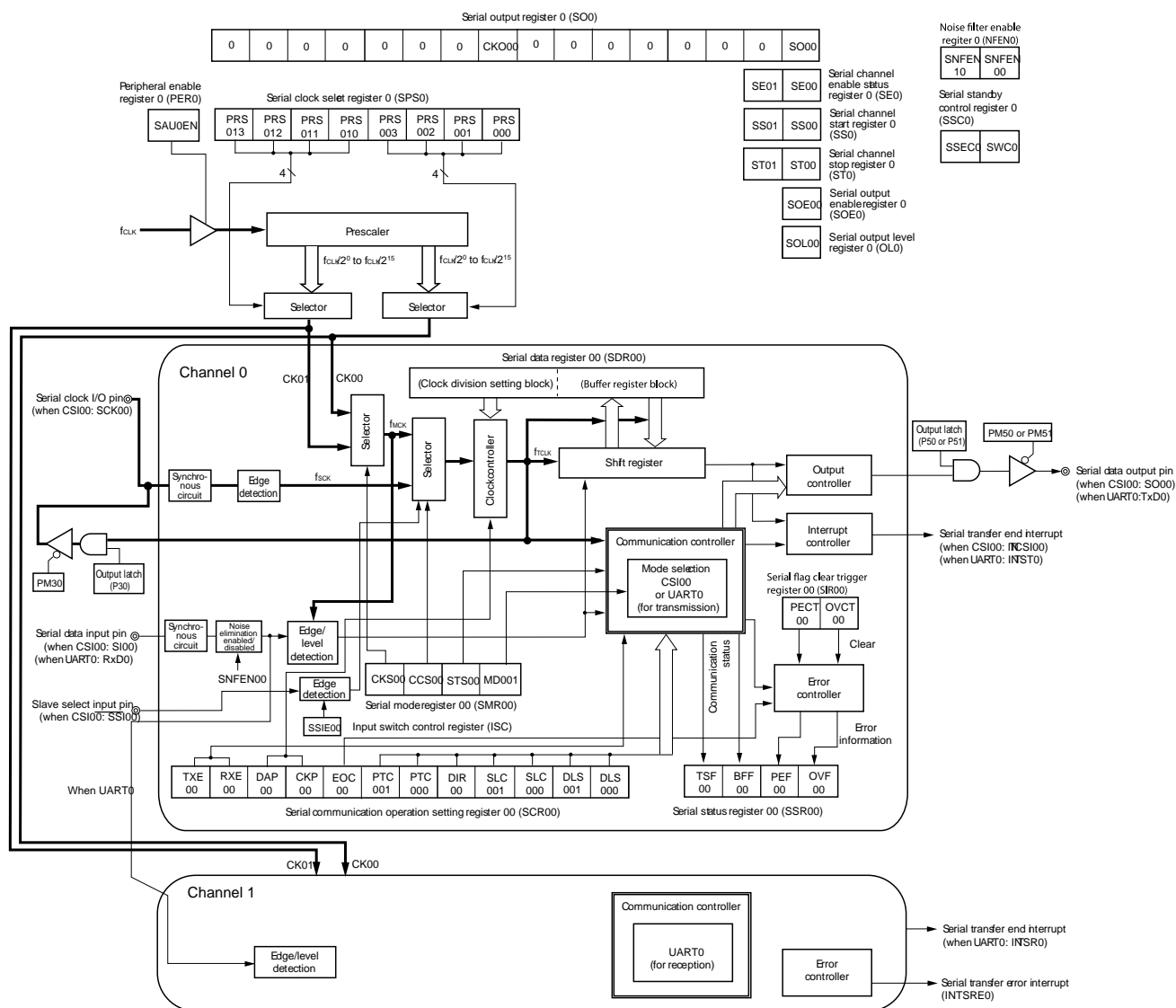
- CSIp communication ... SIOp (CSIp data register)
- UARTq reception ... RXDq (UARTq receive data register)
- UARTq transmission ... TXDq (UARTq transmit data register)

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00), q: UART number (q = 0)

Figure 11-1 shows the block diagram of the serial array unit 0.

Figure 11-1. Block Diagram of Serial Array Unit 0

<R>



### 11.2.1 Shift register

This is a 9-bit register that converts parallel data into serial data or vice versa.

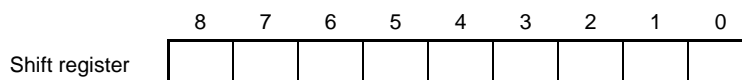
In case of the UART communication of nine bits of data, nine bits (bits 0 to 8) are used<sup>Note 1</sup>.

During reception, it converts data input to the serial pin into parallel data.

When data is transmitted, the value set to this register is output as serial data from the serial output pin.

The shift register cannot be directly manipulated by program.

To read or write the shift register, use the lower 8/9 bits of serial data register mn (SDRmn).



### 11.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)

The SDRmn register is the transmit/receive data register (16 bits) of channel n. Bits 8 to 0 (lower 9 bits) or bits 7 to 0 (lower 8 bits) function as a transmit/receive buffer register, and bits 15 to 9 are used as a register that sets the division ratio of the operation clock ( $f_{MCK}$ ).

When data is received, parallel data converted by the shift register is stored in the lower 8/9 bits. When data is to be transmitted, set transmit to be transferred to the shift register to the lower 8/9 bits.

The data stored in the lower 8/9 bits of this register is as follows, depending on the setting of bits 0 and 1 (DLSmn0, DLSmn1) of serial communication operation setting register mn (SCRmn), regardless of the output sequence of the data.

- 7-bit data length (stored in bits 0 to 6 of SDRmn register)
- 8-bit data length (stored in bits 0 to 7 of SDRmn register)
- 9-bit data length (stored in bits 0 to 8 of SDRmn register)

The SDRmn register can be read or written in 16-bit units.

The lower 8/9 bits of the SDRmn register can be read or written<sup>Note</sup> as the following SFR, depending on the communication mode.

- CSIp communication ... SIOp (CSIp data register)
- UARTq reception ... RXDq (UARTq receive data register)
- UARTq transmission ... TXDq (UARTq transmit data register)

Reset signal generation clears the SDRmn register to 0000H.

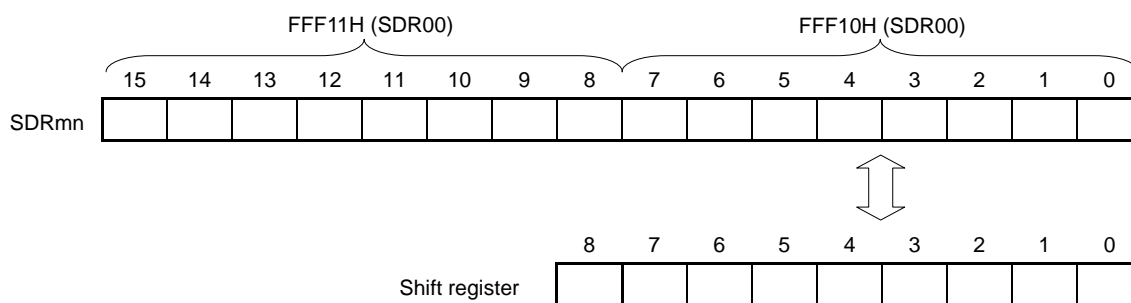
**Note** Writing in 8-bit units is prohibited when the operation is stopped (SEmn = 0).

**Remarks 1.** After data is received, "0" is stored in bits 0 to 8 in bit portions that exceed the data length.

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00), q: UART number (q = 0)

**Figure 11-2. Format of Serial Data Register mn (SDRmn) (mn = 00)**

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01)



**Remark** For the function of the higher 7 bits of the SDRmn register, see **11.3 Registers Controlling Serial Array Unit**.

### 11.3 Registers Controlling Serial Array Unit

Serial array unit is controlled by the following registers.

- Peripheral enable register 0 (PER0)
- Serial clock select register m (SPSm)
- Serial mode register mn (SMRmn)
- Serial communication operation setting register mn (SCRmn)
- Serial data register mn (SDRmn)
- Serial flag clear trigger register mn (SIRmn)
- Serial status register mn (SSRmn)
- Serial channel start register m (SSm)
- Serial channel stop register m (STm)
- Serial channel enable status register m (SEm)
- Serial output enable register m (SOEm)
- Serial output level register m (SOLm)
- Serial output register m (SOM)
- Serial standby control register m (SSCm)
- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0)
- Port mode register 3 (PM3)
- Port register 3 (P3)

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1)

### 11.3.1 Peripheral enable register 0 (PER0)

PER0 is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial array unit 0 is used, be sure to set bit 2 (SAU0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the PER0 register to 00H.

**Figure 11-3. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

SAUmEN	Control of serial array unit m input clock supply
0	Stops supply of input clock. <ul style="list-style-type: none"> <li>• SFR used by serial array unit m cannot be written.</li> <li>• Serial array unit m is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by serial array unit m can be read/written.</li> </ul>

<R>

- Cautions 1.** When setting serial array unit m, be sure to first set the following registers with the SAUmEN bit set to 1. If SAUmEN = 0, writing to a control register of serial array unit m is ignored, and, even if the register is read, only the default value is read (except for the input switch control register (ISC), noise filter enable register 0 (NFEN0), port mode register 3 (PM3), and port register 3 (P3)).
- Serial clock select register m (SPSm)
  - Serial mode register mn (SMRmn)
  - Serial communication operation setting register mn (SCRmn)
  - Serial data register mn (SDRmn)
  - Serial flag clear trigger register mn (SIRmn)
  - Serial status register mn (SSRmn)
  - Serial channel start register m (SSm)
  - Serial channel stop register m (STm)
  - Serial channel enable status register m (SEm)
  - Serial output enable register m (SOEm)
  - Serial output level register m (SOLm)
  - Serial output register m (SOM)
  - Serial standby control register m (SSCm)
- 2.** Be sure to clear bits 7, 3, and 1 to "0".

### 11.3.2 Serial clock select register m (SPSm)

The SPSm register is a 16-bit register that is used to select two types of operation clocks (CKm0, CKm1) that are commonly supplied to each channel. CKm1 is selected by bits 7 to 4 of the SPSm register, and CKm0 is selected by bits 3 to 0.

Rewriting the SPSm register is prohibited when the register is in operation (when SEMn = 1).

The SPSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SPSm register can be set with an 8-bit memory manipulation instruction with SPSmL.

Reset signal generation clears the SPSm register to 0000H.

**Figure 11-4. Format of Serial Clock Select Register m (SPSm)**

Address: F0126H, F0127H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SPSm	0	0	0	0	0	0	0	0	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00

PRS mk3	PRS mk2	PRS mk1	PRS mk0	Section of operation clock (CKmk0) <sup>Note 1</sup>					
					f <sub>CLK</sub> = 2 MHz	f <sub>CLK</sub> = 5 MHz	f <sub>CLK</sub> = 10 MHz	f <sub>CLK</sub> = 20 MHz	f <sub>CLK</sub> = 32 MHz
0	0	0	0	f <sub>CLK</sub>	2 MHz	5 MHz	10 MHz	20 MHz	32 MHz
0	0	0	1	f <sub>CLK</sub> /2	1 MHz	2.5 MHz	5 MHz	10 MHz	16 MHz
0	0	1	0	f <sub>CLK</sub> /2 <sup>2</sup>	500 kHz	1.25 MHz	2.5 MHz	5 MHz	8 MHz
0	0	1	1	f <sub>CLK</sub> /2 <sup>3</sup>	250 kHz	625 kHz	1.25 MHz	2.5 MHz	4 MHz
0	1	0	0	f <sub>CLK</sub> /2 <sup>4</sup>	125 kHz	313 kHz	625 kHz	1.25 MHz	2 MHz
0	1	0	1	f <sub>CLK</sub> /2 <sup>5</sup>	62.5 kHz	156 kHz	313 kHz	625 kHz	1 MHz
0	1	1	0	f <sub>CLK</sub> /2 <sup>6</sup>	31.3 kHz	78.1 kHz	156 kHz	313 kHz	500 kHz
0	1	1	1	f <sub>CLK</sub> /2 <sup>7</sup>	15.6 kHz	39.1 kHz	78.1 kHz	156 kHz	250 kHz
1	0	0	0	f <sub>CLK</sub> /2 <sup>8</sup>	7.81 kHz	19.5 kHz	39.1 kHz	78.1 kHz	125 kHz
1	0	0	1	f <sub>CLK</sub> /2 <sup>9</sup>	3.91 kHz	9.77 kHz	19.5 kHz	39.1 kHz	62.5 kHz
1	0	1	0	f <sub>CLK</sub> /2 <sup>10</sup>	1.95 kHz	4.88 kHz	9.77 kHz	19.5 kHz	31.3 kHz
1	0	1	1	f <sub>CLK</sub> /2 <sup>11</sup>	977 Hz	2.44 kHz	4.88 kHz	9.77 kHz	15.6 kHz
1	1	0	0	f <sub>CLK</sub> /2 <sup>12</sup>	488 Hz	1.22 kHz	2.44 kHz	4.88 kHz	7.81 kHz
1	1	0	1	f <sub>CLK</sub> /2 <sup>13</sup>	244 Hz	610 Hz	1.22 kHz	2.44 kHz	3.91 kHz
1	1	1	0	f <sub>CLK</sub> /2 <sup>14</sup>	122 Hz	305 Hz	610 Hz	1.22 kHz	1.95 kHz
1	1	1	1	f <sub>CLK</sub> /2 <sup>15</sup>	61 Hz	153 kHz	305 Hz	610 Hz	977 Hz

**Note** When changing the clock selected for f<sub>CLK</sub> (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array units (SAUs).

**Caution** Be sure to clear bits 15 to 8 to "0".

- Remarks**
1. f<sub>CLK</sub>: CPU/peripheral hardware clock frequency
  2. m: Unit number (m = 0)
  3. k = 0, 1

### 11.3.3 Serial mode register mn (SMRmn)

The SMRmn register is a register that sets an operation mode of channel n. It is also used to select an operation clock ( $f_{MCK}$ ), specify whether the serial clock ( $f_{SCK}$ ) may be input or not, set a start trigger, an operation mode (CSI or UART), and an interrupt source. This register is also used to invert the level of the receive data only in the UART mode.

Rewriting the SMRmn register is prohibited when the register is in operation (when  $SE_{mn} = 1$ ). However, the MDmn0 bit can be rewritten during operation.

The SMRmn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets the SMRmn register to 0020H.

**Figure 11-5. Format of Serial Mode Register mn (SMRmn) (1/2)**

Address: F0110H, F0111H (SMR00), F0112H, F0113H (SMR01) After reset: 0020H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STS mn <sup>Note</sup>	0	SIS mn0 <sup>Note</sup> e	1	0	0	0	MD mn1	MD mn0

CKS mn	Selection of operation clock ( $f_{MCK}$ ) of channel n
0	Operation clock CKm0 set by the SPSm register
1	Operation clock CKm1 set by the SPSm register
Operation clock ( $f_{MCK}$ ) is used by the edge detector. In addition, depending on the setting of the CCSmn bit and the higher 7 bits of the SDRmn register, a transfer clock ( $f_{TCLK}$ ) is generated.	

CCS mn	Selection of transfer clock ( $f_{TCLK}$ ) of channel n
0	Divided operation clock $f_{MCK}$ specified by the CKSmn bit
1	Clock input $f_{SCK}$ from the SCKp pin (slave transfer in CSI mode)
Transfer clock $f_{TCLK}$ is used for the shift register, communication controller, output controller, interrupt controller, and error controller. When $CCS_{mn} = 0$ , the division ratio of operation clock ( $f_{MCK}$ ) is set by the higher 7 bits of the SDRmn register.	

STS mn	Selection of start trigger source
0	Only software trigger is valid (selected for CSI and UART transmission).
1	Valid edge of the RxDq pin (selected for UART reception)
Transfer is started when the above source is satisfied after 1 is set to the SSm register.	

**Note** The SMR01 register only.

**Caution** Be sure to clear bits 13 to 9, 7, and 4 to 2 (or bits 13 to 6, and 4 to 2 for the SMR00 register) to “0”. Be sure to set bit 5 to “1”.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00), q: UART number (q = 0)



**Figure 11-5. Format of Serial Mode Register mn (SMRmn) (2/2)**

Address: F0110H, F0111H (SMR00), F0112H, F0113H (SMR01) After reset: 0020H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMRmn	CKS mn	CCS mn	0	0	0	0	0	STS mn <sup>Note</sup>	0	SIS mn0 <sup>Note</sup> e	1	0	0	0	MD mn1	MD mn0

SIS mn0	Controls inversion of level of receive data of channel n in UART mode
0	Falling edge is detected as the start bit. The input communication data is captured as is.
1	Rising edge is detected as the start bit. The input communication data is inverted and captured.

MD mn1	Setting of operation mode of channel n
0	CSI mode
1	UART mode

MD mn0	Selection of interrupt source of channel n
0	Transfer end interrupt
1	Buffer empty interrupt (Occurs when data is transferred from the SDRmn register to the shift register.)
For successive transmission, the next transmit data is written by setting the MDmn0 bit to 1 when SDRmn data has run out.	

**Note** The SMR01 register only.

**Caution** Be sure to clear bits 13 to 9, 7, and 4 to 2 (or bits 13 to 6, and 4 to 2 for the SMR00 register) to “0”. Be sure to set bit 5 to “1”.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00), q: UART number (q = 0)

#### 11.3.4 Serial communication operation setting register mn (SCRmn)

The SCRmn register is a communication operation setting register of channel n. It is used to set a data transmission/reception mode, phase of data and clock, whether an error signal is to be masked or not, parity bit, start bit, stop bit, and data length.

Rewriting the SCRmn register is prohibited when the register is in operation (when SEMn = 1).

The SCRmn register can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets the SCRmn register to 0087H.

**Figure 11-6. Format of Serial Communication Operation Setting Register mn (SCRmn) (1/2)**

Address: F0118H, F0119H (SCR00), F011AH, F011BH (SCR01) After reset: 0087H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLCm n1 <sup>Note 1</sup>	SLC mn0	0	1	DLSm n1	DLS mn0

TXE mn	RXE mn	Setting of operation mode of channel n
0	0	Disable communication.
0	1	Reception only
1	0	Transmission only
1	1	Transmission/reception

DAP mn	CKP mn	Selection of data and clock phase in CSI mode	Type
0	0		1
0	1		2
1	0		3
1	1		4

Be sure to set DAPmn, CKPmn = 0, 0 in the UART mode.

<R>

EOC mn	Mask control of error interrupt signal (INTSREx (x = 0))
0	Disables generation of error interrupt INTSREx (INTSRx is generated).
1	Enables generation of error interrupt INTSREx (INTSRx is not generated if an error occurs).
Set EOCmn = 0 in the CSI mode and during UART transmission <sup>Note 2</sup> .	

**Notes 1.** The SCR00 register only.

**2.** When using CSImn not with EOCmn = 0, error interrupt INTSREn may be generated.

**Caution** Be sure to clear bits 3, 6, and 11 to “0” (Also clear bit 5 of the SCR01 register to 0). Be sure to set bit 2 to “1”.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00)

**Figure 11-6. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/2)**

Address: F0118H, F0119H (SCR00), F011AH, F011BH (SCR01) After reset: 0087H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCRmn	TXE mn	RXE mn	DAP mn	CKP mn	0	EOC mn	PTC mn1	PTC mn0	DIR mn	0	SLCm n1 <sup>Note 1</sup>	SLC mn0	0	1	DLSm n1	DLS mn0

PTC mn1	PTC mn0	Setting of parity bit in UART mode			
		Transmission		Reception	
0	0	Does not output the parity bit.		Receives without parity	
0	1	Outputs 0 parity <sup>Note 2</sup> .		No parity judgment	
1	0	Outputs even parity.		Judged as even parity.	
1	1	Outputs odd parity.		Judges as odd parity.	
Be sure to set PTCmn1, PTCmn0 = 0, 0 in the CSI mode.					

DIR mn	Selection of data transfer sequence in CSI and UART modes			
	0	Inputs/outputs data with MSB first.		
	1	Inputs/outputs data with LSB first.		

SLCm n1 <sup>Note 1</sup>	SLC mn0	Setting of stop bit in UART mode			
		0	0	No stop bit	
		0	1	Stop bit length = 1 bit	
		1	0	Stop bit length = 2 bits (mn = 00 only)	
		1	1	Setting prohibited	
When the transfer end interrupt is selected, the interrupt is generated when all stop bits have been completely transferred. Set 1 bit (SLCmn1, SLCmn0 = 0, 1) during UART reception. Set no stop bit (SLCmn1, SLCmn0 = 0, 0) in the CSI mode. Set 1 bit (SLCmn1, SLCmn0 = 0, 1) or 2 bits (SLCmn1, SLCmn0 = 1, 0) during UART transmission.					

&lt;R&gt;

DLSm n1 <sup>Note 2</sup>	DLS mn0	Setting of data length in CSI and UART modes			
		0	1	9-bit data length (stored in bits 0 to 8 of the SDRmn register) (settable in UART mode only)	
		1	0	7-bit data length (stored in bits 0 to 6 of the SDRmn register)	
		1	1	8-bit data length (stored in bits 0 to 7 of the SDRmn register)	
		Other than above		Setting prohibited	

**Notes 1.** The SCR00 register only.**2.** 0 is always added regardless of the data contents.**Caution** Be sure to clear bits 3, 6, and 11 to "0" (Also clear bit 5 of the SCR01 register to 0). Be sure to set bit 2 to "1".**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), p: CSI number (p = 00)

### 11.3.5 Higher 7 bits of the serial data register mn (SDRmn)

The SDRmn register is the transmit/receive data register (16 bits) of channel n.

Bits 8 to 0 (lower 9 bits) of SDR00, SDR01 function as a transmit/receive buffer register, and bits 15 to 9 are used as a register that sets the division ratio of the operation clock ( $f_{MCK}$ ,  $f_{SCK}$ ).

If the CCSmn bit of serial mode register mn (SMRmn) is cleared to 0, the clock set by dividing the operating clock by the higher 7 bits of the SDRmn register is used as the transfer clock.

<R> If the CCSmn bit of serial mode register mn (SMRmn) is set to 1, set bits 15 to 9 (upper 7 bits) of SDR00 and SDR01 to 0000000B. The input clock fSCK (slave transfer in CSI mode) from the SCKp pin is used as the transfer clock.

The lower 8/9 bits of the SDRmn register function as a transmit/receive buffer register. During reception, the parallel data converted by the shift register is stored in the lower 9 bits, and during transmission, the data to be transmitted to the shift register is set to the lower 9 bits.

The SDRmn register can be read or written in 16-bit units.

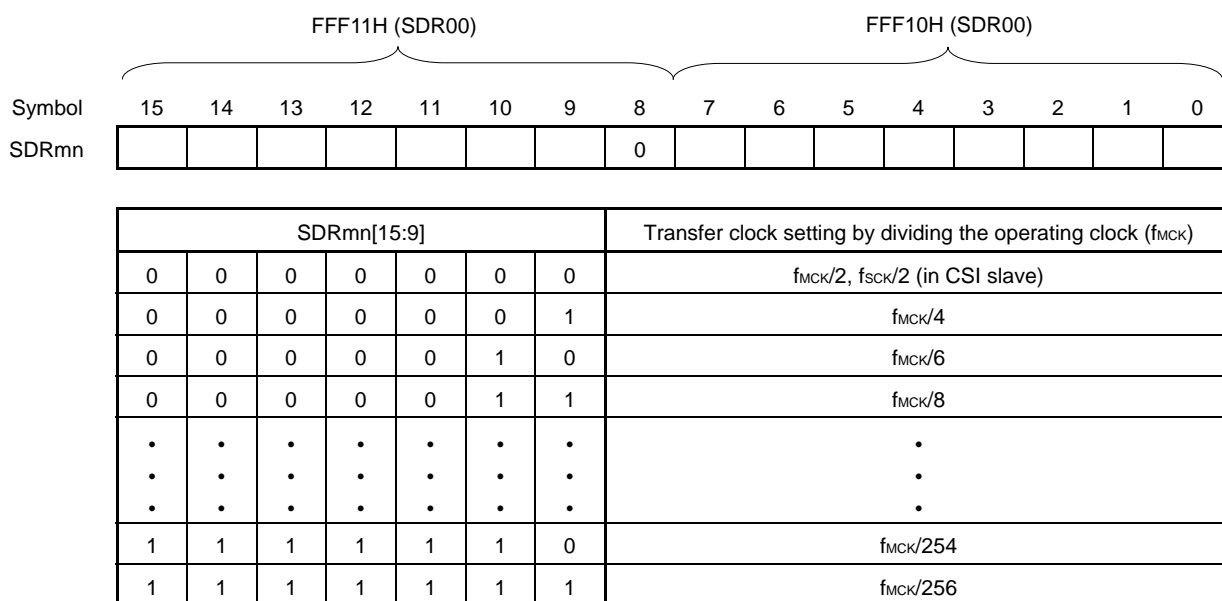
However, the higher 7 bits can be written or read only when the operation is stopped ( $SE_{mn} = 0$ ). During operation ( $SE_{mn} = 1$ ), a value is written only to the lower 9 bits of the SDRmn register. When the SDRmn register is read during operation, 0 is always read.

Reset signal generation clears the SDRmn register to 0000H.

**Figure 11-7. Format of Serial Data Register mn (SDRmn)**

Address: FFF10H, FFF11H (SDR00), FFF12H, FFF13H (SDR01)

After reset: 0000H R/W



**Cautions** 1. Setting SDRmn[15:9] = (0000000B, 0000001B) is prohibited when UART is used.

2. Do not write eight bits to the lower eight bits if operation is stopped ( $SE_{mn} = 0$ ). (If these bits are written to, the higher seven bits are cleared to 0.)

**Remarks** 1. For the function of the lower 9 bits of the SDRmn register, see 11.2 Configuration of Serial Array Unit.

2. m: Unit number (m = 0), n: Channel number (n = 0, 1)

### 11.3.6 Serial flag clear trigger register mn (SIRmn)

The SIRmn register is a trigger register that is used to clear each error flag of channel n.

When each bit (FECTmn, PECTmn, OVCTmn) of this register is set to 1, the corresponding bit (FEFmn, PEFmn, OVFmn) of serial status register mn is cleared to 0. Because the SIRmn register is a trigger register, it is cleared immediately when the corresponding bit of the SSRmn register is cleared.

The SIRmn register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SIRmn register can be set with an 8-bit memory manipulation instruction with SIRmnL.

Reset signal generation clears the SIRmn register to 0000H.

**Figure 11-8. Format of Serial Flag Clear Trigger Register mn (SIRmn)**

Address: F0108H, F0109H (SIR00), F010AH, F010BH (SIR01) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIRmn	0	0	0	0	0	0	0	0	0	0	0	0	0	FECT mn <sup>Note</sup>	PEC Tmn	OVC Tmn

FEC Tmn	Clear trigger of framing error of channel n														
0	Not cleared														
1	Clears the FEFmn bit of the SSRmn register to 0.														

PEC Tmn	Clear trigger of parity error flag of channel n														
0	Not cleared														
1	Clears the PEFmn bit of the SSRmn register to 0.														

OVC Tmn	Clear trigger of overrun error flag of channel n														
0	Not cleared														
1	Clears the OVFmn bit of the SSRmn register to 0.														

**Note** The SIR01 register only.

**Caution** Be sure to clear bits 15 to 3 (or bits 15 to 2 for the SIR00 register) to "0".

- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0, 1)
  2. When the SIRmn register is read, 0000H is always read.

### 11.3.7 Serial status register mn (SSRmn)

The SSRmn register is a register that indicates the communication status and error occurrence status of channel n. The errors indicated by this register are a framing error, parity error, and overrun error.

The SSRmn register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSRmn register can be set with an 8-bit memory manipulation instruction with SSRmnL.

Reset signal generation clears the SSRmn register to 0000H.

**Figure 11-9. Format of Serial Status Register mn (SSRmn) (1/2)**

Address: F0100H, F0101H (SSR00), F0102H, F0103H (SSR01) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEFm n <sup>Note</sup>	PEF mn	OVF mn

TSF mn	Communication status indication flag of channel n
0	Communication is stopped or suspended.
1	Communication is in progress.
<p>&lt;Clear conditions&gt;</p> <ul style="list-style-type: none"> <li>The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is suspended).</li> <li>Communication ends.</li> </ul> <p>&lt;Set condition&gt;</p> <ul style="list-style-type: none"> <li>Communication starts.</li> </ul>	

BFF mn	Buffer register status indication flag of channel n
0	Valid data is not stored in the SDRmn register.
1	Valid data is stored in the SDRmn register.
<p>&lt;Clear conditions&gt;</p> <ul style="list-style-type: none"> <li>Transferring transmit data from the SDRmn register to the shift register ends during transmission.</li> <li>Reading receive data from the SDRmn register ends during reception.</li> <li>The STmn bit of the STm register is set to 1 (communication is stopped) or the SSmn bit of the SSm register is set to 1 (communication is enabled).</li> </ul> <p>&lt;Set conditions&gt;</p> <ul style="list-style-type: none"> <li>Transmit data is written to the SDRmn register while the TXEmn bit of the SCRmn register is set to 1 (transmission or transmission and reception mode in each communication mode).</li> <li>Receive data is stored in the SDRmn register while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission and reception mode in each communication mode).</li> <li>A reception error occurs.</li> </ul>	

**Note** The SSR01 register only.

**Caution** If data is written to the SDRmn register when BFFmn = 1, the transmit/receive data stored in the register is discarded and an overrun error (OVEmn = 1) is detected.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1)

Figure 11-9. Format of Serial Status Register mn (SSRmn) (2/2)

Address: F0100H, F0101H (SSR00), F0102H, F0103H (SSR01) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSRmn	0	0	0	0	0	0	0	0	0	TSF mn	BFF mn	0	0	FEFm n <sup>Note</sup>	PEF mn	OVF mn

FEFm n <sup>Note</sup>	Framing error detection flag of channel n
0	No error occurs.
1	An error occurs (during UART reception).
<Clear condition> <ul style="list-style-type: none"> <li>• 1 is written to the FECTmn bit of the SIRmn register.</li> </ul> <Set condition> <ul style="list-style-type: none"> <li>• A stop bit is not detected when UART reception ends.</li> </ul>	

PEF mn	Parity/ACK error detection flag of channel n
0	No error occurs.
1	Parity error occurs (during UART reception) or ACK is not detected (during I <sup>2</sup> C transmission).
<Clear condition> <ul style="list-style-type: none"> <li>• 1 is written to the PECTmn bit of the SIRmn register.</li> </ul> <Set condition> <ul style="list-style-type: none"> <li>• The parity of the transmit data and the parity bit do not match when UART reception ends (parity error).</li> </ul>	

OVF mn	Overflow error detection flag of channel n
0	No error occurs.
1	An error occurs
<Clear condition> <ul style="list-style-type: none"> <li>• 1 is written to the OVCTmn bit of the SIRmn register.</li> </ul> <Set condition> <ul style="list-style-type: none"> <li>• Even though receive data is stored in the SDRmn register, that data is not read and transmit data or the next receive data is written while the RXEmn bit of the SCRmn register is set to 1 (reception or transmission and reception mode in each communication mode).</li> <li>• Transmit data is not ready for slave transmission or transmission and reception in CSI mode.</li> </ul>	

**Note** The SSR01 register only.

<R> **Caution** When the CSI is performing reception operations in the SNOOZE mode (SWCm = 1), the OVFmn flag will not change.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1)

### 11.3.8 Serial channel start register m (SSm)

The SSm register is a trigger register that is used to enable starting communication/count by each channel.

When 1 is written a bit of this register (SSmn), the corresponding bit (SEmn) of serial channel enable status register m (SEm) is set to 1 (Operation is enabled). Because the SSmn bit is a trigger bit, it is cleared immediately when SEmn = 1.

The SSm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSm register can be set with an 1-bit or 8-bit memory manipulation instruction with SSmL.

Reset signal generation clears the SSm register to 0000H.

**Figure 11-10. Format of Serial Channel Start Register m (SSm)**

Address: F0122H, F0123H (SS0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SS0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS01	SS00

SSmn	Operation start trigger of channel n
0	No trigger operation
1	Sets the SEmn bit to 1 and enters the communication wait status <sup>Note</sup> .

**Note** If set the SSmn = 1 to during a communication operation, will wait status to stop the communication.

At this time, holding status value of control register and shift register, SCKmn and SOMn pins, and FEFmn, PEFmn, OVfmn flags.

**Cautions 1.** Be sure to clear bits 15 to 2 to "0".

**2.** For the UART reception, set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after 4 or more f<sub>MCK</sub> clocks have elapsed.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 1)

**2.** When the SSm register is read, 0000H is always read.



### 11.3.9 Serial channel stop register m (STm)

The STm register is a trigger register that is used to enable stopping communication/count by each channel.

When 1 is written a bit of this register (STmn), the corresponding bit (SEmn) of serial channel enable status register m (SEm) is cleared to 0 (operation is stopped). Because the STmn bit is a trigger bit, it is cleared immediately when SEmn = 0.

The STm register can set written by a 16-bit memory manipulation instruction.

The lower 8 bits of the STm register can be set with a 1-bit or 8-bit memory manipulation instruction with STmL.

Reset signal generation clears the STm register to 0000H.

**Figure 11-11. Format of Serial Channel Stop Register m (STm)**

Address: F0124H, F0125H (ST0) After reset: 0000H W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ST0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ST01	ST00

STm n	Operation stop trigger of channel n														
0	No trigger operation														
1	Clears the SEmn bit to 0 and stops the communication operation <sup>Note</sup> .														

**Note** Holding status value of the control register and shift register, the SCKmn and SOMn pins, and FEFmn, PEFmn, OVFmn flags.

**Caution** Be sure to clear bits 15 to 2 to "0".

- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0, 1)
  2. When the STm register is read, 0000H is always read.

### 11.3.10 Serial channel enable status register m (SEm)

The SEm register indicates whether data transmission/reception operation of each channel is enabled or stopped.

When 1 is written a bit of serial channel start register m (SSm), the corresponding bit of this register is set to 1. When 1 is written a bit of serial channel stop register m (STm), the corresponding bit is cleared to 0.

Channel n that is enabled to operate cannot rewrite by software the value of the CKOmn bit (serial clock output of channel n) of serial output register m (SOM) to be described below, and a value reflected by a communication operation is output from the serial clock pin.

Channel n that stops operation can set the value of the CKOmn bit of the SOM register by software and output its value from the serial clock pin. In this way, any waveform, such as that of a start condition/stop condition, can be created by software.

The SEm register can be read by a 16-bit memory manipulation instruction.

The lower 8 bits of the SEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SEmL.

Reset signal generation clears the SEm register to 0000H.

**Figure 11-12. Format of Serial Channel Enable Status Register m (SEm)**

Address: F0120H, F0121H (SE0) After reset: 0000H R

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SE0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SE01	SE00

SEm n	Indication of operation enable/stop status of channel n														
0	Operation stops														
1	Operation is enabled.														

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1)

### 11.3.11 Serial output enable register m (SOEm)

The SOEm register is a register that is used to enable or stop output of the serial communication operation of each channel.

Channel n that enables serial output cannot rewrite by software the value of the SOMn bit of serial output register m (SOM) to be described below, and a value reflected by a communication operation is output from the serial data output pin.

For channel n, whose serial output is stopped, the SOMn bit value of the SOM register can be set by software, and that value can be output from the serial data output pin. In this way, any waveform of the start condition and stop condition can be created by software.

The SOEm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SOEm register can be set with a 1-bit or 8-bit memory manipulation instruction with SOEmL. Reset signal generation clears the SOEm register to 0000H.

**Figure 11-13. Format of Serial Output Enable Register m (SOEm)**

Address: F012AH, F012BH    After reset: 0000H    R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
SOE0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOE00

SOEmn	Serial output enable/stop of channel n
0	Stops output by serial communication operation.
1	Enables output by serial communication operation.

**Caution** Be sure to clear bits 15 to 1 to "0".

**Remark** m: Unit number (m = 0), n: Channel number (n = 0)

### 11.3.12 Serial output register m (SOM)

The SOM register is a buffer register for serial output of each channel.

The value of the SOMn bit of this register is output from the serial data output pin of channel n.

The value of the CKOMn bit of this register is output from the serial clock output pin of channel n.

The SOMn bit of this register can be rewritten by software only when serial output is disabled (SOEmn = 0). When serial output is enabled (SOEmn = 1), rewriting by software is ignored, and the value of the register can be changed only by a serial communication operation.

The CKOMn bit of this register can be rewritten by software only when the channel operation is stopped (SEmn = 0). While channel operation is enabled (SEmn = 1), rewriting by software is ignored, and the value of the CKOMn bit can be changed only by a serial communication operation.

<R> To use a pin for the serial interface as a port function pin other than a serial interface function pin, set the corresponding the CKOMn and SOMn bits to 1.

The SOM register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears the SOM register to 0303H.

**Figure 11-14. Format of Serial Output Register m (SOM)**

Address: F0128H, F0129H (SO0) After reset: 0303H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SO0	0	0	0	0	0	0	1	CKO 00	0	0	0	0	0	0	1	SO 00

CKO mn	Serial clock output of channel n
0	Serial clock output value is "0".
1	Serial clock output value is "1".

SO mn	Serial data output of channel n
0	Serial data output value is "0".
1	Serial data output value is "1".

<R> **Caution** Be sure to clear bits 15 to 10 and 7 to 2 of the SO0 register to "0"  
Be sure to set bits 9 and 1 of the SO0 register to "1".

**Remark** m: Unit number (m = 0), n: Channel number (n = 0)

**11.3.13 Serial output level register m (SOLm)**

The SOLm register is a register that is used to set inversion of the data output level of each channel.

This register can be set only in the UART mode. Be sure to set 0 for corresponding bit in the CSI mode.

Inverting channel n by using this register is reflected on pin output only when serial output is enabled (SOEmn = 1). When serial output is disabled (SOEmn = 0), the value of the SOMn bit is output as is.

Rewriting the SOLm register is prohibited when the register is in operation (when SEMn = 1).

The SOLm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SOLm register can be set with an 8-bit memory manipulation instruction with SOLmL.

Reset signal generation clears the SOLm register to 0000H.

**Figure 11-15. Format of Serial Output Level Register m (SOLm)**

Address: F0134H, F0135H (SOL0) After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOL0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOL00

SOLmn	Selects inversion of the level of the transmit data of channel n in UART mode
0	Communication data is output as is.
1	Communication data is inverted and output.

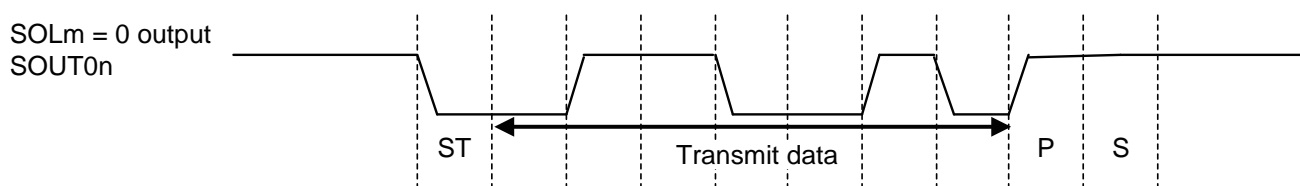
**Caution** Be sure to clear bits 15 to 1 to “0”.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0)

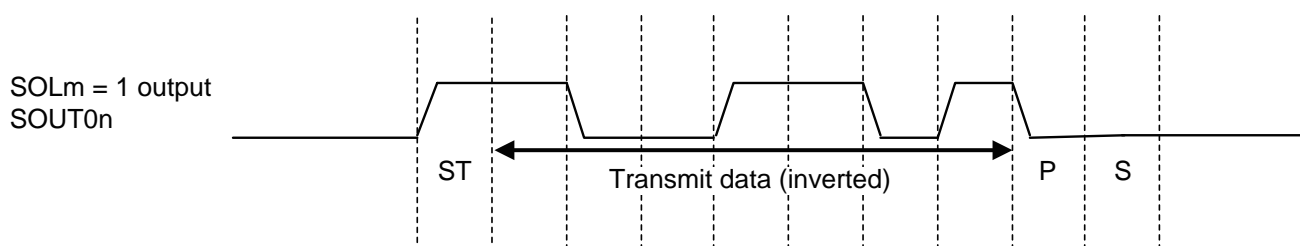
Figure 14 - 16 shows examples in which the level of transmit data is reversed during UART transmission.

**Figure 11-16. Examples of Reverse Transmit Data**

(a) Non-reverse Output (SOLmn = 0)



(b) Reverse Output (SOLmn = 1)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 2)

### 11.3.14 Serial standby control register m (SSCm)

The SSC0 register is used to control the startup of reception (the SNOOZE mode) while in the STOP mode when receiving CSI00 or UART0 serial data.

The SSCm register can be set by a 16-bit memory manipulation instruction.

The lower 8 bits of the SSCm register can be set with an 8-bit memory manipulation instruction with SSCmL.

Reset signal generation clears the SSCm register to 0000H.

**Caution** The maximum transfer rate in the SNOOZE mode is as follows.

<R>

- When using CSI00: Up to 1 Mbps
- When using UART0: 4800 bps only

Figure 11-17. Format of Serial Standby Control Register m (SSCm)

Address: F0138H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SS ECm	SWC m

<R>

SSECm	Selection of whether to enable or disable the generation of communication error interrupts in the SNOOZE mode
0	Enable the generation of error interrupts (INTSRE0)
1	Disable the generation of error interrupts (INTSRE0)
<ul style="list-style-type: none"> <li>• The SSECm bit can be set to 1 or 0 only when both the SWCm and EOCmn bits are set to 1 during UART reception in the SNOOZE mode. In other cases, clear the SSECm bit to 0.</li> <li>• Setting SSECm, SWCm = 1, 0 is prohibited.</li> </ul>	

SWCm	Setting of the SNOOZE mode
0	Do not use the SNOOZE mode function.
1	Use the SNOOZE mode function.
<ul style="list-style-type: none"> <li>• When there is a hardware trigger signal in the STOP mode, the STOP mode is exited, and A/D conversion is performed without operating the CPU (the SNOOZE mode).</li> <li>• The SNOOZE mode function can only be specified when the high-speed on-chip oscillator clock is selected for the CPU/peripheral hardware clock (<math>f_{CLK}</math>). If any other clock is selected, specifying this mode is prohibited.</li> <li>• Even when using SNOOZE mode, be sure to set the SWCm bit to 0 in normal operation mode and change it to 1 just before shifting to STOP mode.</li> </ul> <p>Also, be sure to change the SWCm bit to 0 after returning from STOP mode to normal operation mode.</p>	

**Caution** Setting SSECm, SWCm = 1, 0 is prohibited.

Figure 11-18. Interrupt in UART Reception Operation in SNOOZE Mode

<R>

EOCmn Bit	SSECm Bit	Reception Ended Successfully	Reception Ended in an Error
0	0	INTSRx is generated.	INTSRx is generated.
0	1	INTSRx is generated.	INTSRx is generated.
1	0	INTSRx is generated.	INTSRx is generated.
1	1	INTSRx is generated.	No interrupt is generated.

### 11.3.15 Input switch control register (ISC)

The SSIE00 bit of the ISC register is used to control the  $\overline{\text{SSI00}}$  pin input when CSI00 communication and slave mode are applied. While a high level is being input to the  $\overline{\text{SSI00}}$  pin, no transmission/reception operation is performed even if a serial clock is input, and the SO00 pin outputs high impedance.

While a low level is being input to the  $\overline{\text{SSI00}}$  pin, a transmission/reception operation is performed according to each mode setting if a serial clock is input.

The ISC register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the ISC register to 00H.

**Figure 11-19. Format of Input Switch Control Register (ISC)**

Address: F0073H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
ISC	SSIE00	0	0	0	0	0	0	0

SSIE00	$\overline{\text{SSI00}}$ input setting in CSI communication and slave mode
0	$\overline{\text{SSI00}}$ pin input is invalid.
1	$\overline{\text{SSI00}}$ pin input is valid.

**Caution** Be sure to clear bits 6 to 0 to "0".

### 11.3.16 Noise filter enable register 0 (NFEN0)

The NFEN0 register is used to set whether the noise filter can be used for the input signal from the serial data input pin to each channel.

Disable the noise filter of the pin used for CSI, by clearing the corresponding bit of this register to 0.

Enable the noise filter of the pin used for UART communication, by setting the corresponding bit of this register to 1.

When the noise filter is enabled, CPU/peripheral hardware clock ( $f_{CLK}$ ) is synchronized with 2-clock match detection.

When the noise filter is OFF, only synchronization is performed with the CPU/peripheral hardware clock ( $f_{MCK}$ ).

The NFEN0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears the NFEN0 register to 00H.

**Figure 11-20. Format of Noise Filter Enable Register 0 (NFEN0)**

Address: F0070H	After reset: 00H	R/W						
Symbol	7	6	5	4	3	2	1	0
NFEN0	0	0	0	0	0	0	0	SNFEN00
SNFEN00	Use of noise filter of RxD0 pin (RXD0/TOOLRXD/SDA00/SI00/P11)							
0	Noise filter OFF							
1	Noise filter ON							
Set the SNFEN00 bit to 1 to use the RxD0 pin. Clear the SNFEN00 bit to 0 to use the other than RxD0 pin.								

**Caution** Be sure to clear bits 7 to 1 to "0".



### 11.3.17 Port mode register 3 (PM3)

This register sets input/output of port 3 in 1-bit units.

When using the ports (such as P30/INTP2/TxD0/TOOLTxD/SO0) to be shared with the serial data output pin or serial clock output pin for serial data output or serial clock output, set the port mode register (PMxx) bit corresponding to each port to 0. And set the port register (Pxx) bit corresponding to each port to 1.

Example: When using P30/INTP2/TxD0/TOOLTxD/SO0 for serial data output

Set the PM30 bit of the port mode register 3 to "0".

Set the P30 bit of the port register 3 to "1".

When using the ports (such as P30/INTP2/TxD0/TOOLTxD/SO0) to be shared with the serial data input pin or serial clock input pin for serial data input or serial clock input, set the port mode register (PMxx) bit corresponding to each port to 1. At this time, the port register (Pxx) bit may be 0 or 1.

Example: When using P30/INTP2/TxD0/TOOLTxD/SO0 for serial data input

Set the PM30 bit of port mode register 3 to "1".

Set the P30 bit of port register 3 to 0 or "1".

The PM3 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets the PM3 register to FFH.

Refer to Table 4-11 to see which PMxx registers are provided for each product.

**Figure 11-21. Format of Port Mode Register 3 (PM3)**

Address: FFF23H After reset: FFH R/W

Symbol	7	6	5	4	3	2	1	0
PM3	1	1	PM35 <sup>Note</sup>	PM34 <sup>Note</sup>	PM33	PM32	PM31	PM30

PM3n	P3n pin I/O mode selection (n = 0 to 5)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

**Note** These are not provided in 24-pin products.

**Caution** Be sure to set bits 7 and 6 to "1".

## 11.4 Operation Stop Mode

Each serial interface of serial array unit has the operation stop mode.

In this mode, serial communication cannot be executed, thus reducing the power consumption.

In addition, the pin for serial interface can be used as port function pins in this mode.

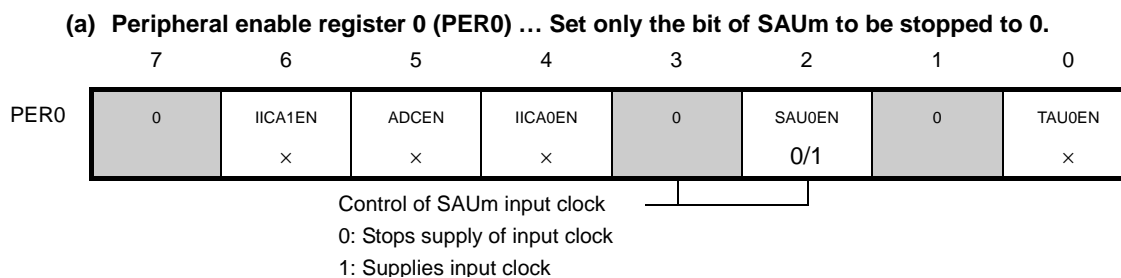
### 11.4.1 Stopping the operation by units

The stopping of the operation by units is set by using peripheral enable register 0 (PER0).

The PER0 register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

To stop the operation of serial array unit 0, set bit 2 (SAU0EN) to 0.

**Figure 11-22. Peripheral Enable Register 0 (PER0) Setting When Stopping the Operation by Units**



**Cautions 1.** If SAUmEN = 0, writing to a control register of serial array unit m is ignored, and, even if the register is read, only the default value is read.

Note that this does not apply to the following registers.

- Input switch control register (ISC)
- Noise filter enable register 0 (NFEN0)
- Port mode register 3 (PM3)
- Port register 3 (P3)

**2.** Be sure to clear bits 7, 3, and 1 to “0”.

**Remark** x: Bits not used with serial array units (depending on the settings of other peripheral functions)

□: Setting disabled (set to the initial value)

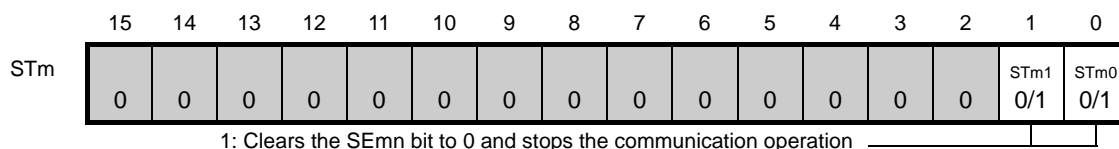
0/1: Set to 0 or 1 depending on the usage of the user

### 11.4.2 Stopping the operation by channels

The stopping of the operation by channels is set using each of the following registers.

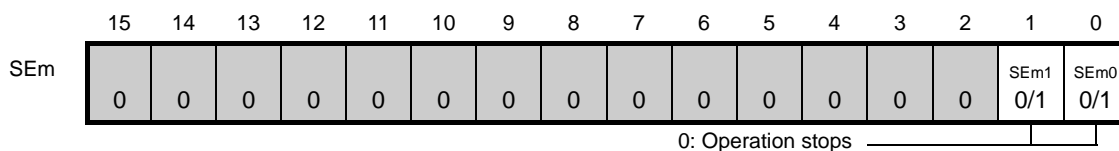
**Figure 11-23. Each Register Setting When Stopping the Operation by Channels**

- (a) **Serial channel stop register m (STm) ... This register is a trigger register that is used to enable stopping communication/count by each channel.**



\* Because the ST<sub>m</sub>n bit is a trigger bit, it is cleared immediately when SE<sub>m</sub>n = 0.

- (b) **Serial Channel Enable Status Register m (SEm) ... This register indicates whether data transmission/reception operation of each channel is enabled or stopped.**



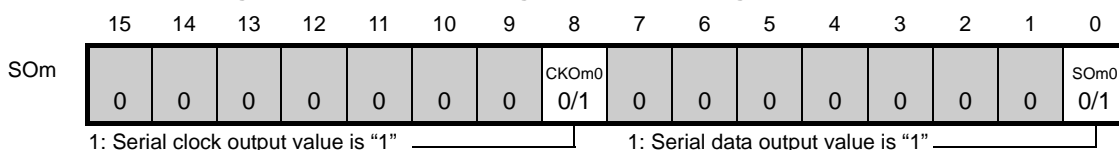
\* The SE<sub>m</sub> register is a read-only status register, whose operation is stopped by using the ST<sub>m</sub> register. With a channel whose operation is stopped, the value of the CKO<sub>m</sub>n bit of the SO<sub>m</sub> register can be set by software.

- (c) **Serial output enable register m (SOEm) ... This register is a register that is used to enable or stop output of the serial communication operation of each channel.**



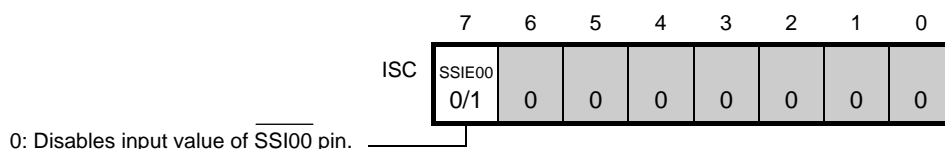
\* For channel n, whose serial output is stopped, the SO<sub>m</sub>n bit value of the SO<sub>m</sub> register can be set by software.

- (d) **Serial output register m (SOm) ... This register is a buffer register for serial output of each channel.**



\* When using pins corresponding to each channel as port function pins, set the corresponding CKO<sub>m</sub>n, SO<sub>m</sub>n bits to "1".

- (e) **Input switch control register (ISC) ... This register controls the  $\overline{\text{SSI00}}$  pin of CSI00 slave channel (channel 0 of unit 0).**



**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0, 1)

**2.** : Setting disabled (set to the initial value), 0/1: Set to 0 or 1 depending on the usage of the user

## 11.5 Operation of 3-Wire Serial I/O (CSI00) Communication

This is a clocked communication function that uses three lines: serial clock (SCK) and serial data (SI and SO) lines.

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable
- Level setting of transmit/receive data

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate <sup>Note</sup>

<R>

During master communication: Max.  $f_{MCK}/2$

During slave communication: Max.  $f_{MCK}/6$

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

In addition, CSI00 supports the SNOOZE mode. When SCK input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

CSI00 (channel 0) supports the slave select function. For details, see **11.6 Clock Synchronous Serial Communication with Slave Select Input Function**.

[Slave select function]

- Slave select input (SSI00 pin is used)

<R>

**Note** Use the clocks within a range satisfying the SCK cycle time ( $t_{KCY}$ ) characteristics (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

The channels supporting 3-wire serial I/O (CSI00) are channel 0 of SAU0.

Unit	Channel	Used as CSI	Used as UART
0	0	CSI00 (supporting slave select input function)	UART0
	1	–	

3-wire serial I/O (CSI00) performs the following seven types of communication operations.

- Master transmission (See 11.5.1.)
- Master reception (See 11.5.2.)
- Master transmission/reception (See 11.5.3.)
- Slave transmission (See 11.5.4.)
- Slave reception (See 11.5.5.)
- Slave transmission/reception (See 11.5.6.)
- SNOOZE mode function (See 11.5.7.)

### 11.5.1 Master transmission

Master transmission is that the R7F0C010 output a transfer clock and transmits data to another device.

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SO00
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	None
Transfer data length	7 or 8 bits
Transfer rate <sup>Note</sup>	Max. $f_{MCK}/2$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$ : System clock frequency
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

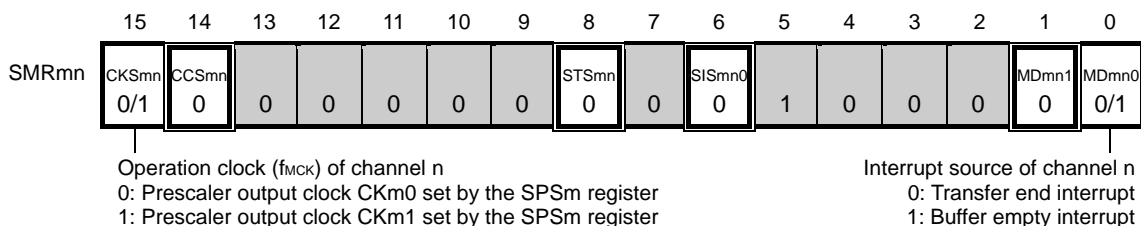
**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

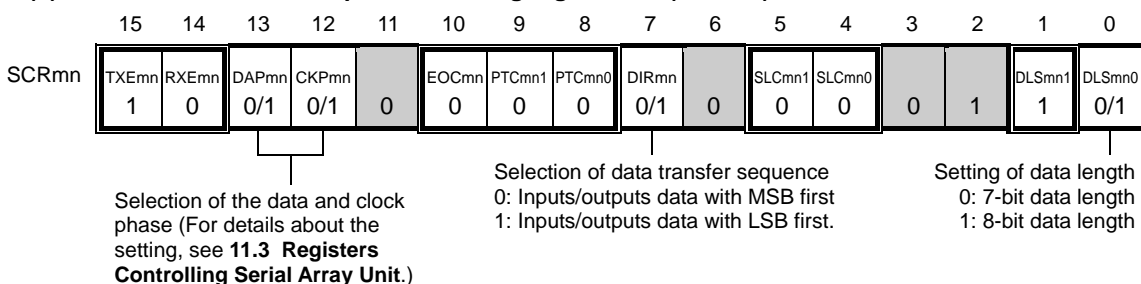
(1) Register setting

Figure 11-24. Example of Contents of Registers for Master Transmission of 3-Wire Serial I/O (CSI00)

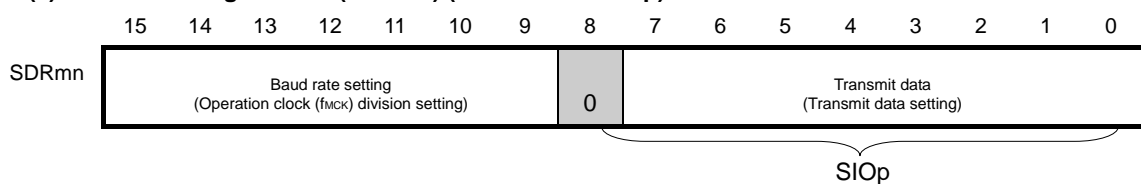
(a) Serial mode register mn (SMRmn)



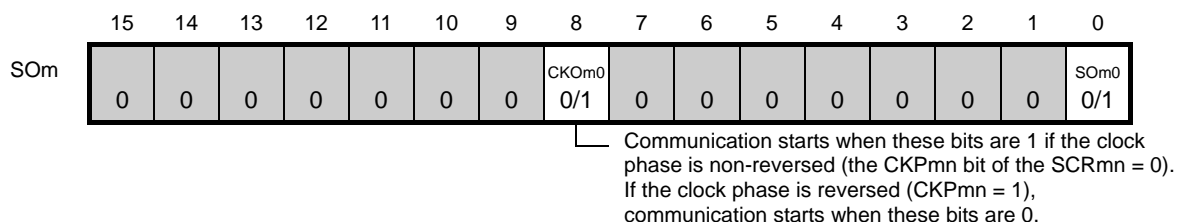
(b) Serial communication operation setting register mn (SCRmn)



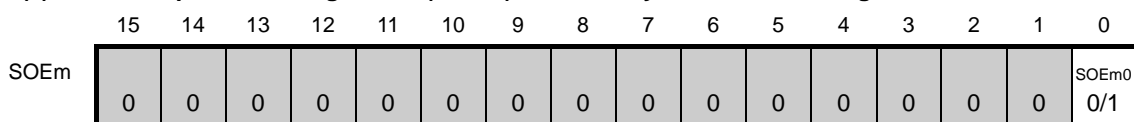
(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



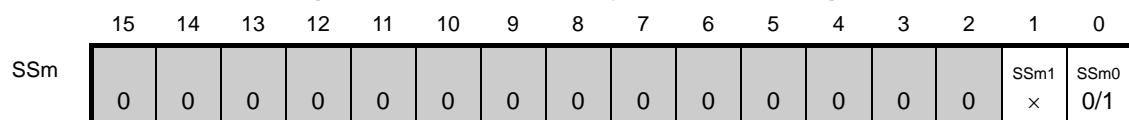
(d) Serial output register m (SOM) ... Sets only the bits of the target channel.



(e) Serial output enable register m (SOEm) ... Sets only the bits of the target channel to 1.



(f) Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.

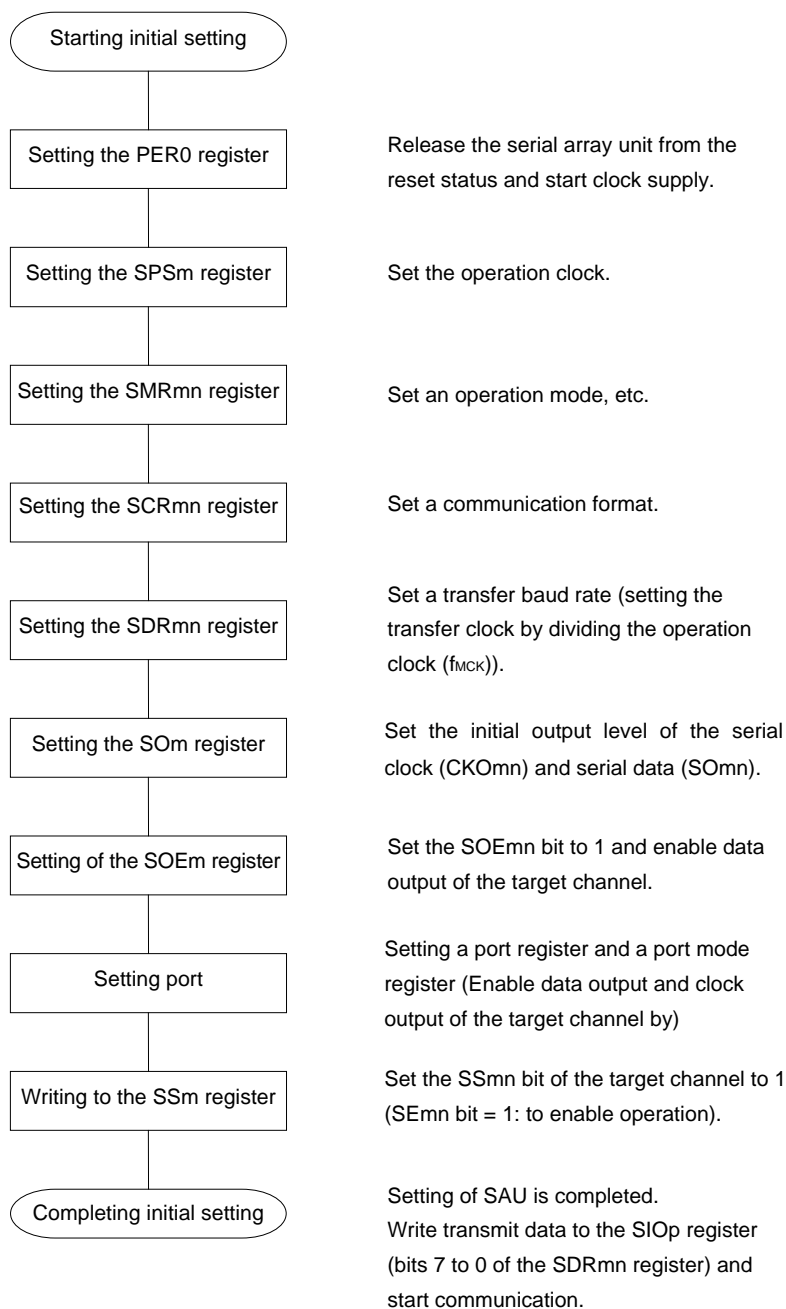


- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00
  - : Setting is fixed in the CSI master transmission mode, ■: Setting disabled (set to the initial value)  
0/1: Set to 0 or 1 depending on the usage of the user



## (2) Operation procedure

Figure 11-25. Initial Setting Procedure for Master Transmission



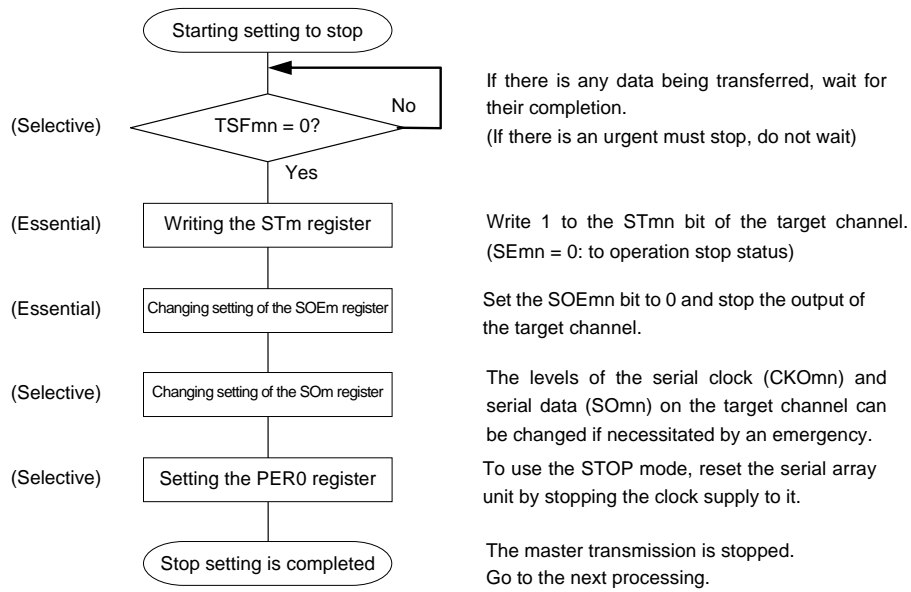
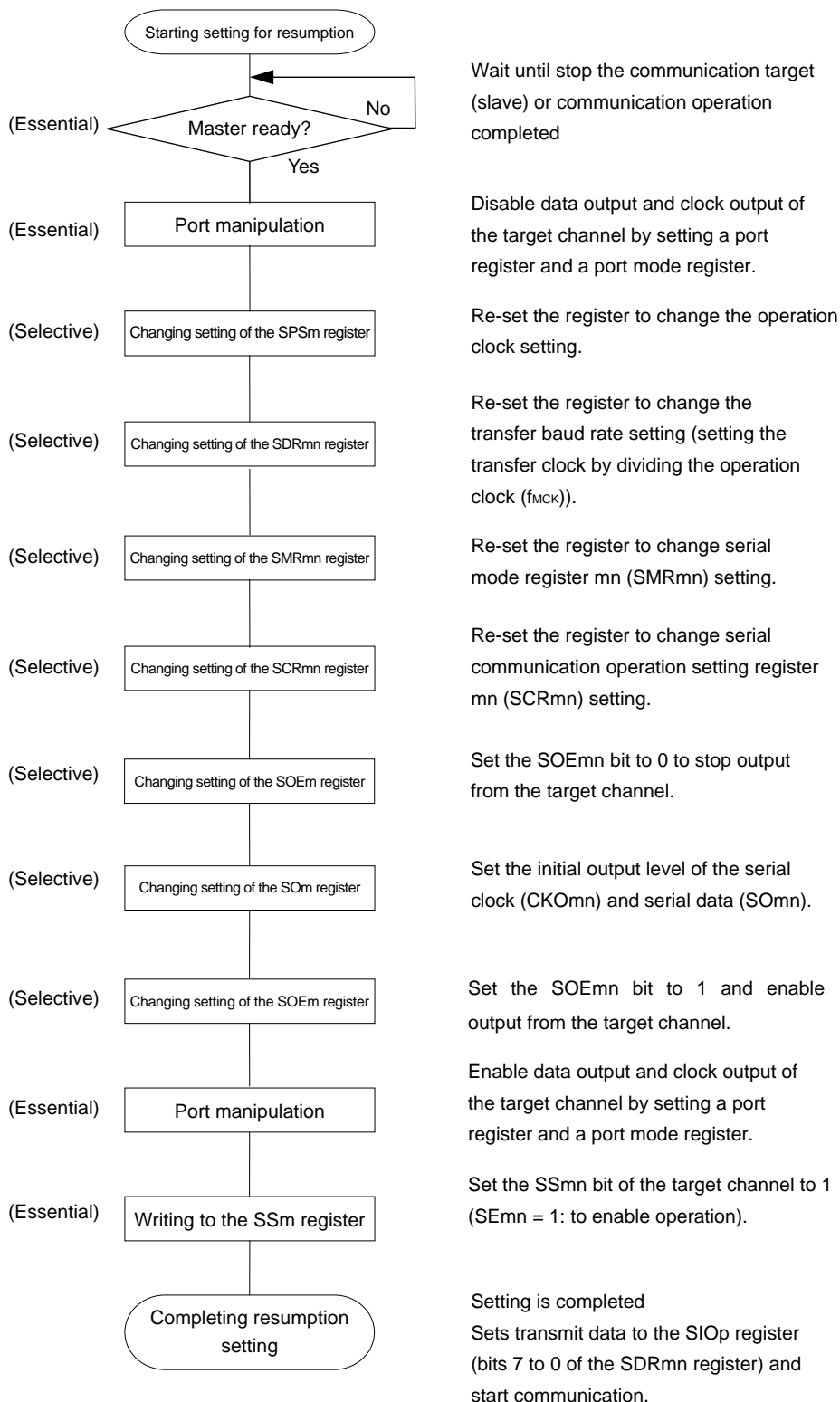
**Figure 11-26. Procedure for Stopping Master Transmission**

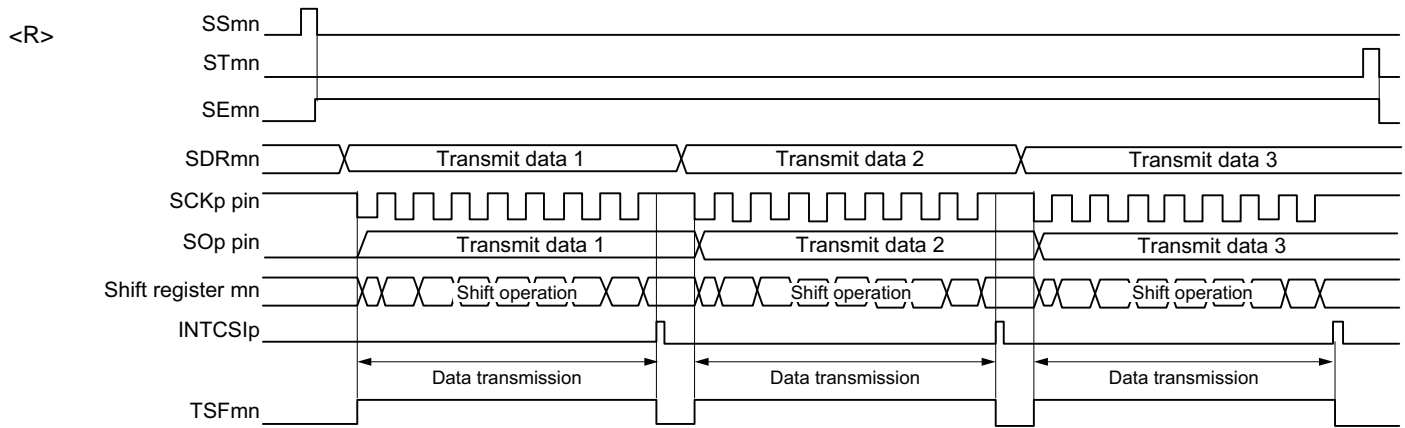
Figure 11-27. Procedure for Resuming Master Transmission



**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

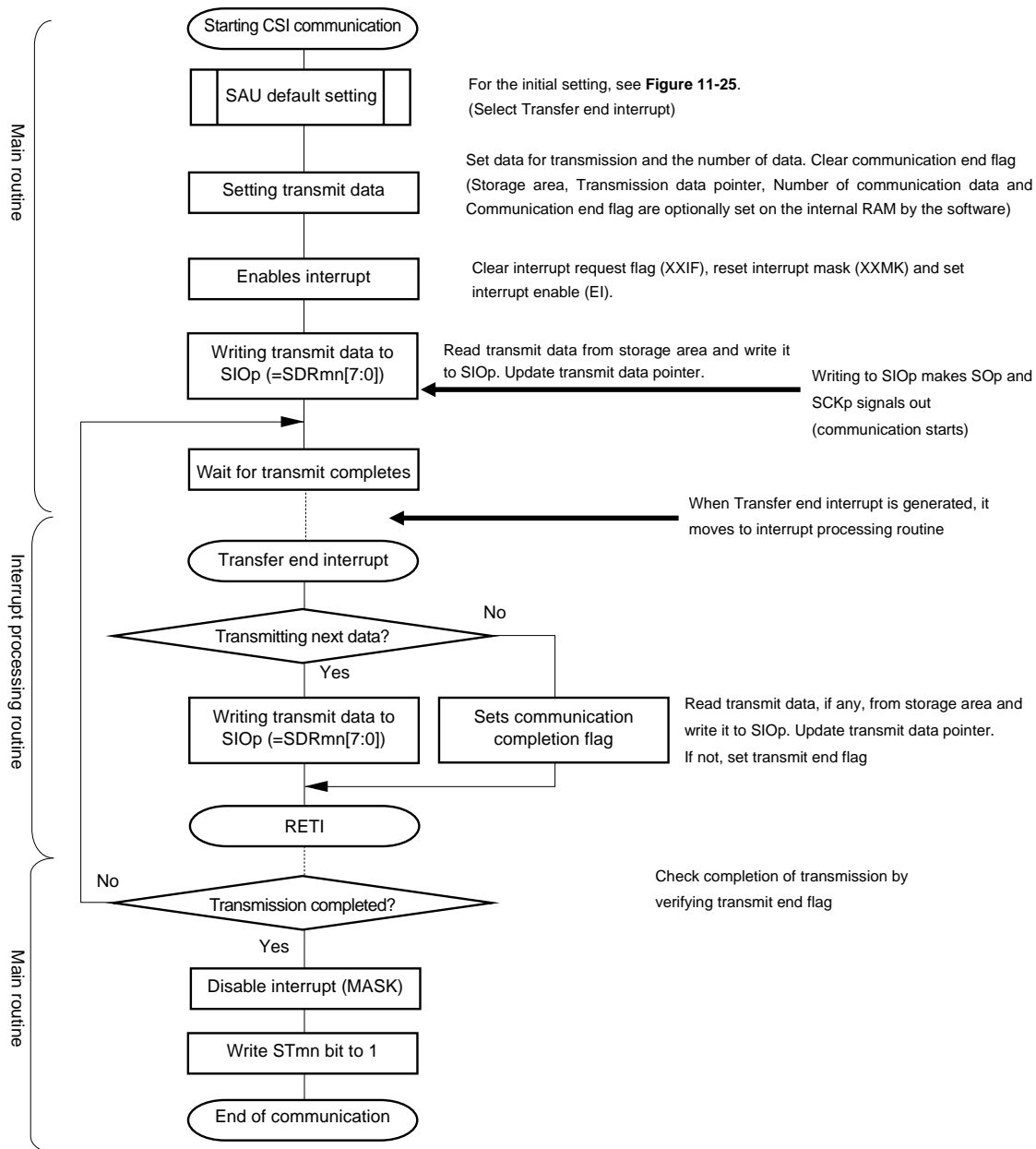
(3) Processing flow (in single-transmission mode)

Figure 11-28. Timing Chart of Master Transmission (in Single-Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

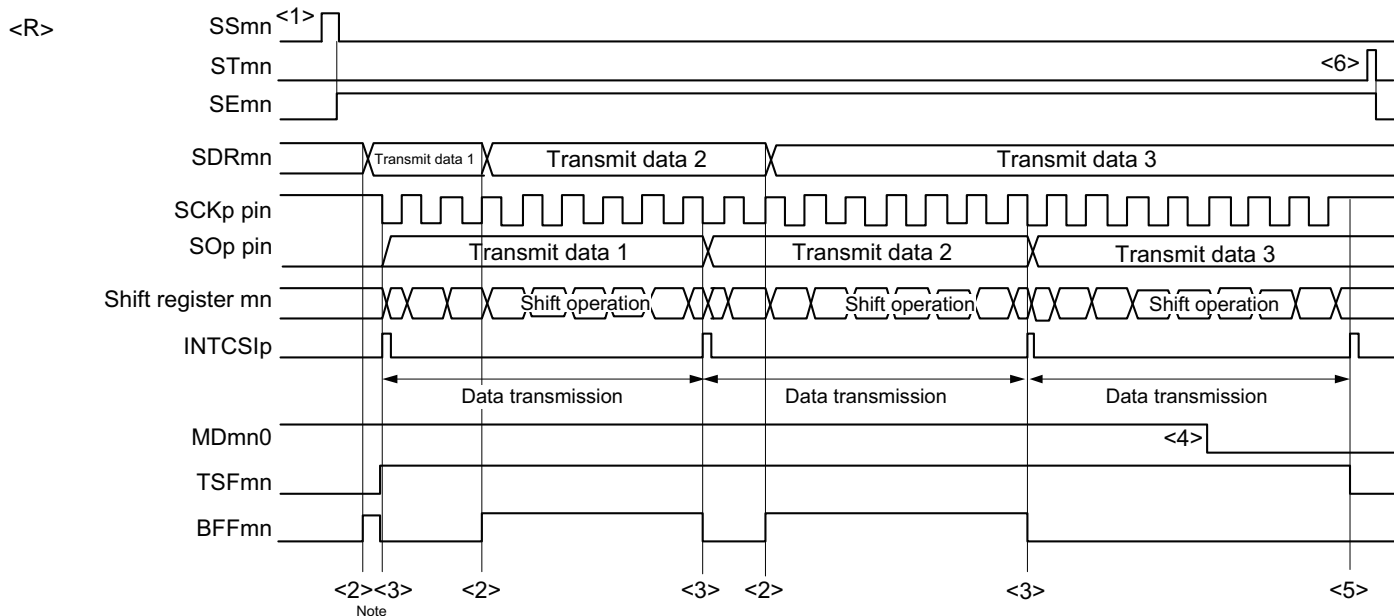
Figure 11-29. Flowchart of Master Transmission (in Single-Transmission Mode)



<R>

(4) Processing flow (in continuous transmission mode)

Figure 11-30. Timing Chart of Master Transmission (in Continuous Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



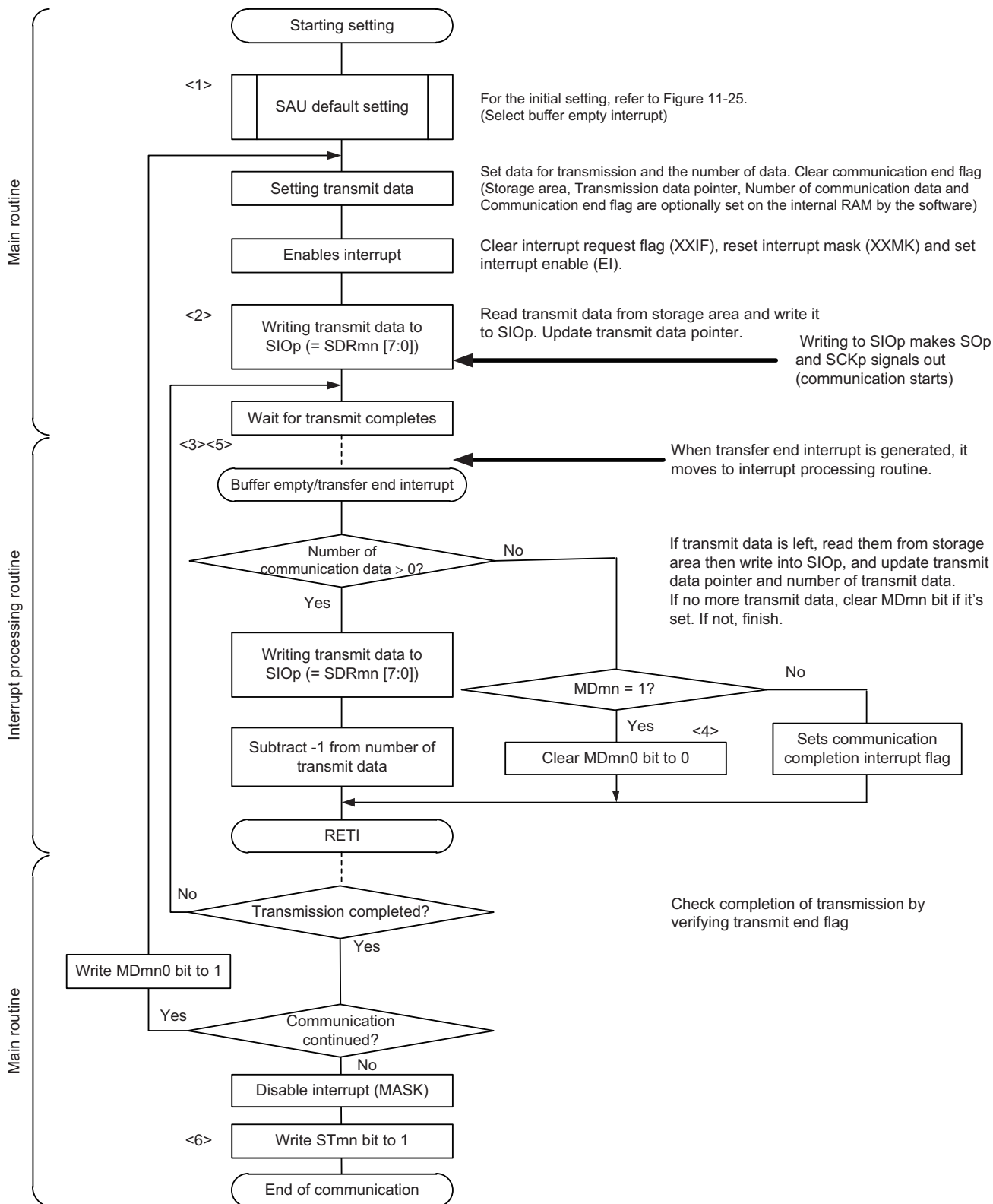
**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it will be rewritten before the transfer end interrupt of the last transmit data.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

Figure 11-31. Flowchart of Master Transmission (in Continuous Transmission Mode)

<R>



**Remark** <1> to <6> in the figure correspond to <1> to <6> in **Figure 11-30 Timing Chart of Master Transmission (in Continuous Transmission Mode)**.

### 11.5.2 Master reception

Master reception is that the R7F0C010 output a transfer clock and receives data from other device.

<R>

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SI00
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overflow error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate <sup>Note</sup>	Max. $f_{MCK}/2$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$ : System clock frequency
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

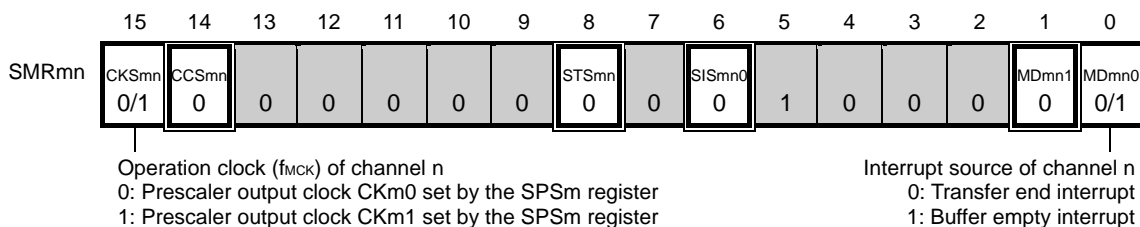
**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00



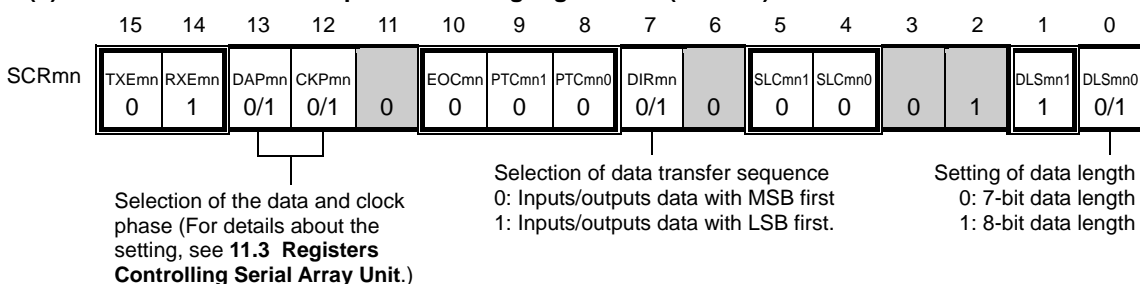
(1) Register setting

Figure 11-32. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00)

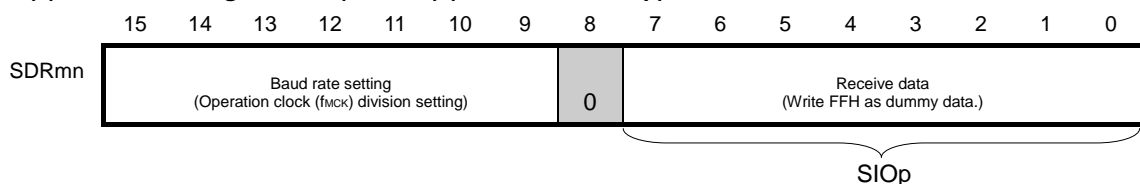
(a) Serial mode register mn (SMRmn)



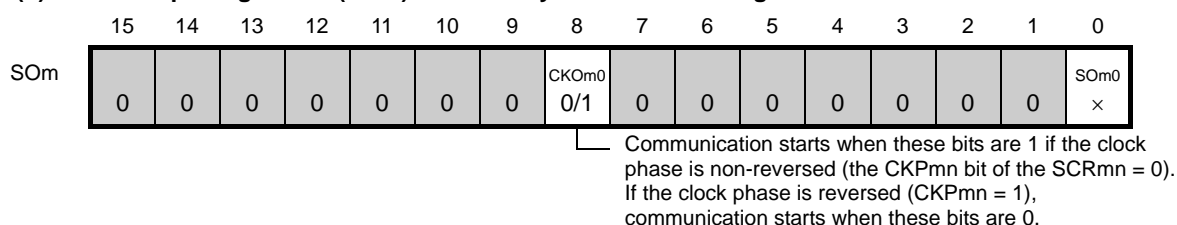
(b) Serial communication operation setting register mn (SCRmn)



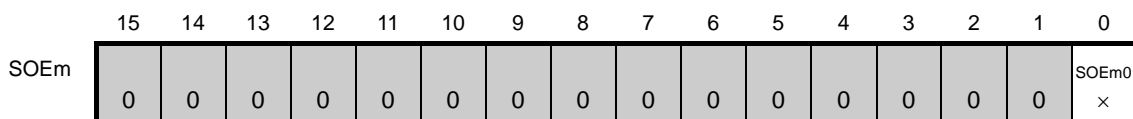
(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



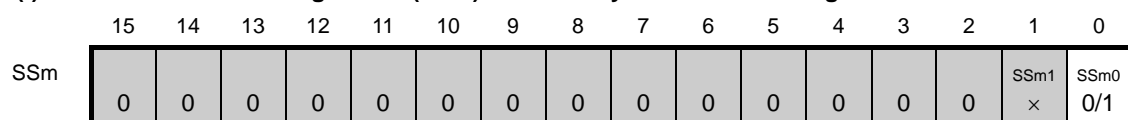
(d) Serial output register m (SOM) ... Sets only the bits of the target channel.



(e) Serial output enable register m (SOEm) ...The register that not used in this mode.



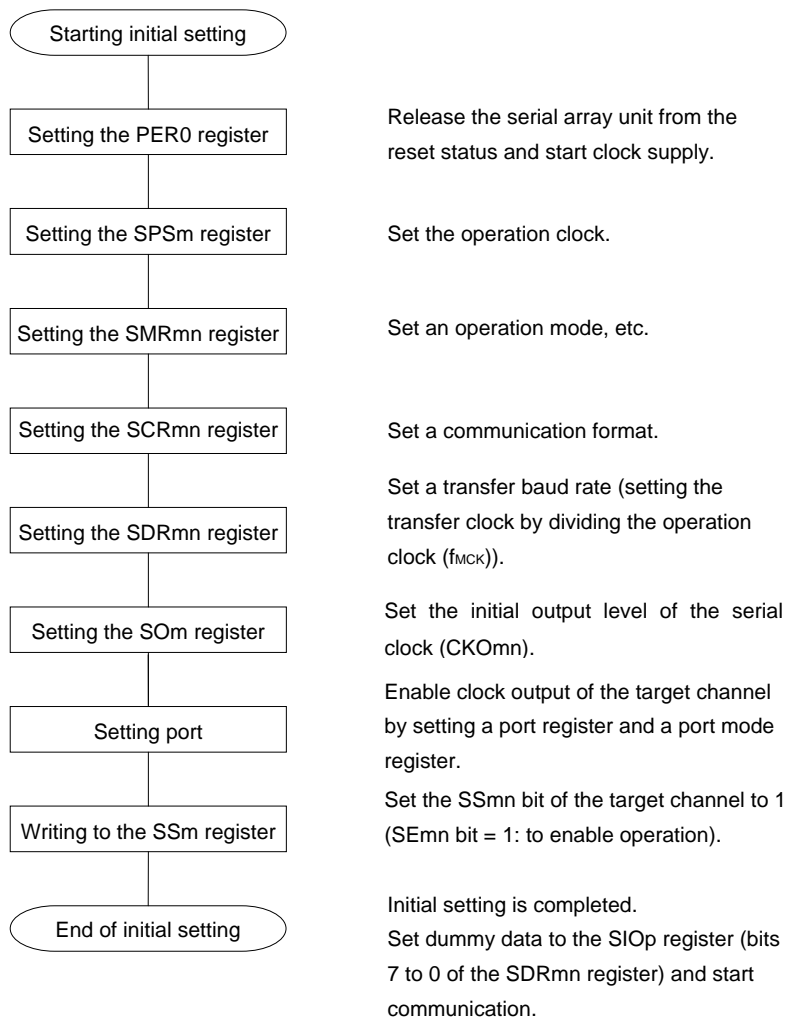
(f) Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.



- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00
  - : Setting is fixed in the CSI master reception mode, ■: Setting disabled (set to the initial value)  
 x: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

(2) Operation procedure

Figure 11-33. Initial Setting Procedure for Master Reception



<R>

**Figure 11-34. Procedure for Stopping Master Reception**

<R>

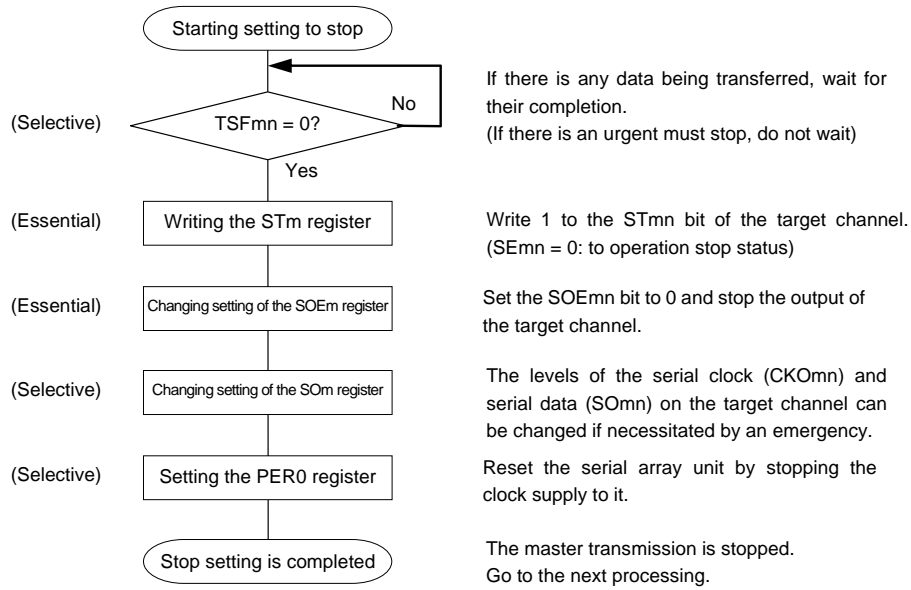
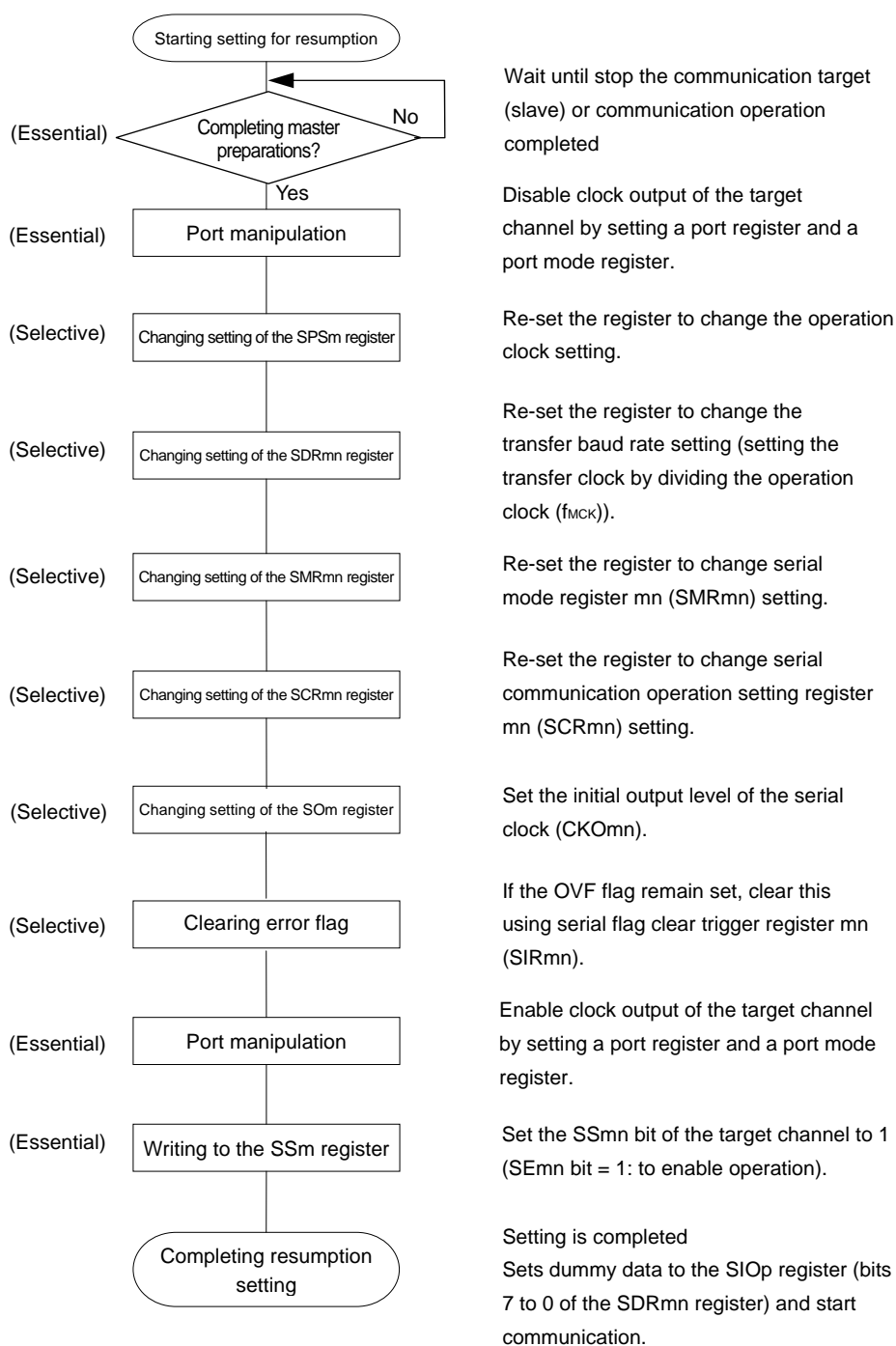


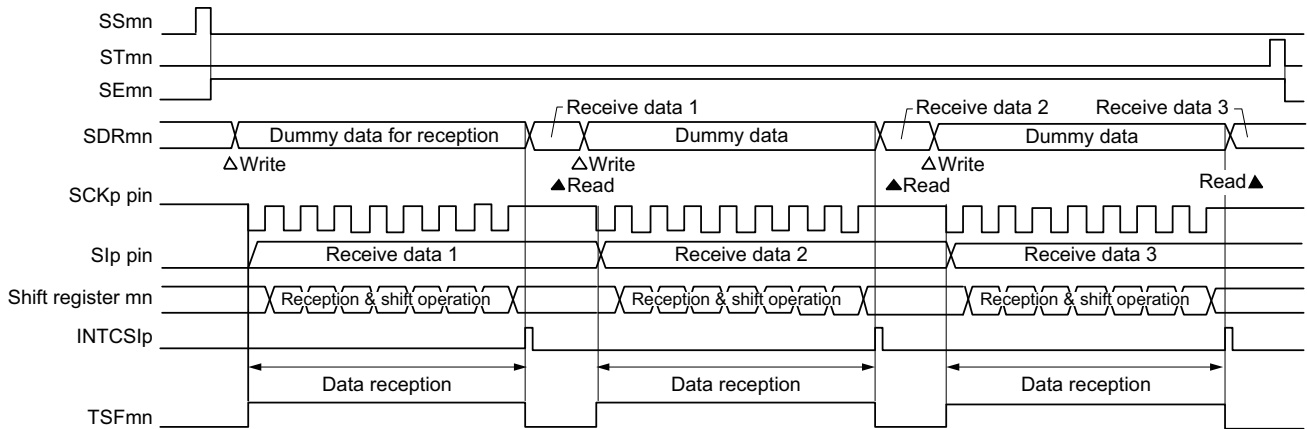
Figure 11-35. Procedure for Resuming Master Reception



**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

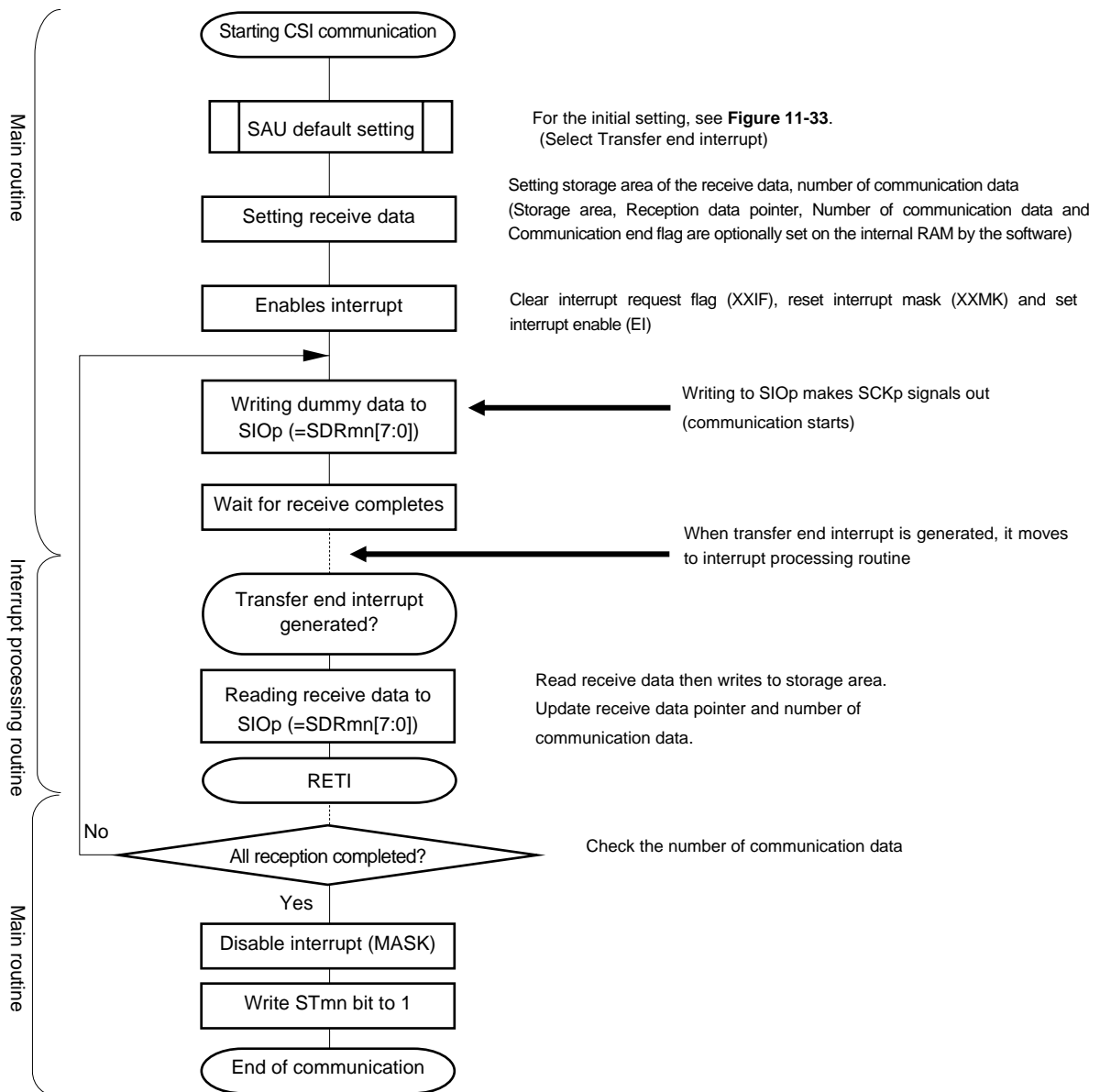
(3) Processing flow (in single-reception mode)

Figure 11-36. Timing Chart of Master Reception (in Single-Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

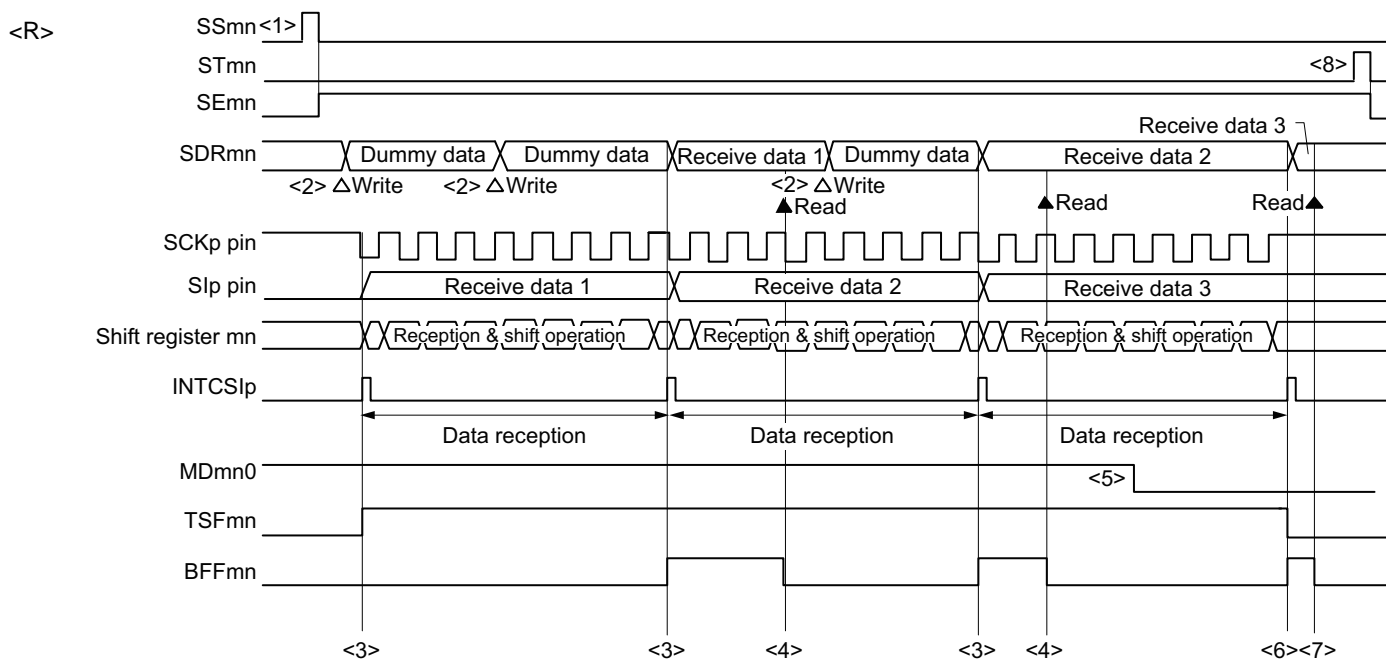
Figure 11-37. Flowchart of Master Reception (in Single-Reception Mode)



<R>

(4) Processing flow (in continuous reception mode)

Figure 11-38. Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)

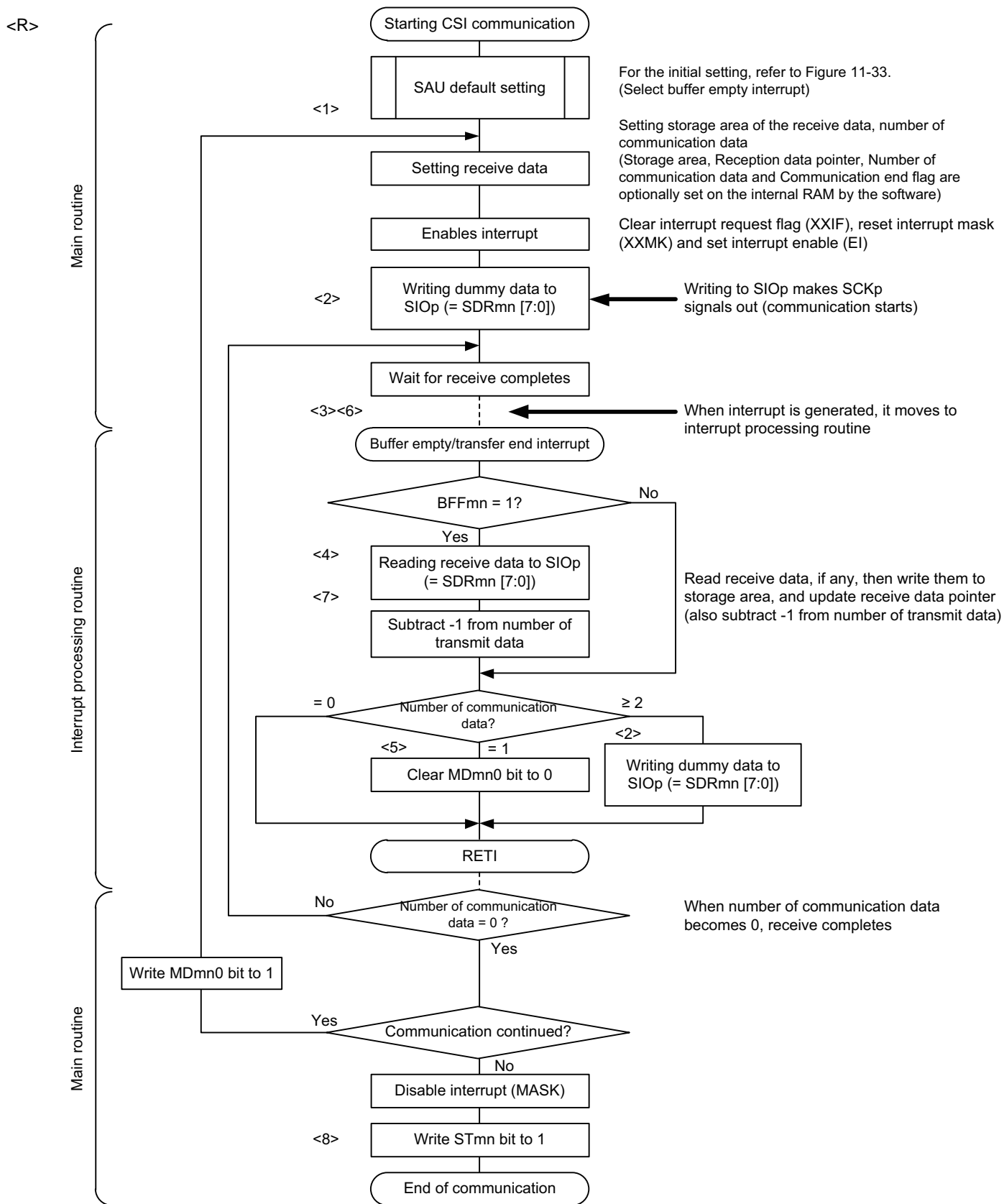


**Caution** The MDmn0 bit can be rewritten even during operation. However, rewrite it before receive of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last receive data.

**Remarks 1.** <math>\langle 1 \rangle</math> to <math>\langle 8 \rangle</math> in the figure correspond to <math>\langle 1 \rangle</math> to <math>\langle 8 \rangle</math> in **Figure 11-39 Flowchart of Master Reception (in Continuous Reception Mode)**.

**2.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

Figure 11-39. Flowchart of Master Reception (in Continuous Reception Mode)



**Remark** <1> to <8> in the figure correspond to <1> to <8> in Figure 11-38 Timing Chart of Master Reception (in Continuous Reception Mode).



### 11.5.3 Master transmission/reception

Master transmission/reception is that the R7F0C010 output a transfer clock and transmits/receives data to/from other device.

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SI00, SO00
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate <sup>Note</sup>	Max. $f_{MCK}/2$ [Hz] Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [Hz] $f_{CLK}$ : System clock frequency
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data I/O starts at the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

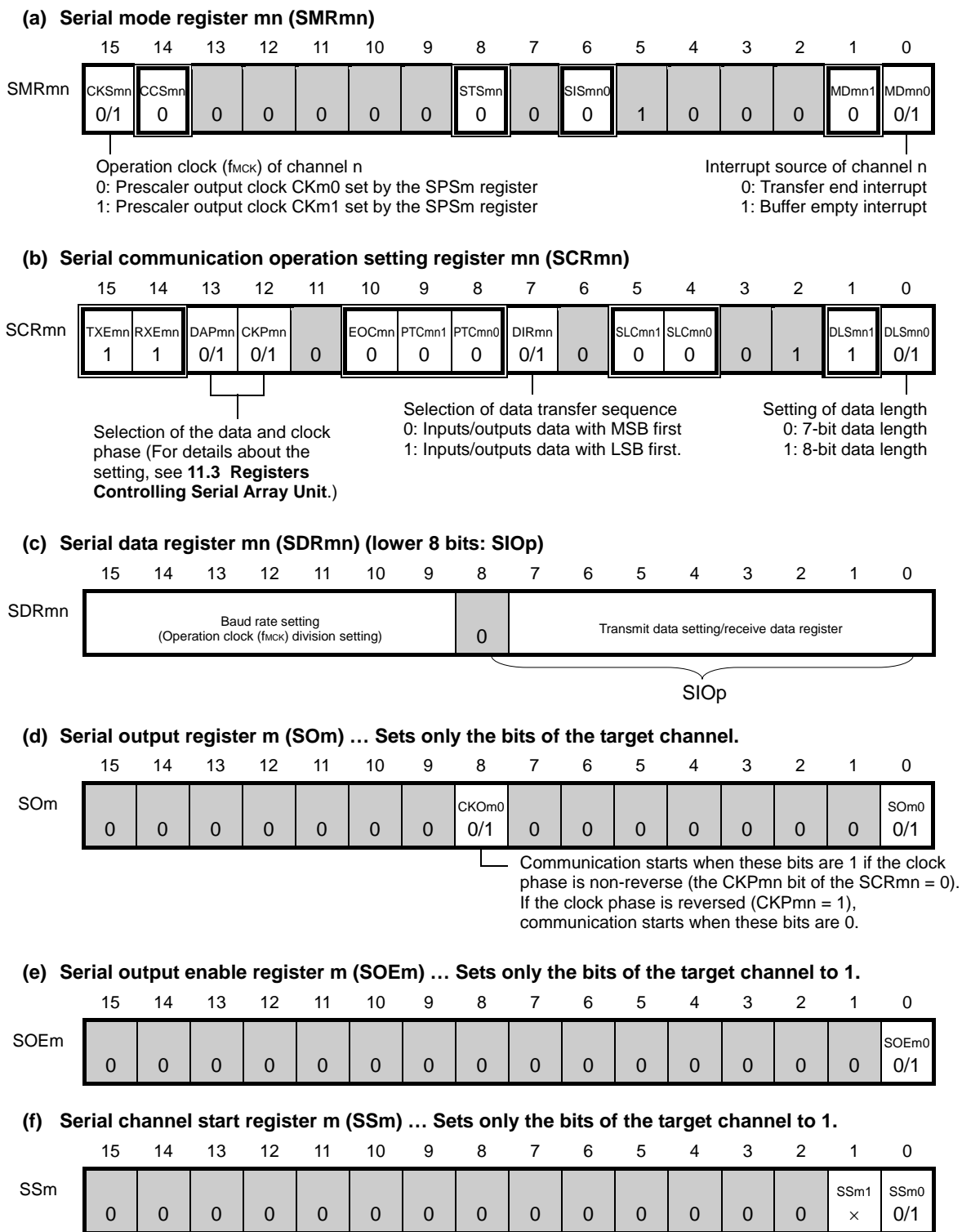
&lt;R&gt;

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

(1) Register setting

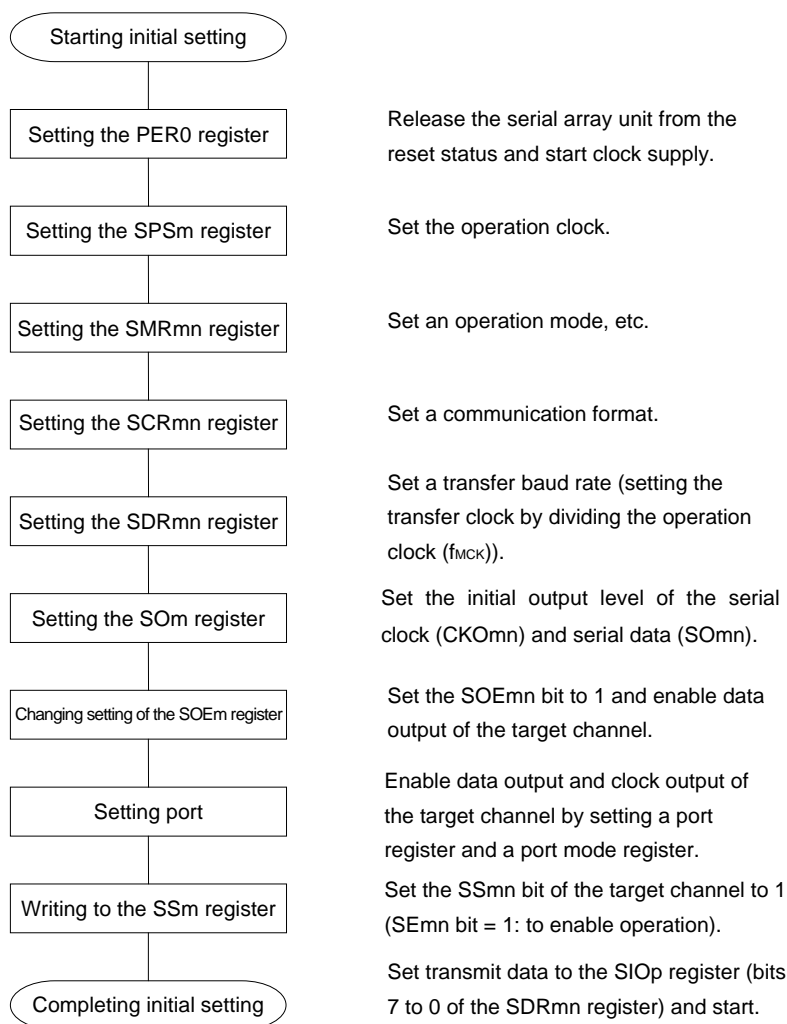
Figure 11-40. Example of Contents of Registers for Master Transmission/Reception of 3-Wire Serial I/O (CSI00)



- Remarks 1. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00
2. □: Setting is fixed in the CSI master transmission/reception mode  
 ■: Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

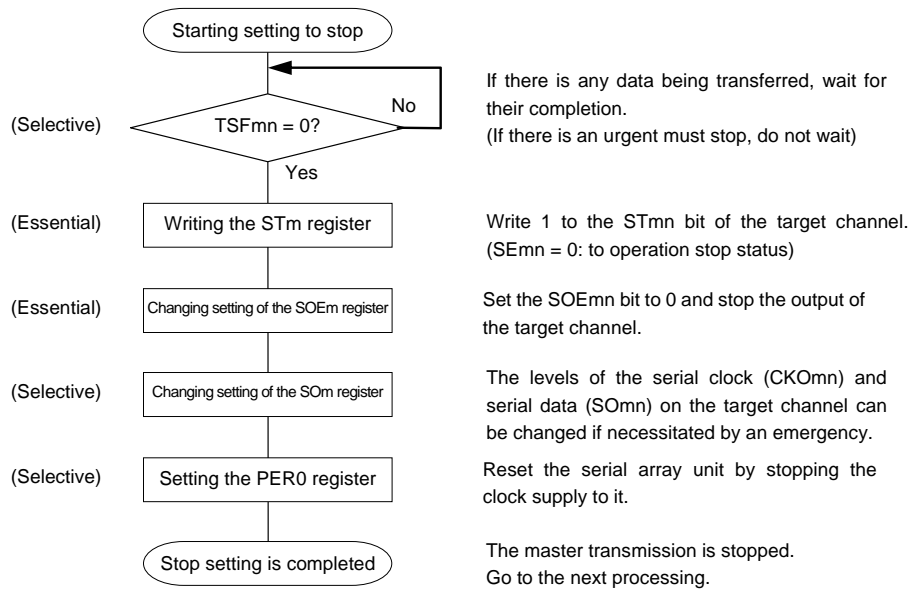
(2) Operation procedure

Figure 11-41. Initial Setting Procedure for Master Transmission/Reception



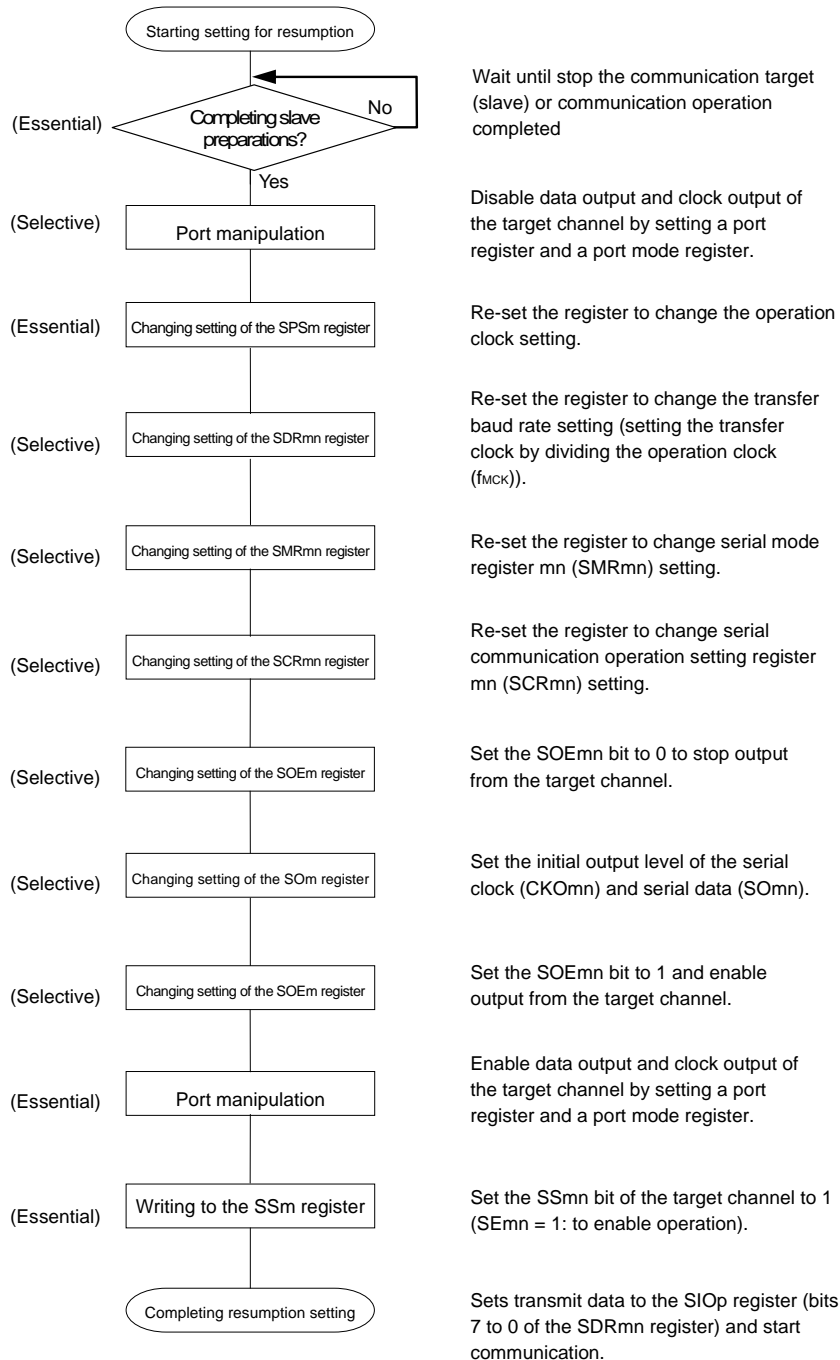
<R>

**Figure 11-42. Procedure for Stopping Master Transmission/Reception**



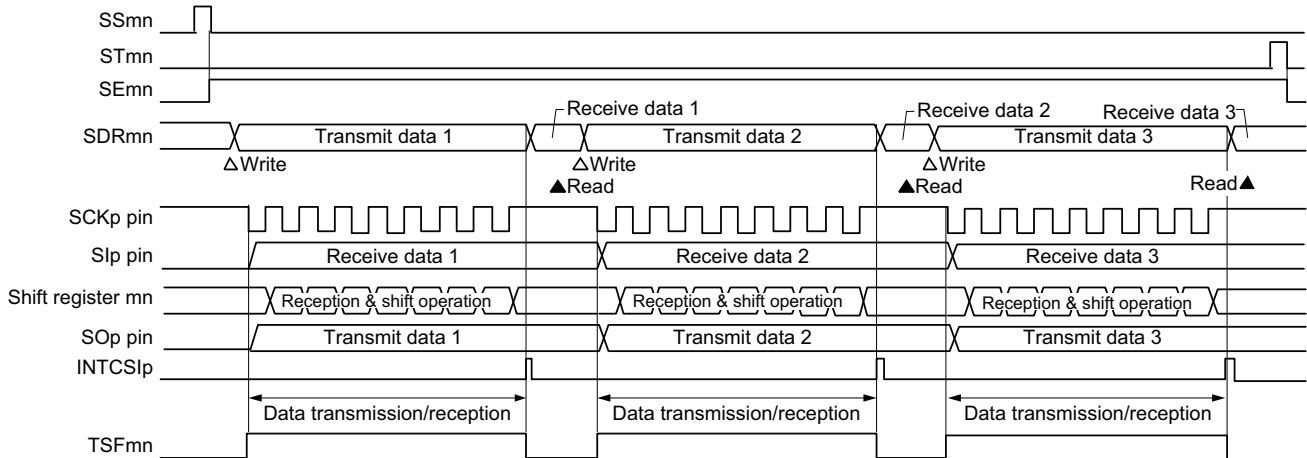
<R>

**Figure 11-43. Procedure for Resuming Master Transmission/Reception**



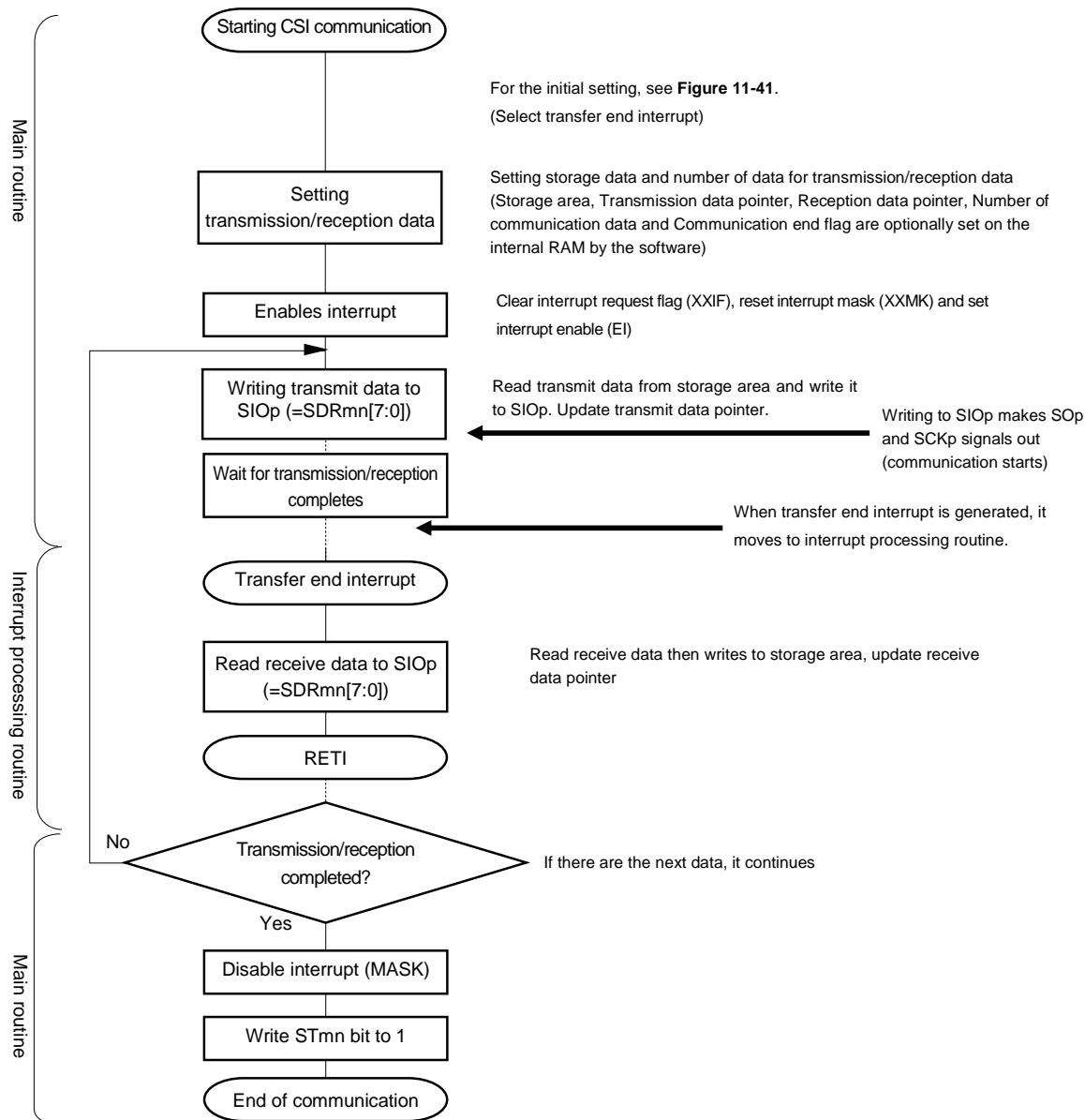
## (3) Processing flow (in single-transmission/reception mode)

<R> **Figure 11-44. Timing Chart of Master Transmission/Reception (in Single-Transmission/Reception Mode)**  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

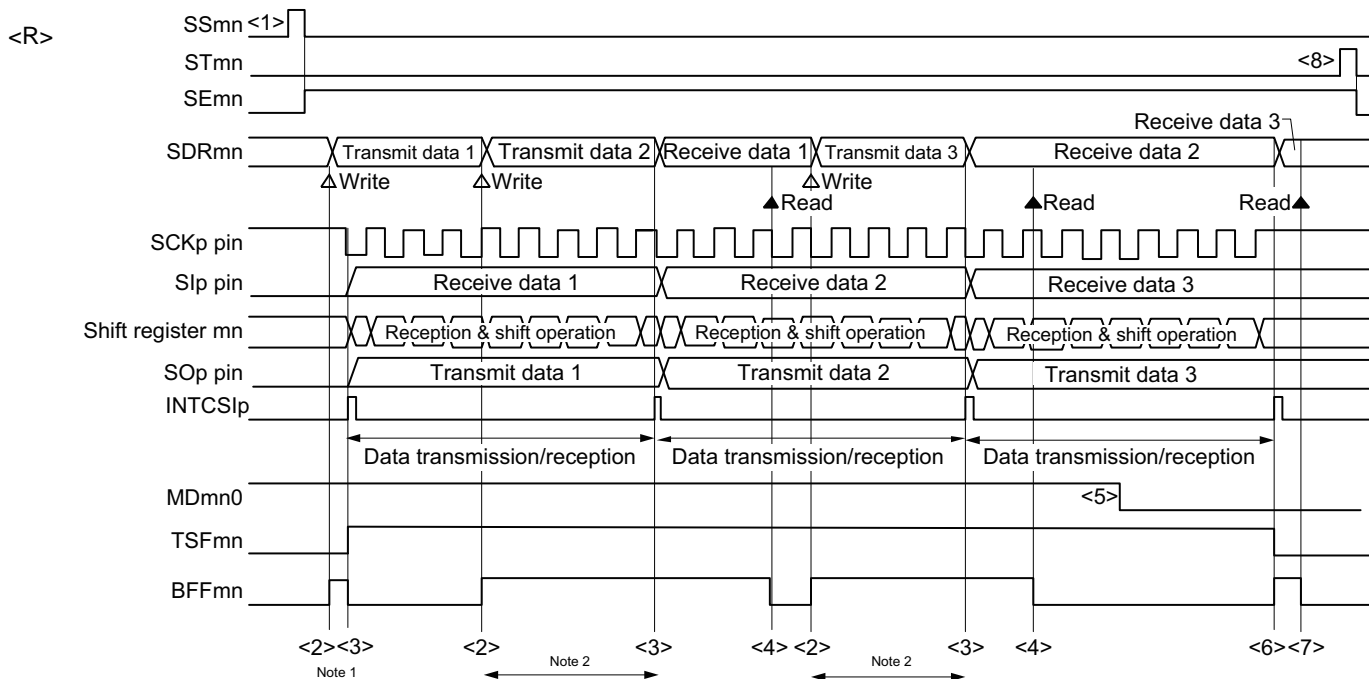
Figure 11-45. Flowchart of Master Transmission/Reception (in Single-Transmission/Reception Mode)



<R>

(4) Processing flow (in continuous transmission/reception mode)

Figure 11-46. Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



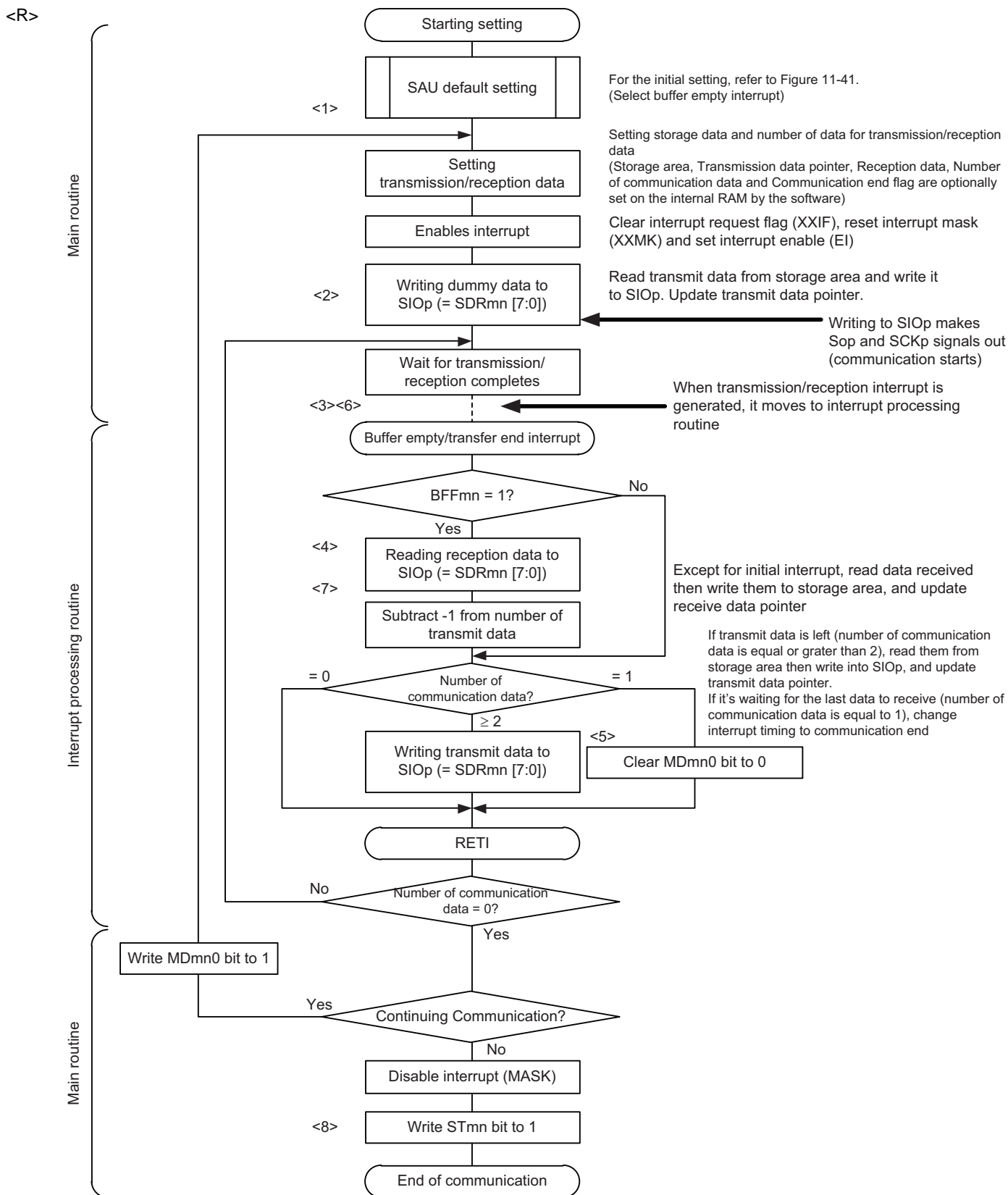
- Notes**
1. If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.
  2. The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.

- Remarks**
1. <1> to <8> in the figure correspond to <1> to <8> in Figure 11-47 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode).
  2. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00



Figure 11-47. Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)



Remark <1> to <8> in the figure correspond to <1> to <8> in Figure 11-46 Timing Chart of Master Transmission/Reception (in Continuous Transmission/Reception Mode).

### 11.5.4 Slave transmission

Slave transmission is that the R7F0C010 transmit data to another device in the state of a transfer clock being input from another device.

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SO00
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] <sup>Notes 1, 2</sup> .
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is  $f_{MCK}/6$  [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{MCK}$ : Operation clock frequency of target channel

$f_{SCK}$ : Serial clock frequency

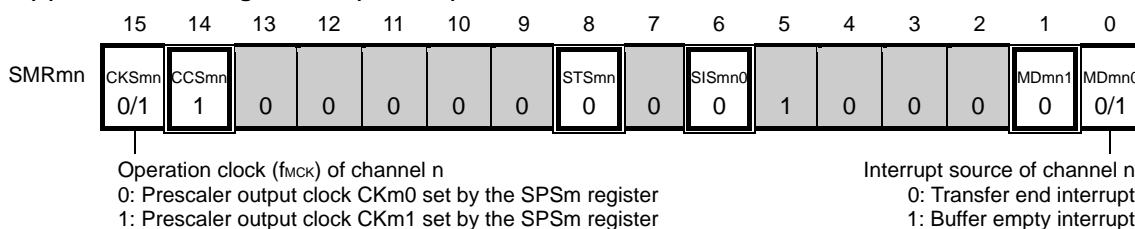
**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

<R>

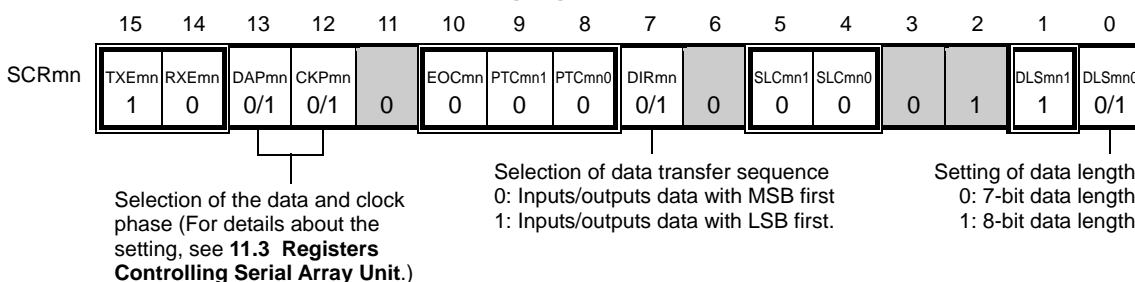
(1) Register setting

Figure 11-48. Example of Contents of Registers for Slave Transmission of 3-Wire Serial I/O (CSI00)

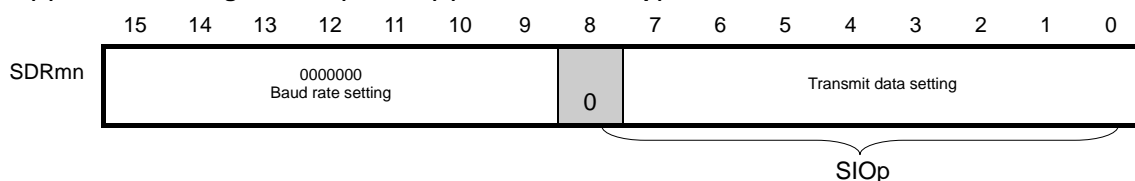
(a) Serial mode register mn (SMRmn)



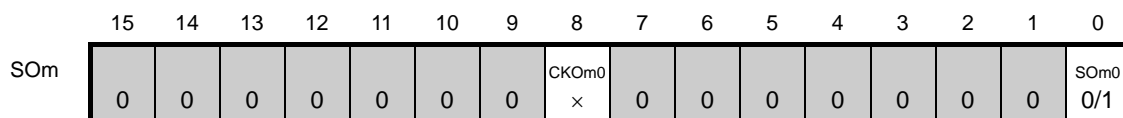
(b) Serial communication operation setting register mn (SCRmn)



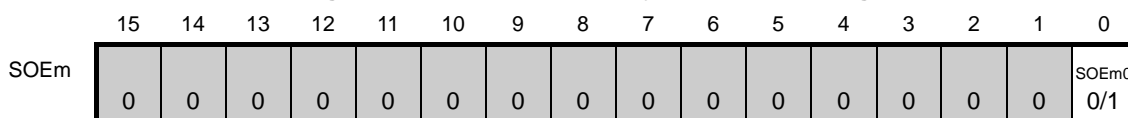
(c) Serial data register mn (SDRmn) (lower 8 bits: SIOp)



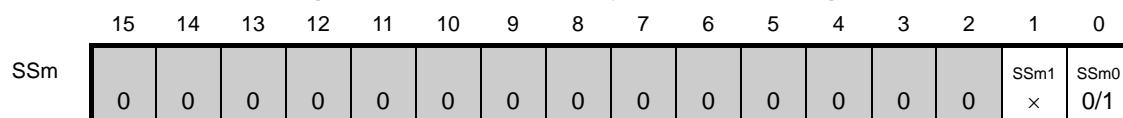
(d) Serial output register m (SOM) ... Sets only the bits of the target channel.



(e) Serial output enable register m (SOEm) ... Sets only the bits of the target channel to 1.



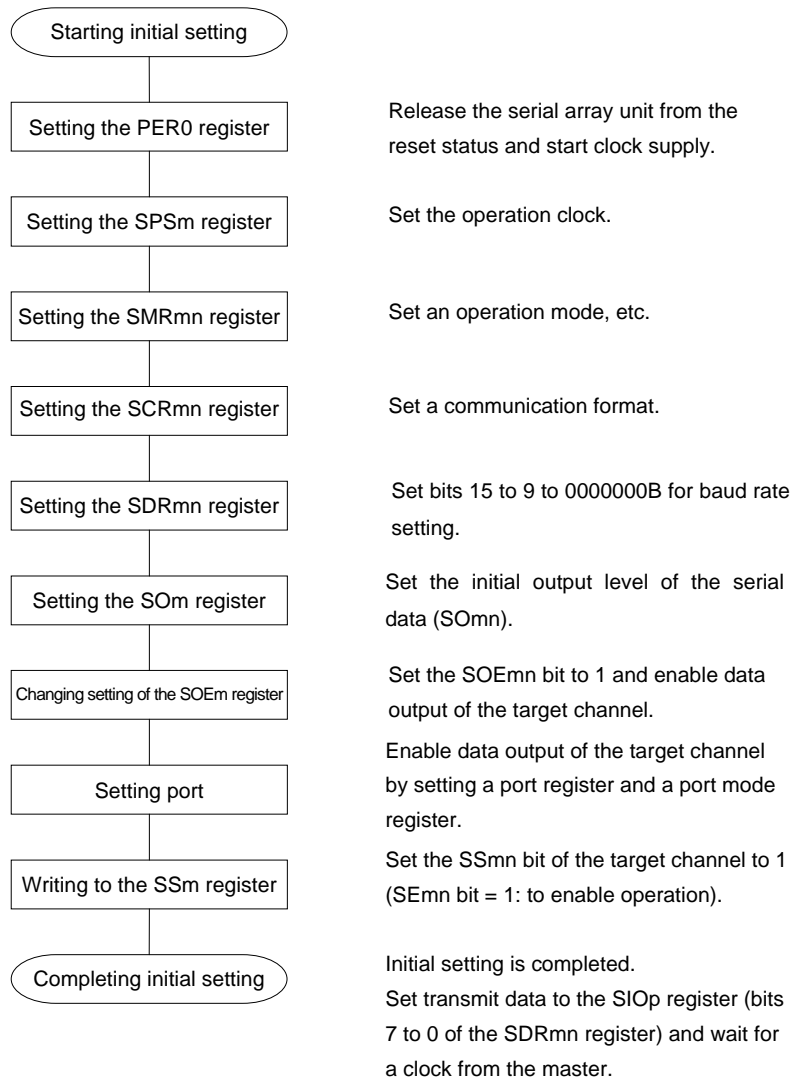
(f) Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.



- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00
  - : Setting is fixed in the CSI slave transmission mode, ◻: Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

## (2) Operation procedure

Figure 11-49. Initial Setting Procedure for Slave Transmission



**Figure 11-50. Procedure for Stopping Slave Transmission**

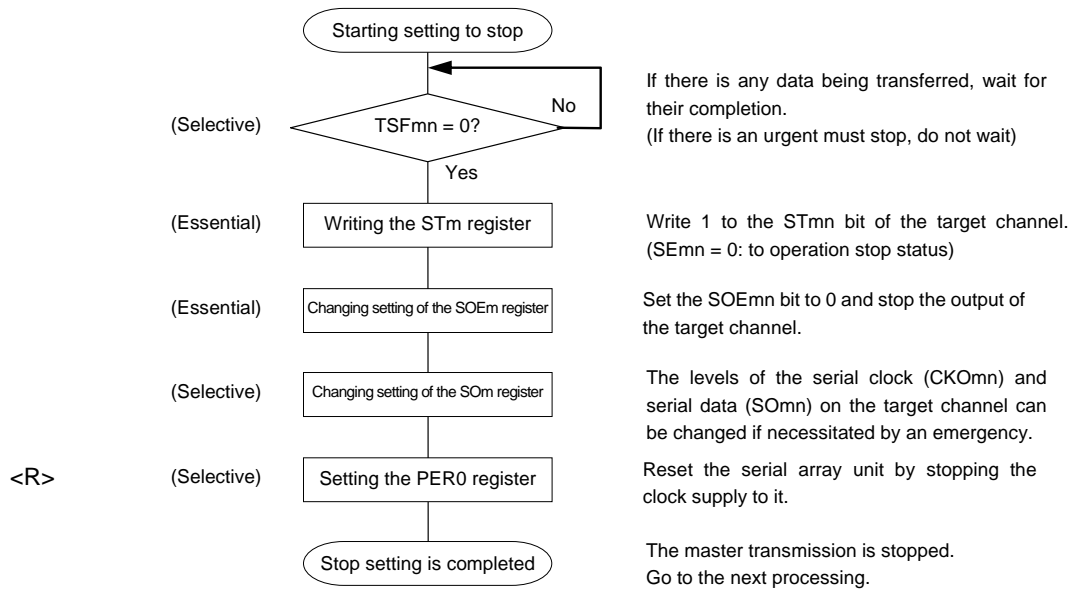
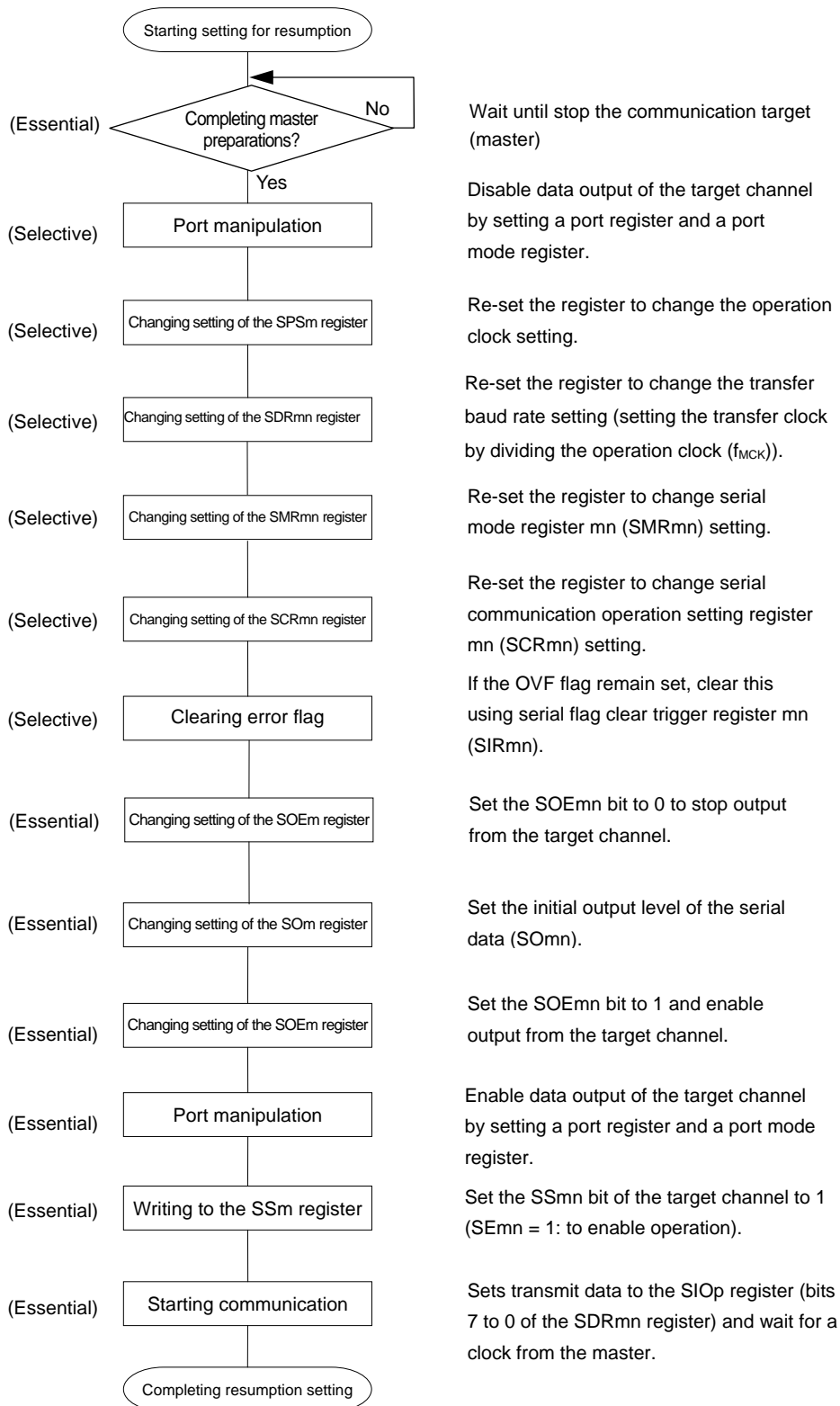


Figure 11-51. Procedure for Resuming Slave Transmission

<R>

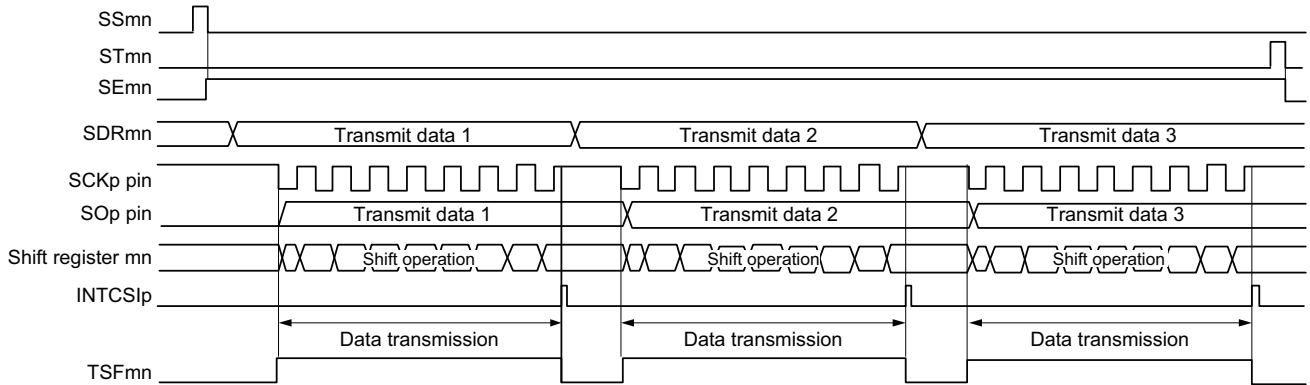


**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

(3) Processing flow (in single-transmission mode)

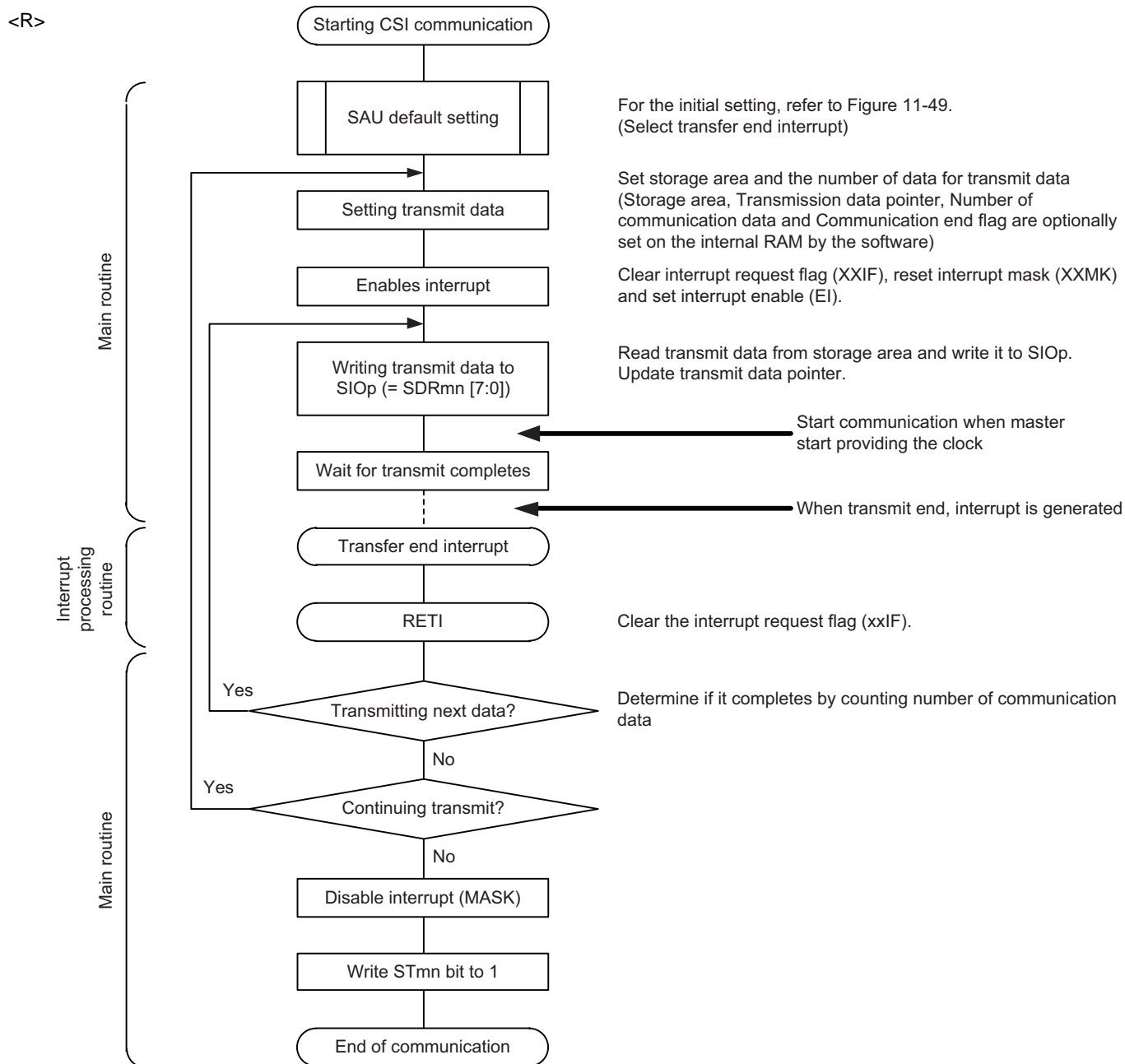
Figure 11-52. Timing Chart of Slave Transmission (in Single-Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)

<R>



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

Figure 11-53. Flowchart of Slave Transmission (in Single-Transmission Mode)

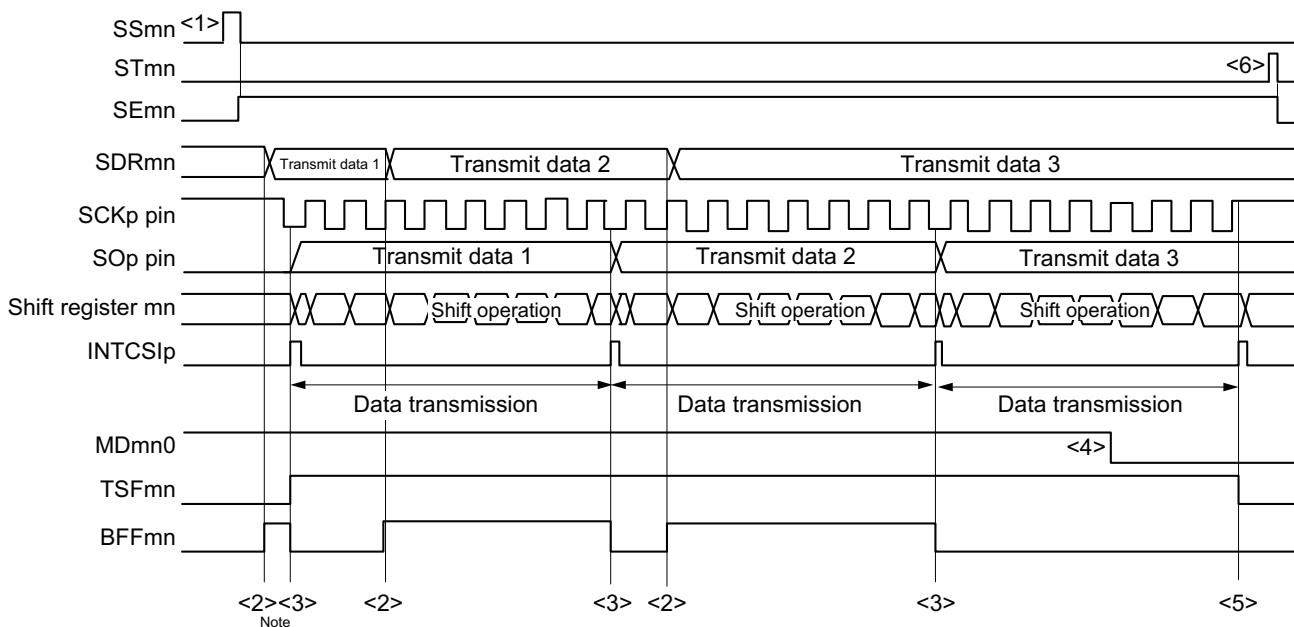




(4) Processing flow (in continuous transmission mode)

Figure 11-54. Timing Chart of Slave Transmission (in Continuous Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)

<R>

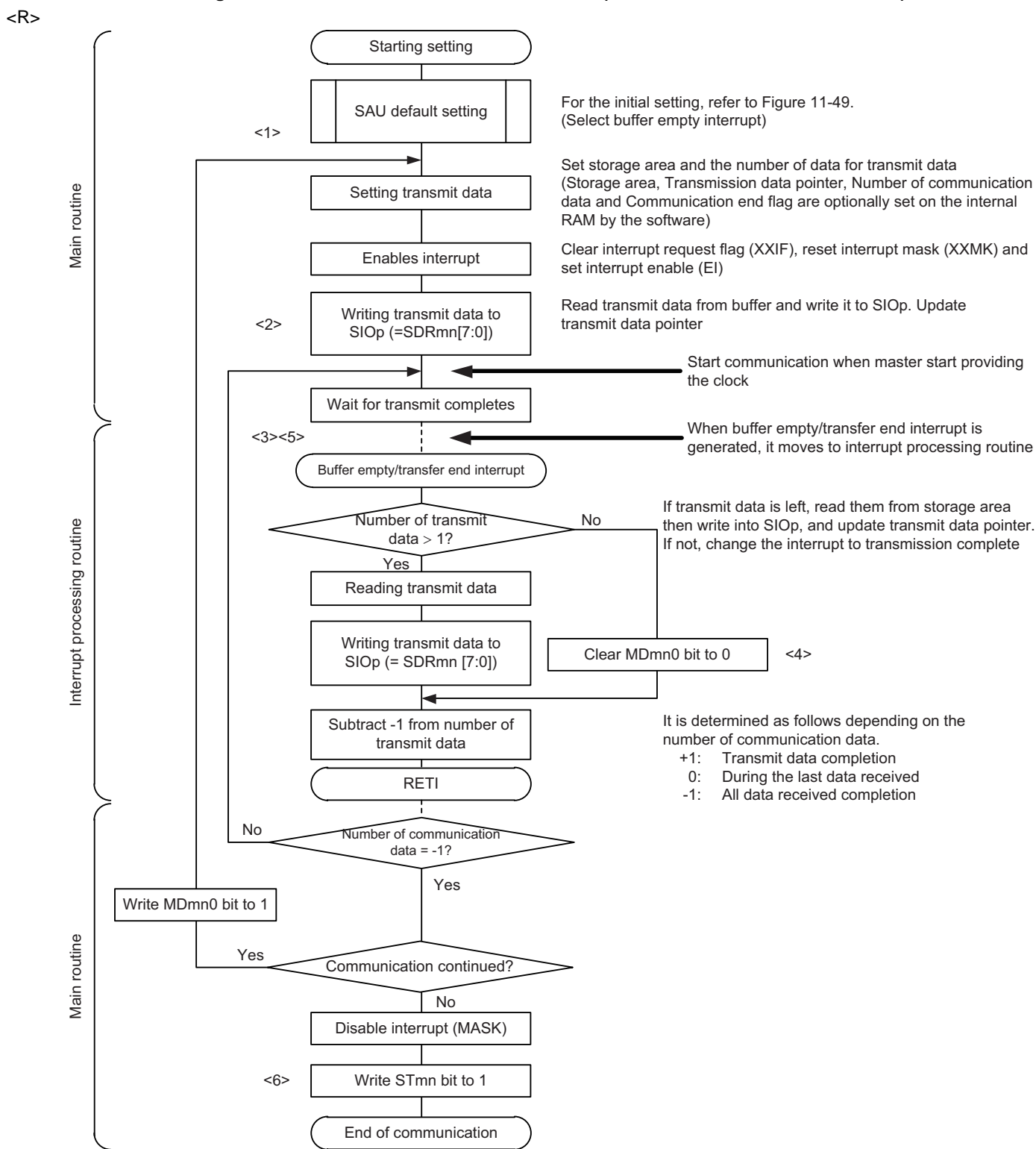


**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

Figure 11-55. Flowchart of Slave Transmission (in Continuous Transmission Mode)



**Remark** <1> to <6> in the figure correspond to <1> to <6> in **Figure 11-54 Timing Chart of Slave Transmission (in Continuous Transmission Mode)**.

### 11.5.5 Slave reception

Slave reception is that the R7F0C010 receive data from another device in the state of a transfer clock being input from another device.

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SI00
Interrupt	INTCSI00
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)
Error detection flag	Overflow error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] <sup>Notes 1, 2</sup>
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is  $f_{MCK}/6$  [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{MCK}$ : Operation clock frequency of target channel

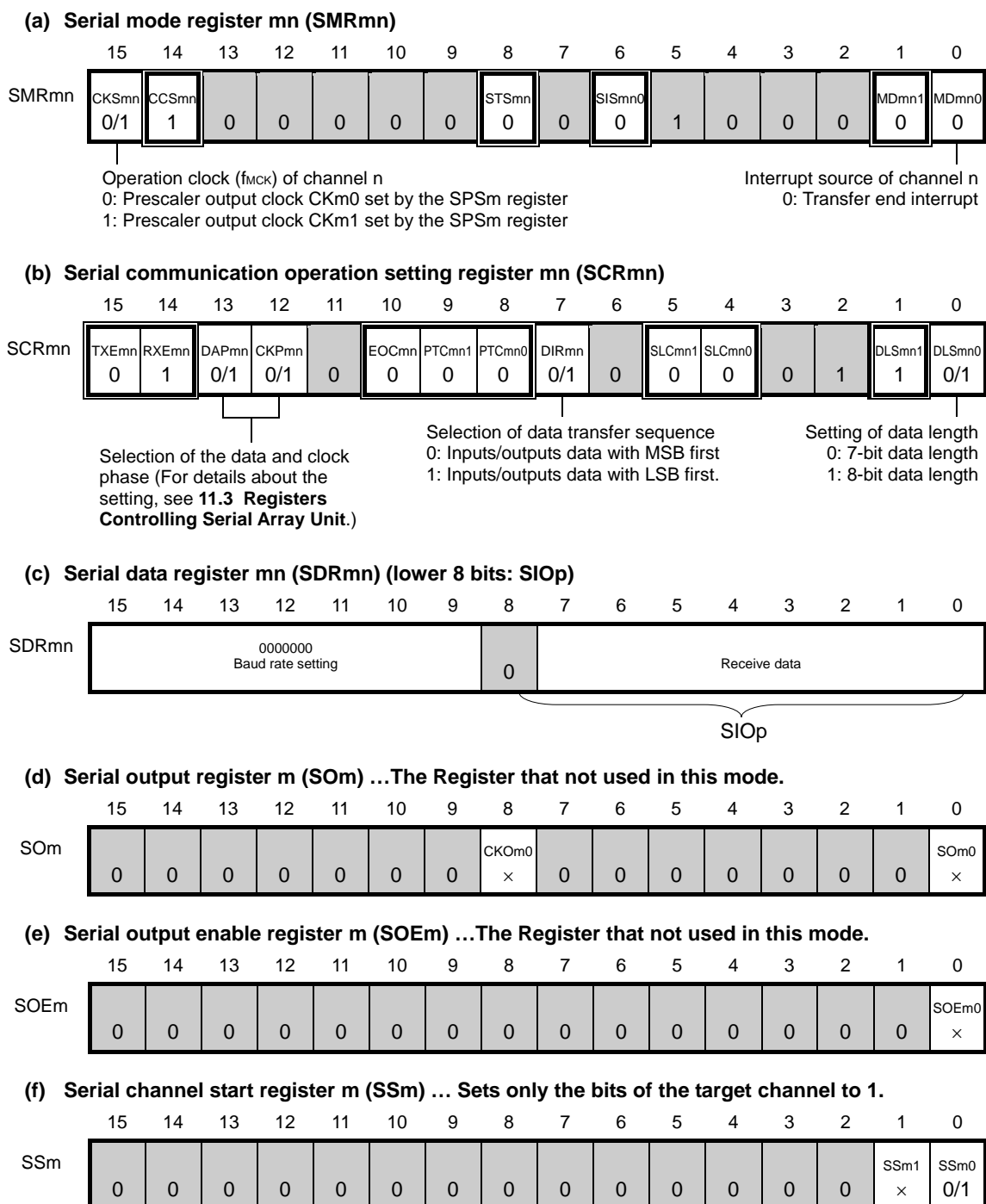
$f_{SCK}$ : Serial clock frequency

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

<R>

## (1) Register setting

Figure 11-56. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00)



- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00
  - : Setting is fixed in the CSI slave transmission mode, □: Setting disabled (set to the initial value)  
 x: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

(2) Operation procedure

Figure 11-57. Initial Setting Procedure for Slave Reception

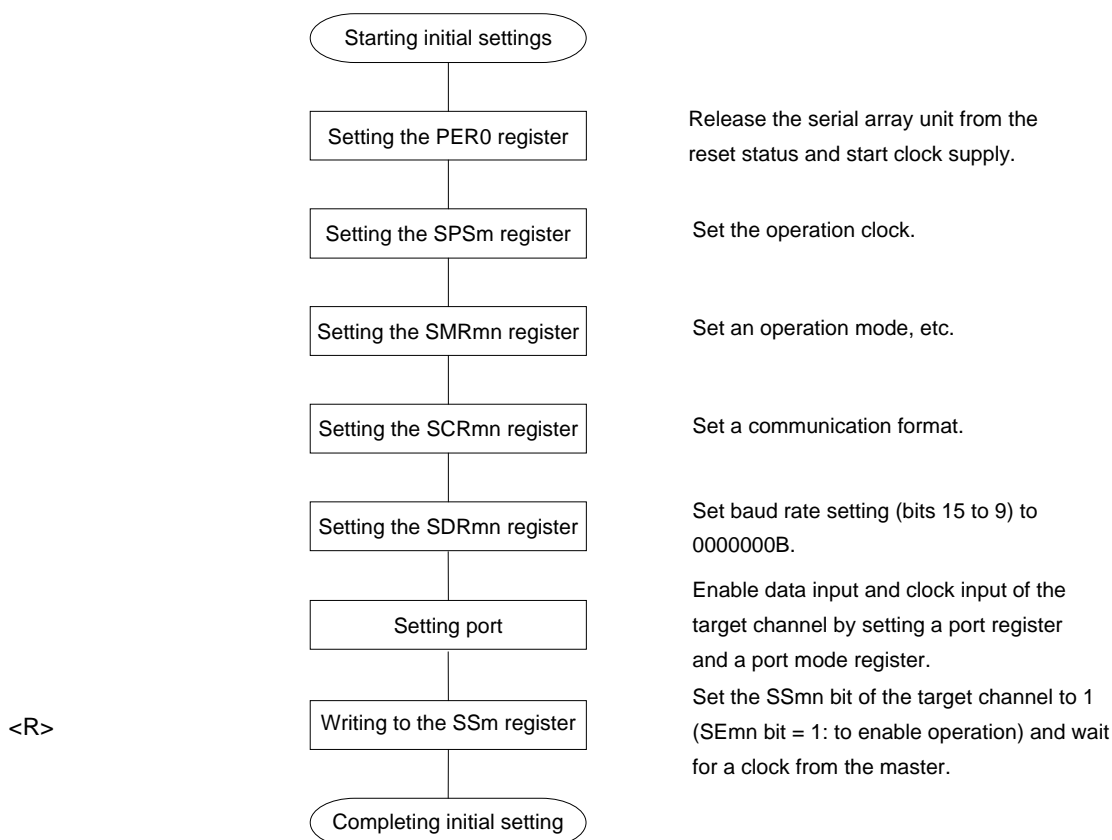
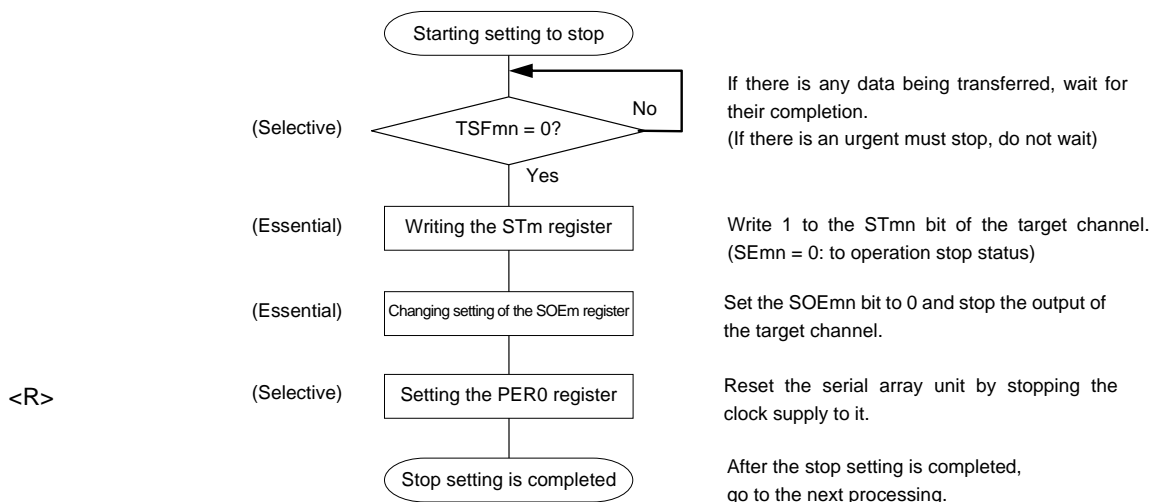
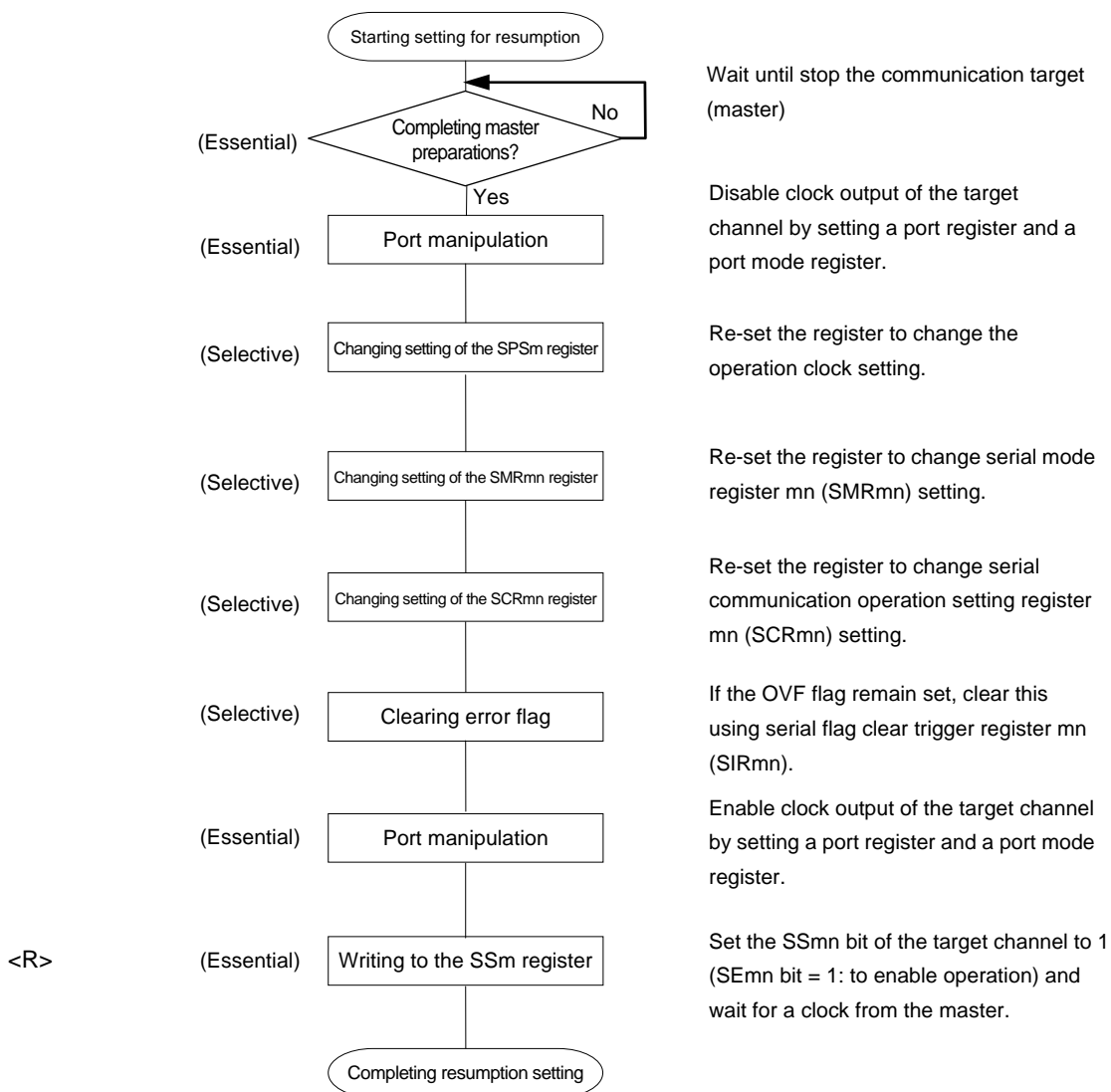


Figure 11-58. Procedure for Stopping Slave Reception



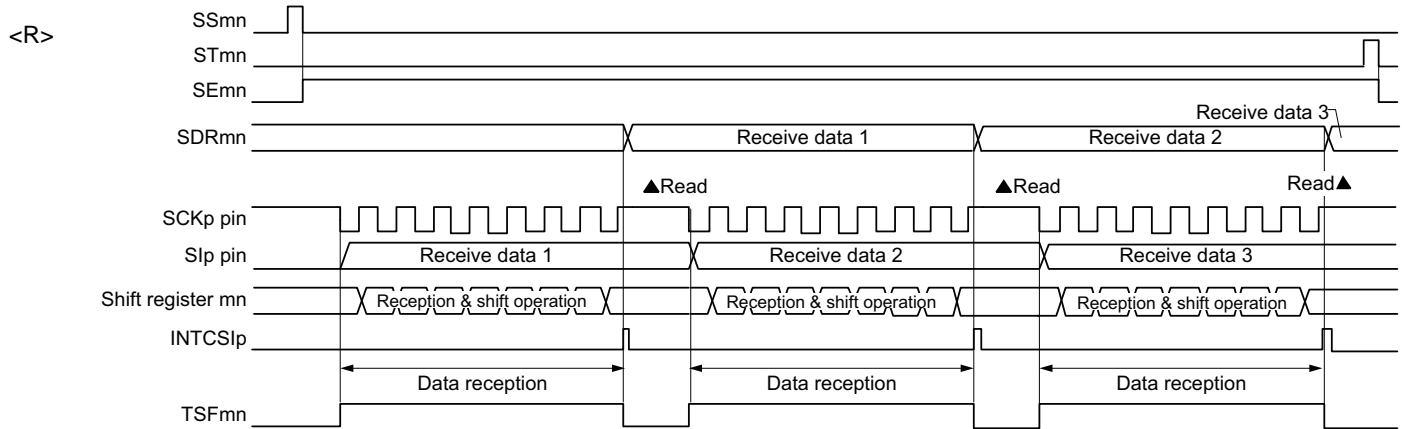
**Figure 11-59. Procedure for Resuming Slave Reception**



**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

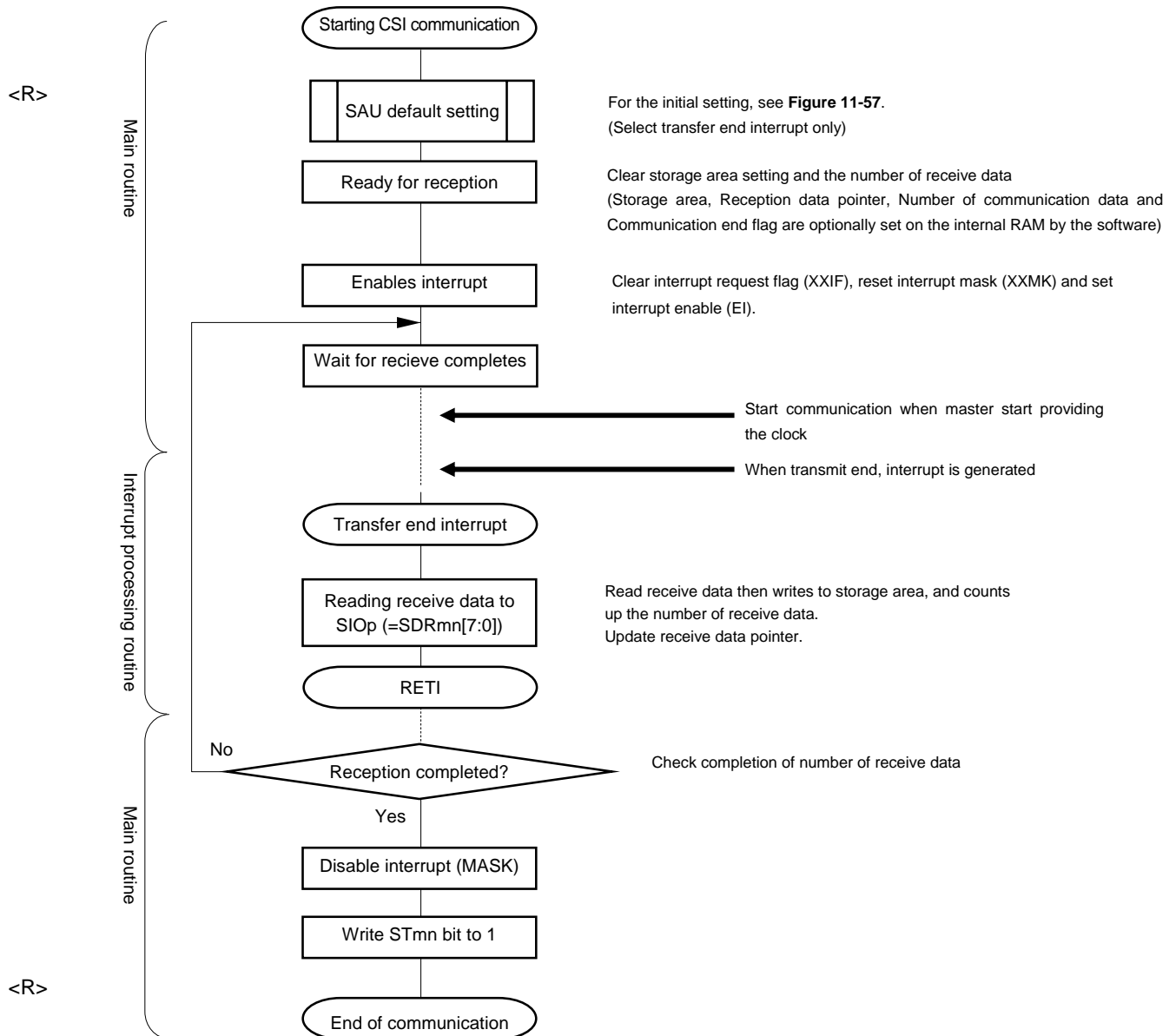
(3) Processing flow (in single-reception mode)

Figure 11-60. Timing Chart of Slave Reception (in Single-Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

Figure 11-61. Flowchart of Slave Reception (in Single-Reception Mode)





### 11.5.6 Slave transmission/reception

Slave transmission/reception is that the R7F0C010 transmit/receive data to/from another device in the state of a transfer clock being input from another device.

3-Wire Serial I/O	CSI00
Target channel	Channel 0 of SAU0
Pins used	SCK00, SI00, S0m0
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] <sup>Notes 1, 2</sup>
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data I/O starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first

**Notes 1.** Because the external serial clock input to the SCK00 pin is sampled internally and used, the fastest transfer rate is  $f_{MCK}/6$  [Hz].

**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{MCK}$ : Operation clock frequency of target channel

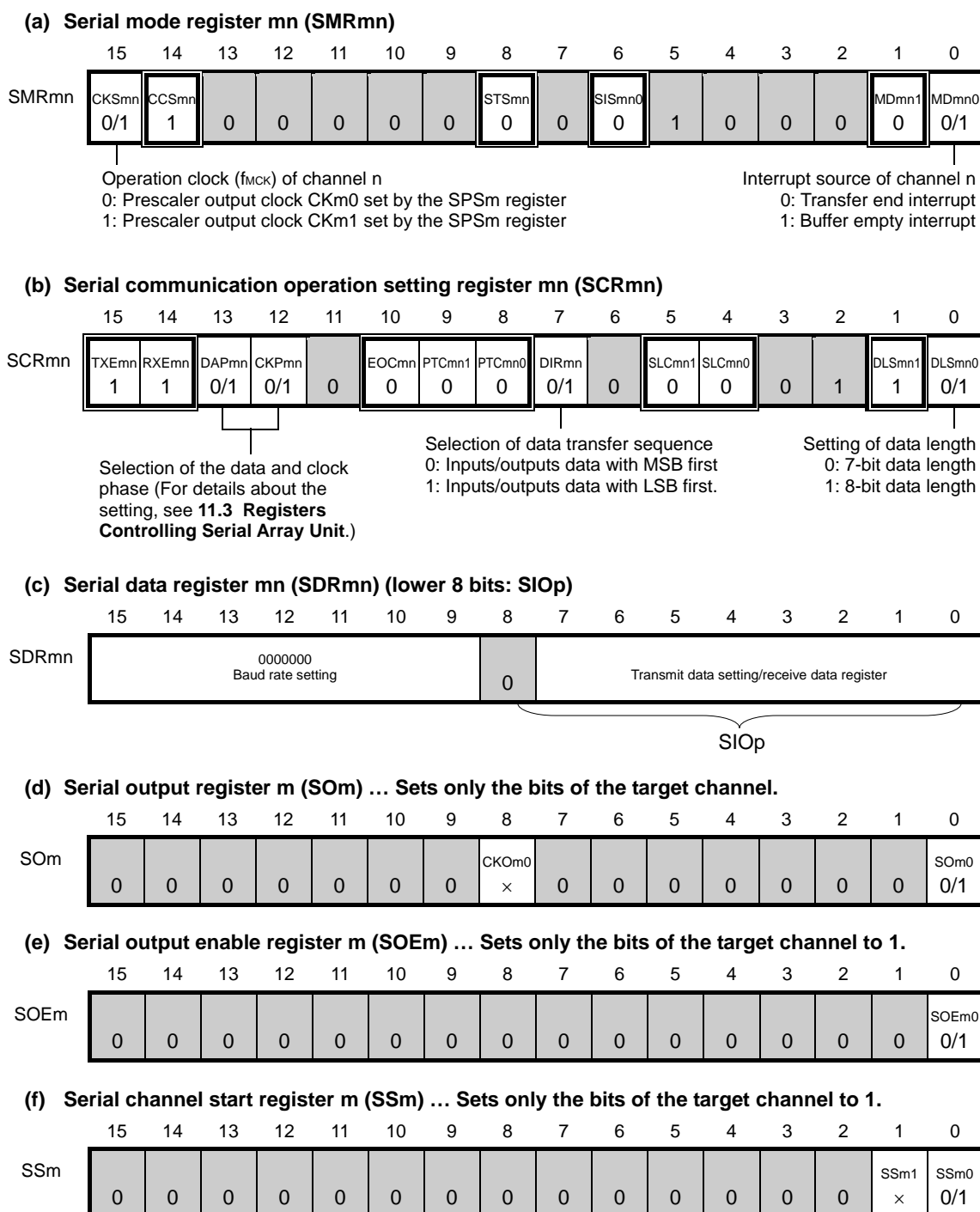
$f_{CLK}$ : Serial clock frequency

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

<R>

## (1) Register setting

Figure 11-62. Example of Contents of Registers for Slave Transmission/Reception of 3-Wire Serial I/O (CSI00)

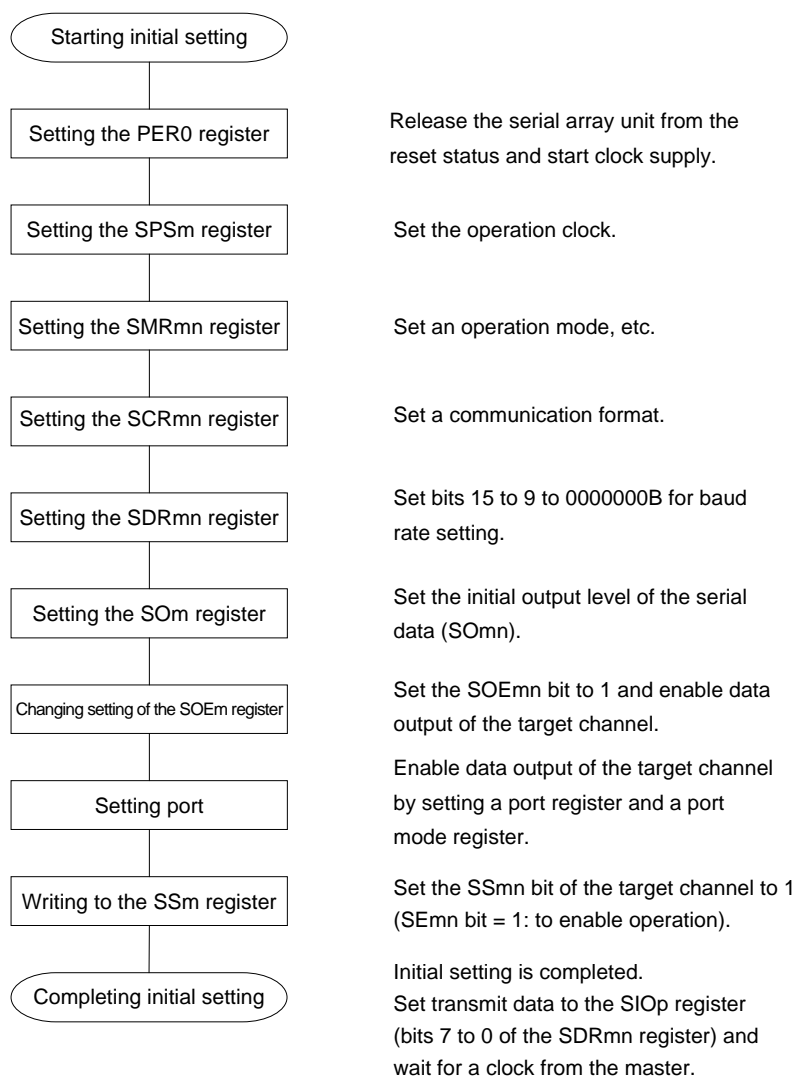


**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

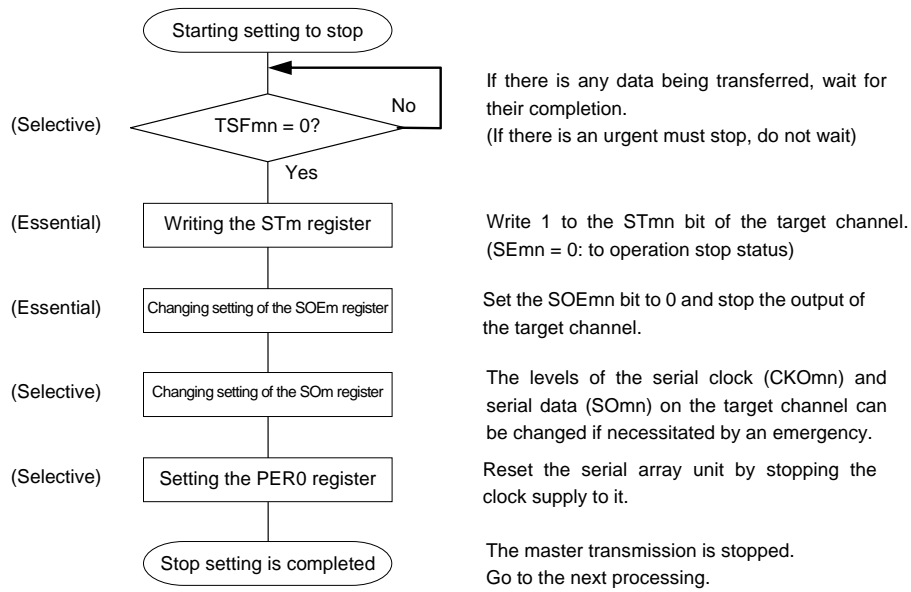
- Remarks**
- m: Unit number ( $m = 0$ ), n: Channel number ( $n = 0$ ), p: CSI number ( $p = 00$ ), mn = 00
  - : Setting is fixed in the CSI slave transmission/reception mode,  
 ■: Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

## (2) Operation procedure

Figure 11-63. Initial Setting Procedure for Slave Transmission/Reception

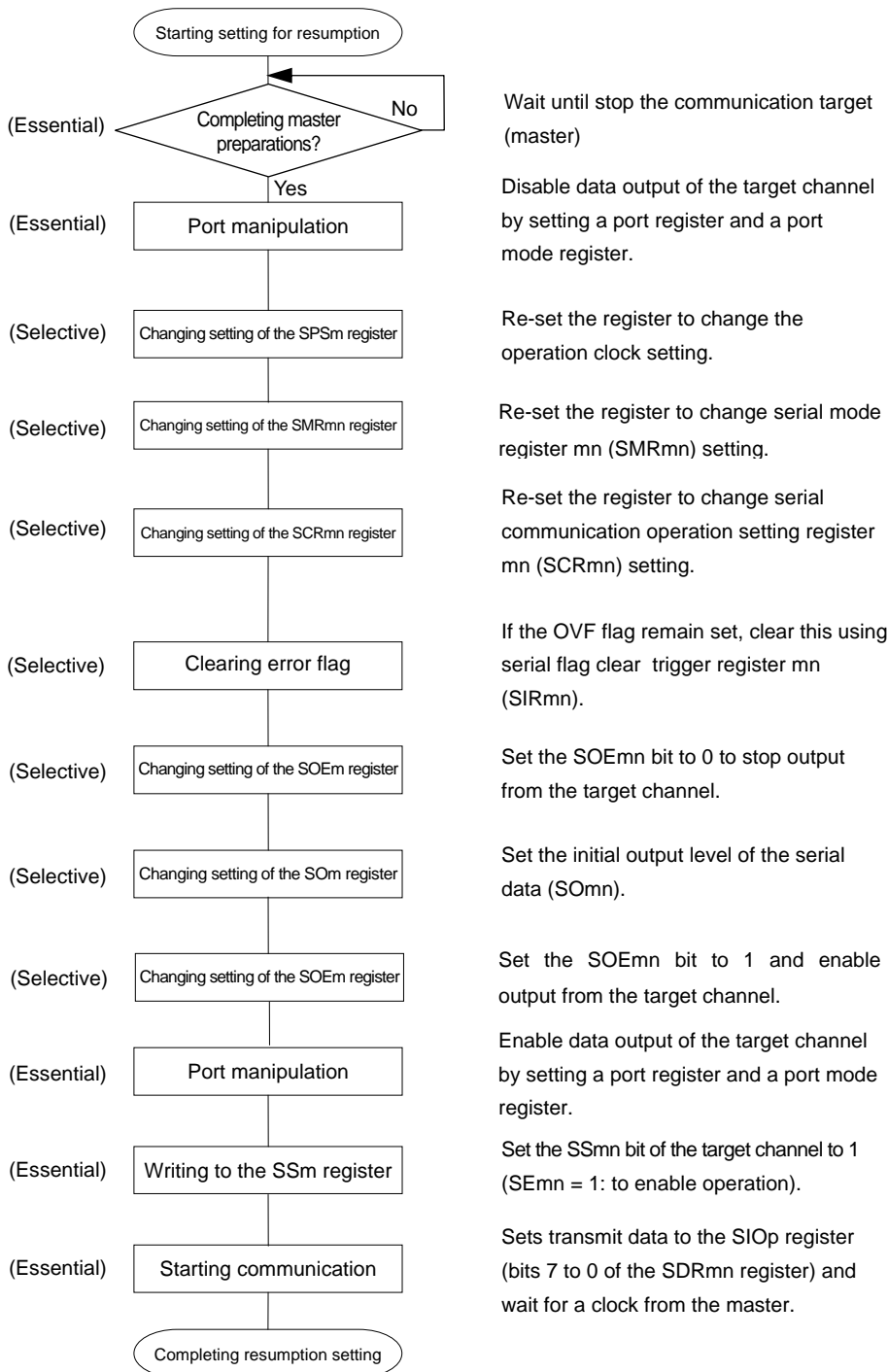


**Figure 11-64. Procedure for Stopping Slave Transmission/Reception**



<R>

Figure 11-65. Procedure for Resuming Slave Transmission/Reception

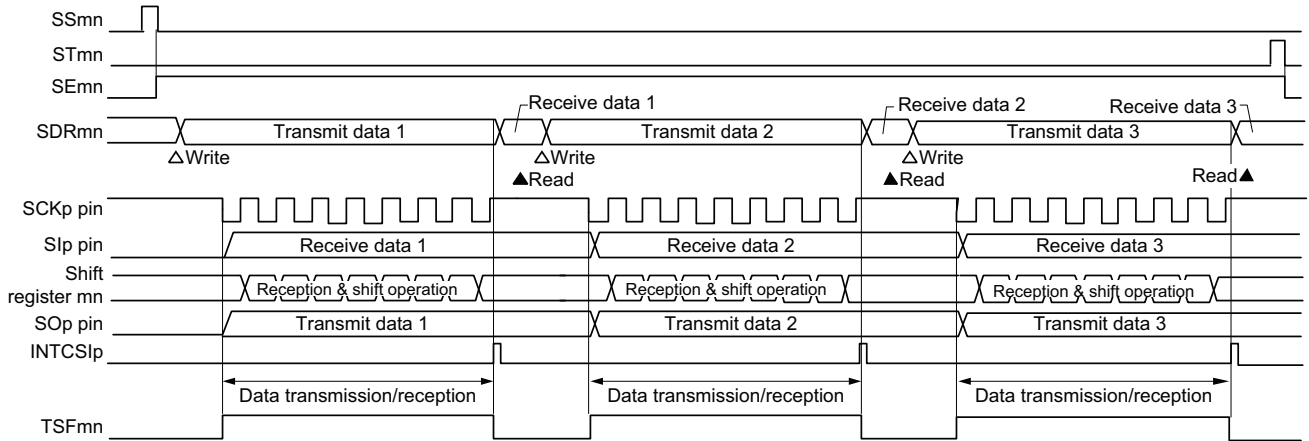


- Cautions**
1. Be sure to set transmit data to the SIOp register before the clock from the master is started.
  2. If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

(3) Processing flow (in single-transmission/reception mode)

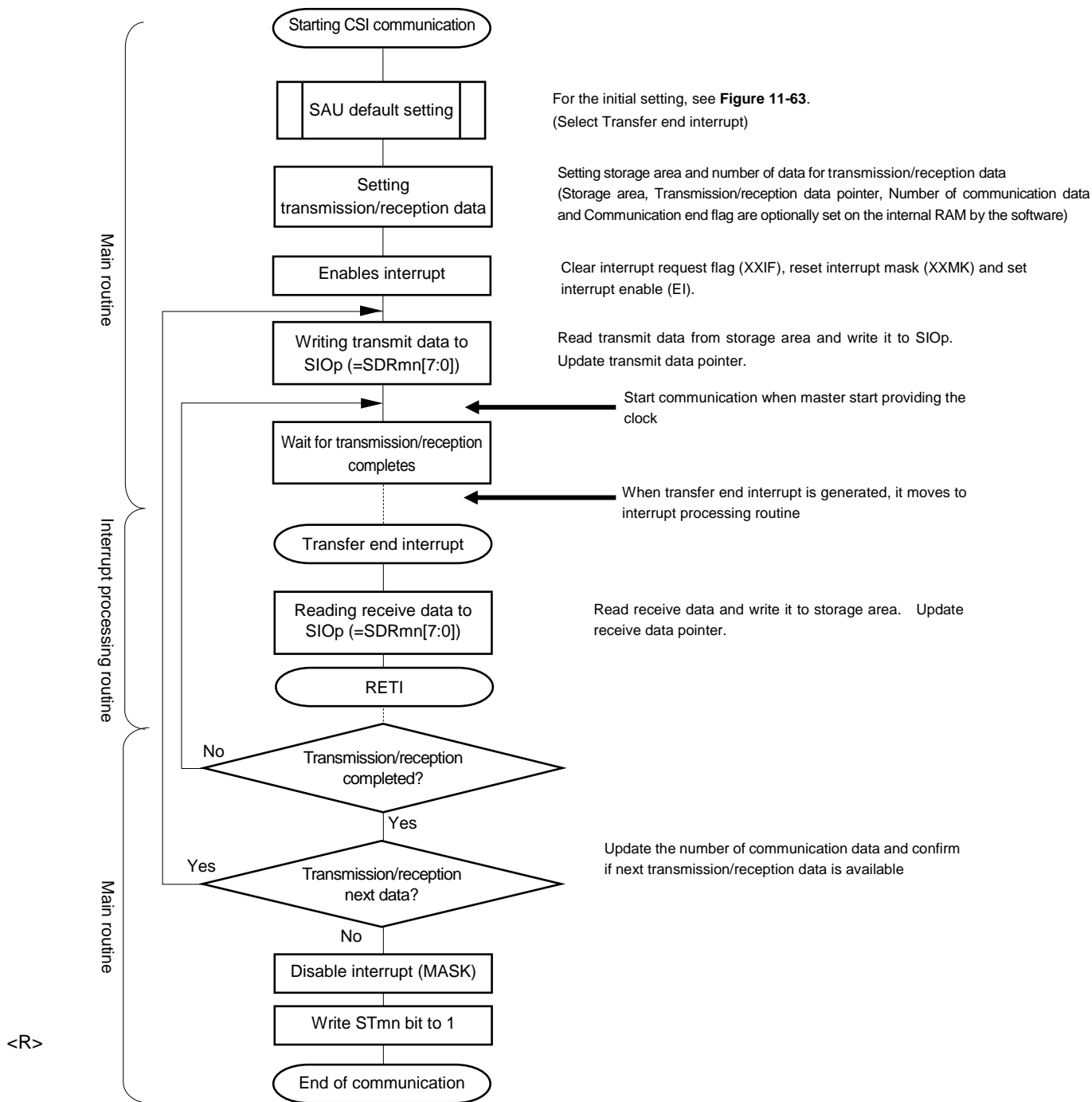
<R>

Figure 11-66. Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00

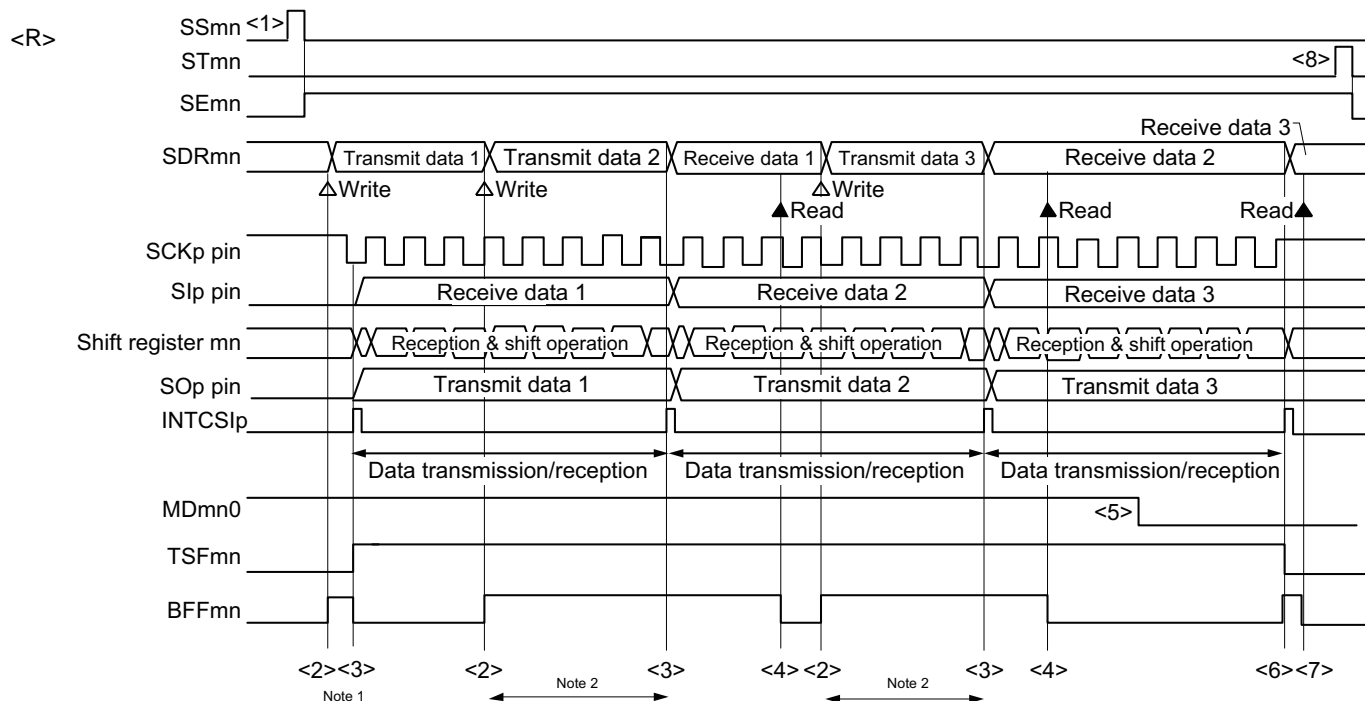
Figure 11-67. Flowchart of Slave Transmission/Reception (in Single-Transmission/Reception Mode)



**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

## (4) Processing flow (in continuous transmission/reception mode)

Figure 11-68. Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Notes** 1. If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

2. The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

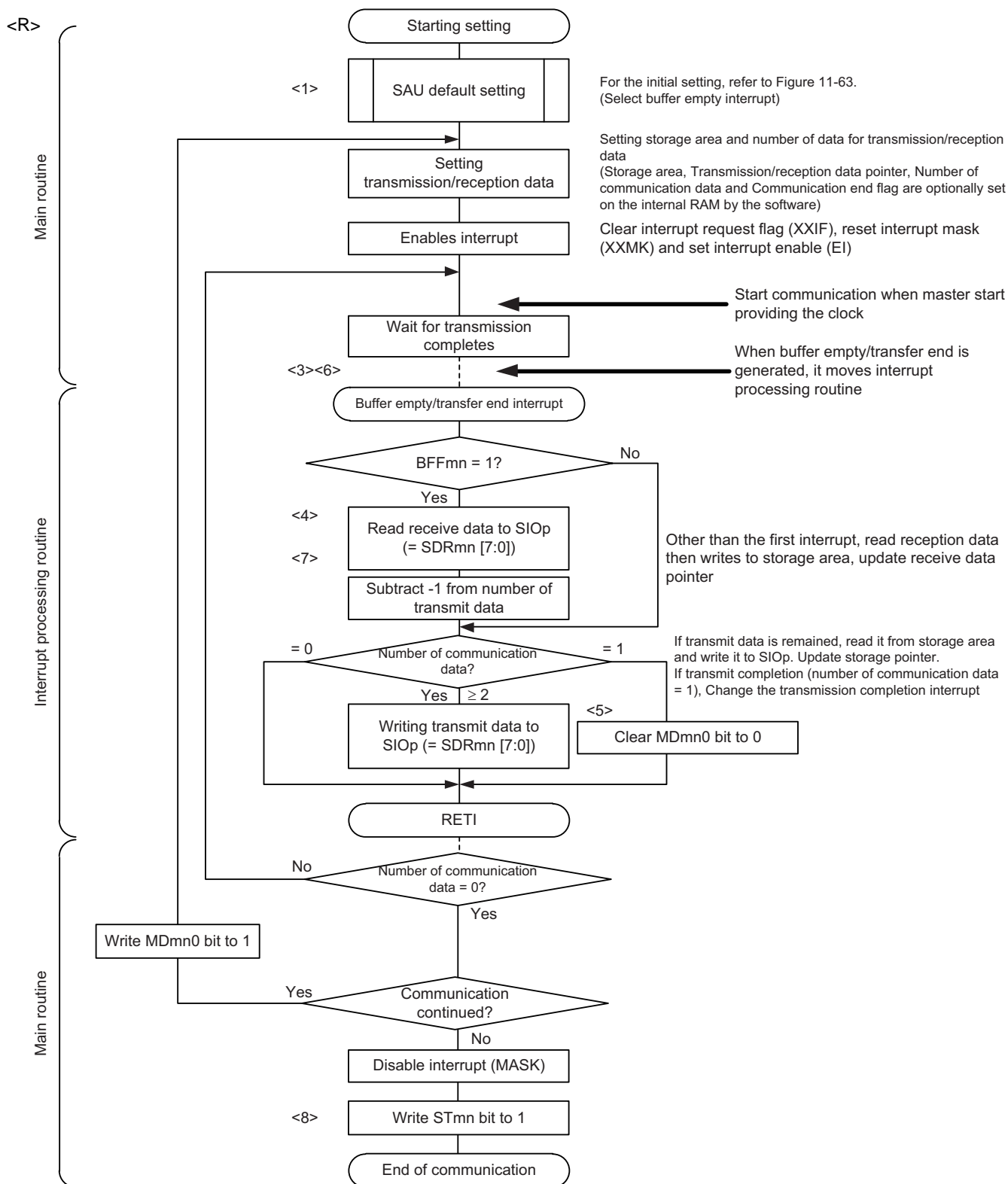
**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.

**Remarks** 1. <1> to <8> in the figure correspond to <1> to <8> in Figure 11-69 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode).

2. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00), mn = 00



Figure 11-69. Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)



**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

**Remark** <1> to <8> in the figure correspond to <1> to <8> in Figure 11-68 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode).

### 11.5.7 SNOOZE mode function (CSI00)

SNOOZE mode makes CSI operate reception by SCKp pin input detection while the STOP mode. Normally CSI stops communication in the STOP mode. But, using the SNOOZE mode makes reception CSI operate unless the CPU operation by detecting SCKp pin input.

<R> When using the CSI in SNOOZE mode, make the following setting before switching to the STOP mode (see **Figure 11-71** and **Figure 11-73 Flowchart of SNOOZE Mode Operation**).

- When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode. After the initial setting has been completed, set the SSm1 bit of serial channel start register m (SSm) to 1.

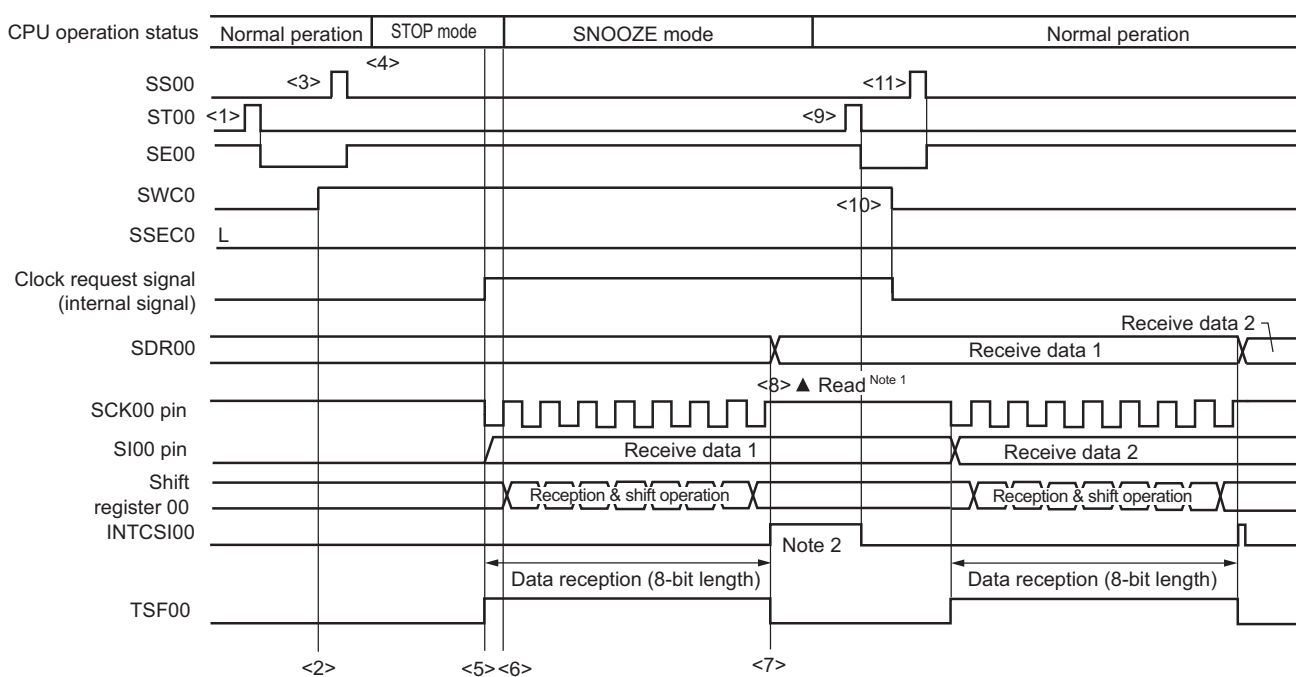
After a transition to the STOP mode, the CSI starts reception operations upon detection of an edge of the SCKp pin.

**Cautions 1.** The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for f<sub>CLK</sub>.

**2.** The maximum transfer rate when using CSIp in the SNOOZE mode is 1 Mbps.

#### (1) SNOOZE mode operation (once startup)

**Figure 11-70. Timing Chart of SNOOZE Mode Operation (Once Startup) (Type 1: DAPmn = 0, CKPmn = 0)**



- Notes**
- Only read received data while SWCm = 1 and before the next edge of the SCKp pin input is detected.
  - The transfer end interrupt (INTCSIp) is cleared either when SWCm is cleared to 0 or when the next edge of the SCKp pin input is detected.

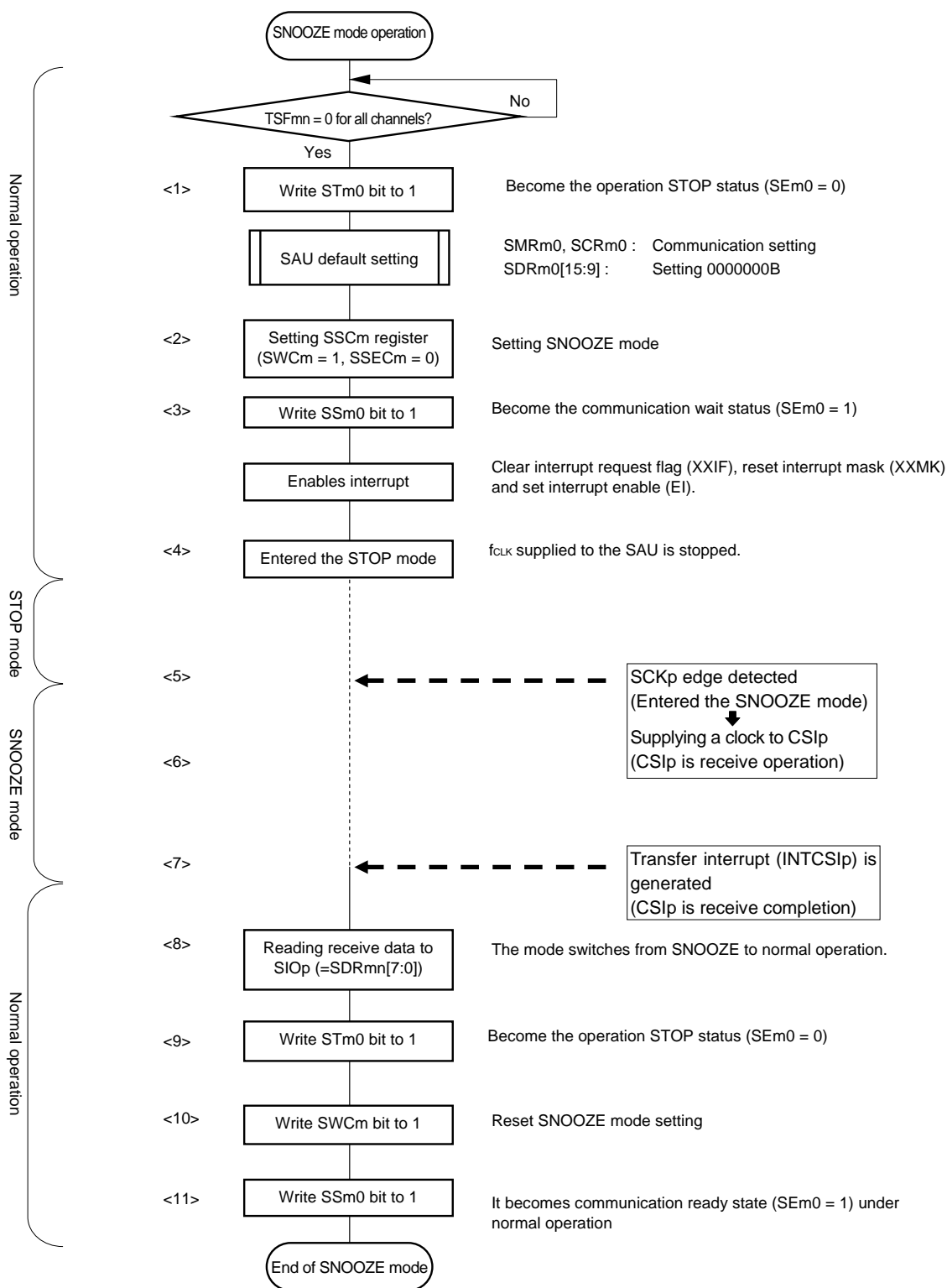
**Caution** Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEM0 bit, and stop the operation).

And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in **Figure 11-71 Flowchart of SNOOZE Mode Operation (Once Startup)**.

**2.** m = 0; p = 00

Figure 11-71. Flowchart of SNOOZE Mode Operation (Once Startup)

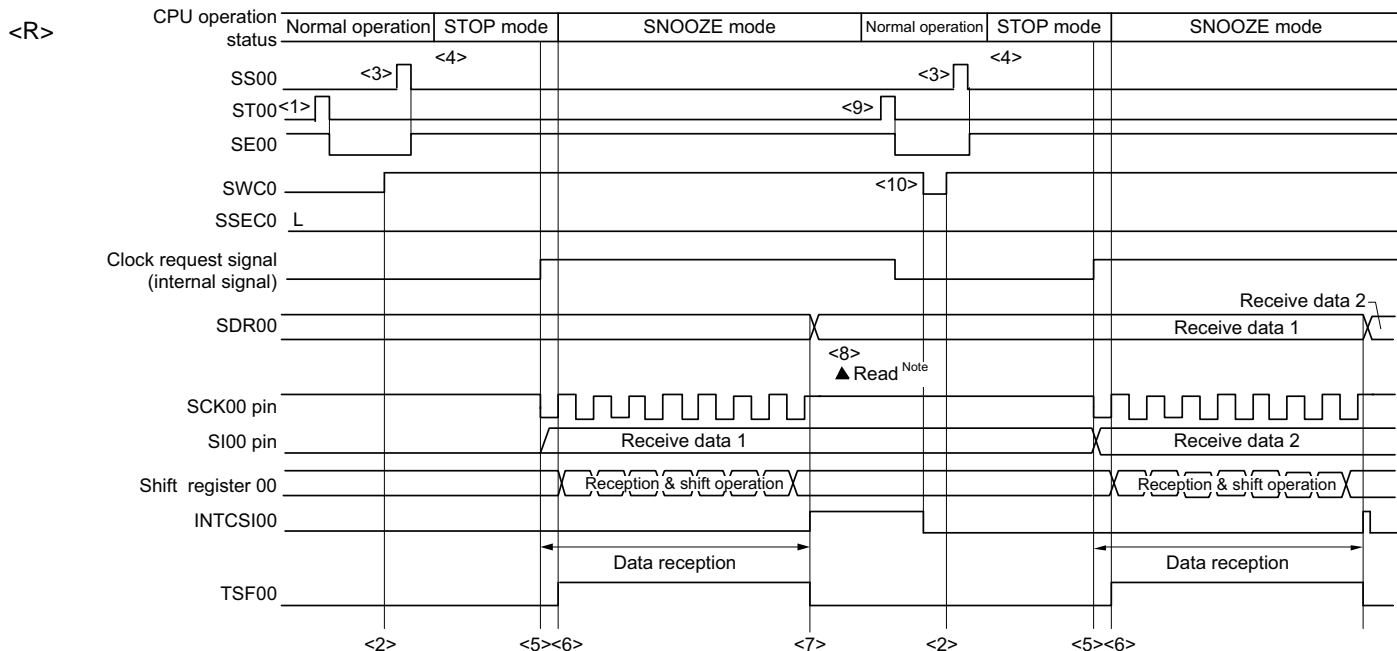


Remarks 1. <1> to <11> in the figure correspond to <1> to <11> in Figure 11-70 Timing Chart of SNOOZE Mode Operation (Once Startup).

2. m = 0; p = 00

(2) SNOOZE mode operation (continuous startup)

Figure 11-72. Timing Chart of SNOOZE Mode Operation (Continuous Startup) (Type 1: DAPmn = 0, CKPmn = 0)



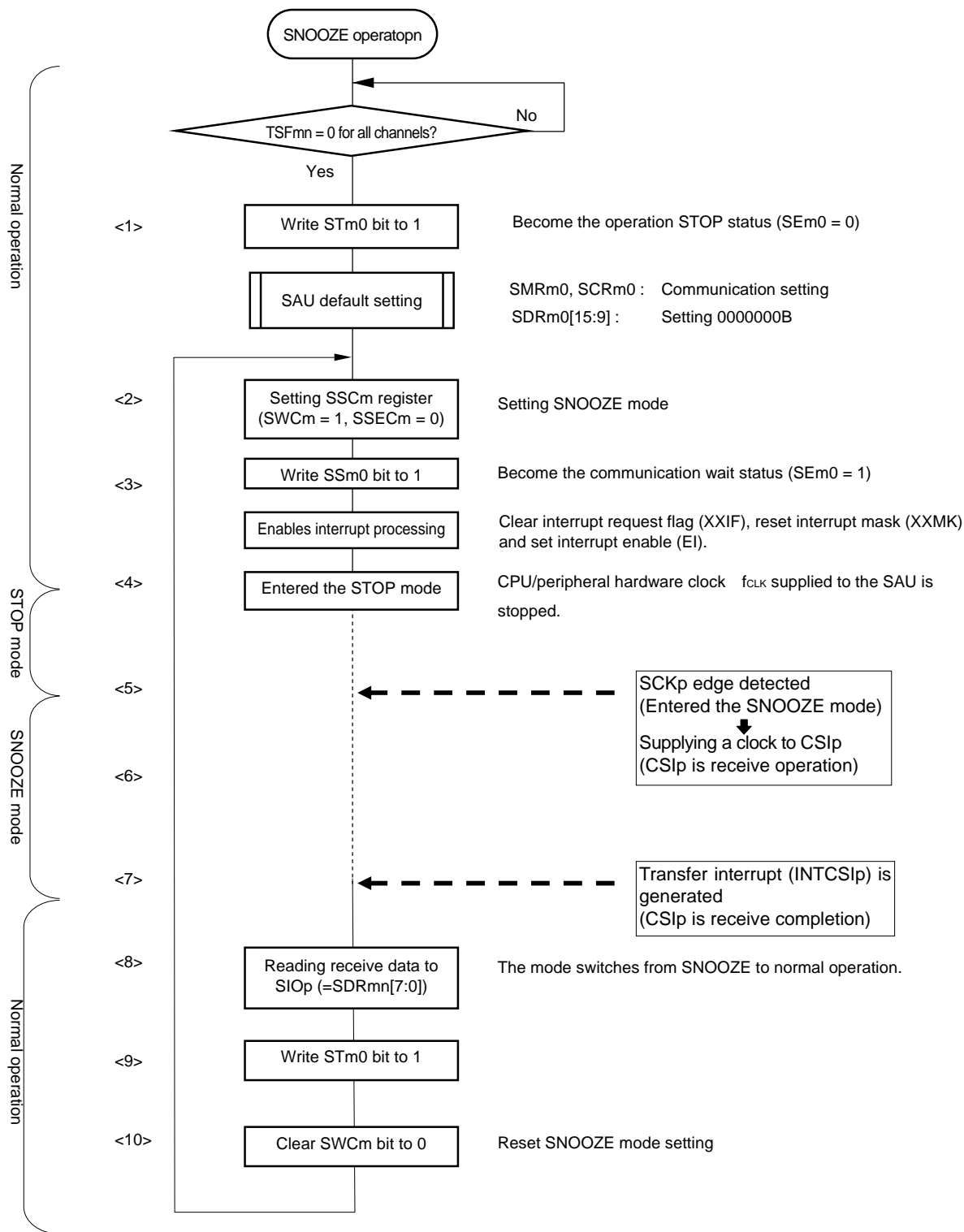
**Note** Only read received data while SWCm = 1 and before the next edge of the SCKp pin input is detected.

- Cautions**
1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm0 bit to 1 (clear the SEm0 bit, and stop the operation). And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE release).
  2. When SWCm = 1, the BFFm1 and OVFM1 flags will not change.

- Remarks**
1. <1> to <10> in the figure correspond to <1> to <10> in Figure 11-73 Flowchart of SNOOZE Mode Operation (Continuous Startup).
  2. m = 0; p = 00

Figure 11-73. Flowchart of SNOOZE Mode Operation (Continuous Startup)

<R>



### 11.5.8 Calculating transfer clock frequency

The transfer clock frequency for 3-wire serial I/O (CSI00) communication can be calculated by the following expressions.

#### (1) Master

$$\text{(Transfer clock frequency)} = \{\text{Operation clock (f}_{\text{MCK}}\text{) frequency of target channel}\} \div (\text{SDRmn}[15:9] + 1) \div 2 \text{ [Hz]}$$

#### (2) Slave

$$\text{(Transfer clock frequency)} = \{\text{Frequency of serial clock (SCK) supplied by master}\}^{\text{Note}} \text{ [Hz]}$$

**Note** The permissible maximum transfer clock frequency is  $f_{\text{MCK}}/6$ .

**Remark** The value of SDRmn[15:9] is the value of bits 15 to 9 of serial data register mn (SDRmn) (0000000B to 1111111B) and therefore is 0 to 127.

The operation clock ( $f_{\text{MCK}}$ ) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

Table 11-2. Selection of Operation Clock for 3-Wire Serial I/O

SMRmn Register	SPSm Register								Operation Clock ( $f_{CLK}$ ) <sup>Note</sup>	
	CKSmn	PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00	$f_{CLK} = 32 \text{ MHz}$
0	X	X	X	X	0	0	0	0	$f_{CLK}$	32 MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16 MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	8 MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	4 MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	2 MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	1 MHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	500 kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	250 kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	125 kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	62.5 kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25 kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63 kHz
	X	X	X	X	1	1	0	0	$f_{CLK}/2^{12}$	7.81 kHz
	X	X	X	X	1	1	0	1	$f_{CLK}/2^{13}$	3.91 kHz
	X	X	X	X	1	1	1	0	$f_{CLK}/2^{14}$	1.95 kHz
X	X	X	X	1	1	1	1	$f_{CLK}/2^{15}$	977 Hz	
1	0	0	0	0	X	X	X	X	$f_{CLK}$	32 MHz
	0	0	0	1	X	X	X	X	$f_{CLK}/2$	16 MHz
	0	0	1	0	X	X	X	X	$f_{CLK}/2^2$	8 MHz
	0	0	1	1	X	X	X	X	$f_{CLK}/2^3$	4 MHz
	0	1	0	0	X	X	X	X	$f_{CLK}/2^4$	2 MHz
	0	1	0	1	X	X	X	X	$f_{CLK}/2^5$	1 MHz
	0	1	1	0	X	X	X	X	$f_{CLK}/2^6$	500 kHz
	0	1	1	1	X	X	X	X	$f_{CLK}/2^7$	250 kHz
	1	0	0	0	X	X	X	X	$f_{CLK}/2^8$	125 kHz
	1	0	0	1	X	X	X	X	$f_{CLK}/2^9$	62.5 kHz
	1	0	1	0	X	X	X	X	$f_{CLK}/2^{10}$	31.25 kHz
	1	0	1	1	X	X	X	X	$f_{CLK}/2^{11}$	15.63 kHz
	1	1	0	0	X	X	X	X	$f_{CLK}/2^{12}$	7.81 kHz
	1	1	0	1	X	X	X	X	$f_{CLK}/2^{13}$	3.91 kHz
	1	1	1	0	X	X	X	X	$f_{CLK}/2^{14}$	1.95 kHz
1	1	1	1	X	X	X	X	$f_{CLK}/2^{15}$	977 Hz	
Other than above									Setting prohibited	

**Note** When changing the clock selected for  $f_{CLK}$  (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

### 11.5.9 Procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication

The procedure for processing errors that occurred during 3-wire serial I/O (CSI00) communication is described in Figure 11-72.

**Figure 11-74. Processing Procedure in Case of Overrun Error**

Software Manipulation	Hardware Status	Remark
Reads serial data register mn (SDRmn).	→ The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data.	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		Error type is identified and the read value is used to clear error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn).	→ Error flag is cleared.	Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), mn = 00



## 11.6 Clock Synchronous Serial Communication with Slave Select Input Function

Channel 0 of SAU0 correspond to the clock synchronous serial communication with slave select input function.

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable
- Level setting of transmit/receive data

[Clock control]

- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel
- Maximum transfer rate <sup>Note</sup>

During slave communication: Max.  $f_{MCK}/6$

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

[Expansion function]

- Slave select function

**Note** Use this operation within a range that satisfies the conditions above and the AC characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

Unit	Channel	Used as CSI	Used as UART
0	0	CSI00 (supporting slave select input function)	UART0
	1	-	

Slave select input function performs the following three types of communication operations.

- Slave transmission (See **11.6.1.**)
- Slave reception (See **11.6.2.**)
- Slave transmission/reception (See **11.6.3.**)

Multiple slaves can be connected to a master and communication can be performed by using the slave select input function. The master outputs a slave select signal to the slave (one) that is the other party of communication, and each slave judges whether it has been selected as the other party of communication and controls the SO pin output. When a slave is selected, transmit data can be communicated from the SO pin to the master. When a slave is not selected, the SO pin is set to high-impedance output. Therefore, short circuit of the SO pin with SO output of other slave can be avoided. Furthermore, when a slave is not selected, no transmission/reception operation is performed even if a serial clock is input from the master.

**Caution** Output the slave select signal by port manipulation.

**Figure 11-75. Example of Slave Select Input Function Configuration**

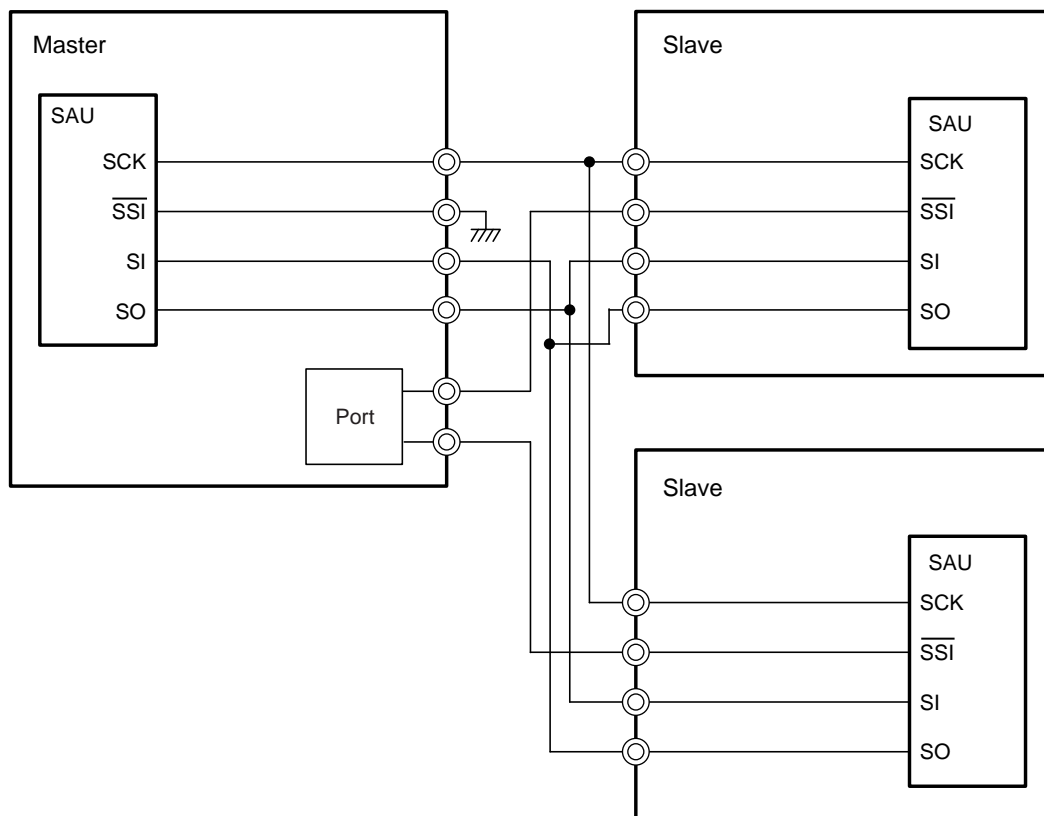
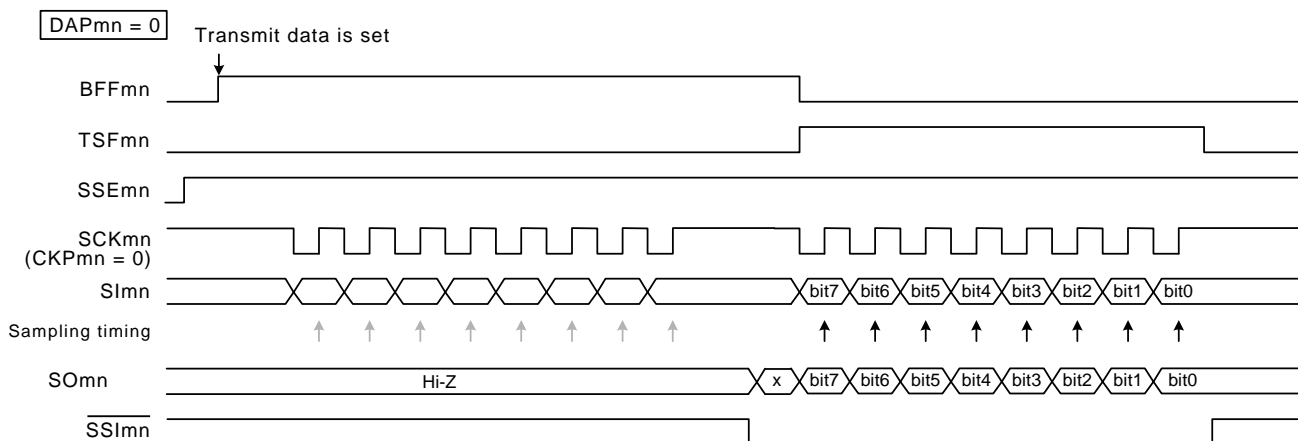
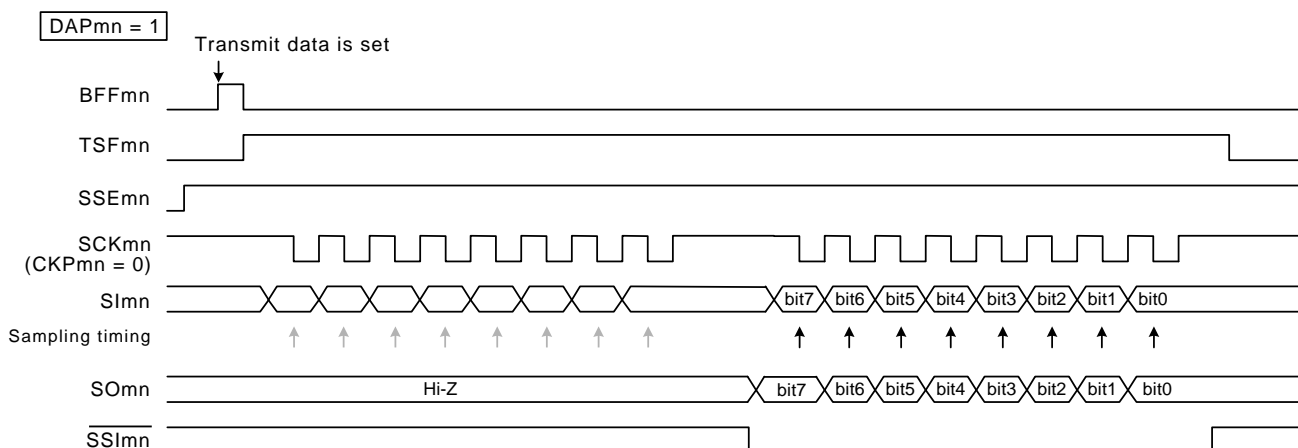


Figure 11-76. Slave Select Input Function Timing Diagram



While  $\overline{SSImn}$  is at high level, transmission is not performed even if the falling edge of  $\overline{SCKmn}$  (serial clock) arrives, and neither is receive data sampled in synchronization with the rising edge. When  $\overline{SSImn}$  goes to low level, data is output (shifted) in synchronization with the falling edge of the serial clock and a reception operation is performed in synchronization with the rising edge.



If  $DAPmn = 1$ , when transmit data is set while  $\overline{SSImn}$  is at high level, the first data (bit 7) is output to the data output. However, no shift operation is performed even if the rising edge of  $\overline{SCKmn}$  (serial clock) arrives, and neither is receive data sampled in synchronization with the falling edge. When  $\overline{SSImn}$  goes to low level, data is output (shifted) in synchronization with the next rising edge and a reception operation is performed in synchronization with the falling edge.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0)

### 11.6.1 Slave transmission

Slave transmission is that the R7F0C010 transmit data to another device in the state of a transfer clock being input from another device.

Slave Select Input Function	CSI00
Target channel	Channel 0 of SAU0
Pins used	$\overline{\text{SCK00}}$ , $\text{SO00}$ , $\overline{\text{SSI00}}$
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{\text{MCK}}/6$ [Hz] <sup>Notes 1, 2</sup> .
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data output starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data output starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first
Slave select Input function	Slave select input function operation selectable

**Notes 1.** Because the external serial clock input to the  $\overline{\text{SCK00}}$  pin is sampled internally and used, the fastest transfer rate is  $f_{\text{MCK}}/6$  [Hz].

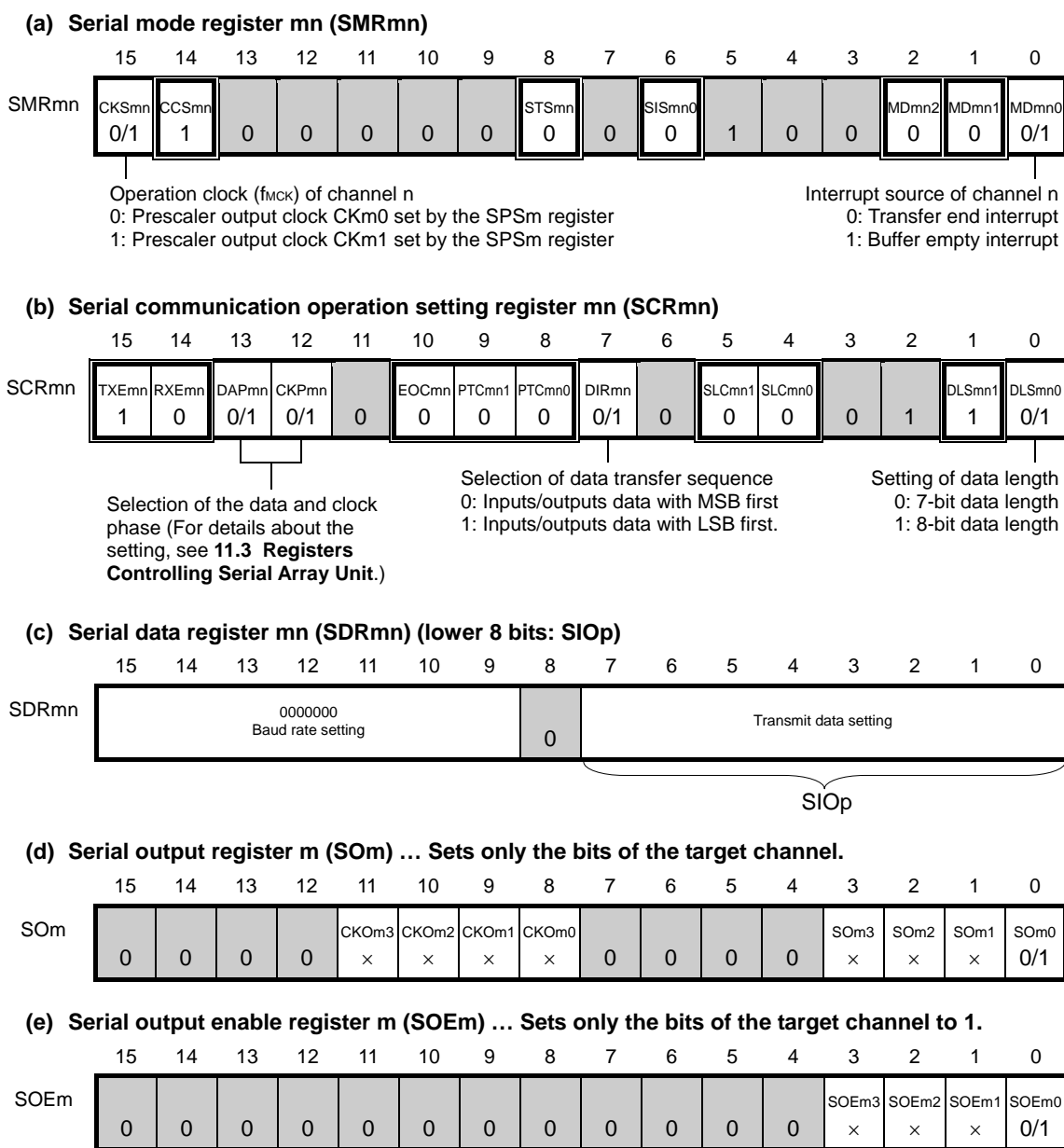
**2.** Use this operation within a range that satisfies the conditions above and the AC characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{\text{MCK}}$ : Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0)

(1) Register setting

Figure 11-77. Example of Contents of Registers for Slave Transmission of Slave Select Input Function (CSI00) (1/2)



- Remarks 1. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)
2. □: Setting is fixed in the CSI slave transmission mode, ■: Setting disabled (set to the initial value)  
 x: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

**Figure 11-77. Example of Contents of Registers for Slave Transmission of Slave Select Input Function (CSI00) (2/2)**

(f) **Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3	SSm2	SSm1	SSm0
													x	x	x	0/1

(g) **Input switch control register (ISC) ... SSI00 input setting in CSI00 slave channel (channel 0 of unit 0).**

	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

0: Disables the input value of the SSI00 pin

1: Enables the input value of the SSI00 pin

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

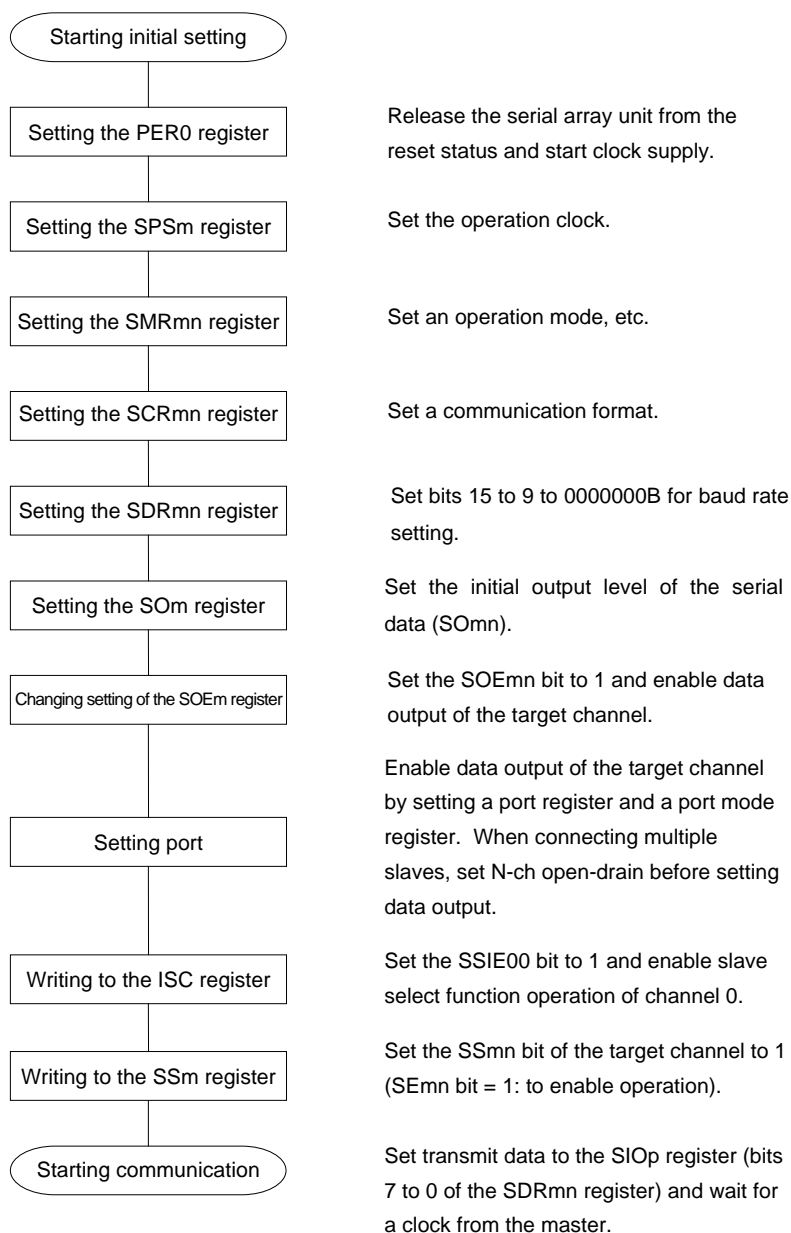
**2.**  : Setting disabled (set to the initial value)

x: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user

(2) Operation procedure

Figure 11-78. Initial Setting Procedure for Slave Transmission

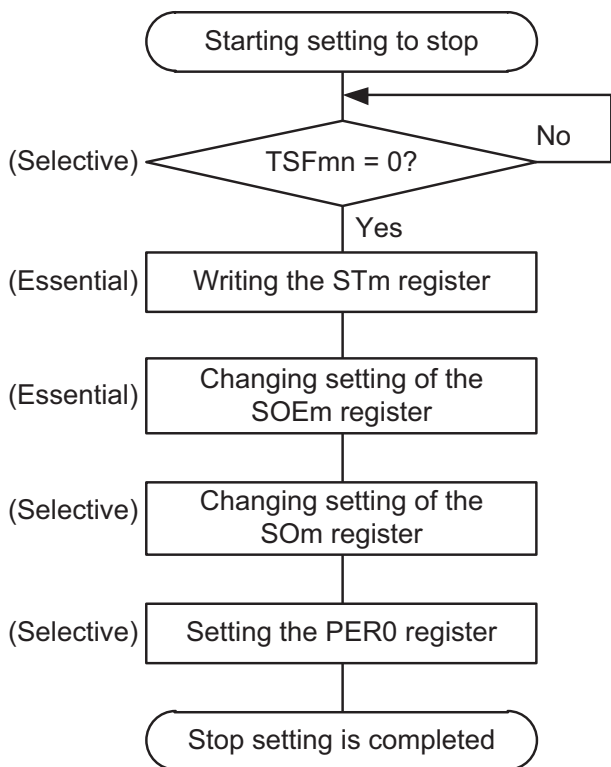


<R>

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-79. Procedure for Stopping Slave Transmission

<R>



If there is any data being transferred, wait for their completion.  
(If there is an urgent must stop, do not wait.)

Write 1 to the STmn bit of the target channel  
(SEmn = 0: to operation stop status).

Set the SOEmn bit to 0 and stop the output of the target channel.

The levels of the serial data (SOMn) on the target channel can be changed if necessitated by an emergency.

Reset the serial array unit by stopping the clock supply to it.

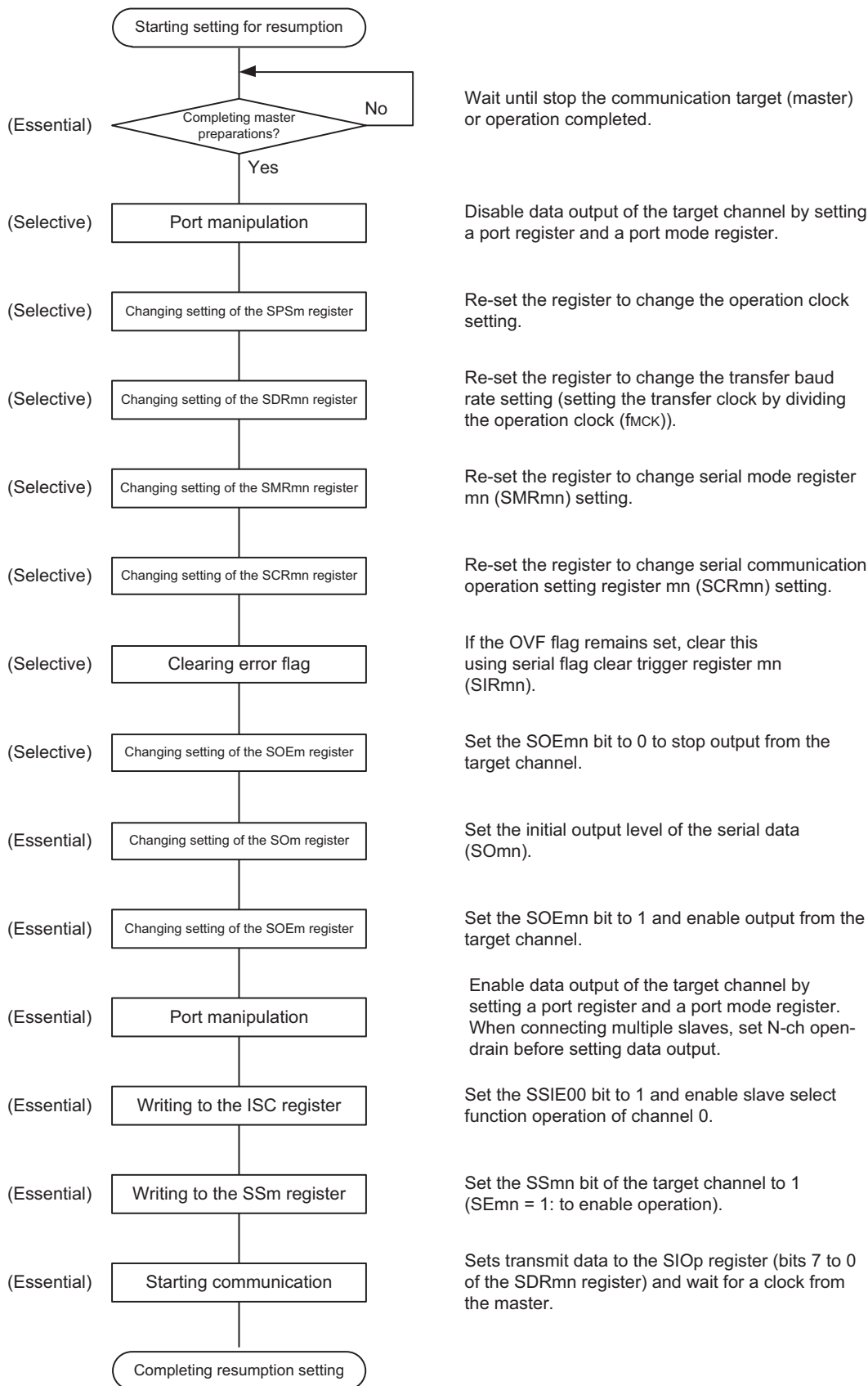
After the stop setting is completed, go to the next processing.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00).



Figure 11-80. Procedure for Resuming Slave Transmission

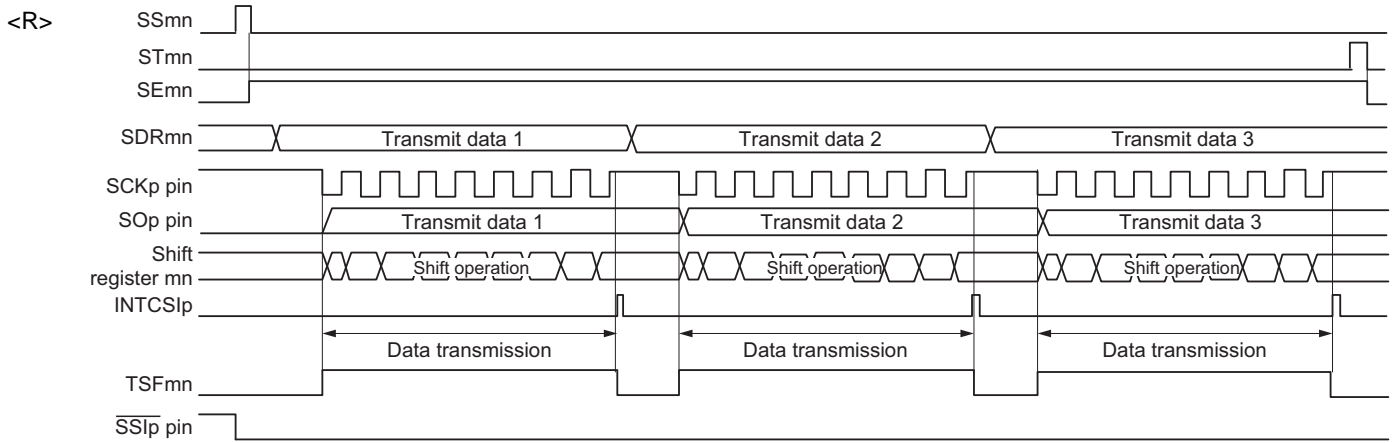
<R>



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

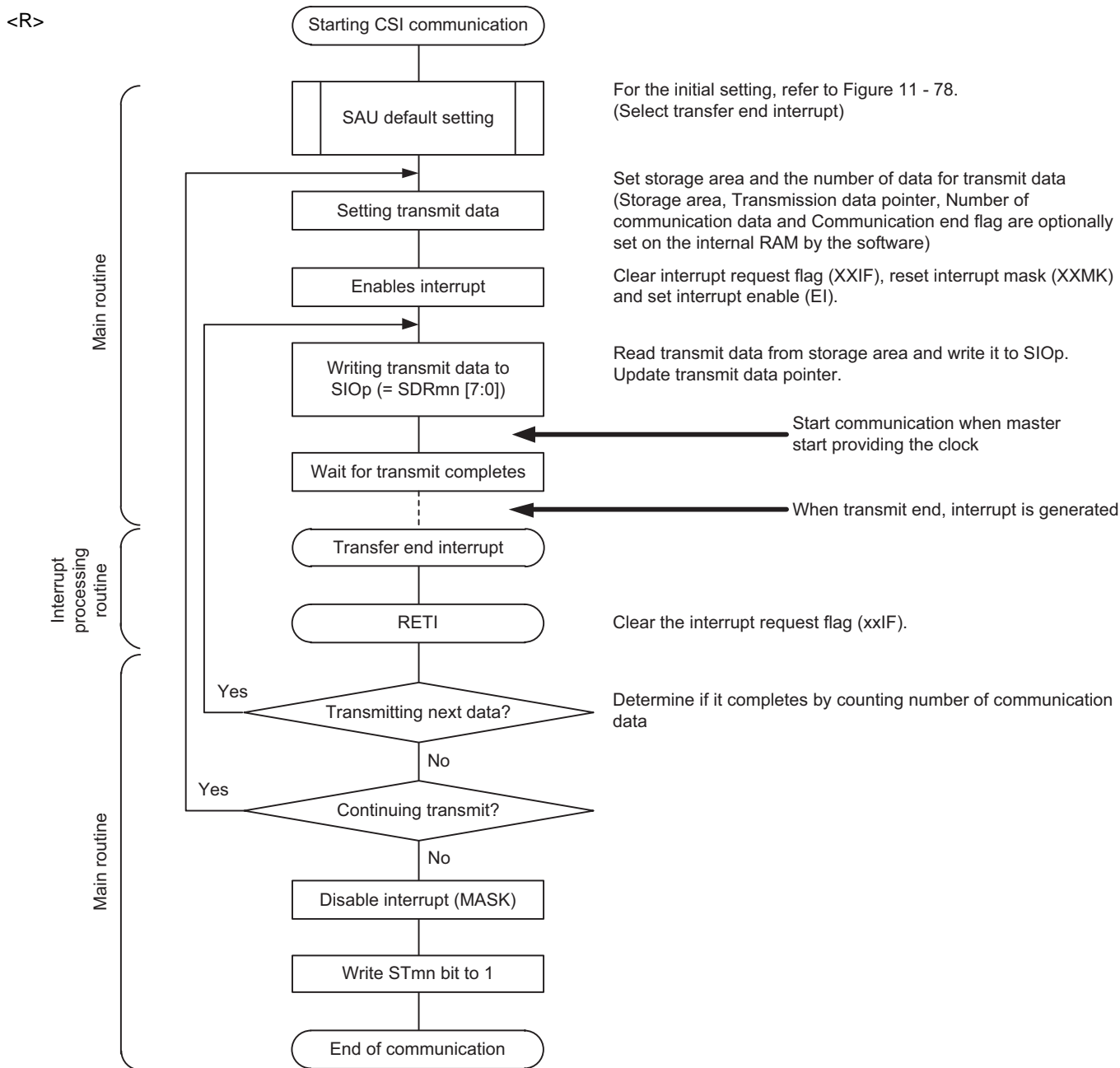
(3) Processing flow (in single-transmission mode)

Figure 11-81. Timing Chart of Slave Transmission (in Single-Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

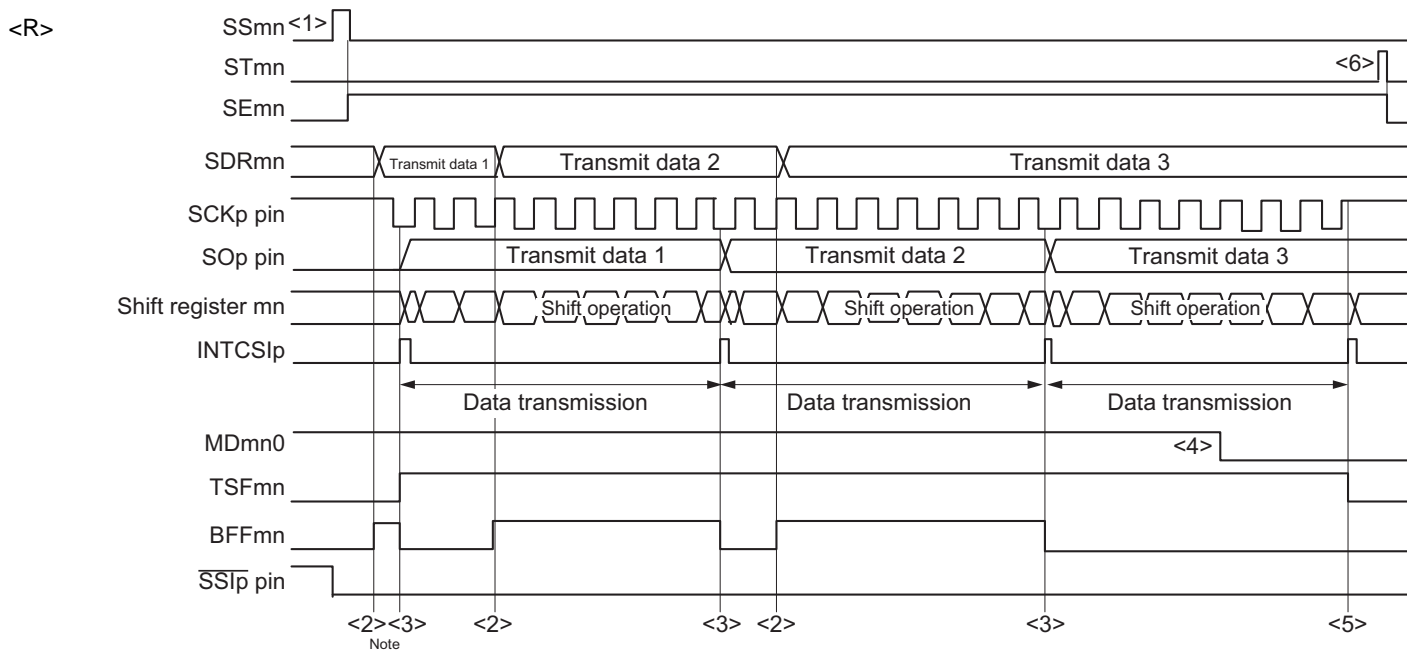
Figure 11-82. Flowchart of Slave Transmission (in Single-Transmission Mode)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

(4) Processing flow (in continuous transmission mode)

Figure 11-83. Timing Chart of Slave Transmission (in Continuous Transmission Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)

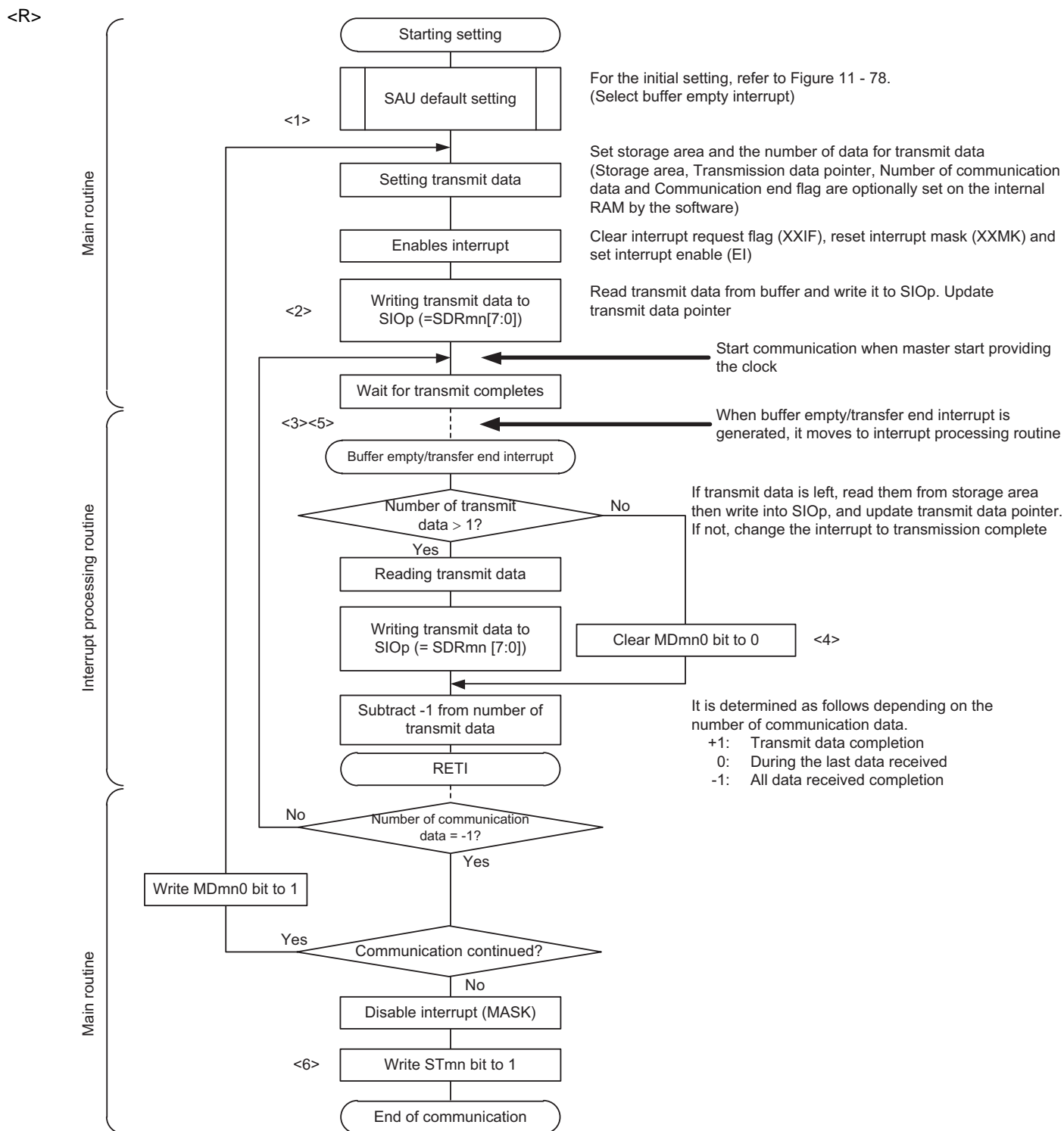


**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-84. Flowchart of Slave Transmission (in Continuous Transmission Mode)



Remarks 1. <1> to <6> in the figure correspond to <1> to <6> in Figure 11-83 Timing Chart of Slave Transmission (in Continuous Transmission Mode).

2. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

### 11.6.2 Slave reception

&lt;R&gt;

Slave reception is that the R7F0C010 data from another device in the state of a transfer clock being input from another device.

Slave Select Input Function	CSI00
Target channel	Channel 0 of SAU0
Pins used	$\overline{SCK00}$ , SI00, $\overline{SSI00}$
Interrupt	INTCSI00
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{MCK}/6$ [Hz] <sup>Notes 1, 2</sup>
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data input starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data input starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first
slave select input function	Slave select input function operation selectable

**Notes 1.** Because the external serial clock input to the  $\overline{SCK00}$  pin is sampled internally and used, the fastest transfer rate is  $f_{MCK}/6$  [Hz].

&lt;R&gt;

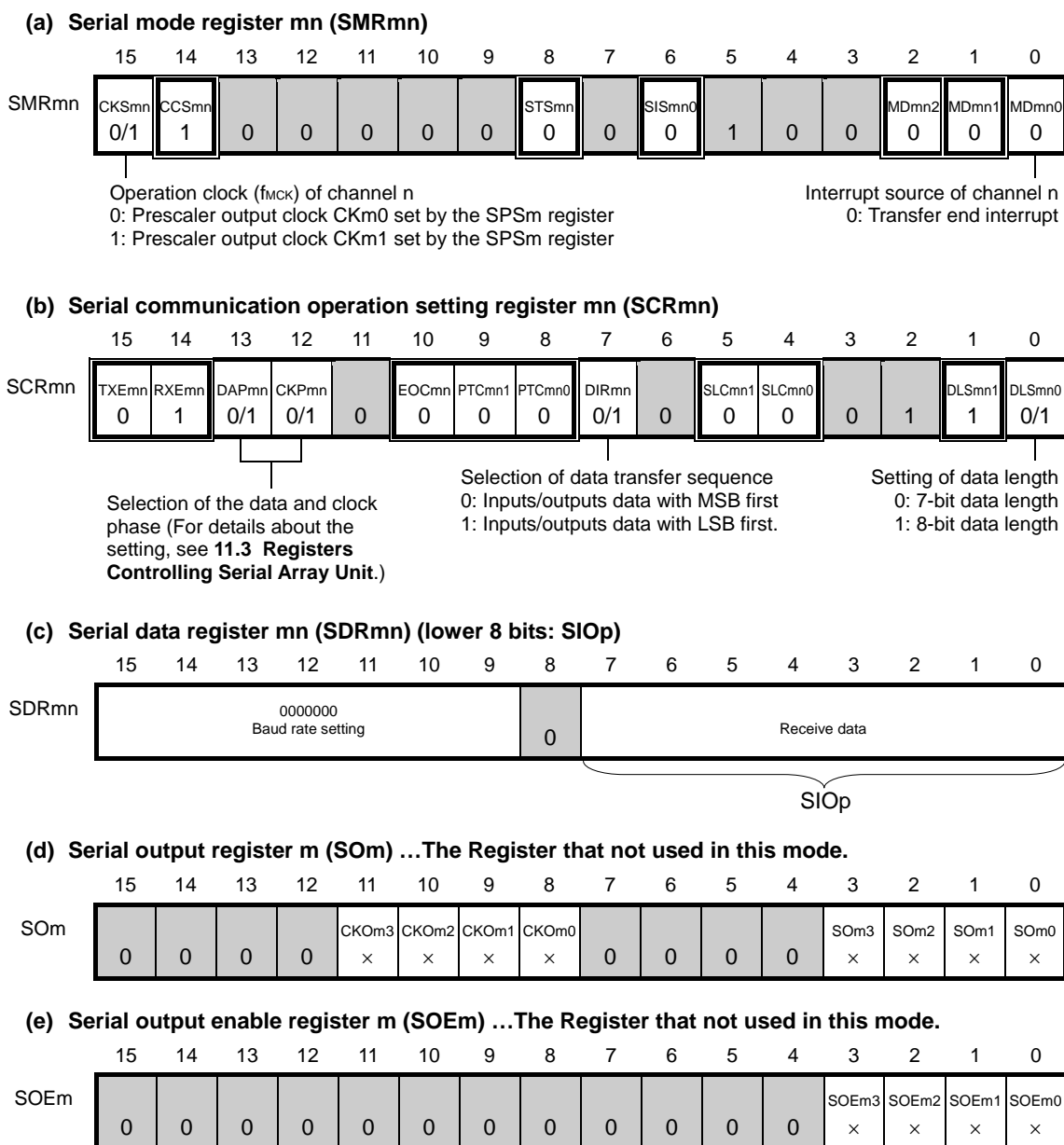
**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{MCK}$ : Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0)

(1) Register setting

Figure 11-85. Example of Contents of Registers for Slave Reception of Slave Select Input Function (CSI00) (1/2)



- Remarks 1. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)
2. □: Setting is fixed in the CSI slave reception mode, □: Setting disabled (set to the initial value)  
 x: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

Figure 11-85. Example of Contents of Registers for Slave Reception of Slave Select Input Function (CSI00) (2/2)

(f) Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3 ×	SSm2 ×	SSm1 ×	SSm0 0/1

(g) Input switch control register (ISC) ...  $\overline{\text{SSI00}}$  input setting in CSI00 slave channel (channel 0 of unit 0).

	7	6	5	4	3	2	1	0
ISC	SSIE00 0/1	0	0	0	0	0	ISC1 0/1	ISC0 0/1

0: Disables the input value of the  $\overline{\text{SSI00}}$  pin

1: Enables the input value of the  $\overline{\text{SSI00}}$  pin

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

**2.**  : Setting disabled (set to the initial value)

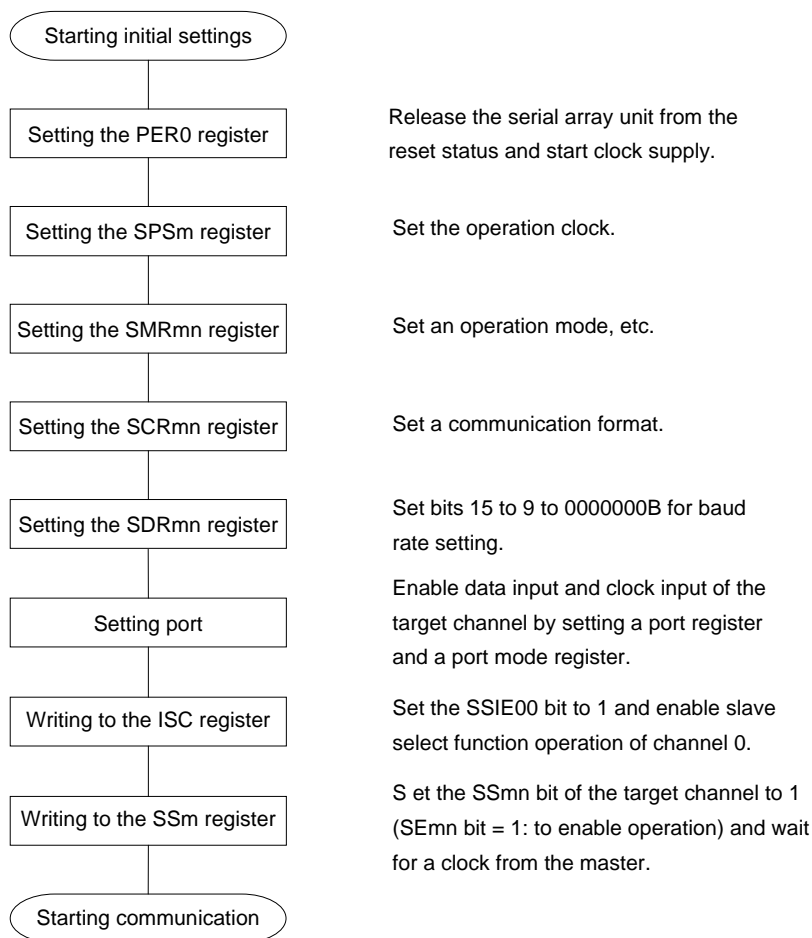
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user



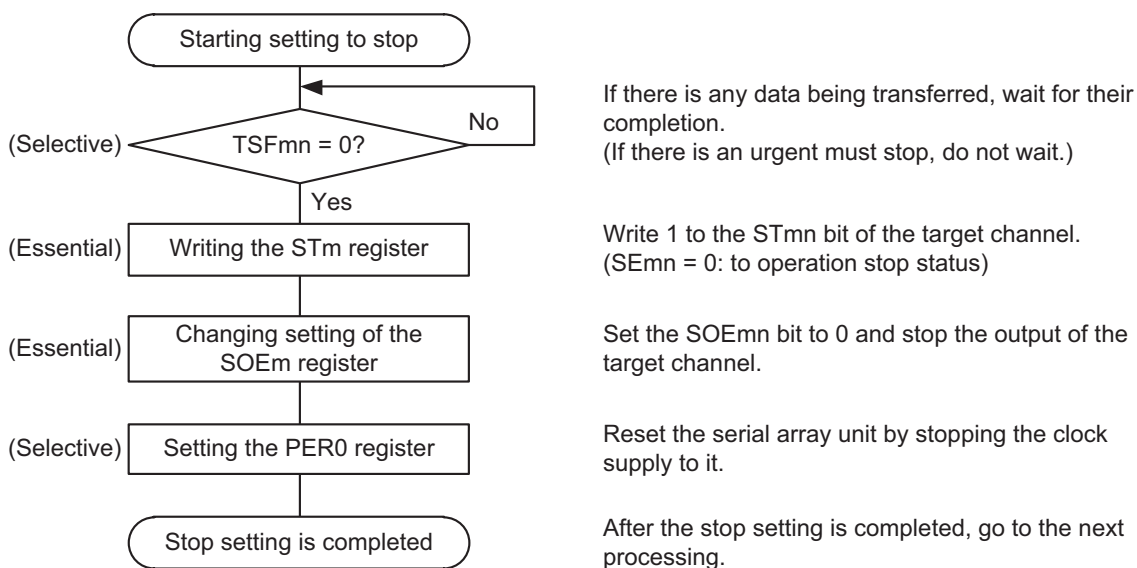
(2) Operation procedure

Figure 11-86. Initial Setting Procedure for Slave Reception



<R>

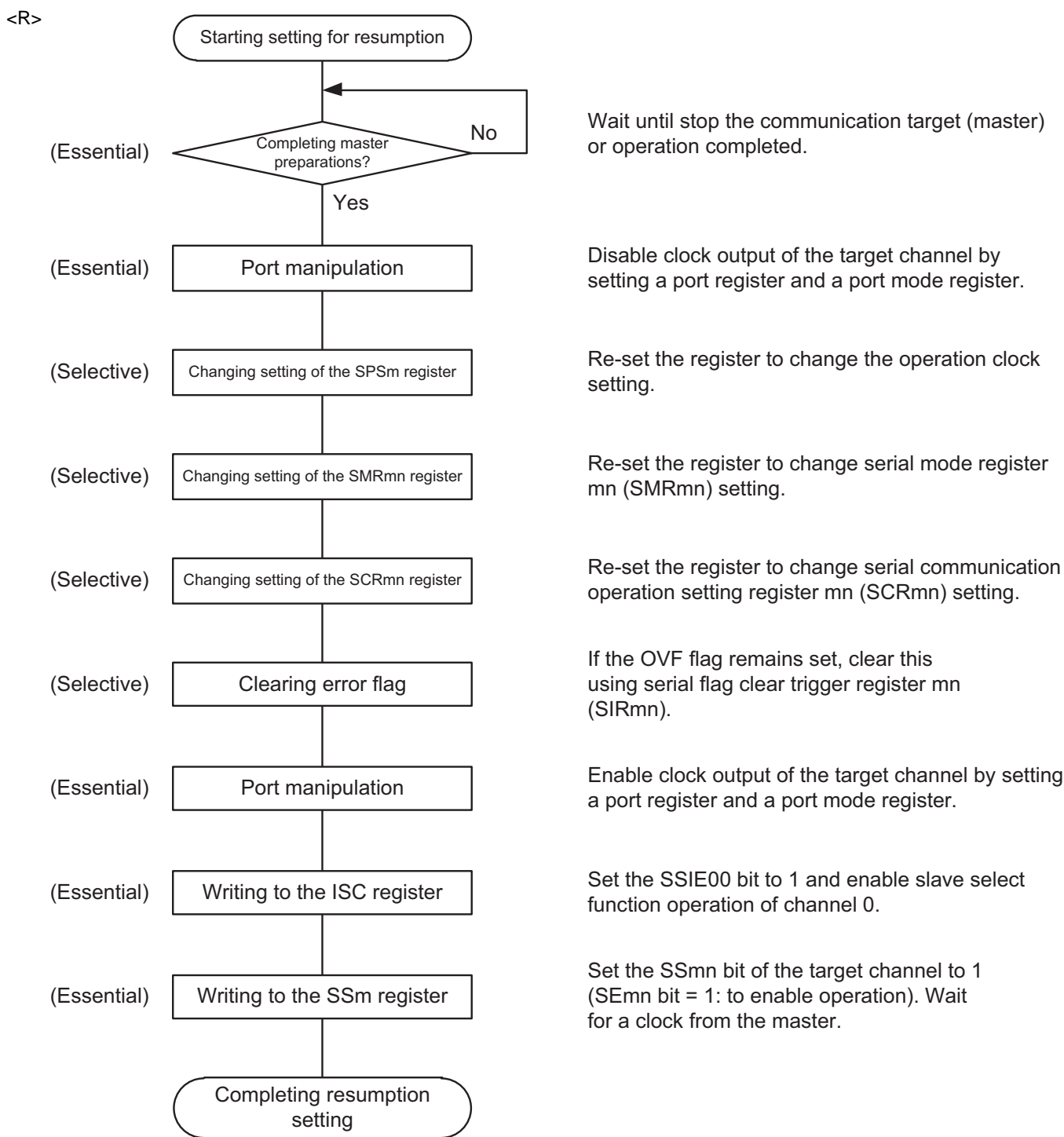
Figure 11-87. Procedure for Stopping Slave Reception



<R>

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

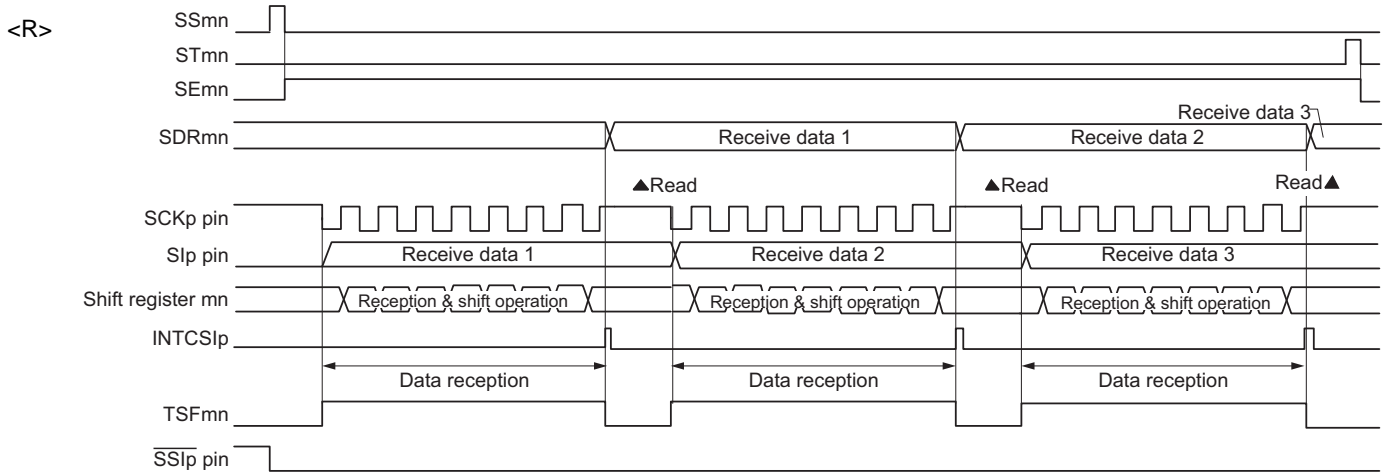
Figure 11-88. Procedure for Resuming Slave Reception



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

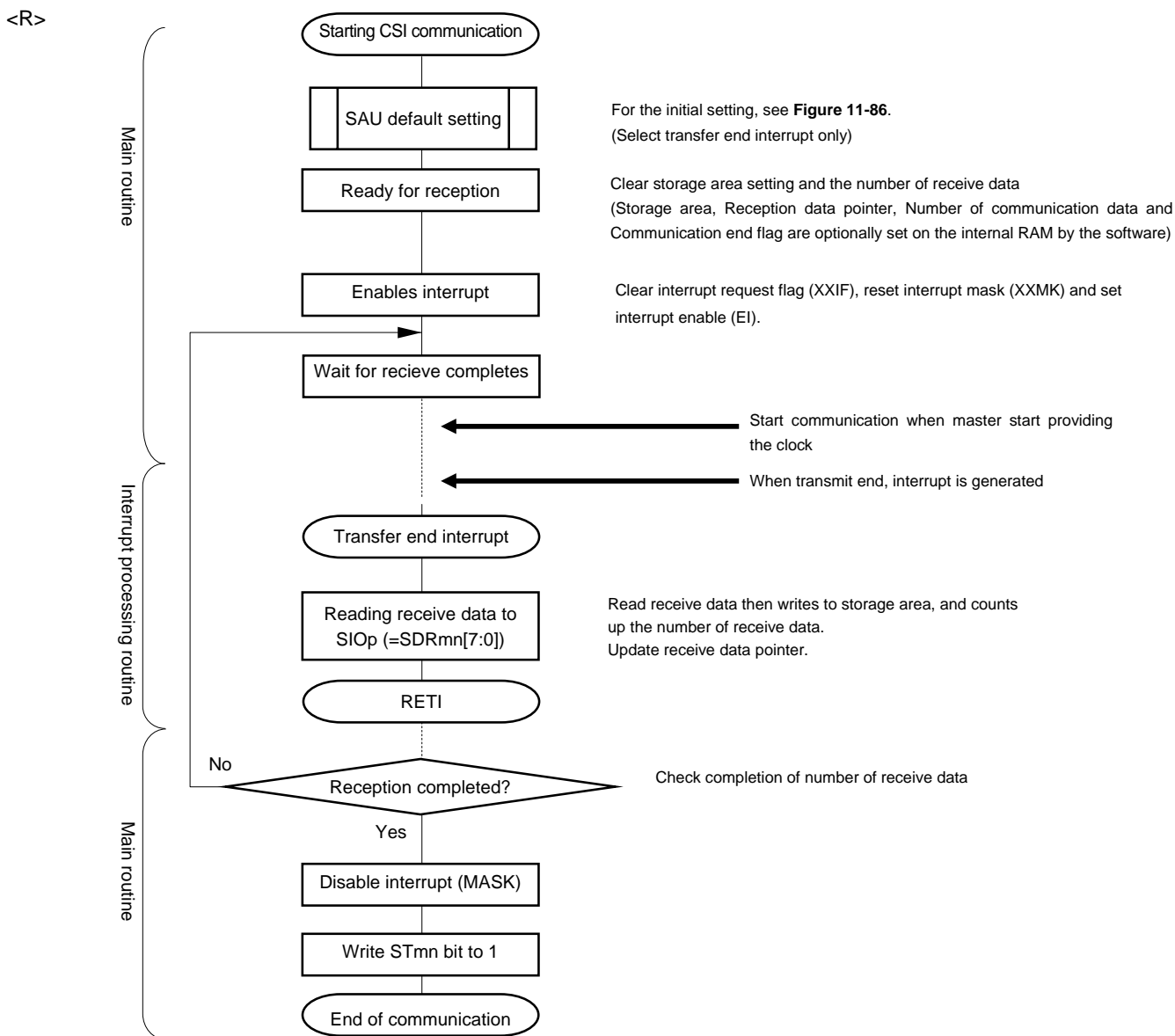
(3) Processing flow (in single-reception mode)

Figure 11-89. Timing Chart of Slave Reception (in Single-Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-90. Flowchart of Slave Reception (in Single-Reception Mode)



### 11.6.3 Slave transmission/reception

Slave transmission/reception is that the R7F0C010 transmit/receive data to/from another device in the state of a transfer clock being input from another device.

Slave Select Input Function	CSI00
Target channel	Channel 0 of SAU0
Pins used	$\overline{\text{SCK00}}$ , SI00, SO00, $\overline{\text{SSI00}}$
Interrupt	INTCSI00
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	Overrun error detection flag (OVFmn) only
Transfer data length	7 or 8 bits
Transfer rate	Max. $f_{\text{MCK}}/6$ [Hz] <sup>Notes 1, 2</sup> .
Data phase	Selectable by the DAPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• DAPmn = 0: Data I/O starts from the start of the operation of the serial clock.</li> <li>• DAPmn = 1: Data I/O starts half a clock before the start of the serial clock operation.</li> </ul>
Clock phase	Selectable by the CKPmn bit of the SCRmn register <ul style="list-style-type: none"> <li>• CKPmn = 0: Non-reverse</li> <li>• CKPmn = 1: Reverse</li> </ul>
Data direction	MSB or LSB first
Slave select input function	Slave select input function operation selectable

**Notes 1.** Because the external serial clock input to the  $\overline{\text{SCK00}}$  pin is sampled internally and used, the fastest transfer rate is  $f_{\text{MCK}}/6$  [Hz].

<R>

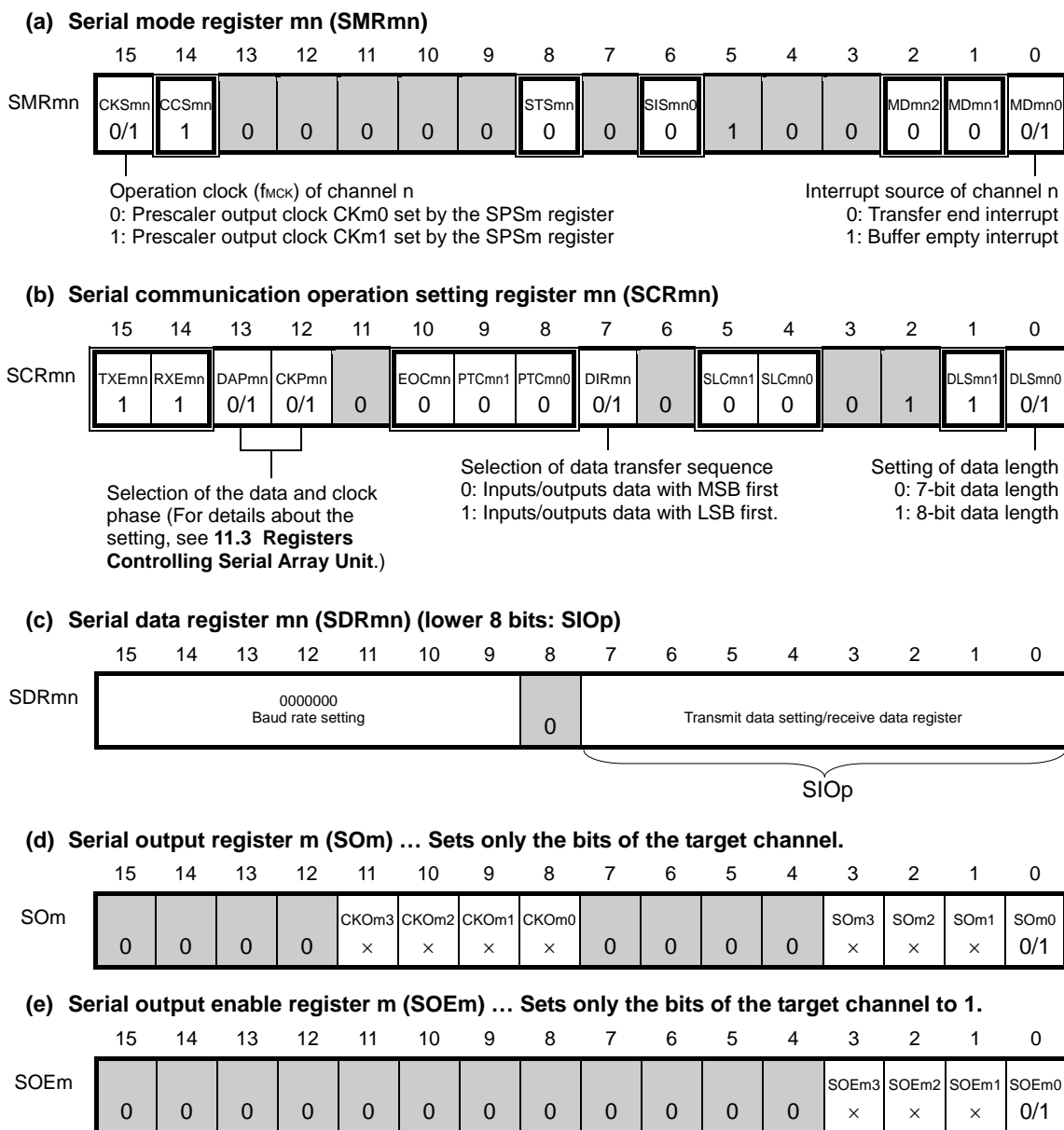
**2.** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristic in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

**Remarks 1.**  $f_{\text{MCK}}$ : Operation clock frequency of target channel

**2.** m: Unit number (m = 0), n: Channel number (n = 0)

(1) Register setting

Figure 11-91. Example of Contents of Registers for Slave Transmission/Reception of Slave Select Input Function (CSI00) (1/2)



**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)
  - |  |   |
|--|---|
| <span style="border: 1px solid black; padding: 0 2px;"> </span>                            | : Setting is fixed in the CSI slave transmission/reception mode                             |
| <span style="background-color: #cccccc; border: 1px solid black; padding: 0 2px;"> </span> | : Setting disabled (set to the initial value)   |
| x  | : Bit that cannot be used in this mode (set to the initial value when not used in any mode) |
| 0/1  | : Set to 0 or 1 depending on the usage of the user  |

**Figure 11-91. Example of Contents of Registers for Slave Transmission/Reception of Slave Select Input Function (CSI00) (2/2)**

(f) **Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	SSm3	SSm2	SSm1	SSm0
													×	×	×	0/1

(g) **Input switch control register (ISC) ... SSI00 input setting in CSI00 slave channel (channel 0 of unit 0).**

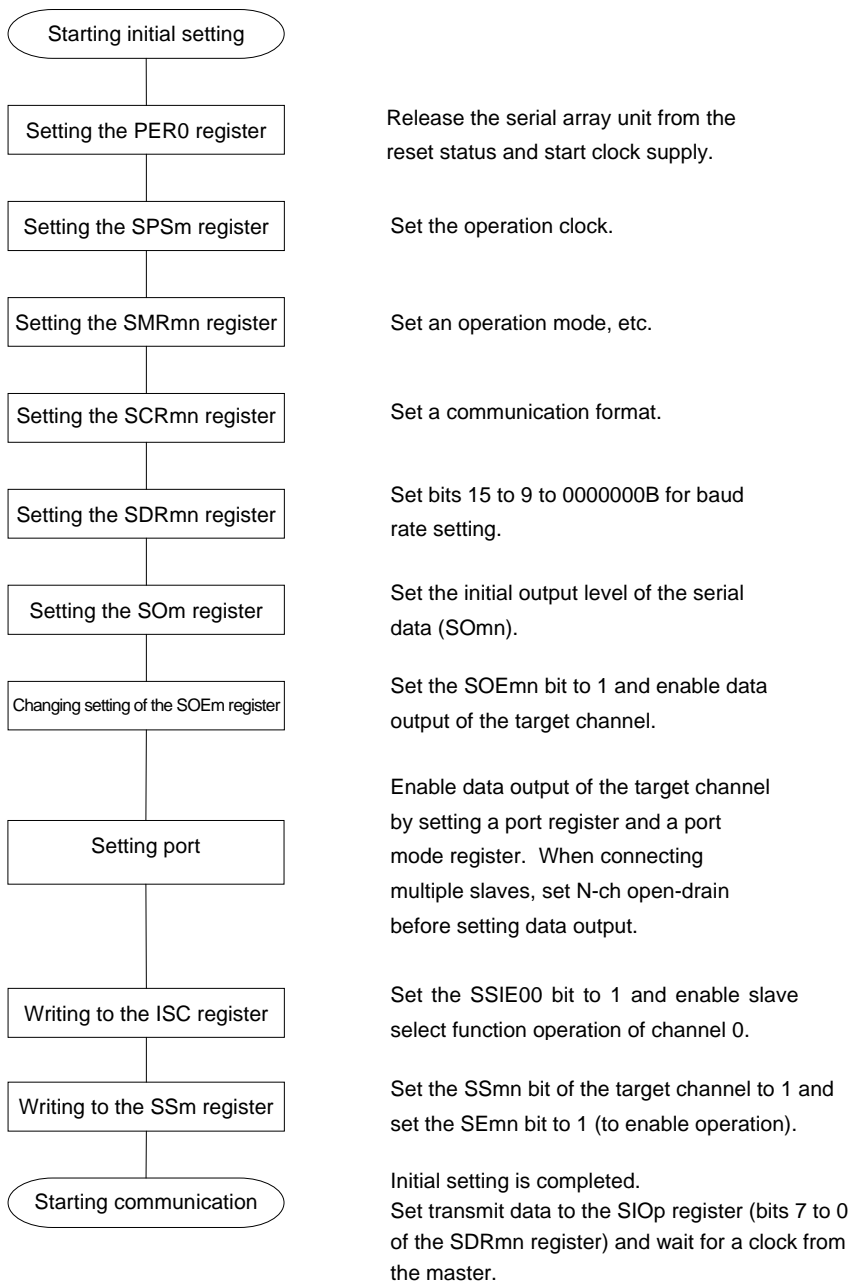
	7	6	5	4	3	2	1	0
ISC	SSIE00						ISC1	ISC0
	0/1	0	0	0	0	0	0/1	0/1

- 0: Disables the input value of the SSI00 pin
- 1: Enables the input value of the SSI00 pin

- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)
  2.  : Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

(2) Operation procedure

Figure 11-92. Initial Setting Procedure for Slave Transmission/Reception



<R>

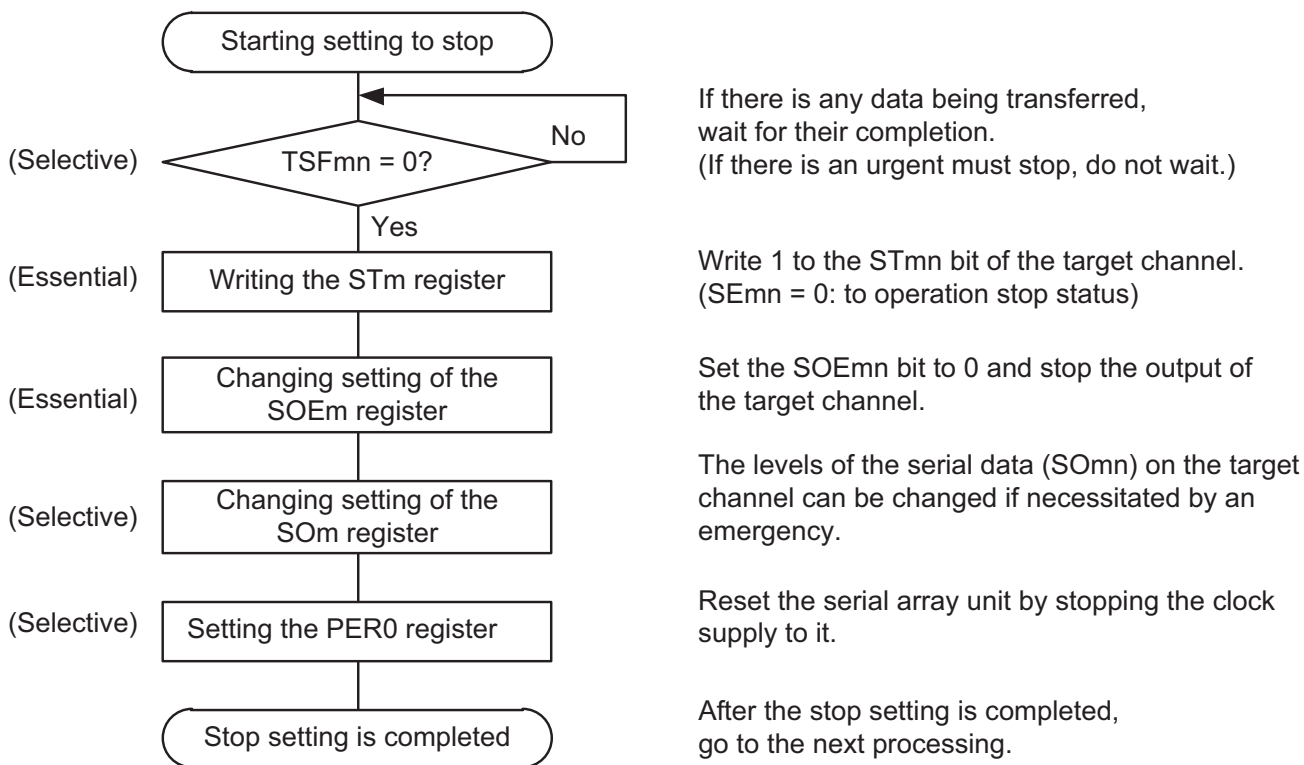
**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)



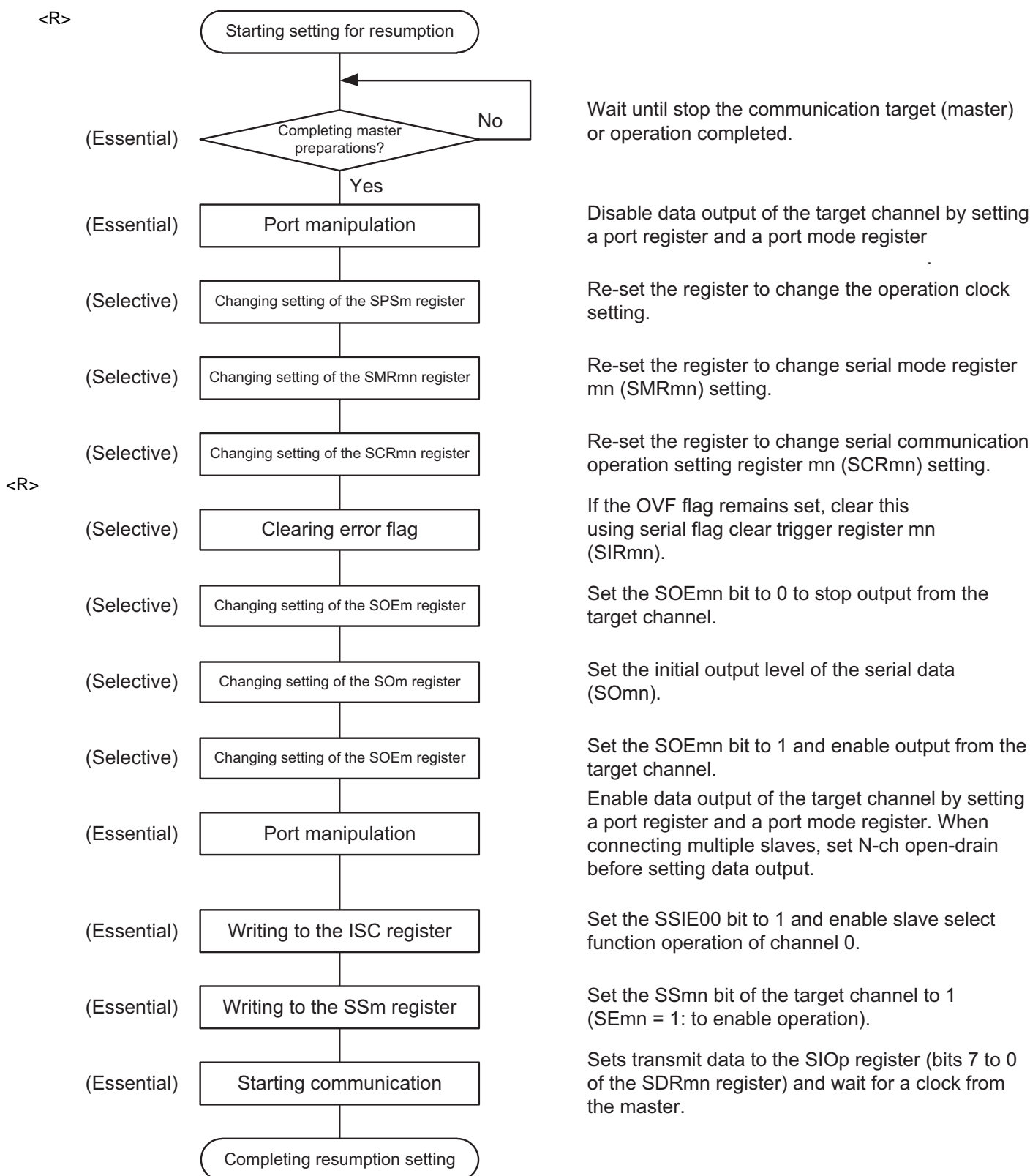
Figure 11-93. Procedure for Stopping Slave Transmission/Reception

<R>



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-94. Procedure for Resuming Slave Transmission/Reception

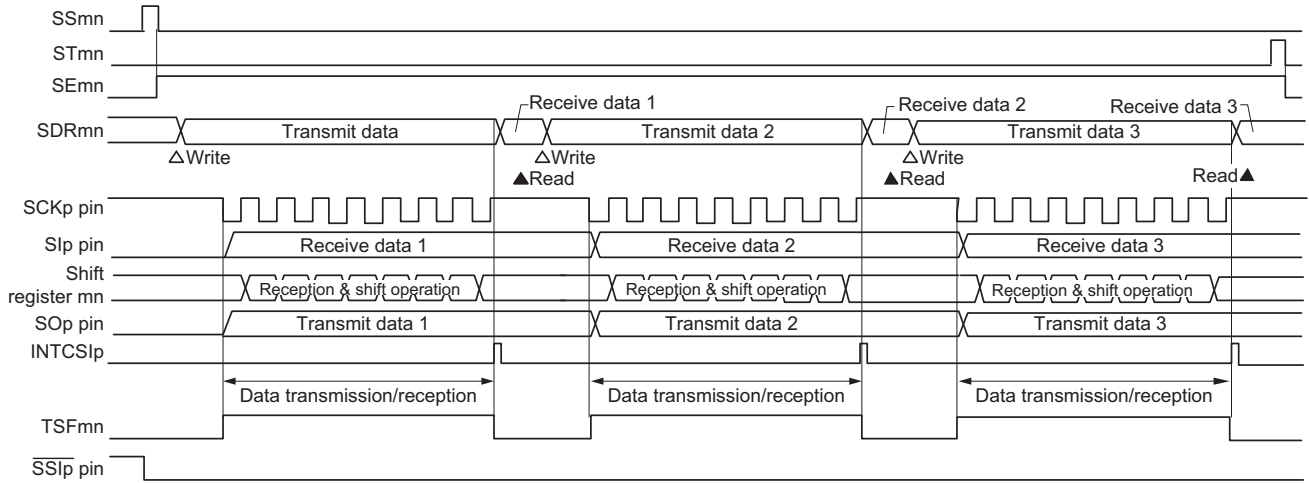


- <R>
- Cautions**
1. Be sure to set transmit data to the SIOp register before the clock from the master is started.
  2. If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (master) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

(3) Processing flow (in single-transmission/reception mode)

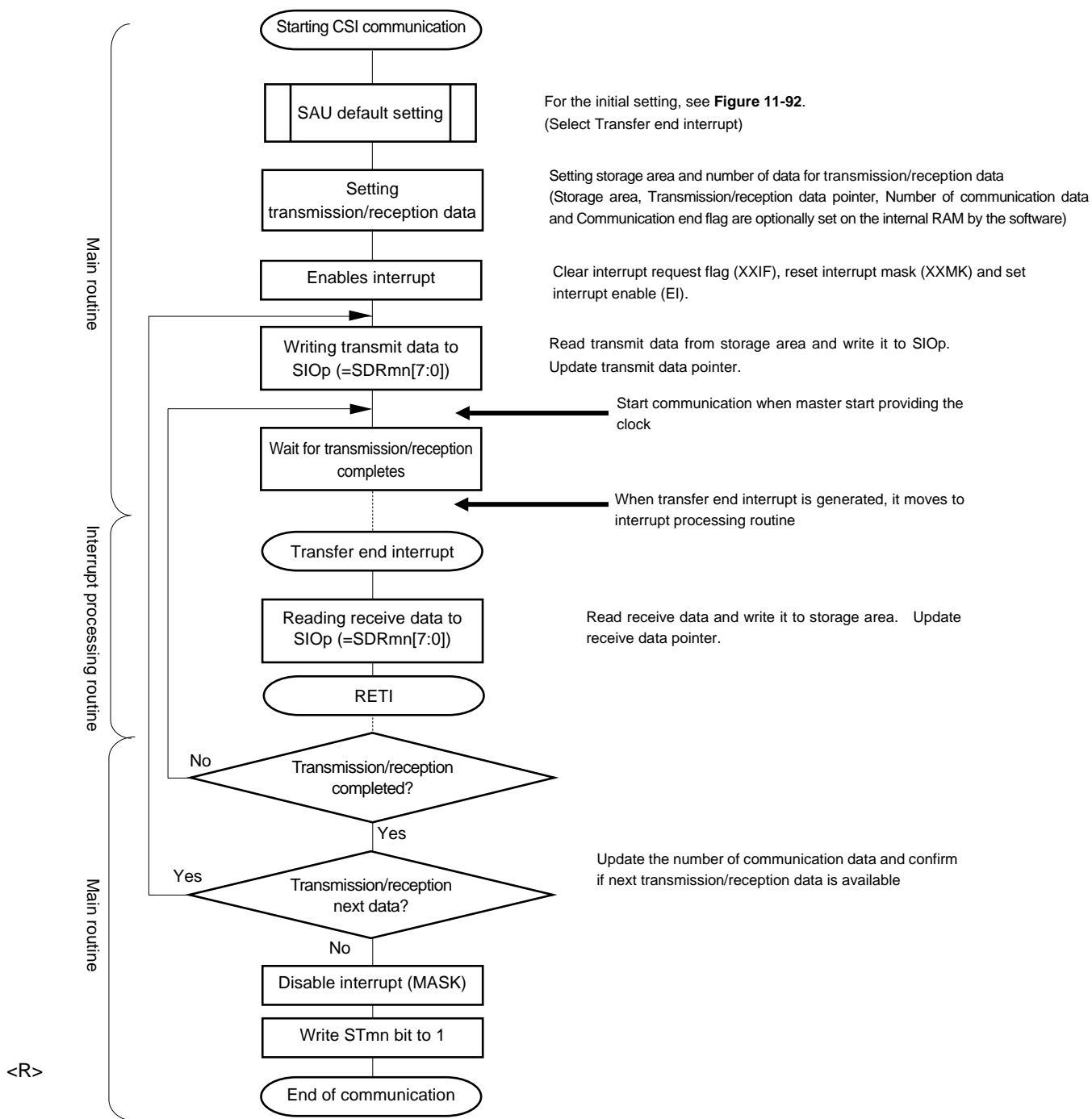
Figure 11-95. Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)

<R>



**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-96. Flowchart of Slave Transmission/Reception (in Single- Transmission/Reception Mode)

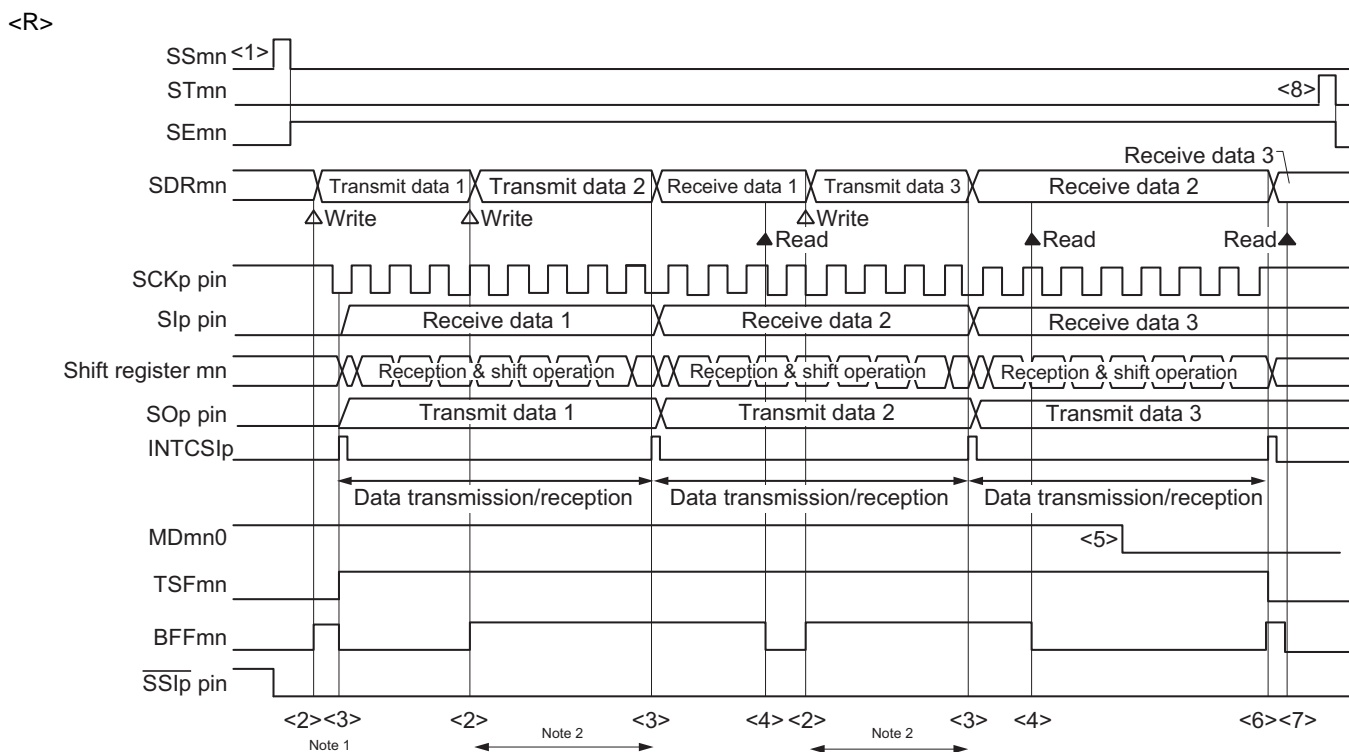


**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

(4) Processing flow (in continuous transmission/reception mode)

Figure 11-97. Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)  
(Type 1: DAPmn = 0, CKPmn = 0)

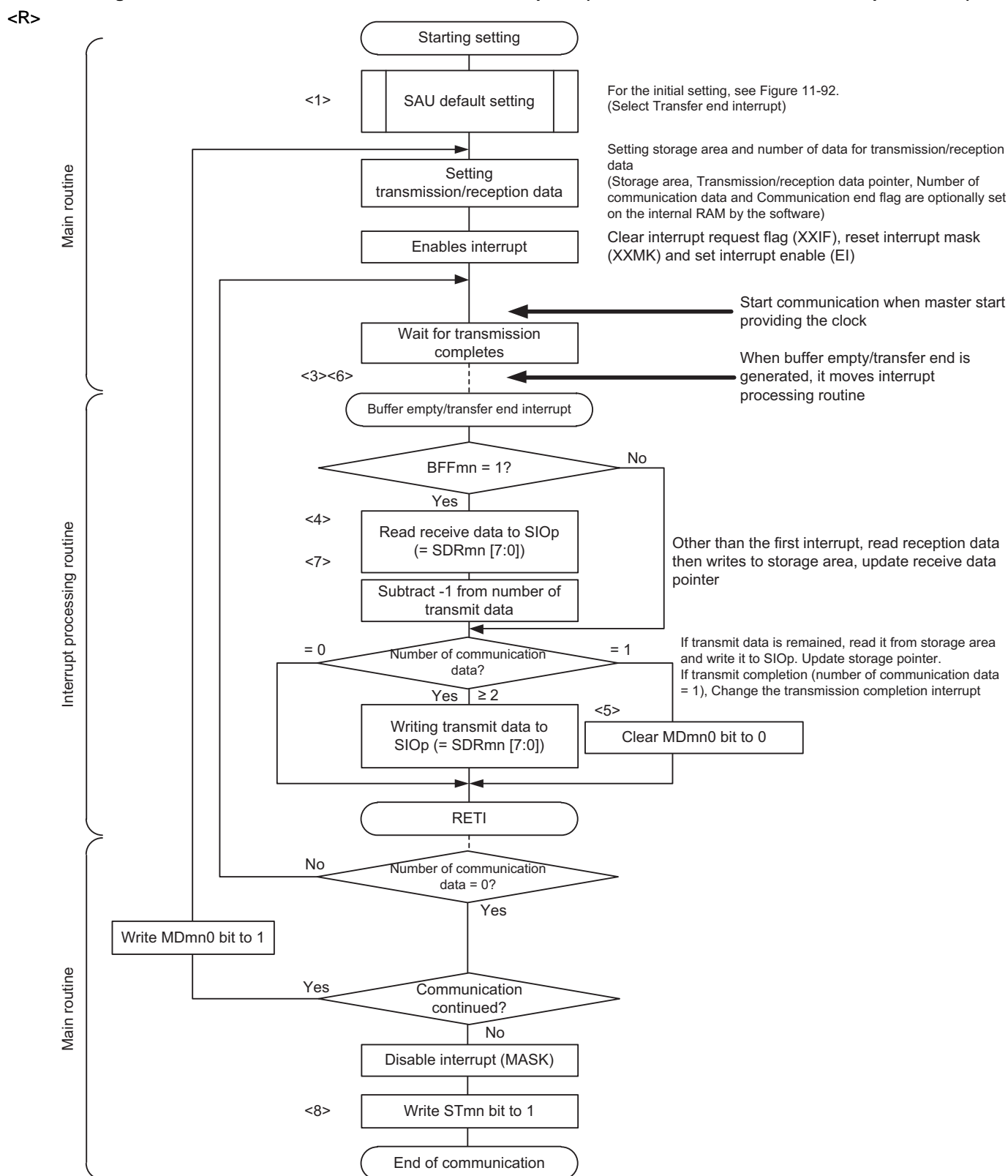


- Notes**
1. If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.
  2. The transmit data can be read by reading the SDRmn register during this period. At this time, the transfer operation is not affected.

**Caution** The MDmn0 bit of serial mode register mn (SMRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it has been rewritten before the transfer end interrupt of the last transmit data.

- Remarks**
1. <1> to <8> in the figure correspond to <1> to <8> in Figure 11-96 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode).
  2. m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Figure 11-98. Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)



**Caution** Be sure to set transmit data to the SIOp register before the clock from the master is started.

**Remarks 1.** <1> to <8> in the figure correspond to <1> to <8> in Figure 11-97 Timing Chart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode).

**2.** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

#### 11.6.4 Calculating transfer clock frequency

The transfer clock frequency for slave select input function (CSI00) communication can be calculated by the following expressions.

<R>

##### (1) Slave

(Transfer clock frequency) = {Frequency of serial clock ( $\overline{SCK}$ ) supplied by master} <sup>Note</sup>	[Hz]
--	------

**Note** The permissible maximum transfer clock frequency is  $f_{MCK}/6$ .

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), p: CSI number (p = 00)

Table 11-3. Selection of Operation Clock for Slave Select Input Function

SMRmn Register	SPSm Register								Operation Clock ( $f_{CLK}$ ) <sup>Note</sup>	
	CKSmn	PRS m13	PRS m12	PRS m11	PRS m10	PRS m03	PRS m02	PRS m01	PRS m00	$f_{CLK} = 32$ MHz
0	X	X	X	X	0	0	0	0	$f_{CLK}$	32 MHz
	X	X	X	X	0	0	0	1	$f_{CLK}/2$	16 MHz
	X	X	X	X	0	0	1	0	$f_{CLK}/2^2$	8 MHz
	X	X	X	X	0	0	1	1	$f_{CLK}/2^3$	4 MHz
	X	X	X	X	0	1	0	0	$f_{CLK}/2^4$	2 MHz
	X	X	X	X	0	1	0	1	$f_{CLK}/2^5$	1 MHz
	X	X	X	X	0	1	1	0	$f_{CLK}/2^6$	500 kHz
	X	X	X	X	0	1	1	1	$f_{CLK}/2^7$	250 kHz
	X	X	X	X	1	0	0	0	$f_{CLK}/2^8$	125 kHz
	X	X	X	X	1	0	0	1	$f_{CLK}/2^9$	62.5 kHz
	X	X	X	X	1	0	1	0	$f_{CLK}/2^{10}$	31.25 kHz
	X	X	X	X	1	0	1	1	$f_{CLK}/2^{11}$	15.63 kHz
	X	X	X	X	1	1	0	0	$f_{CLK}/2^{12}$	7.81 kHz
	X	X	X	X	1	1	0	1	$f_{CLK}/2^{13}$	3.9 kHz
	X	X	X	X	1	1	1	0	$f_{CLK}/2^{14}$	1.95 kHz
X	X	X	X	1	1	1	1	$f_{CLK}/2^{15}$	977 Hz	

**Note** When changing the clock selected for  $f_{CLK}$  (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

- Remarks 1.** X: Don't care  
**2.** m: Unit number (m = 0), n: Channel number (n = 0)



### 11.6.5 Procedure for processing errors that occurred during slave select input function communication

The procedure for processing errors that occurred during slave select input function communication is described in Figure 11-97.

**Figure 11-99. Processing Procedure in Case of Overrun Error**

Software Manipulation	Hardware Status	Remark
Reads serial data register mn (SDRmn).	→ The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data.	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		Error type is identified and the read value is used to clear error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn).	→ Error flag is cleared.	Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0)

## 11.7 Operation of UART (UART0) Communication

This is a start-stop synchronization function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. By using these two communication lines, each data frame, which consist of a start bit, data, parity bit, and stop bit, is transferred asynchronously (using the internal baud rate) between the microcontroller and the other communication party. Full-duplex asynchronous communication UART communication can be performed by using a channel dedicated to transmission (even-numbered channel) and a channel dedicated to reception (odd-numbered channel).

### [Data transmission/reception]

- Data length of 7, 8, or 9 bits
- Select the MSB/LSB first
- Level setting of transmit/receive data (selecting whether to reverse the level)
- Parity bit appending and parity check functions
- Stop bit appending, stop bit check function

### [Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

### [Error detection flag]

- Framing error, parity error, or overrun error

In addition, UART0 reception supports the SNOOZE mode. When RxD0 pin input is detected while in the STOP mode, the SNOOZE mode makes data reception that does not require the CPU possible.

<R>

UART0 uses channels 0 and 1 of SAU0.

Unit	Channel	Used as CSI	Used as UART
0	0	CSI00 (supporting slave select input function)	UART0
	1	–	

If UART0 is selected for channels 0 and 1 of unit 0, for example, these channels cannot be used for CSI00.

**Caution** When using a serial array unit for UART, both the transmitter side (even-numbered channel) and the receiver side (odd-numbered channel) can only be used for UART.

<R>

UART performs the following two types of communication operations.

- UART transmission (See 11.7.1.)
- UART reception (See 11.7.2.)

### 11.7.1 UART transmission

<R> UART transmission is an operation to transmit data from the R7F0C010 to another device asynchronously (start-stop synchronization).

Of two channels used for UART, the even channel is used for UART transmission.

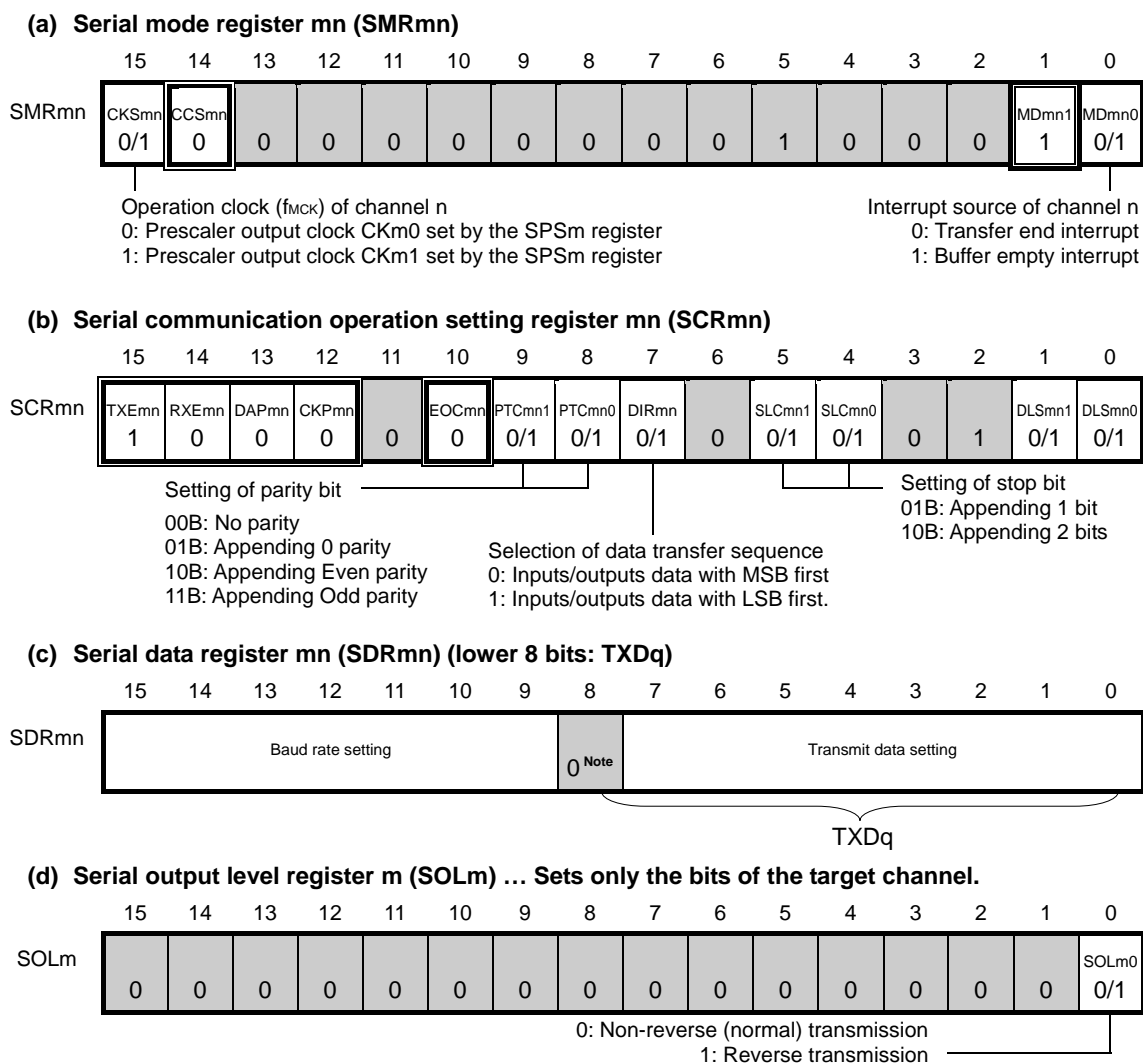
UART	UART0
Target channel	Channel 0 of SAU0
Pins used	TxD0
Interrupt	INTST0
	Transfer end interrupt (in single-transfer mode) or buffer empty interrupt (in continuous transfer mode) can be selected.
Error detection flag	None
Transfer data length	7, 8, or 9 bits
Transfer rate	Max. $f_{MCK}/6$ [bps] (SDR <sub>mn</sub> [15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] <sup>Note</sup>
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)
Parity bit	The following selectable <ul style="list-style-type: none"> <li>• No parity bit</li> <li>• Appending 0 parity</li> <li>• Appending even parity</li> <li>• Appending odd parity</li> </ul>
Stop bit	The following selectable <ul style="list-style-type: none"> <li>• Appending 1 bit</li> <li>• Appending 2 bits</li> </ul>
Data direction	MSB or LSB first

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

- Remarks**
1.  $f_{MCK}$ : Operation clock frequency of target channel  
 $f_{CLK}$ : System clock frequency
  2. m: Unit number (m = 0), n: Channel number (n = 0), mn = 00

(1) Register setting

Figure 11-100. Example of Contents of Registers for UART Transmission of UART (UART0) (1/2)

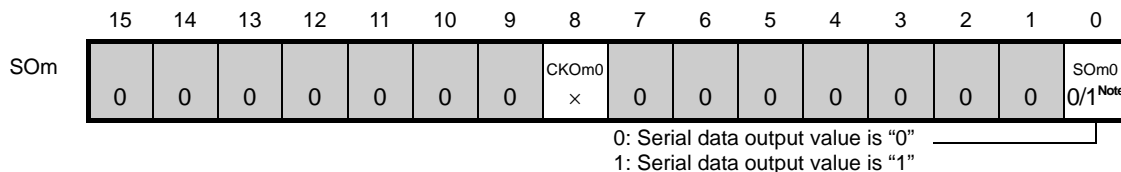


**Note** When UART0 performs 9-bit communication (by setting the DLS001 and DLS000 bits of the SCR00 register to 1), bits 0 to 8 of the SDR00 register are used as the transmission data specification area.

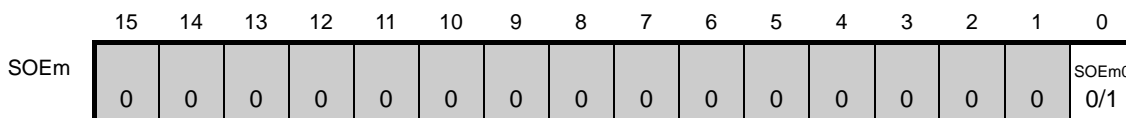
- Remarks**
- m: Unit number (m = 0), n: Channel number (n = 0), q: UART number (q = 0), mn = 00
  - : Setting is fixed in the UART transmission mode, ■: Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

Figure 11-100. Example of Contents of Registers for UART Transmission of UART (UART0) (2/2)

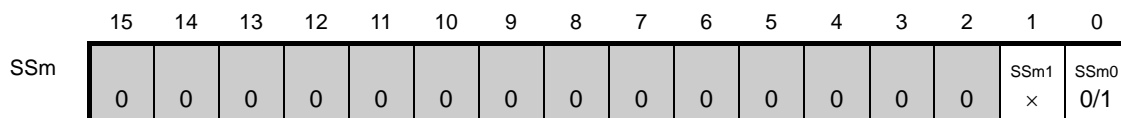
(e) Serial output register m (SOm) ... Sets only the bits of the target channel.



(f) Serial output enable register m (SOEm) ... Sets only the bits of the target channel to 1.



(g) Serial channel start register m (SSm) ... Sets only the bits of the target channel to 1.

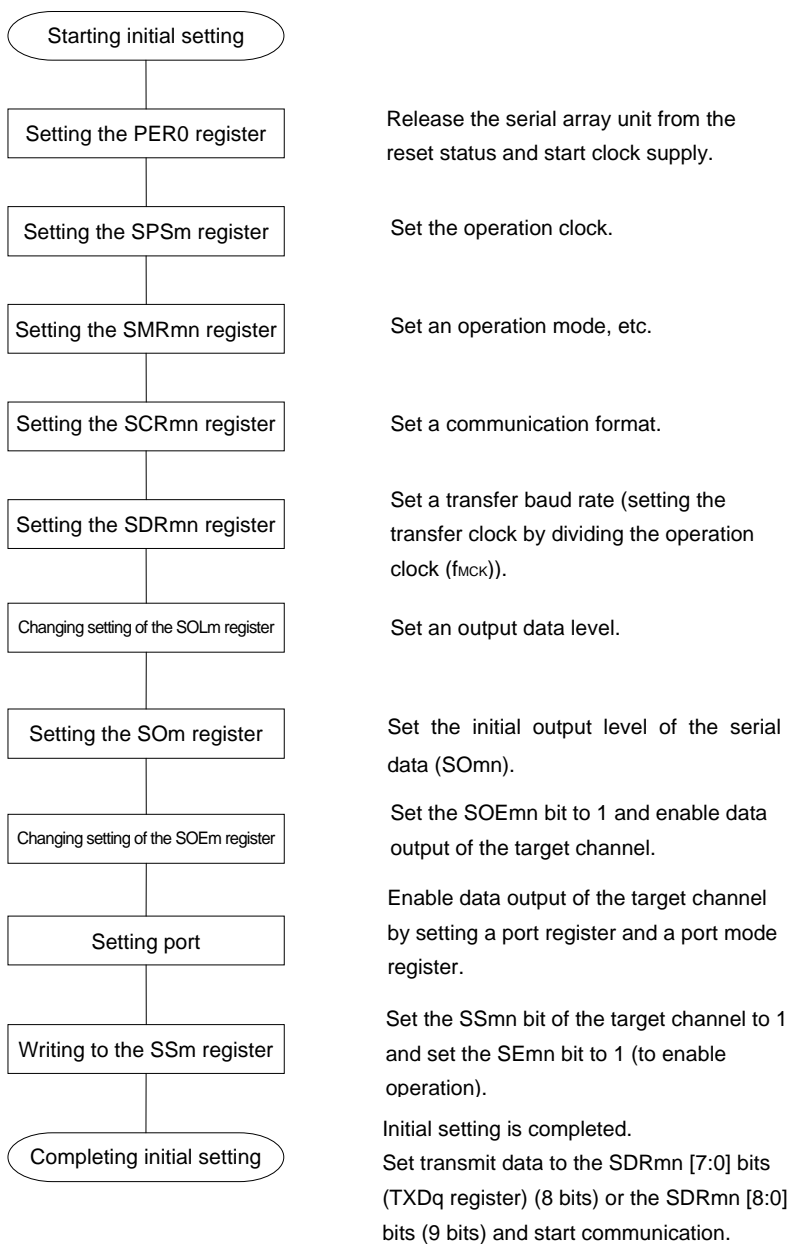


**Note** Before transmission is started, be sure to set to 1 when the SOLmn bit of the target channel is set to 0, and set to 0 when the SOLmn bit of the target channel is set to 1. The value varies depending on the communication data during communication operation.

- Remarks**
1. m: Unit number (m = 0), n: Channel number (n = 0), q: UART number (q = 0), mn = 00
  2.  : Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

(2) Operation procedure

Figure 11-101. Initial Setting Procedure for UART Transmission



<R>

<R>

**Figure 11-102. Procedure for Stopping UART Transmission**

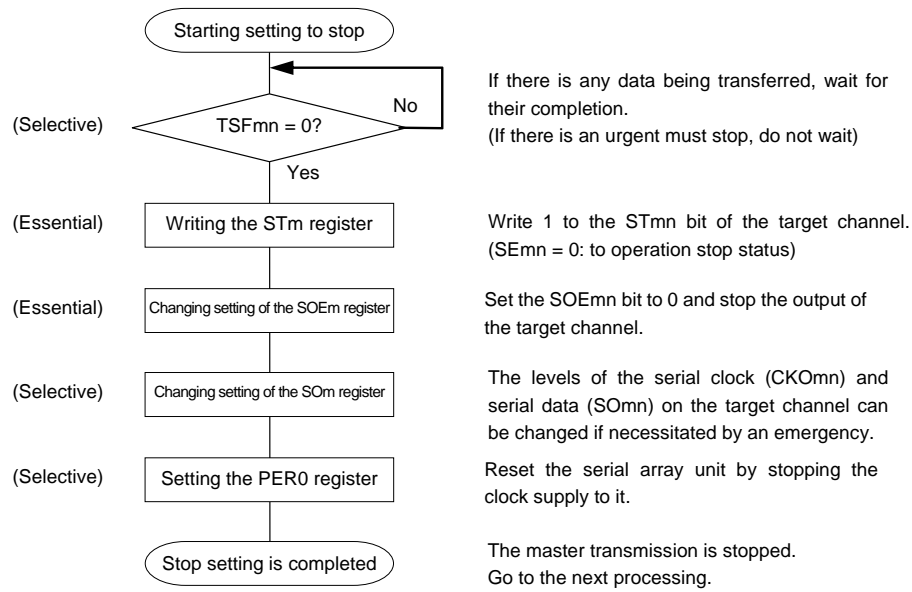
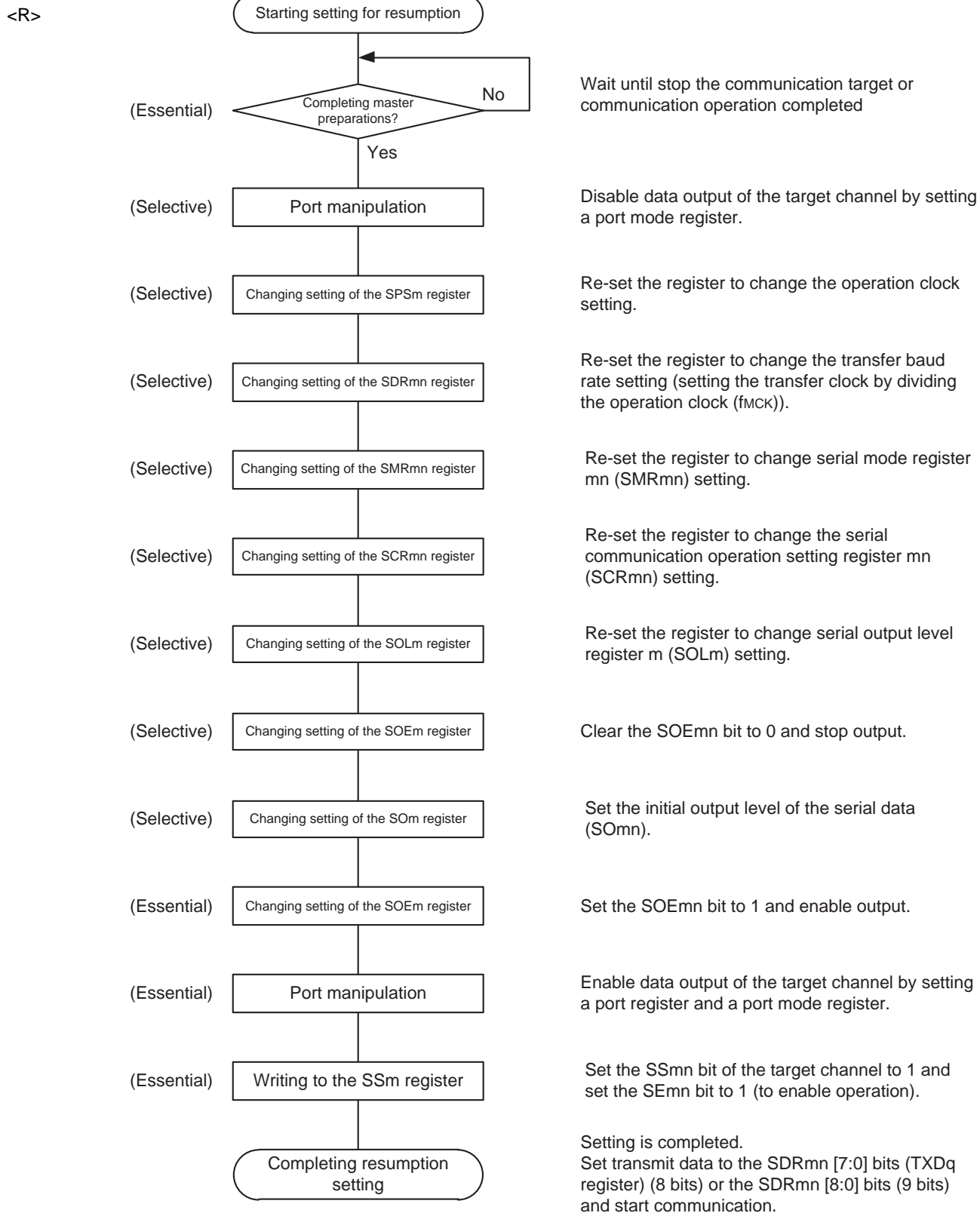




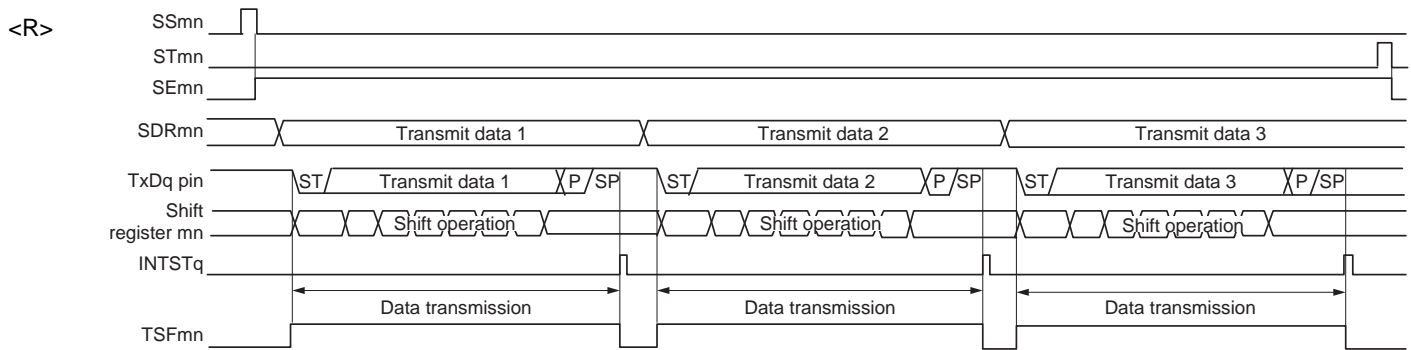
Figure 11-103. Procedure for Resuming UART Transmission



**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target stops or transmission finishes, and then perform initialization instead of restarting the transmission.

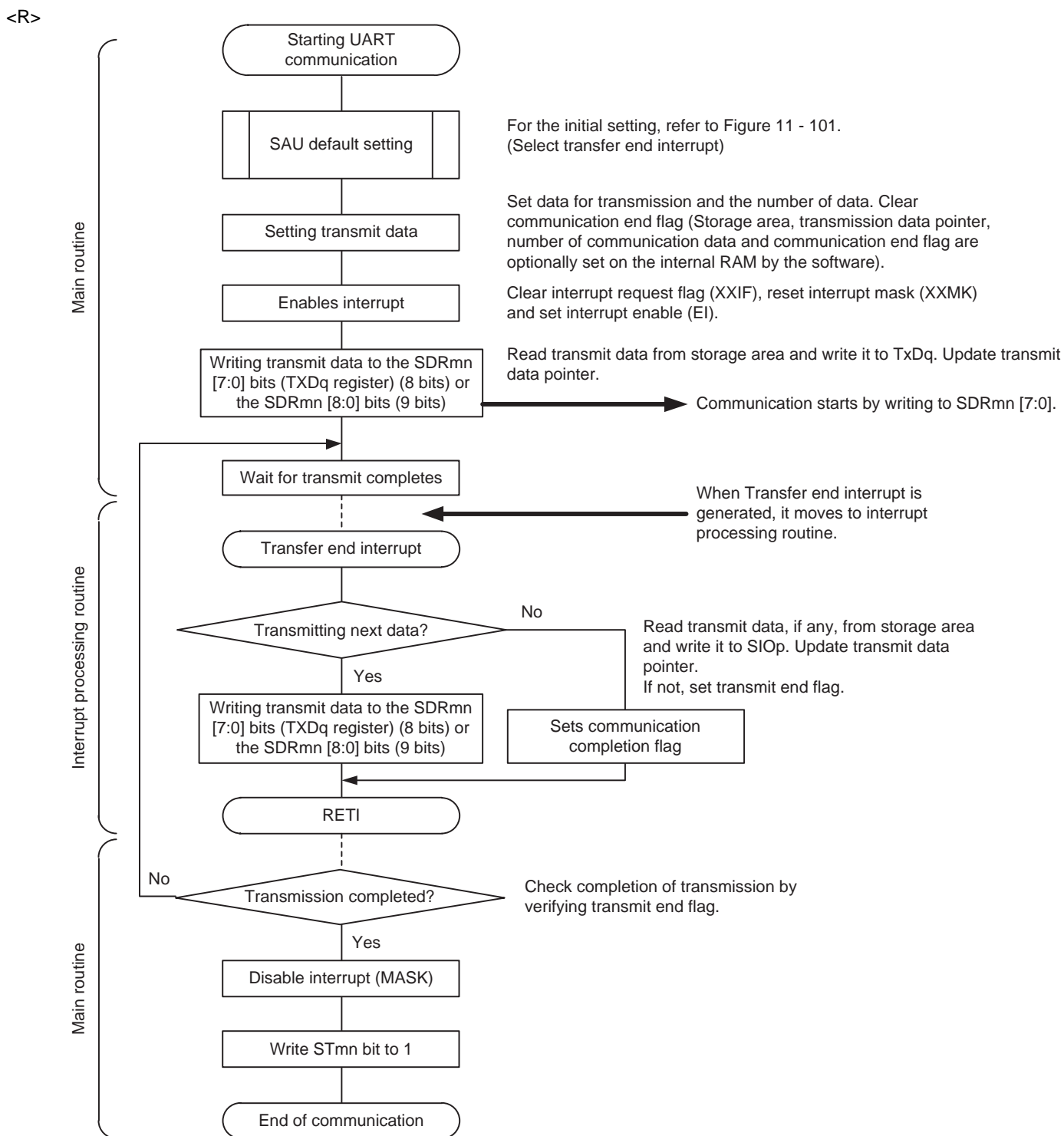
(3) Processing flow (in single-transmission mode)

Figure 11-104. Timing Chart of UART Transmission (in Single-Transmission Mode)



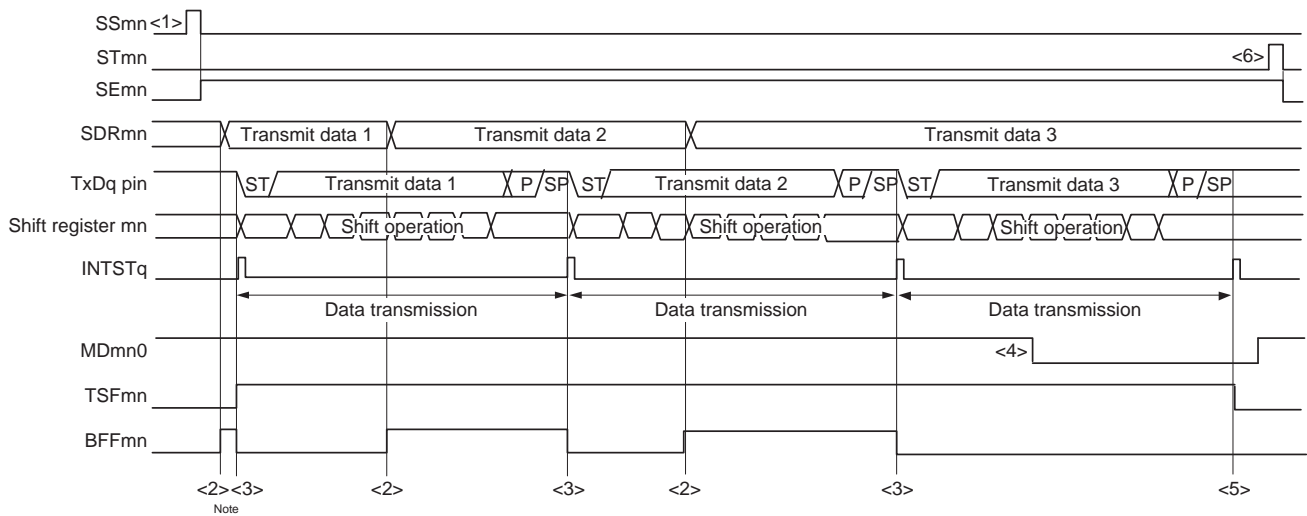
**Remark** m: Unit number (m = 0), n: Channel number (n = 0), q: UART number (q = 0), mn = 00

Figure 11-105. Flowchart of UART Transmission (in Single-Transmission Mode)



## (4) Processing flow (in continuous transmission mode)

Figure 11-106. Timing Chart of UART Transmission (in Continuous Transmission Mode)

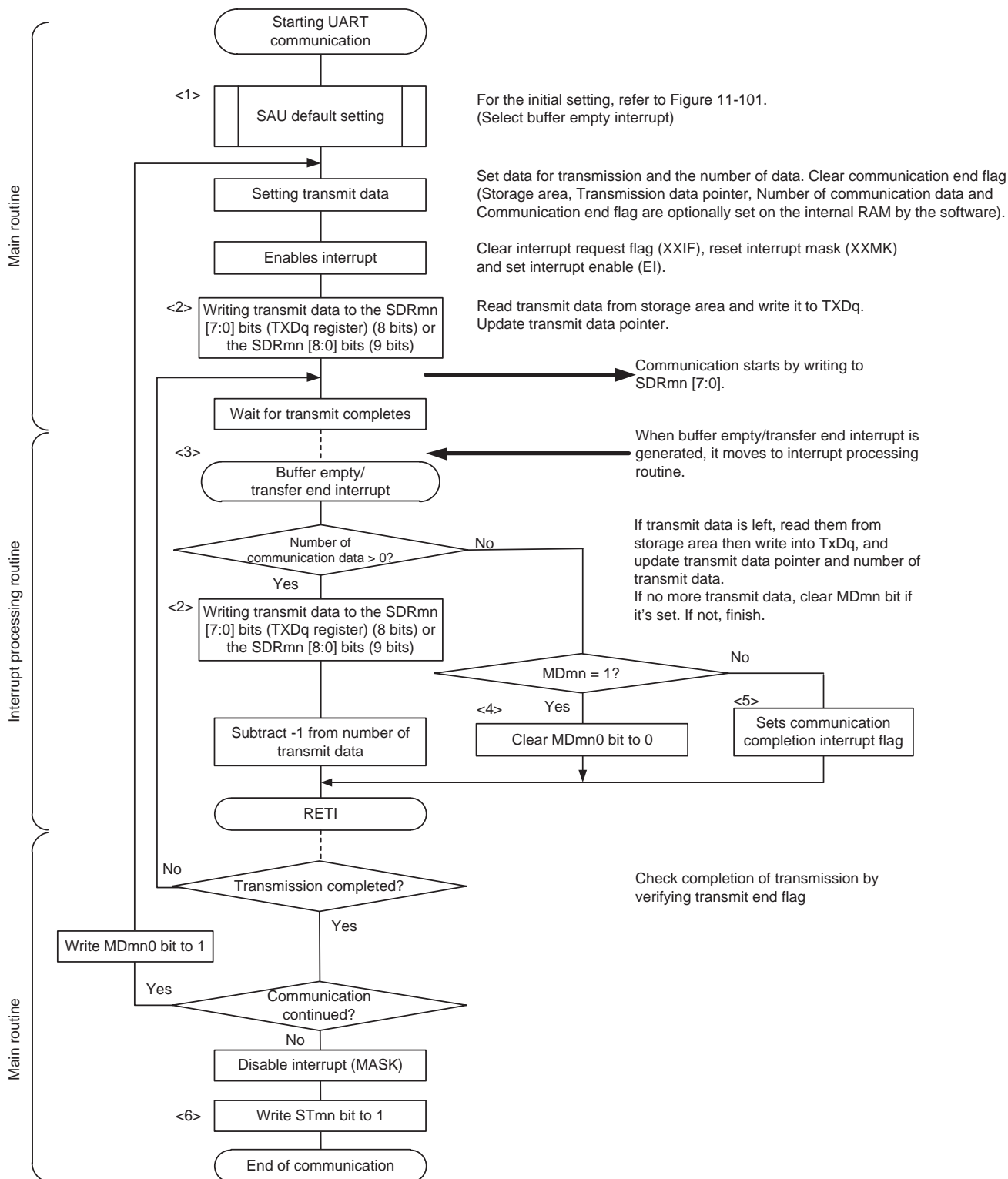


**Note** If transmit data is written to the SDRmn register while the BFFmn bit of serial status register mn (SSRmn) is 1 (valid data is stored in serial data register mn (SDRmn)), the transmit data is overwritten.

**Caution** The MDmn0 bit of serial mode register mn (SSRmn) can be rewritten even during operation. However, rewrite it before transfer of the last bit is started, so that it will be rewritten before the transfer end interrupt of the last transmit data.

**Remark** m: Unit number (m = 0), n: Channel number (n = 0), q: UART number (q = 0), mn = 00

Figure 11-107. Flowchart of UART Transmission (in Continuous Transmission Mode)



<R> **Remark** <1> to <6> in the figure correspond to <1> to <6> in Figure 11-106 Timing Chart of UART Transmission (in Continuous Transmission Mode).

### 11.7.2 UART reception

UART reception is an operation wherein the R7F0C010 asynchronously receives data from another device (start-stop synchronization).

For UART reception, the odd-number channel of the two channels used for UART is used. The SMR register of both the odd- and even-numbered channels must be set.

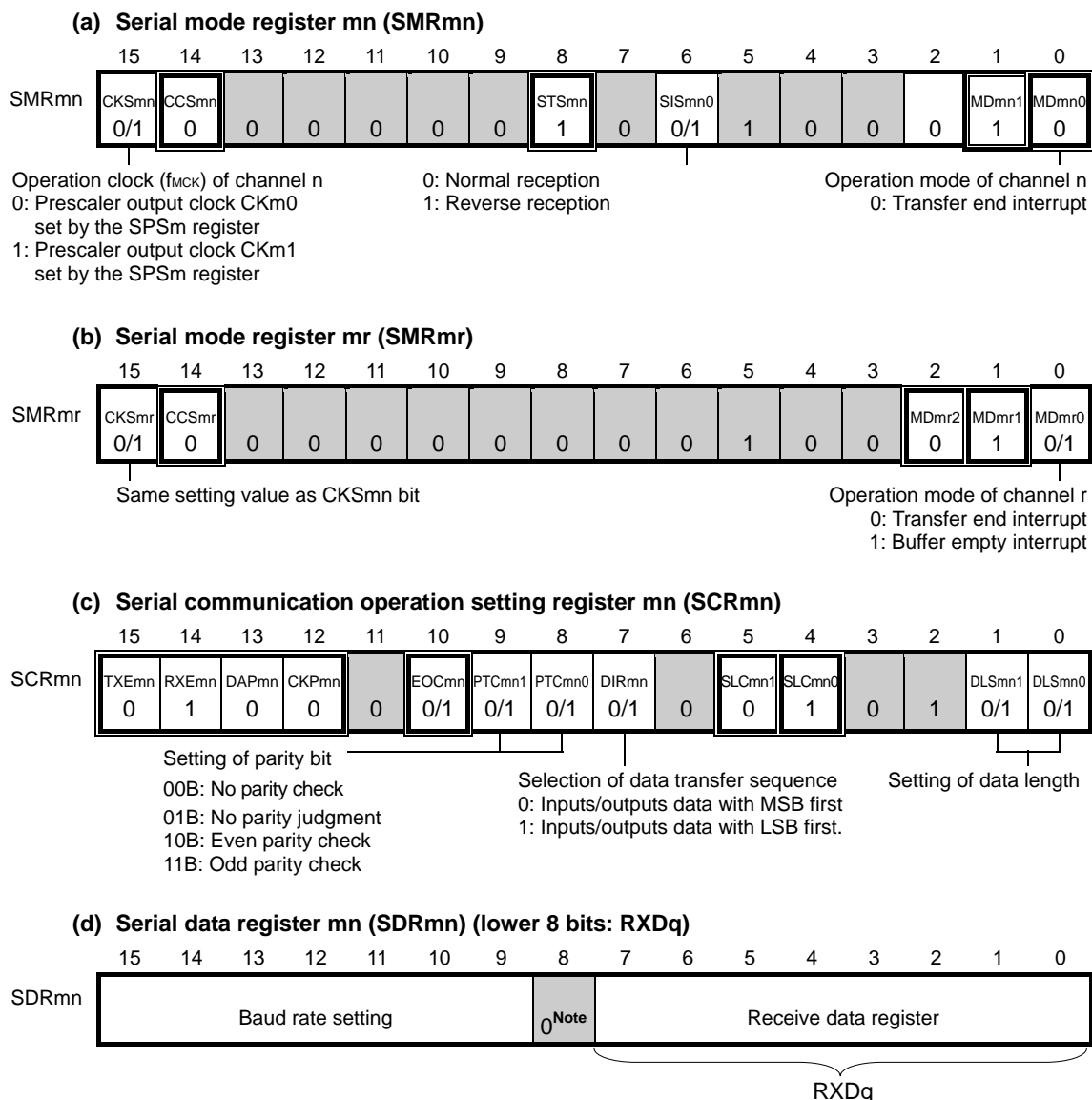
UART	UART0
Target channel	Channel 1 of SAU0
Pins used	RxD0
Interrupt	INTSR0
	Transfer end interrupt only (Setting the buffer empty interrupt is prohibited.)
Error interrupt	INTSRE0
Error detection flag	<ul style="list-style-type: none"> <li>• Framing error detection flag (FEFmn)</li> <li>• Parity error detection flag (PEFmn)</li> <li>• Overrun error detection flag (OVFmn)</li> </ul>
Transfer data length	7, 8 or 9 bits
Transfer rate	Max. $f_{MCK}/6$ [bps] (SDRmn [15:9] = 2 or more), Min. $f_{CLK}/(2 \times 2^{15} \times 128)$ [bps] <sup>Note</sup>
Data phase	Non-reverse output (default: high level) Reverse output (default: low level)
Parity bit	The following selectable <ul style="list-style-type: none"> <li>• No parity bit (no parity check)</li> <li>• No parity judgment (0 parity)</li> <li>• Even parity check</li> <li>• Odd parity check</li> </ul>
Stop bit	1 bit check
Data direction	MSB or LSB first

**Note** Use this operation within a range that satisfies the conditions above and the peripheral functions characteristics in the electrical specifications (see **CHAPTER 27 ELECTRICAL SPECIFICATIONS**).

- Remarks**
1.  $f_{MCK}$ : Operation clock frequency of target channel  
 $f_{CLK}$ : System clock frequency
  2. m: Unit number (m = 0), n: Channel number (n = 1), mn = 01

## (1) Register setting

Figure 11-108. Example of Contents of Registers for UART Reception of UART (UART0) (1/2)



**Note** When UART performs 9-bit communication, bits 0 to 8 of the SDRm1 register are used as the transmission data specification area.

**Caution** For the UART reception, be sure to set the SMRmr register of channel r to UART transmission mode that is to be paired with channel n.

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 1), mn = 01

r: Channel number (r = 0), q: UART number (q = 0)

2.  : Setting is fixed in the UART reception mode,  : Setting disabled (set to the initial value)  
 ×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)  
 0/1: Set to 0 or 1 depending on the usage of the user

Figure 11-108. Example of Contents of Registers for UART Reception of UART (UART0) (2/2)

(e) Serial output register m (SOm) ... The register that not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOm	0	0	0	0	0	0	0	CKOm0 ×	0	0	0	0	0	0	0	SOm0 ×

(f) Serial output enable register m (SOEm) ...The register that not used in this mode.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOEm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SOEm0 ×

(g) Serial channel start register m (SSm) ... Sets only the bits of the target channel is 1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SSm1 0/1	SSm0 ×

<R>

**Remarks 1.** m: Unit number (m = 0), n: Channel number (n = 1), mn = 01

r: Channel number (r = 0), q: UART number (q = 0)

**2.** : Setting is fixed in the UART reception mode, : Setting disabled (set to the initial value)

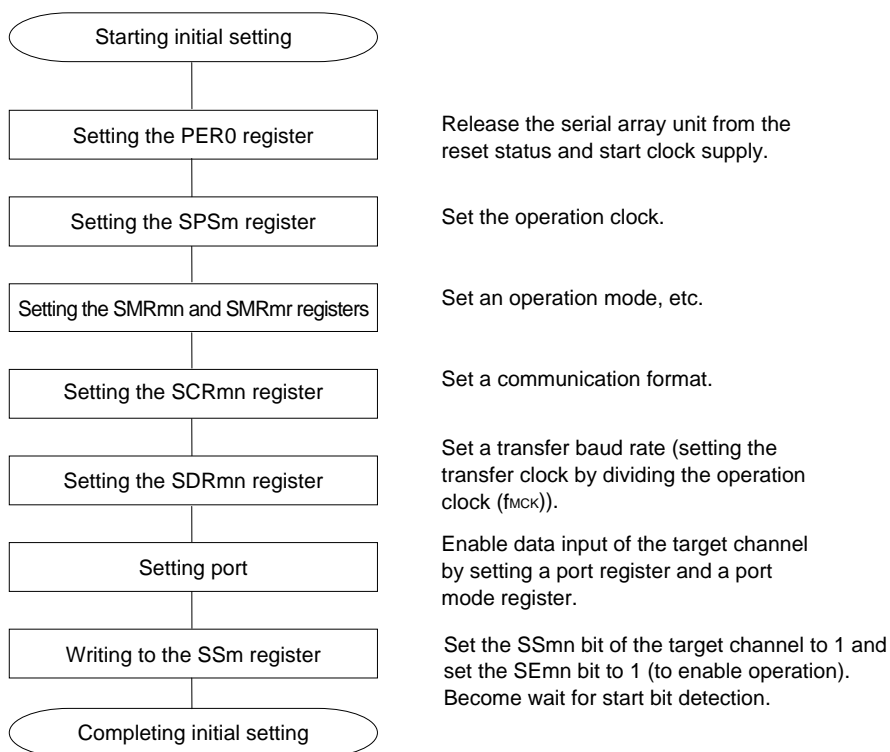
×: Bit that cannot be used in this mode (set to the initial value when not used in any mode)

0/1: Set to 0 or 1 depending on the usage of the user



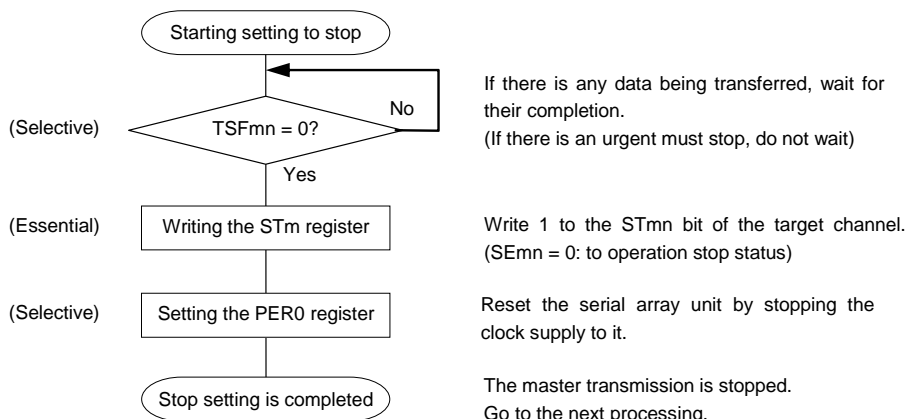
(2) Operation procedure

Figure 11-109. Initial Setting Procedure for UART Reception



**Caution** Set the RXEmn bit of SCRmn register to 1, and then be sure to set SSmn to 1 after 4 or more  $f_{MCK}$  clocks have elapsed.

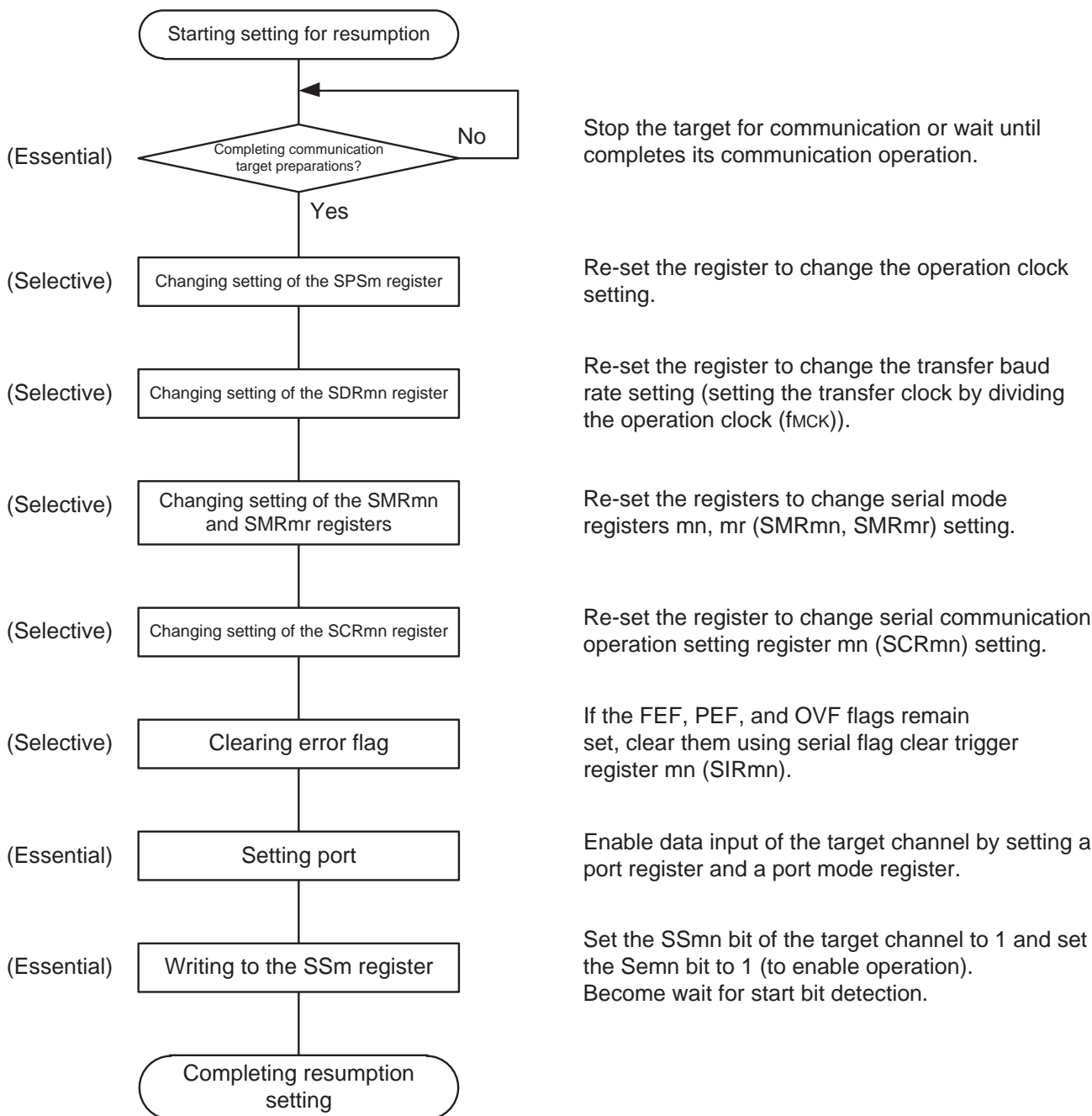
Figure 11-110. Procedure for Stopping UART Reception



<R>

Figure 11-111. Procedure for Resuming UART Reception

<R>



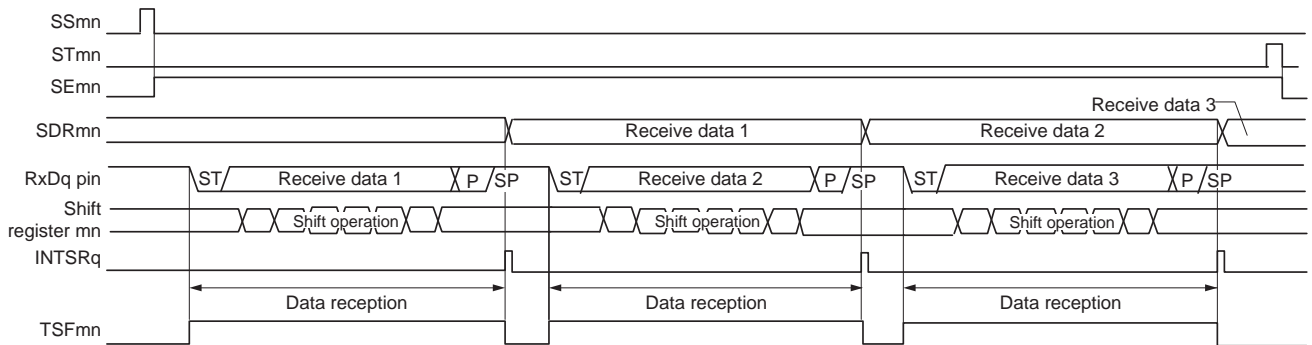
**Caution** After is set RXEmn bit to 1 of SCRmn register, set the SSmn = 1 from an interval of at least four clocks of fmck.

**Remark** If PER0 is rewritten while stopping the master transmission and the clock supply is stopped, wait until the transmission target (slave) stops or transmission finishes, and then perform initialization instead of restarting the transmission.

(3) Processing flow

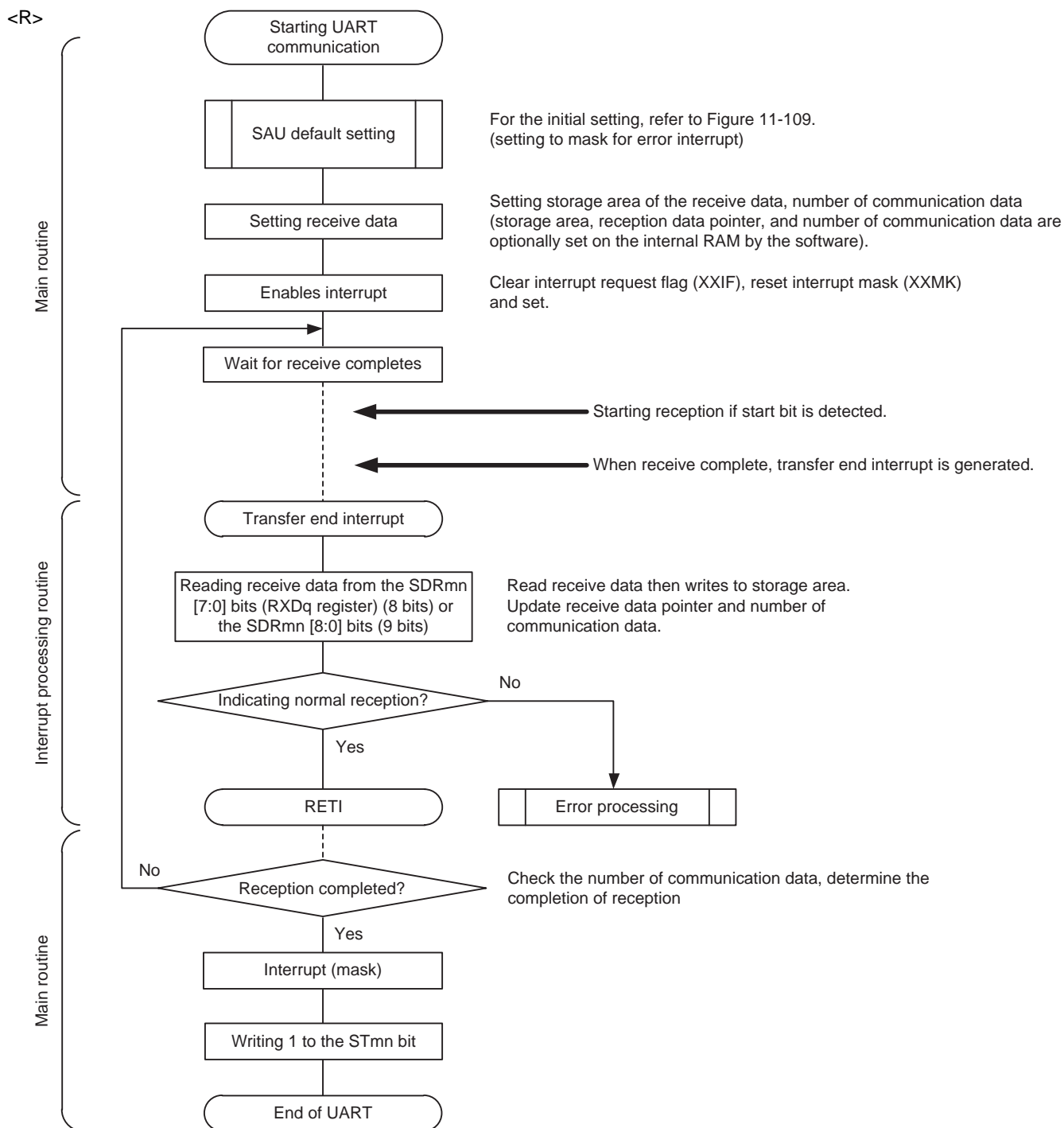
<R>

Figure 11-112. Timing Chart of UART Reception



**Remark** m: Unit number (m = 0), n: Channel number (n = 1), mn = 01  
 r: Channel number (r = 0), q: UART number (q = 0)

Figure 11-113. Flowchart of UART Reception



### 11.7.3 SNOOZE mode function

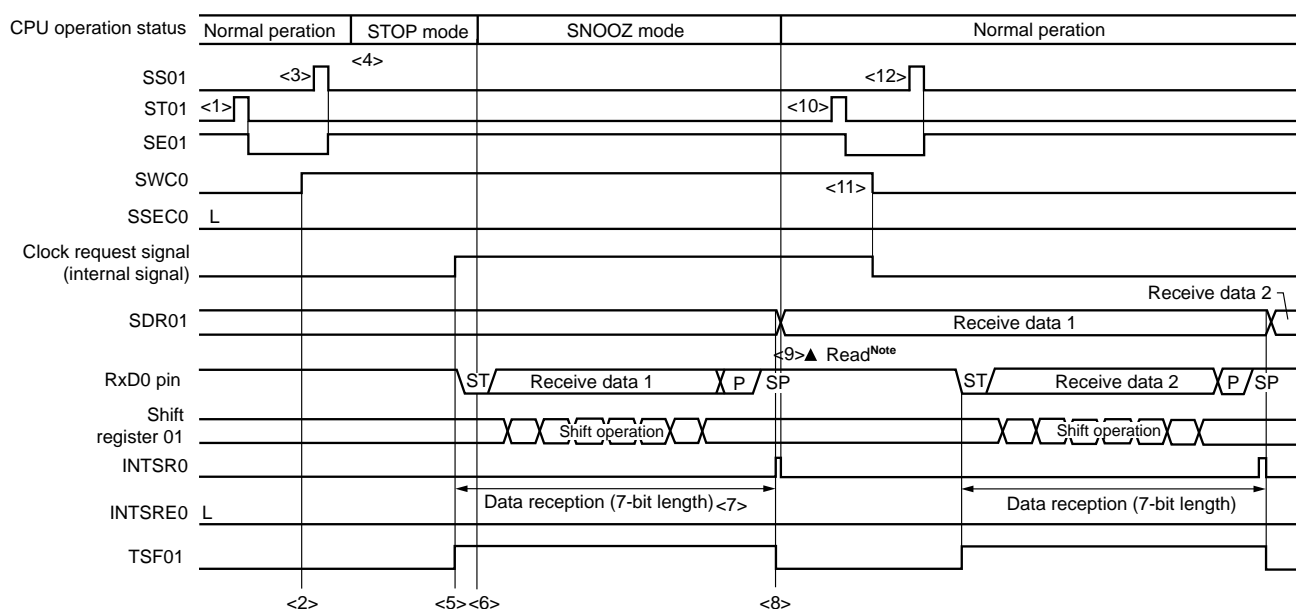
SNOOZE mode makes UART operate reception by RxDq pin input detection while the STOP mode. Normally UART stops communication in the STOP mode. But, using the SNOOZE mode makes reception UART operate unless the CPU operation by detecting RxDq pin input.

When using the SNOOZE mode function, set the SWCm bit of serial standby control register m (SSCm) to 1 just before switching to the STOP mode.

- Cautions 1.** The SNOOZE mode can only be specified when the high-speed on-chip oscillator clock is selected for fCLK.
- 2.** The maximum transfer rate when using UARTq in the SNOOZE mode is 9600 bps.

#### (1) SNOOZE mode operation (Normal operation)

Figure 11-114. Timing Chart of SNOOZE Mode Operation (Normal Operation Mode)



**Note** Read the received data when SWCm is 1

**Caution** Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEM1 bit, and stop the operation). And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in Figure 11-116 Flowchart of SNOOZE Mode Operation (Normal Operation/Abnormal Operation <1>).

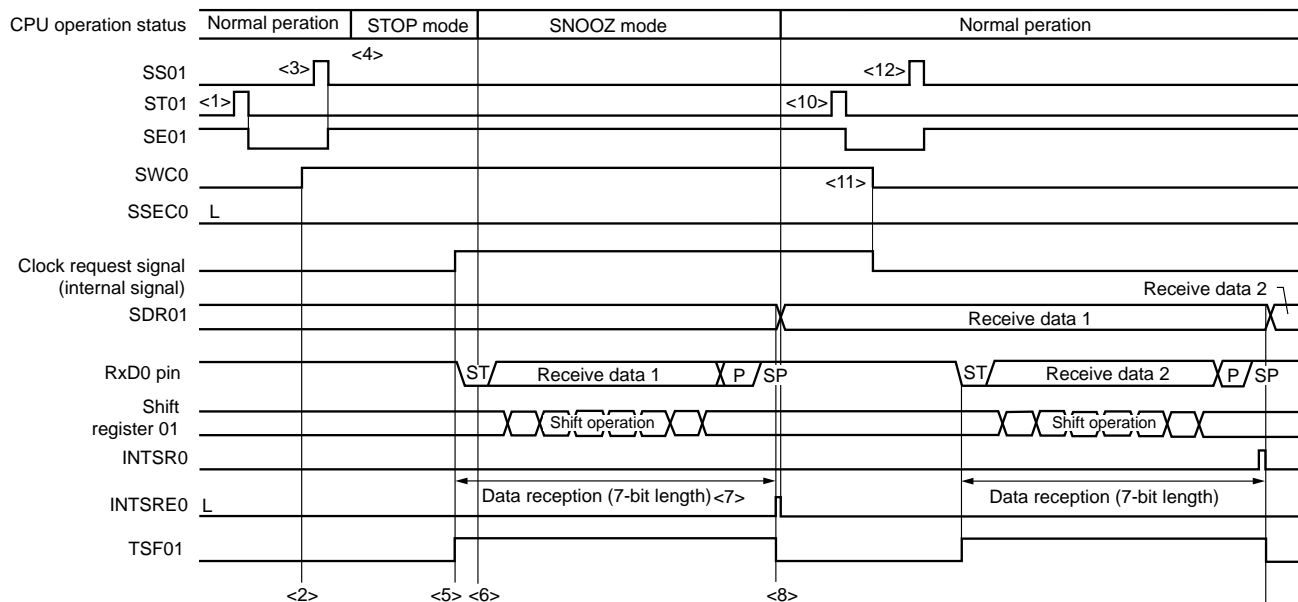
**2.** m = 0; q = 0

**(2) SNOOZE mode operation (Abnormal Operation <1>)**

Abnormal operation <1> is the operation performed when a communication error occurs while SSECm = 0. Because SSECm = 0, an error interrupt (INTSREq) is generated when a communication error occurs.

**Figure 11-115. Timing Chart of SNOOZE Mode Operation (Abnormal Operation <1>)**

<R>

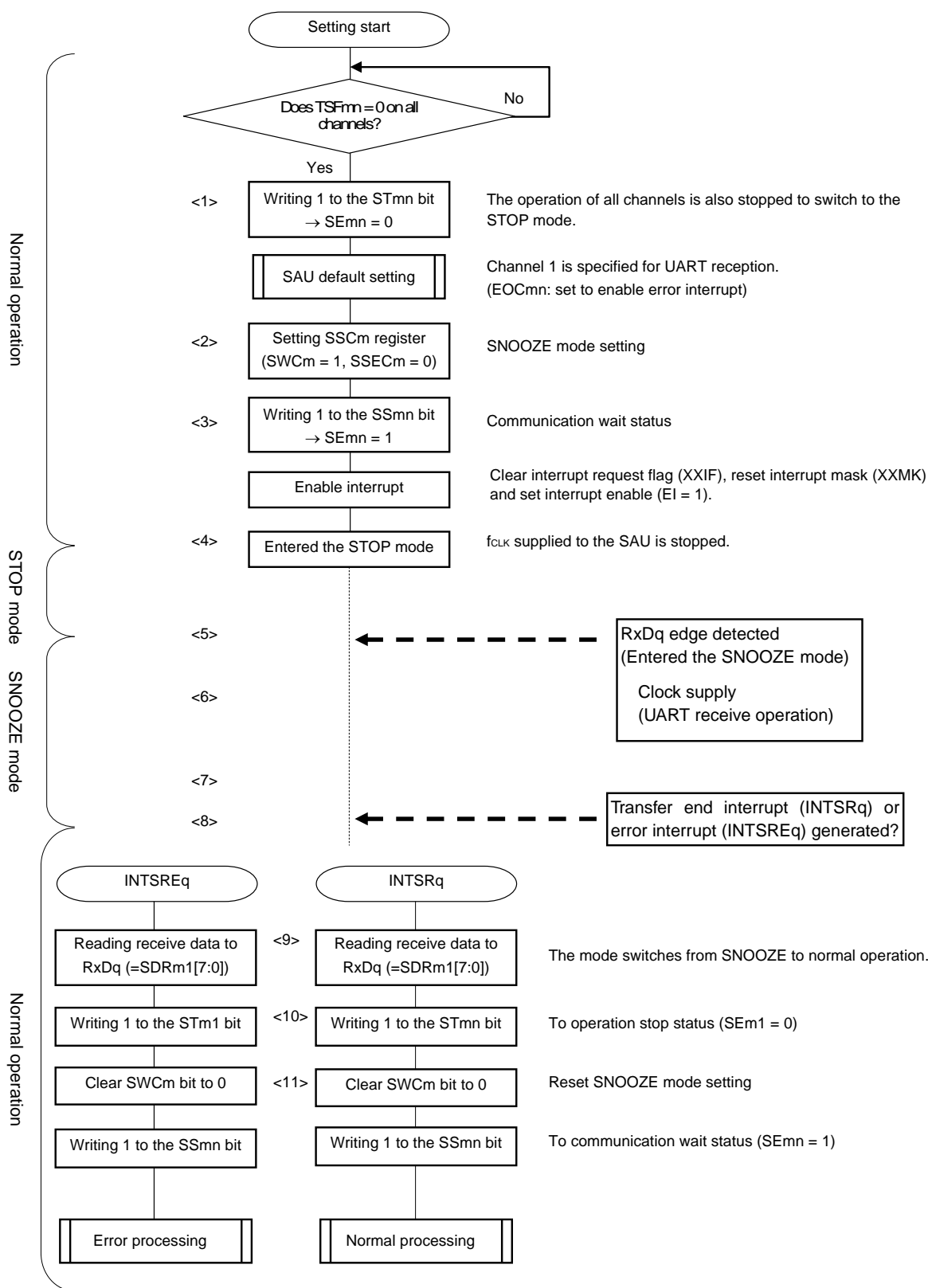


**Caution** Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the STm1 bit to 1 (clear the SEM1 bit, and stop the operation). And after completion the receive operation, also clearing SWCm bit to 0 (SNOOZE mode release).

**Remarks 1.** <1> to <11> in the figure correspond to <1> to <11> in Figure 11-116 Flowchart of SNOOZE Mode Operation (Normal Operation/Abnormal Operation <1>).

**2.** m = 0; q = 0

Figure 11-116. Flowchart of SNOOZE Mode Operation (Normal Operation/Abnormal Operation <1>)



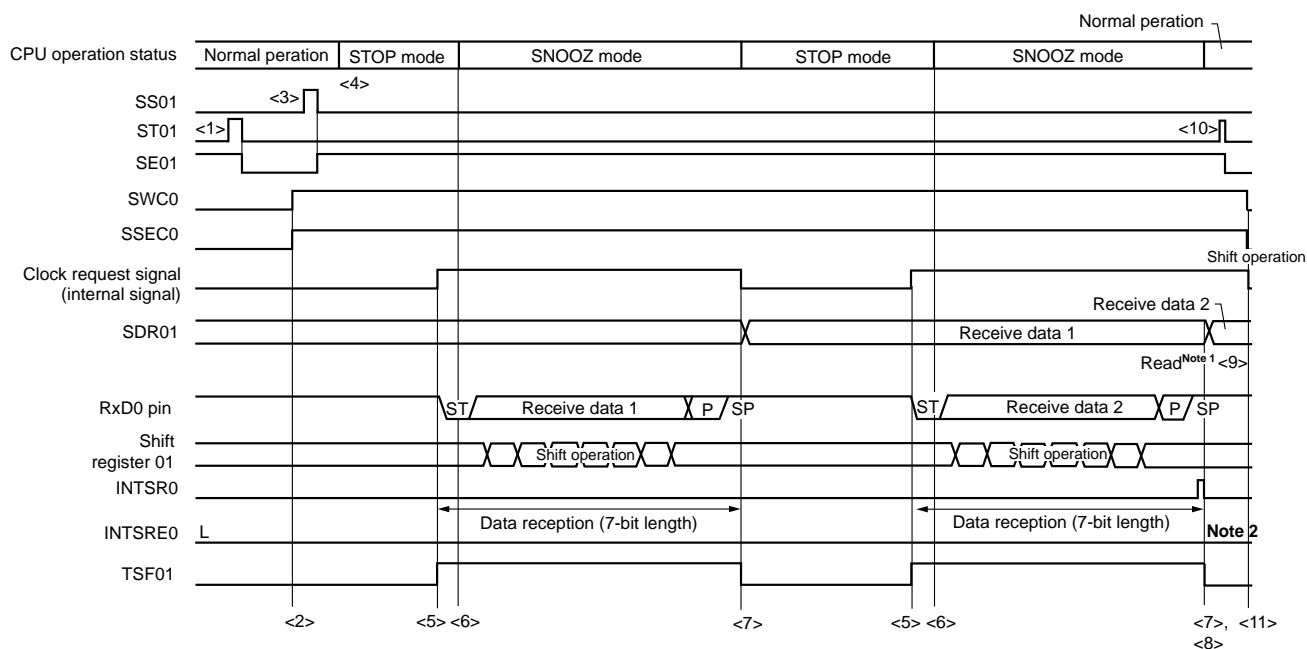
Remarks 1. <1> to <11> in the figure correspond to <1> to <11> in Figure 11-114 Timing Chart of SNOOZE Mode Operation (Normal Operation Mode) and Figure 11-115 Timing Chart of SNOOZE Mode Operation (Abnormal Operation <1>).

2. m = 0; q = 0

**(3) SNOOZE mode operation (Abnormal Operation <2>)**

Abnormal operation <2> is the operation performed when a communication error occurs while  $SSECm = 1$ . Because  $SSECm = 1$ , an error interrupt (INTSREq) is not generated when a communication error occurs.

&lt;R&gt;

**Figure 11-117. Timing Chart of SNOOZE Mode Operation (Abnormal Operation <2>)**

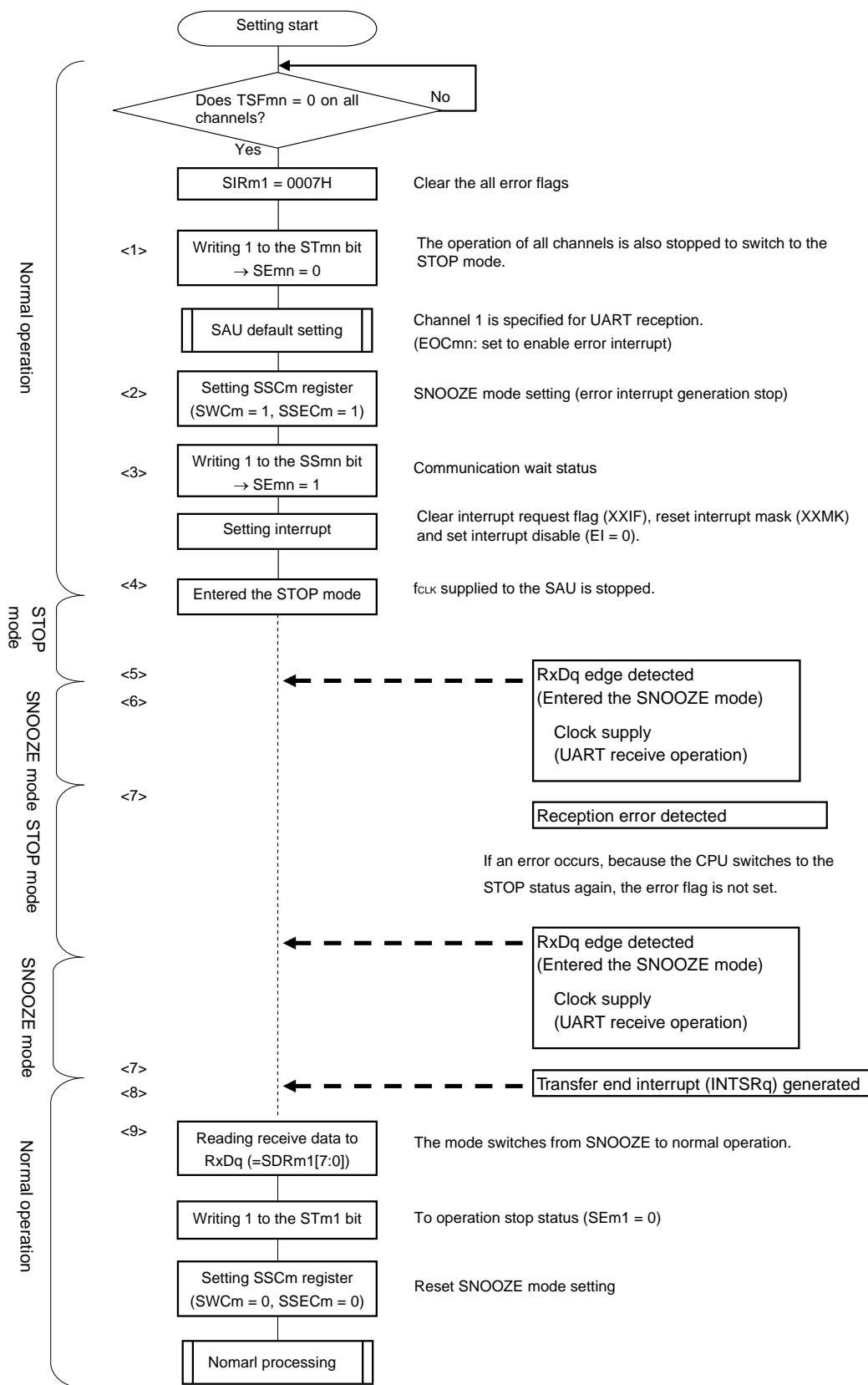
- Notes**
1. Only read received data while  $SWCm = 1$ .
  2. After UARTq successfully finishes reception in the SNOOZE mode, it is possible to continue to perform normal reception operations without changing the settings, but, because  $SSECm = 1$ , the  $PEFm1$  and  $FEFm1$  bits are not set even if a framing error or parity error occurs. In addition, no error interrupt (INTSREq) is generated.

- Cautions**
1. Before switching to the SNOOZE mode or after reception operation in the SNOOZE mode finishes, set the  $STm1$  bit to 1 (clear the  $SEM1$  bit, and stop the operation). And after completion the receive operation, also clearing  $SWCm$  bit to 0 (SNOOZE mode release).
  2. When using the SNOOZE mode while  $SSECm$  is set to 1, no overrun errors occur. Therefore, when using the SNOOZE mode, read bits 7 to 0 ( $RxDq$ ) of the  $SDRm1$  register before switching to the STOP mode.

- Remarks**
1. <1> to <9> in the figure correspond to <1> to <9> in Figure 11-118 Flowchart of SNOOZE Mode Operation (Abnormal Operation <2>).
  2.  $m = 0$ ;  $q = 0$



Figure 11-118. Flowchart of SNOOZE Mode Operation (Abnormal Operation <2>)



(Caution and Remarks are listed on the next page.)

**Caution** When using the SNOOZE mode while SSECm is set to 1, no overrun errors occur. Therefore, when using the SNOOZE mode, read bits 7 to 0 (RxDq) of the SDRm1 register before switching to the STOP mode.

**Remarks 1.** <1> to <9> in the figure correspond to <1> to <9> in **Figure 11-117 Timing Chart of SNOOZE Mode Operation (Abnormal Operation <2>)**.

**2.** m = 0; q = 0

### 11.7.4 Calculating baud rate

#### (1) Baud rate calculation expression

The baud rate for UART (UART0) communication can be calculated by the following expressions.

$$\text{(Baud rate)} = \{\text{Operation clock (f}_{\text{MCK}}\text{) frequency of target channel}\} \div (\text{SDRmn}[15:9] + 1) \div 2 \text{ [bps]}$$

**Caution** Setting serial data register mn (SDRmn) SDRmn[15:9] = (0000000B, 0000001B) is prohibited.

**Remarks 1.** When UART is used, the value of SDRmn[15:9] is the value of bits 15 to 9 of the SDRmn register (0000010B to 1111111B) and therefore is 2 to 127.

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 1), mn = 00, 01

The operation clock (f<sub>MCK</sub>) is determined by serial clock select register m (SPSm) and bit 15 (CKSmn) of serial mode register mn (SMRmn).

Table 11-4. Selection of Operation Clock for UART

SMRmn Register	SPSm Register								Operation Clock (f <sub>CLK</sub> ) <sup>Note</sup>	
	CKSmn	PRSm13	PRSm12	PRSm11	PRSm10	PRSm03	PRSm02	PRSm01	PRSm00	f <sub>CLK</sub> = 32 MHz
0	X	X	X	X	0	0	0	0	f <sub>CLK</sub>	32 MHz
	X	X	X	X	0	0	0	1	f <sub>CLK</sub> /2	16 MHz
	X	X	X	X	0	0	1	0	f <sub>CLK</sub> /2 <sup>2</sup>	8 MHz
	X	X	X	X	0	0	1	1	f <sub>CLK</sub> /2 <sup>3</sup>	4 MHz
	X	X	X	X	0	1	0	0	f <sub>CLK</sub> /2 <sup>4</sup>	2 MHz
	X	X	X	X	0	1	0	1	f <sub>CLK</sub> /2 <sup>5</sup>	1 MHz
	X	X	X	X	0	1	1	0	f <sub>CLK</sub> /2 <sup>6</sup>	500 kHz
	X	X	X	X	0	1	1	1	f <sub>CLK</sub> /2 <sup>7</sup>	250 kHz
	X	X	X	X	1	0	0	0	f <sub>CLK</sub> /2 <sup>8</sup>	125 kHz
	X	X	X	X	1	0	0	1	f <sub>CLK</sub> /2 <sup>9</sup>	62.5 kHz
	X	X	X	X	1	0	1	0	f <sub>CLK</sub> /2 <sup>10</sup>	31.25 kHz
	X	X	X	X	1	0	1	1	f <sub>CLK</sub> /2 <sup>11</sup>	15.63 kHz
	X	X	X	X	1	1	0	0	f <sub>CLK</sub> /2 <sup>12</sup>	7.81 kHz
	X	X	X	X	1	1	0	1	f <sub>CLK</sub> /2 <sup>13</sup>	3.91 kHz
	X	X	X	X	1	1	1	0	f <sub>CLK</sub> /2 <sup>14</sup>	1.95 kHz
X	X	X	X	1	1	1	1	f <sub>CLK</sub> /2 <sup>15</sup>	977 Hz	
1	0	0	0	0	X	X	X	X	f <sub>CLK</sub>	32 MHz
	0	0	0	1	X	X	X	X	f <sub>CLK</sub> /2	16 MHz
	0	0	1	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>2</sup>	8 MHz
	0	0	1	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>3</sup>	4 MHz
	0	1	0	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>4</sup>	2 MHz
	0	1	0	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>5</sup>	1 MHz
	0	1	1	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>6</sup>	500 kHz
	0	1	1	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>7</sup>	250 kHz
	1	0	0	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>8</sup>	125 kHz
	1	0	0	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>9</sup>	62.5 kHz
	1	0	1	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>10</sup>	31.25 kHz
	1	0	1	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>11</sup>	15.63 kHz
	1	1	0	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>12</sup>	7.81 kHz
	1	1	0	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>13</sup>	3.91 kHz
	1	1	1	0	X	X	X	X	f <sub>CLK</sub> /2 <sup>14</sup>	1.95 kHz
1	1	1	1	X	X	X	X	f <sub>CLK</sub> /2 <sup>15</sup>	977 Hz	
Other than above									Setting prohibited	

**Note** When changing the clock selected for f<sub>CLK</sub> (by changing the system clock control register (CKC) value), do so after having stopped (serial channel stop register m (STm) = 000FH) the operation of the serial array unit (SAU).

**Remarks 1.** X: Don't care

**2.** m: Unit number (m = 0), n: Channel number (n = 0, 1), mn = 00, 01

**(2) Baud rate error during transmission**

The baud rate error of UART (UART0) communication during transmission can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

$$(\text{Baud rate error}) = (\text{Calculated baud rate value}) \div (\text{Target baud rate}) \times 100 - 100 [\%]$$

Here is an example of setting a UART baud rate at  $f_{\text{CLK}} = 32 \text{ MHz}$ .

UART Baud Rate (Target Baud Rate)	$f_{\text{CLK}} = 32 \text{ MHz}$			
	Operation Clock ( $f_{\text{MCK}}$ )	SDRmn[15:9]	Calculated Baud Rate	Error from Target Baud Rate
300 bps	$f_{\text{CLK}}/2^9$	103	300.48 bps	+0.16 %
600 bps	$f_{\text{CLK}}/2^8$	103	600.96 bps	+0.16 %
1200 bps	$f_{\text{CLK}}/2^7$	103	1201.92 bps	+0.16 %
2400 bps	$f_{\text{CLK}}/2^6$	103	2403.85 bps	+0.16 %
4800 bps	$f_{\text{CLK}}/2^5$	103	4807.69 bps	+0.16 %
9600 bps	$f_{\text{CLK}}/2^4$	103	9615.38 bps	+0.16 %
19200 bps	$f_{\text{CLK}}/2^3$	103	19230.8 bps	+0.16 %
31250 bps	$f_{\text{CLK}}/2^3$	63	31250.0 bps	$\pm 0.0 \%$
38400 bps	$f_{\text{CLK}}/2^2$	103	38461.5 bps	+0.16 %
76800 bps	$f_{\text{CLK}}/2$	103	76923.1 bps	+0.16 %
153600 bps	$f_{\text{CLK}}$	103	153846 bps	+0.16 %
312500 bps	$f_{\text{CLK}}$	50	313725.5 bps	+0.39 %

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), mn = 00, 01

**(3) Permissible baud rate range for reception**

The permissible baud rate range for reception during UART (UART0) communication can be calculated by the following expression. Make sure that the baud rate at the transmission side is within the permissible baud rate range at the reception side.

$$\text{(Maximum receivable baud rate)} = \frac{2 \times k \times \text{Nfr}}{2 \times k \times \text{Nfr} - k + 2} \times \text{Brate}$$

$$\text{(Minimum receivable baud rate)} = \frac{2 \times k \times (\text{Nfr} - 1)}{2 \times k \times \text{Nfr} - k - 2} \times \text{Brate}$$

Brate: Calculated baud rate value at the reception side (See 11.7.4 (1) Baud rate calculation expression.)

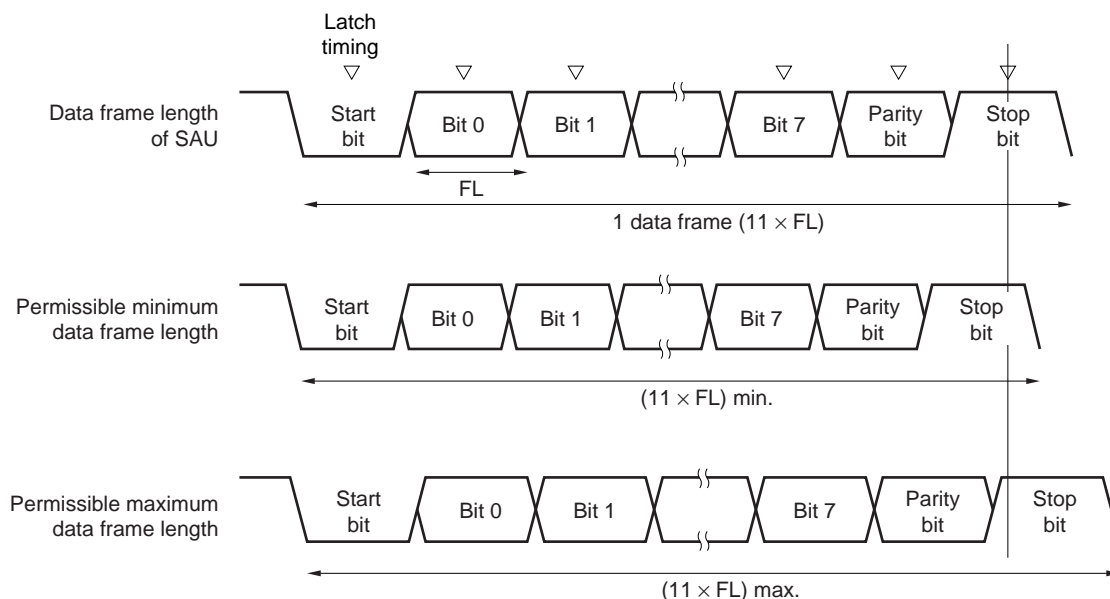
k: SDRmn[15:9] + 1

Nfr: 1 data frame length [bits]

= (Start bit) + (Data length) + (Parity bit) + (Stop bit)

**Remark** m: Unit number (m = 0), n: Channel number (n = 1), mn = 01

**Figure 11-119. Permissible Baud Rate Range for Reception (1 Data Frame Length = 11 Bits)**



As shown in Figure 11-119, the timing of latching receive data is determined by the division ratio set by bits 15 to 9 of serial data register mn (SDRmn) after the start bit is detected. If the last data (stop bit) is received before this latch timing, the data can be correctly received.

### 11.7.5 Procedure for processing errors that occurred during UART (UART0) communication

The procedure for processing errors that occurred during UART (UART0) communication is described in Figures 11-120 and 11-121.

**Figure 11-120. Processing Procedure in Case of Parity Error or Overrun Error**

Software Manipulation	Hardware Status	Remark
Reads serial data register mn (SDRmn).	→ The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data.	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		Error type is identified and the read value is used to clear error flag.
Writes 1 to serial flag clear trigger register mn (SIRmn).	→ Error flag is cleared.	Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification.

**Figure 11-121. Processing Procedure in Case of Framing Error**

Software Manipulation	Hardware Status	Remark
Reads serial data register mn (SDRmn).	→ The BFFmn bit of the SSRmn register is set to 0 and channel n is enabled to receive data.	This is to prevent an overrun error if the next reception is completed during error processing.
Reads serial status register mn (SSRmn).		Error type is identified and the read value is used to clear error flag.
Writes serial flag clear trigger register mn (SIRmn).	→ Error flag is cleared.	Error can be cleared only during reading, by writing the value read from the SSRmn register to the SIRmn register without modification.
Sets the STmn bit of serial channel stop register m (STm) to 1.	→ The SEmn bit of serial channel enable status register m (SEm) is set to 0 and channel n stops operating.	
Synchronization with other party of communication		Synchronization with the other party of communication is re-established and communication is resumed because it is considered that a framing error has occurred because the start bit has been shifted.
Sets the SSmn bit of serial channel start register m (SSm) to 1.	→ The SEmn bit of serial channel enable status register m (SEm) is set to 1 and channel n is enabled to operate.	

**Remark** m: Unit number (m = 0), n: Channel number (n = 0, 1), mn = 00, 01

## CHAPTER 12 SERIAL INTERFACE IICA

The R7F0C010 have two units of serial interface IICA and support two slave addresses.

When using IICA0 and IICA1 as slaves, the corresponding slave can communicate with the master when either one of the slave addresses matches an address received from the I<sup>2</sup>C bus.

### 12.1 Functions of Serial Interface IICA

Serial interface IICA has the following three modes.

#### (1) Operation stop mode

This mode is used when serial transfers are not performed. It can therefore be used to reduce power consumption.

#### (2) I<sup>2</sup>C bus mode (multimaster supported)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCLAn) line and a serial data bus (SDAAn) line.

This mode complies with the I<sup>2</sup>C bus format and the master device can generate “start condition”, “address”, “transfer direction specification”, “data”, and “stop condition” data to the slave device, via the serial data bus. The slave device automatically detects these received status and data by hardware. This function can simplify the part of application program that controls the I<sup>2</sup>C bus.

Since the SCLAn and SDAAn pins are used for open drain outputs, serial interface IICA requires pull-up resistors for the serial clock line and the serial data bus line.

#### (3) Wakeup mode

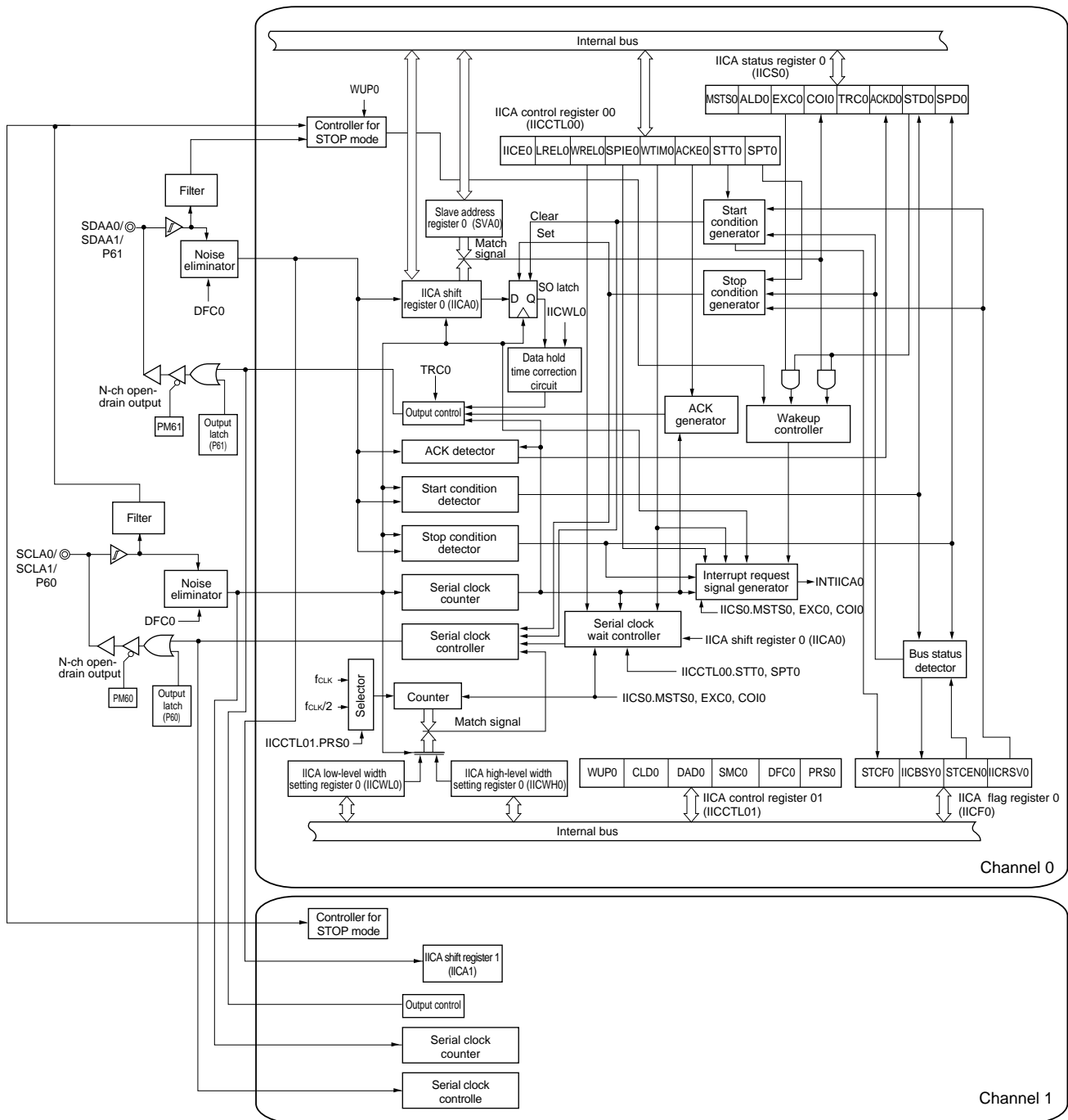
The STOP mode can be released by generating an interrupt request signal (INTIICAn) when an extension code from the master device or a local address has been received while in STOP mode. This can be set by using the WUPn bit of IICA control register n1 (IICCTLn1).

Figure 12-1 shows a block diagram of serial interface IICA.

**Remark** n = 0, 1



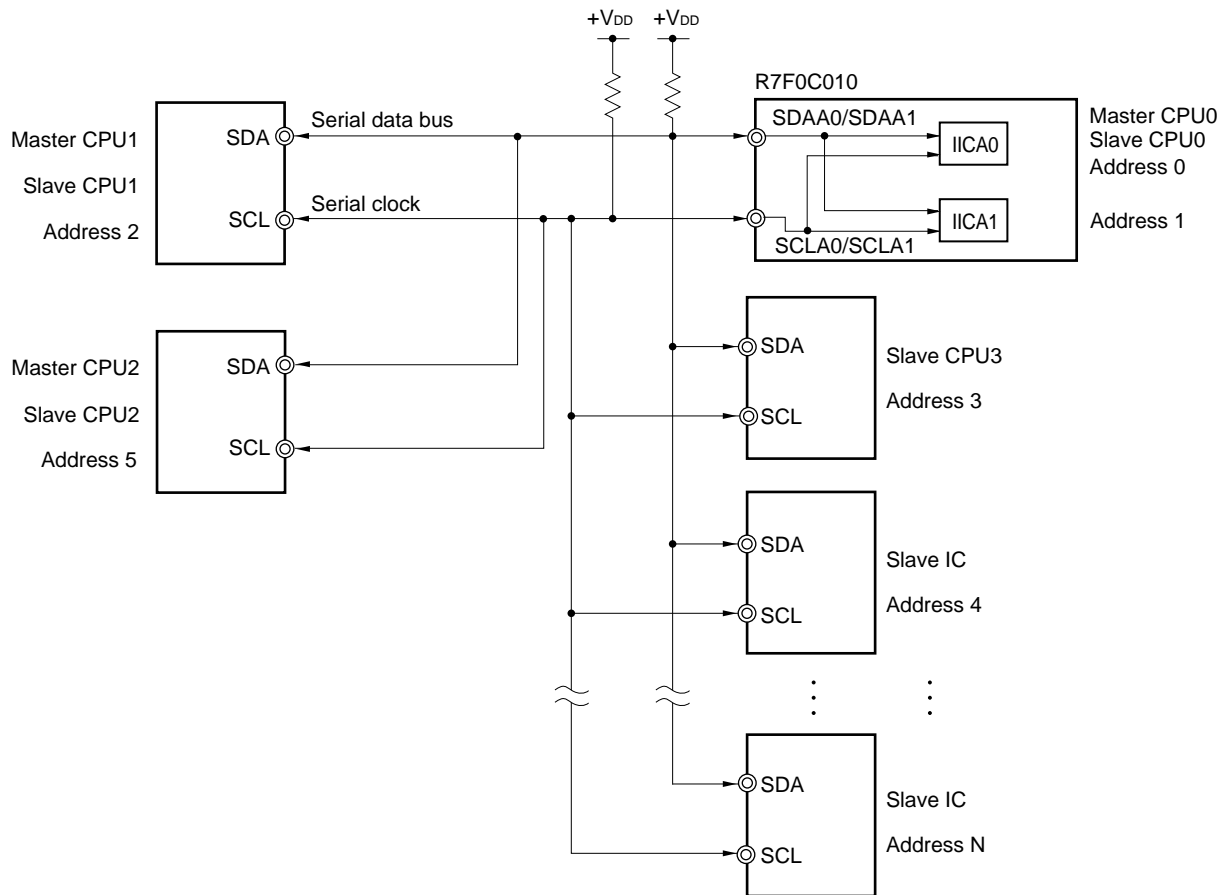
Figure 12-1. Block Diagram of Serial Interface IICA0, IICA1



**Caution** The R7F0C010 can wait simultaneously because they have channels 0 and 1. However, they cannot communicate simultaneously as they share the P60/SCLA0/SCLA1 and P61/SDAA0/SDAA1 pins.

Figure 12-2 shows a serial bus configuration example.

**Figure 12-2. Serial Bus Configuration Example Using I<sup>2</sup>C Bus**



**Remark** The R7F0C010 can communicate supporting two addresses as they have two units of serial interface IICA.

## 12.2 Configuration of Serial Interface IICA

Serial interface IICA includes the following hardware.

**Table 12-1. Configuration of Serial Interface IICA**

Item	Configuration
Registers	IICA shift register n (IICAn) Slave address register n (SVAn)
Control registers	Peripheral enable register 0 (PER0) IICA control register n0 (IICCTLn0) IICA status register n (IICSn) IICA flag register n (IICFn) IICA control register n1 (IICCTLn1) IICA low-level width setting register n (IICWLn) IICA high-level width setting register n (IICWHn) Port mode register 6 (PM6) Port register 6 (P6)

**Remark** n = 0, 1

### (1) IICA shift register n (IICAn)

The IICAn register is used to convert 8-bit serial data to 8-bit parallel data and vice versa in synchronization with the serial clock. The IICAn register can be used for both transmission and reception.

The actual transmit and receive operations can be controlled by writing and reading operations to the IICAn register. Cancel the wait state and start data transfer by writing data to the IICAn register during the wait period.

The IICAn register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears IICAn to 00H.

**Figure 12-3. Format of IICA Shift Register n (IICAn)**

Address: FFF50H (IICA0), FFF54H (IICA1) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IICAn								

- Cautions**
1. Do not write data to the IICAn register during data transfer.
  2. Write or read the IICAn register only during the wait period. Accessing the IICAn register in a communication state other than during the wait period is prohibited. When the device serves as the master, however, the IICAn register can be written only once after the communication trigger bit (STTn) is set to 1.
  3. When communication is reserved, write data to the IICAn register after the interrupt triggered by a stop condition is detected.

**Remark** n = 0, 1

**(2) Slave address register n (SVAn)**

This register stores seven bits of local addresses {A6, A5, A4, A3, A2, A1, A0} when in slave mode. The SVAn register can be set by an 8-bit memory manipulation instruction. However, rewriting to this register is prohibited while STDn = 1 (while the start condition is detected). Reset signal generation clears the SVAn register to 00H.

**Figure 12-4. Format of Slave Address Register n (SVAn)**

Address: F0234H (SVA0), F023DH (SVA1) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
SVAn	A6	A5	A4	A3	A2	A1	A0	0 <sup>Note</sup>

**Note** Bit 0 is fixed to 0.

**(3) SO latch**

The SO latch is used to retain the SDAAn pin's output level.

**(4) Wakeup controller**

This circuit generates an interrupt request (INTIICAn) when the address received by this register matches the address value set to the slave address register n (SVAn) or when an extension code is received.

**(5) Serial clock counter**

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

**(6) Interrupt request signal generator**

This circuit controls the generation of interrupt request signals (INTIICAn).

An I<sup>2</sup>C interrupt request is generated by the following two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by the WTIMn bit)
- Interrupt request generated when a stop condition is detected (set by the SPIEn bit)

**Remark** WTIMn bit: Bit 3 of IICA control register n0 (IICCTLn0)

SPIEn bit: Bit 4 of IICA control register n0 (IICCTLn0)

**(7) Serial clock controller**

In master mode, this circuit generates the clock output via the SCLAn pin from a sampling clock.

**(8) Serial clock wait controller**

This circuit controls the wait timing.

**(9) ACK generator, stop condition detector, start condition detector, and ACK detector**

These circuits generate and detect each status.

**(10) Data hold time correction circuit**

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

**Remark** n = 0, 1

**(11) Start condition generator**

This circuit generates a start condition when the STTn bit is set to 1.

However, in the communication reservation disabled status (IICRSVn bit = 1), when the bus is not released (IICBSYn bit = 1), start condition requests are ignored and the STCFn bit is set to 1.

**(12) Stop condition generator**

This circuit generates a stop condition when the SPTn bit is set to 1.

**(13) Bus status detector**

This circuit detects whether or not the bus is released by detecting start conditions and stop conditions.

However, as the bus status cannot be detected immediately following operation, the initial status is set by the STCENn bit.

- Remarks 1.**
- |              |  |
|--------------|--|
| STTn bit:    | Bit 1 of IICA control register n0 (IICCTLn0) |
| SPTn bit:    | Bit 0 of IICA control register n0 (IICCTLn0) |
| IICRSVn bit: | Bit 0 of IICA flag register n (IICFn)        |
| IICBSYn bit: | Bit 6 of IICA flag register n (IICFn)        |
| STCFn bit:   | Bit 7 of IICA flag register n (IICFn)        |
| STCENn bit:  | Bit 1 of IICA flag register n (IICFn)        |
- 2.** n = 0, 1

## 12.3 Registers Controlling Serial Interface IICA

Serial interface IICA is controlled by the following eight registers.

- Peripheral enable register 0 (PER0)
- IICA control register n0 (IICCTLn0)
- IICA flag register n (IICFn)
- IICA status register n (IICSn)
- IICA control register n1 (IICCTLn1)
- IICA low-level width setting register n (IICWLn)
- IICA high-level width setting register n (IICWHn)
- Port mode register 6 (PM6)
- Port register 6 (P6)

### 12.3.1 Peripheral enable register 0 (PER0)

This register is used to enable or disable supplying the clock to the peripheral hardware. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When serial interface IICAn is used, be sure to set bits 6 and 4 (IICA1EN, IICA0EN) of this register to 1.

The PER0 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 12-5. Format of Peripheral Enable Register 0 (PER0)**

Address: F00F0H After reset: 00H R/W

Symbol	7	<6>	<5>	<4>	3	<2>	1	<0>
PER0	0	IICA1EN	ADCEN	IICA0EN	0	SAU0EN	0	TAU0EN

IICAnEN	Control of serial interface IICAn input clock supply
0	Stops input clock supply. <ul style="list-style-type: none"> <li>• SFR used by serial interface IICAn cannot be written.</li> <li>• Serial interface IICAn is in the reset status.</li> </ul>
1	Enables input clock supply. <ul style="list-style-type: none"> <li>• SFR used by serial interface IICAn can be read/written.</li> </ul>

- Cautions**
1. When setting serial interface IICAn, be sure to set the IICAnEN bit to 1 first. If IICAnEN = 0, writing to a control register of serial interface IICAn is ignored, and, even if the register is read, only the default value is read (except for port mode register 6 (PM6) and port register 6 (P6)).
  2. When using IICA of this product as a master, setting both the IICA0EN and IICA1EN bits to 1 is prohibited because only one channel of IICA must be enabled to operate.
  3. To make IICA of this product correspond to two slave addresses, both the IICA0EN and IICA1EN bits must be set to 1.
  4. Be sure to clear bits 1, 3, and 7 to "0".

**Remark** n = 0, 1

### 12.3.2 IICA control register n0 (IICCTLn0)

This register is used to enable/stop I<sup>2</sup>C operations, set wait timing, and set other I<sup>2</sup>C operations.

The IICCTLn0 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, set the SPIEn, WTIMn, and ACKEn bits while IICEn = 0 or during the wait period. These bits can be set at the same time when the IICEn bit is set from “0” to “1”.

Reset signal generation clears this register to 00H.

**Remark** n = 0, 1

Figure 12-6. Format of IICA Control Register n0 (IICCTLn0) (1/4)

Address: F0230H (IICCTL00), F0238H (IICCTL10) After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICCTLn0	IICEn	LRELn	WRELn	SPIEn	WTIMn	ACKEn	STTn	SPTn

IICEn	I <sup>2</sup> C operation enable
0	Stop operation. Reset the IICA status register n (IICSn) <sup>Note 1</sup> . Stop internal operation.
1	Enable operation.
Be sure to set this bit (1) while the SCLAn and SDAAn lines are at high level.	
Condition for clearing (IICEn = 0)	
<ul style="list-style-type: none"> <li>• Cleared by instruction</li> <li>• Reset</li> </ul>	Condition for setting (IICEn = 1)
<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>	

LRELn <sup>Notes 2,3</sup>	Exit from communications
0	Normal operation
1	This exits from the current communications and sets standby mode. This setting is automatically cleared to 0 after being executed. Its uses include cases in which a locally irrelevant extension code has been received. The SCLAn and SDAAn lines are set to high impedance. The following flags of IICA control register n0 (IICCTLn0) and the IICA status register n (IICSn) are cleared to 0. • STTn • SPTn • MSTSn • EXCn • COIn • TRCn • ACKDn • STDn
The standby mode following exit from communications remains in effect until the following communications entry conditions are met.	
<ul style="list-style-type: none"> <li>• After a stop condition is detected, restart is in master mode.</li> <li>• An address match or extension code reception occurs after the start condition.</li> </ul>	
Condition for clearing (LRELn = 0)	
<ul style="list-style-type: none"> <li>• Automatically cleared after execution</li> <li>• Reset</li> </ul>	Condition for setting (LRELn = 1)
<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>	

WRELn <sup>Notes 2,3</sup>	Wait cancellation
0	Do not cancel wait
1	Cancel wait. This setting is automatically cleared after wait is canceled.
When the WRELn bit is set (wait canceled) during the wait period at the ninth clock pulse in the transmission status (TRCn = 1), the SDAAn line goes into the high impedance state (TRCn = 0).	
Condition for clearing (WRELn = 0)	
<ul style="list-style-type: none"> <li>• Automatically cleared after execution</li> <li>• Reset</li> </ul>	Condition for setting (WRELn = 1)
<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>	

**Notes 1.** The IICA status register n (IICSn), the STCFn and IICBSYn bits of the IICA flag register n (IICFn), and the CLDn and DADn bits of IICA control register n1 (IICCTLn1) are reset.

**2.** The signal of this bit is invalid while IICEn is 0.

**3.** When the LRELn and WRELn bits are read, 0 is always read.

**Caution** If the operation of I<sup>2</sup>C is enabled (IICEn = 1) when the SCLAn line is high level, the SDAAn line is low level, and the digital filter is turned on (DFCn bit of IICCTLn1 register = 1), a start condition will be inadvertently detected immediately. In this case, set (1) the LRELn bit by using a 1-bit memory manipulation instruction immediately after enabling operation of I<sup>2</sup>C (IICEn = 1).

**Remark** n = 0, 1



Figure 12-6. Format of IICA Control Register n0 (IICCTLn0) (2/4)

SPIEn <sup>Note 1</sup>	Enable/disable generation of interrupt request when stop condition is detected	
0	Disable	
1	Enable	
If the WUPn bit of IICA control register n1 (IICCTLn1) is 1, no stop condition interrupt will be generated even if SPIEn = 1.		
Condition for clearing (SPIEn = 0)		Condition for setting (SPIEn = 1)
<ul style="list-style-type: none"> <li>• Cleared by instruction</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>

WTIMn <sup>Note 1</sup>	Control of wait and interrupt request generation	
0	Interrupt request is generated at the eighth clock's falling edge. Master mode: After output of eight clocks, clock output is set to low level and wait is set. Slave mode: After input of eight clocks, the clock is set to low level and wait is set for master device.	
1	Interrupt request is generated at the ninth clock's falling edge. Master mode: After output of nine clocks, clock output is set to low level and wait is set. Slave mode: After input of nine clocks, the clock is set to low level and wait is set for master device.	
An interrupt is generated at the falling edge of the ninth clock during address transfer independently of the setting of this bit. The setting of this bit is valid when the address transfer is completed. When in master mode, a wait is inserted at the falling edge of the ninth clock during address transfers. For a slave device that has received a local address, a wait is inserted at the falling edge of the ninth clock after an acknowledge (ACK) is issued. However, when the slave device has received an extension code, a wait is inserted at the falling edge of the eighth clock.		
Condition for clearing (WTIMn = 0)		Condition for setting (WTIMn = 1)
<ul style="list-style-type: none"> <li>• Cleared by instruction</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>

ACKEn <sup>Notes 1, 2</sup>	Acknowledgment control	
0	Disable acknowledgment.	
1	Enable acknowledgment. During the ninth clock period, the SDAAn line is set to low level.	
Condition for clearing (ACKEn = 0)		Condition for setting (ACKEn = 1)
<ul style="list-style-type: none"> <li>• Cleared by instruction</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>

- Notes 1.** The signal of this bit is invalid while IICEn is 0. Set this bit during that period.
- 2.** The set value is invalid during address transfer and if the code is not an extension code. When the device serves as a slave and the addresses match, an acknowledgment is generated regardless of the set value.

**Remark** n = 0, 1

Figure 12-6. Format of IICA Control Register n0 (IICCTLn0) (3/4)

STTn <sup>Note</sup>	Start condition trigger				
0	Do not generate a start condition.				
1	<p>When bus is released (in standby state, when IICBSYn = 0): If this bit is set (1), a start condition is generated (startup as the master).</p> <p>When a third party is communicating:</p> <ul style="list-style-type: none"> <li>• When communication reservation function is enabled (IICRSVn = 0) Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released.</li> <li>• When communication reservation function is disabled (IICRSVn = 1) Even if this bit is set (1), the STTn bit is cleared and the STTn clear flag (STCFn) is set (1). No start condition is generated.</li> </ul> <p>In the wait state (when master device): Generates a restart condition after releasing the wait.</p>				
<p>Cautions concerning set timing</p> <ul style="list-style-type: none"> <li>• For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when the ACKEn bit has been cleared to 0 and slave has been notified of final reception.</li> <li>• For master transmission: A start condition cannot be generated normally during the acknowledge period. Set to 1 during the wait period that follows output of the ninth clock.</li> <li>• Cannot be set to 1 at the same time as stop condition trigger (SPTn).</li> <li>• Once STTn is set (1), setting it again (1) before the clear condition is met is not allowed.</li> </ul>					
<table border="1"> <thead> <tr> <th>Condition for clearing (STTn = 0)</th> <th>Condition for setting (STTn = 1)</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> <li>• Cleared by setting the STTn bit to 1 while communication reservation is prohibited.</li> <li>• Cleared by loss in arbitration</li> <li>• Cleared after start condition is generated by master device</li> <li>• Cleared by LRELn = 1 (exit from communications)</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul> </td> <td> <ul style="list-style-type: none"> <li>• Set by instruction</li> </ul> </td> </tr> </tbody> </table>		Condition for clearing (STTn = 0)	Condition for setting (STTn = 1)	<ul style="list-style-type: none"> <li>• Cleared by setting the STTn bit to 1 while communication reservation is prohibited.</li> <li>• Cleared by loss in arbitration</li> <li>• Cleared after start condition is generated by master device</li> <li>• Cleared by LRELn = 1 (exit from communications)</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul>	<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>
Condition for clearing (STTn = 0)	Condition for setting (STTn = 1)				
<ul style="list-style-type: none"> <li>• Cleared by setting the STTn bit to 1 while communication reservation is prohibited.</li> <li>• Cleared by loss in arbitration</li> <li>• Cleared after start condition is generated by master device</li> <li>• Cleared by LRELn = 1 (exit from communications)</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul>	<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>				

**Note** The signal of this bit is invalid while IICEn is 0.

- Remarks**
1. Bit 1 (STTn) becomes 0 when it is read after data setting.
  2. IICRSVn: Bit 0 of IIC flag register n (IICFn)  
STCFn: Bit 7 of IIC flag register n (IICFn)
  3. n = 0, 1

Figure 12-6. Format of IICA Control Register n0 (IICCTLn0) (4/4)

SPTn	Stop condition trigger	
0	Stop condition is not generated.	
1	Stop condition is generated (termination of master device's transfer).	
Cautions concerning set timing <ul style="list-style-type: none"> <li>• For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when the ACKEn bit has been cleared to 0 and slave has been notified of final reception.</li> <li>• For master transmission: A stop condition cannot be generated normally during the acknowledge period. Therefore, set it during the wait period that follows output of the ninth clock.</li> <li>• Cannot be set to 1 at the same time as start condition trigger (STTn).</li> <li>• The SPTn bit can be set to 1 only when in master mode.</li> <li>• When the WTIMn bit has been cleared to 0, if the SPTn bit is set to 1 during the wait period that follows output of eight clocks, note that a stop condition will be generated during the high-level period of the ninth clock. The WTIMn bit should be changed from 0 to 1 during the wait period following the output of eight clocks, and the SPTn bit should be set to 1 during the wait period that follows the output of the ninth clock.</li> <li>• Once SPTn is set (1), setting it again (1) before the clear condition is met is not allowed.</li> </ul>		
Condition for clearing (SPTn = 0)		Condition for setting (SPTn = 1)
<ul style="list-style-type: none"> <li>• Cleared by loss in arbitration</li> <li>• Automatically cleared after stop condition is detected</li> <li>• Cleared by LRELn = 1 (exit from communications)</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• Set by instruction</li> </ul>

**Caution** When bit 3 (TRCn) of the IICA status register n (IICSn) is set to 1 (transmission status), bit 5 (WRELn) of IICA control register n0 (IICCTLn0) is set to 1 during the ninth clock and wait is canceled, after which the TRCn bit is cleared (reception status) and the SDAAn line is set to high impedance. Release the wait performed while the TRCn bit is 1 (transmission status) by writing to the IICA shift register n.

**Remarks**

1. Bit 0 (SPTn) becomes 0 when it is read after data setting.
2. n = 0, 1

### 12.3.3 IICA status register n (IICSn)

This register indicates the status of I<sup>2</sup>C.

The IICSn register is read by a 1-bit or 8-bit memory manipulation instruction only when STTn = 1 and during the wait period.

Reset signal generation clears this register to 00H.

**Caution** Reading the IICSn register while the address match wakeup function is enabled (WUPn = 1) in STOP mode is prohibited. When the WUPn bit is changed from 1 to 0 (wakeup operation is stopped), regardless of the INTIICAn interrupt request, the change in status is not reflected until the next start condition or stop condition is detected. To use the wakeup function, therefore, enable (SPIEn = 1) the interrupt generated by detecting a stop condition and read the IICSn register after the interrupt has been detected.

**Remark** STTn: Bit 1 of IICA control register n0 (IICCTLn0)  
WUPn: Bit 7 of IICA control register n1 (IICCTLn1)

**Figure 12-7. Format of IICA Status Register n (IICSn) (1/3)**

Address: FFF51H (IICS0), FFF55H (IICS1) After reset: 00H R

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IICSn	MSTS <sub>n</sub>	ALD <sub>n</sub>	EXC <sub>n</sub>	COIn	TRC <sub>n</sub>	ACKD <sub>n</sub>	STD <sub>n</sub>	SPD <sub>n</sub>

MSTS <sub>n</sub>	Master status check flag
0	Slave device status or communication standby status
1	Master device communication status
Condition for clearing (MSTS <sub>n</sub> = 0)	
<ul style="list-style-type: none"> <li>When a stop condition is detected</li> <li>When ALD<sub>n</sub> = 1 (arbitration loss)</li> <li>Cleared by LREL<sub>n</sub> = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>	
Condition for setting (MSTS <sub>n</sub> = 1)	
<ul style="list-style-type: none"> <li>When a start condition is generated</li> </ul>	

ALD <sub>n</sub>	Detection of arbitration loss
0	This status means either that there was no arbitration or that the arbitration result was a "win".
1	This status indicates the arbitration result was a "loss". The MSTS <sub>n</sub> bit is cleared.
Condition for clearing (ALD <sub>n</sub> = 0)	
<ul style="list-style-type: none"> <li>Automatically cleared after the IICSn register is read<sup>Note</sup></li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>	
Condition for setting (ALD <sub>n</sub> = 1)	
<ul style="list-style-type: none"> <li>When the arbitration result is a "loss".</li> </ul>	

**Note** This register is also cleared when a 1-bit memory manipulation instruction is executed for bits other than the IICSn register. Therefore, when using the ALD<sub>n</sub> bit, read the data of this bit before the data of the other bits.

**Remarks** 1. LREL<sub>n</sub>: Bit 6 of IICA control register n0 (IICCTLn0)  
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)  
2. n = 0, 1

Figure 12-7. Format of IICA Status Register n (IICSn) (2/3)

EXCn	Detection of extension code reception	
0	Extension code was not received.	
1	Extension code was received.	
Condition for clearing (EXCn = 0)		Condition for setting (EXCn = 1)
<ul style="list-style-type: none"> <li>When a start condition is detected</li> <li>When a stop condition is detected</li> <li>Cleared by LRELn = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>When the higher four bits of the received address data is either "0000" or "1111" (set at the rising edge of the eighth clock).</li> </ul>

COIn	Detection of matching addresses	
0	Addresses do not match.	
1	Addresses match.	
Condition for clearing (COIn = 0)		Condition for setting (COIn = 1)
<ul style="list-style-type: none"> <li>When a start condition is detected</li> <li>When a stop condition is detected</li> <li>Cleared by LRELn = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>When the received address matches the local address (slave address register n (SVAn)) (set at the rising edge of the eighth clock).</li> </ul>

TRCn	Detection of transmit/receive status	
0	Receive status (other than transmit status). The SDAAn line is set for high impedance.	
1	Transmit status. The value in the SOn latch is enabled for output to the SDAAn line (valid starting at the falling edge of the first byte's ninth clock).	
Condition for clearing (TRCn = 0)		Condition for setting (TRCn = 1)
<p>&lt;Both master and slave&gt;</p> <ul style="list-style-type: none"> <li>When a stop condition is detected</li> <li>Cleared by LRELn = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Cleared by WRELn = 1<sup>Note</sup> (wait cancel)</li> <li>When the ALDn bit changes from 0 to 1 (arbitration loss)</li> <li>Reset</li> <li>When not used for communication (MSTSn, EXCn, COIn = 0)</li> </ul> <p>&lt;Master&gt;</p> <ul style="list-style-type: none"> <li>When "1" is output to the first byte's LSB (transfer direction specification bit)</li> </ul> <p>&lt;Slave&gt;</p> <ul style="list-style-type: none"> <li>When a start condition is detected</li> <li>When "0" is input to the first byte's LSB (transfer direction specification bit)</li> </ul>		<p>&lt;Master&gt;</p> <ul style="list-style-type: none"> <li>When a start condition is generated</li> <li>When 0 (master transmission) is output to the LSB (transfer direction specification bit) of the first byte (during address transfer)</li> </ul> <p>&lt;Slave&gt;</p> <ul style="list-style-type: none"> <li>When 1 (slave transmission) is input to the LSB (transfer direction specification bit) of the first byte from the master (during address transfer)</li> </ul>

**Note** When bit 3 (TRCn) of the IICA status register n (IICSn) is set to 1 (transmission status), bit 5 (WRELn) of IICA control register n0 (IICCTLn0) is set to 1 during the ninth clock and wait is canceled, after which the TRCn bit is cleared (reception status) and the SDAAn line is set to high impedance. Release the wait performed while the TRCn bit is 1 (transmission status) by writing to the IICA shift register n.

**Remarks**

1. LRELn: Bit 6 of IICA control register n0 (IICCTLn0)  
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)
2. n = 0, 1

Figure 12-7. Format of IICA Status Register n (IICSn) (3/3)

ACKDn	Detection of acknowledge ( $\overline{\text{ACK}}$ )	
0	Acknowledge was not detected.	
1	Acknowledge was detected.	
Condition for clearing (ACKDn = 0)		Condition for setting (ACKDn = 1)
<ul style="list-style-type: none"> <li>When a stop condition is detected</li> <li>At the rising edge of the next byte's first clock</li> <li>Cleared by LRELn = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>After the SDAAn line is set to low level at the rising edge of SCLAn line's ninth clock</li> </ul>

STDn	Detection of start condition	
0	Start condition was not detected.	
1	Start condition was detected. This indicates that the address transfer period is in effect.	
Condition for clearing (STDn = 0)		Condition for setting (STDn = 1)
<ul style="list-style-type: none"> <li>When a stop condition is detected</li> <li>At the rising edge of the next byte's first clock following address transfer</li> <li>Cleared by LRELn = 1 (exit from communications)</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>When a start condition is detected</li> </ul>

SPDn	Detection of stop condition	
0	Stop condition was not detected.	
1	Stop condition was detected. The master device's communication is terminated and the bus is released.	
Condition for clearing (SPDn = 0)		Condition for setting (SPDn = 1)
<ul style="list-style-type: none"> <li>At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition</li> <li>When the WUPn bit changes from 1 to 0</li> <li>When the IICEn bit changes from 1 to 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>When a stop condition is detected</li> </ul>

**Remarks 1.** LRELn: Bit 6 of IICA control register n0 (IICCTLn0)

IICEn: Bit 7 of IICA control register n0 (IICCTLn0)

2. n = 0, 1

### 12.3.4 IICA flag register n (IICFn)

This register sets the operation mode of I<sup>2</sup>C and indicates the status of the I<sup>2</sup>C bus.

The IICFn register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the STTn clear flag (STCFn) and I<sup>2</sup>C bus status flag (IICBSYn) bits are read-only.

The IICRSVn bit can be used to enable/disable the communication reservation function.

The STCENn bit can be used to set the initial value of the IICBSYn bit.

The IICRSVn and STCENn bits can be written only when the operation of I<sup>2</sup>C is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) = 0). When operation is enabled, the IICFn register can be read.

Reset signal generation clears this register to 00H.

Figure 12-8. Format of IICA Flag Register n (IICFn)

Address: FFF52H (IICF0), FFF56H (IICF1)    After reset: 00H    R/W<sup>Note</sup>

Symbol	<7>	<6>	5	4	3	2	<1>	<0>
IICFn	STCFn	IICBSYn	0	0	0	0	STCENn	IICRSVn

STCFn	STTn clear flag	
0	Generate start condition	
1	Start condition generation unsuccessful: clear the STTn flag	
Condition for clearing (STCFn = 0)		Condition for setting (STCFn = 1)
<ul style="list-style-type: none"> <li>Cleared by STTn = 1</li> <li>When IICEn = 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>Generating start condition unsuccessful and the STTn bit cleared to 0 when communication reservation is disabled (IICRSVn = 1).</li> </ul>

IICBSYn	I <sup>2</sup> C bus status flag	
0	Bus release status (communication initial status when STCENn = 1)	
1	Bus communication status (communication initial status when STCENn = 0)	
Condition for clearing (IICBSYn = 0)		Condition for setting (IICBSYn = 1)
<ul style="list-style-type: none"> <li>Detection of stop condition</li> <li>When IICEn = 0 (operation stop)</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>Detection of start condition</li> <li>Setting of the IICEn bit when STCENn = 0</li> </ul>

STCENn	Initial start enable trigger	
0	After operation is enabled (IICEn = 1), enable generation of a start condition upon detection of a stop condition.	
1	After operation is enabled (IICEn = 1), enable generation of a start condition without detecting a stop condition.	
Condition for clearing (STCENn = 0)		Condition for setting (STCENn = 1)
<ul style="list-style-type: none"> <li>Cleared by instruction</li> <li>Detection of start condition</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>Set by instruction</li> </ul>

IICRSVn	Communication reservation function disable bit	
0	Enable communication reservation	
1	Disable communication reservation	
Condition for clearing (IICRSVn = 0)		Condition for setting (IICRSVn = 1)
<ul style="list-style-type: none"> <li>Cleared by instruction</li> <li>Reset</li> </ul>		<ul style="list-style-type: none"> <li>Set by instruction</li> </ul>

**Note** Bits 6 and 7 are read-only.

- Cautions**
1. Write to the STCENn bit only when the operation is stopped (IICEn = 0).
  2. As the bus release status (IICBSYn = 0) is recognized regardless of the actual bus status when STCENn = 1, when generating the first start condition (STTn = 1), it is necessary to verify that no third party communications are in progress in order to prevent such communications from being destroyed.
  3. Write to IICRSVn only when the operation is stopped (IICEn = 0).

- Remarks**
1. STTn: Bit 1 of IICA control register n0 (IICCTLn0)  
IICEn: Bit 7 of IICA control register n0 (IICCTLn0)
  2. n = 0, 1

**12.3.5 IICA control register n1 (IICCTLn1)**

This register is used to set the operation mode of I<sup>2</sup>C and detect the statuses of the SCLAn and SDAAn pins.

The IICCTLn1 register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the CLDn and DADn bits are read-only.

Set the IICCTLn1 register, except the WUPn bit, while operation of I<sup>2</sup>C is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation clears this register to 00H.

**Figure 12-9. Format of IICA Control Register n1 (IICCTLn1) (1/2)**

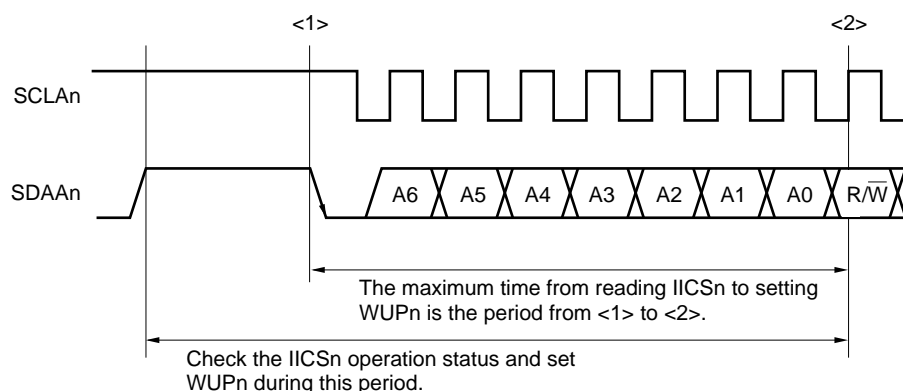
Address: F0231H (IICCTL01), F0239H (IICCTL11) After reset: 00H R/W<sup>Note 1</sup>

Symbol	<7>	6	<5>	<4>	<3>	<2>	1	<0>
IICCTLn1	WUPn	0	CLDn	DADn	SMCn	DFCn	0	PRSn

WUPn	Control of address match wakeup
0	Stops operation of address match wakeup function in STOP mode.
1	Enables operation of address match wakeup function in STOP mode.
<p>To shift to STOP mode when WUPn = 1, execute the STOP instruction at least three clocks after setting (1) the WUPn bit (see <b>Figure 12-22 Flow When Setting WUPn = 1</b>).</p> <p>Clear (0) the WUPn bit after the address has matched or an extension code has been received. The subsequent communication can be entered by the clearing (0) WUPn bit. (The wait must be released and transmit data must be written after the WUPn bit has been cleared (0).)</p> <p>The interrupt timing when the address has matched or when an extension code has been received, while WUPn = 1, is identical to the interrupt timing when WUPn = 0. (A delay of the difference of sampling by the clock will occur.) Furthermore, when WUPn = 1, a stop condition interrupt is not generated even if the SPIEn bit is set to 1.</p>	
Condition for clearing (WUPn = 0)	Condition for setting (WUPn = 1)
<ul style="list-style-type: none"> <li>• Cleared by instruction (after address match or extension code reception)</li> </ul>	<ul style="list-style-type: none"> <li>• Set by instruction (when the MSTSn, EXCn, and COIn bits are "0", and the STDn bit also "0" (communication not entered))<sup>Note 2</sup></li> </ul>

**Notes 1.** Bits 4 and 5 are read-only.

**2.** The status of the IICA status register n (IICCSn) must be checked and the WUPn bit must be set during the period shown below.



**Remark** n = 0, 1



Figure 12-9. Format of IICA Control Register n1 (IICCTLn1) (2/2)

CLDn	Detection of SCLAn pin level (valid only when IICEn = 1)	
0	The SCLAn pin was detected at low level.	
1	The SCLAn pin was detected at high level.	
Condition for clearing (CLDn = 0)		Condition for setting (CLDn = 1)
<ul style="list-style-type: none"> <li>• When the SCLAn pin is at low level</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• When the SCLAn pin is at high level</li> </ul>

DADn	Detection of SDAAn pin level (valid only when IICEn = 1)	
0	The SDAAn pin was detected at low level.	
1	The SDAAn pin was detected at high level.	
Condition for clearing (DADn = 0)		Condition for setting (DADn = 1)
<ul style="list-style-type: none"> <li>• When the SDAAn pin is at low level</li> <li>• When IICEn = 0 (operation stop)</li> <li>• Reset</li> </ul>		<ul style="list-style-type: none"> <li>• When the SDAAn pin is at high level</li> </ul>

SMCn	Operation mode switching	
0	Operates in standard mode (fastest transfer rate: 100 kbps).	
1	Operates in fast mode (fastest transfer rate: 400 kbps) or fast mode plus (fastest transfer rate: 1 Mbps).	

DFCn	Digital filter operation control	
0	Digital filter off.	
1	Digital filter on.	
<p>Digital filter can be used only in fast mode and fast mode plus.            In fast mode and fast mode plus, the transfer clock does not vary, regardless of the DFCn bit being set (1) or cleared (0).            The digital filter is used for noise elimination in fast mode and fast mode plus.</p>		

PRSn	Division of the operation clock	
0	Selects $f_{CLK}$ as operation clock.	
1	Selects $f_{CLK}/2$ as operation clock.	

**Caution** The fastest operation frequency of the operation clock of the serial interface IICA is 20 MHz (Max.). If the  $f_{CLK}$  exceeds 20 MHz, set the clock to  $f_{CLK}/2$  by setting the PRSn bit to 1.

**Remarks**

1. IICEn: Bit 7 of IICA control register n0 (IICCTLn0)
2. n = 0, 1

### 12.3.6 IICA low-level width setting register n (IICWLn)

This register is used to set the low-level width ( $t_{LOW}$ ) of the SCLAn pin signal that is output by serial interface IICA. The data hold time is decided by value the higher 6 bits of IICWL register.

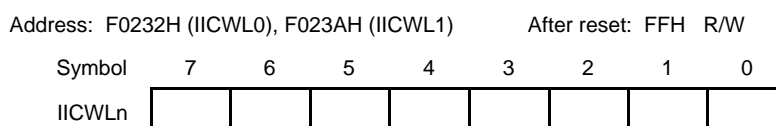
The IICWLn register can be set by an 8-bit memory manipulation instruction.

Set the IICWLn register while operation of I<sup>2</sup>C is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation sets this register to FFH.

For details about setting the IICWLn register, see **12.4.2 Setting transfer clock by using IICWLn and IICWHn registers.**

**Figure 12-10. Format of IICA Low-Level Width Setting Register n (IICWLn)**



### 12.3.7 IICA high-level width setting register n (IICWHn)

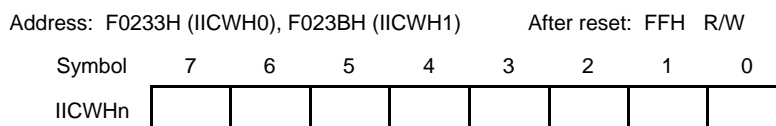
This register is used to set the high-level width of the SCLAn pin signal that is output by serial interface IICA.

The IICWHn register can be set by an 8-bit memory manipulation instruction.

Set the IICWHn register while operation of I<sup>2</sup>C is disabled (bit 7 (IICEn) of IICA control register n0 (IICCTLn0) is 0).

Reset signal generation sets this register to FFH.

**Figure 12-11. Format of IICA High-Level Width Setting Register n (IICWHn)**



**Remarks 1.** For how to set the transfer clock by using the IICWLn and IICWHn registers, see **12.4.2 Setting transfer clock by using IICWLn and IICWHn registers.**

**2.** n = 0, 1

### 12.3.8 Port mode register 6 (PM6)

This register sets the input/output of port 6 in 1-bit units.

When using the P60/SCLA0/SCLA1 pin as clock I/O and the P61/SDAA0/SDAA1 pin as serial data I/O, clear PM60 and PM61 to 0, and set the output latches of P60 and P61 to 1.

Set the IICEn bit (bit 7 of IICA control register n0 (IICCTLn0)) to 1 before setting the output mode because the P60/SCLA0/SCLA1 and P61/SDAA0/SDAA1 pins output a low level (fixed) when the IICEn bit is 0.

The PM6 register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

**Figure 12-12. Format of Port Mode Register 6 (PM6)**

Address: FFF26H    After reset: FFH    R/W

Symbol	7	6	5	4	3	2	1	0
PM6	1	1	1	1	1	1	PM61	PM60

PM6n	P6n pin I/O mode selection (n = 0, 1)
0	Output mode (output buffer on)
1	Input mode (output buffer off)

## 12.4 I<sup>2</sup>C Bus Mode Functions

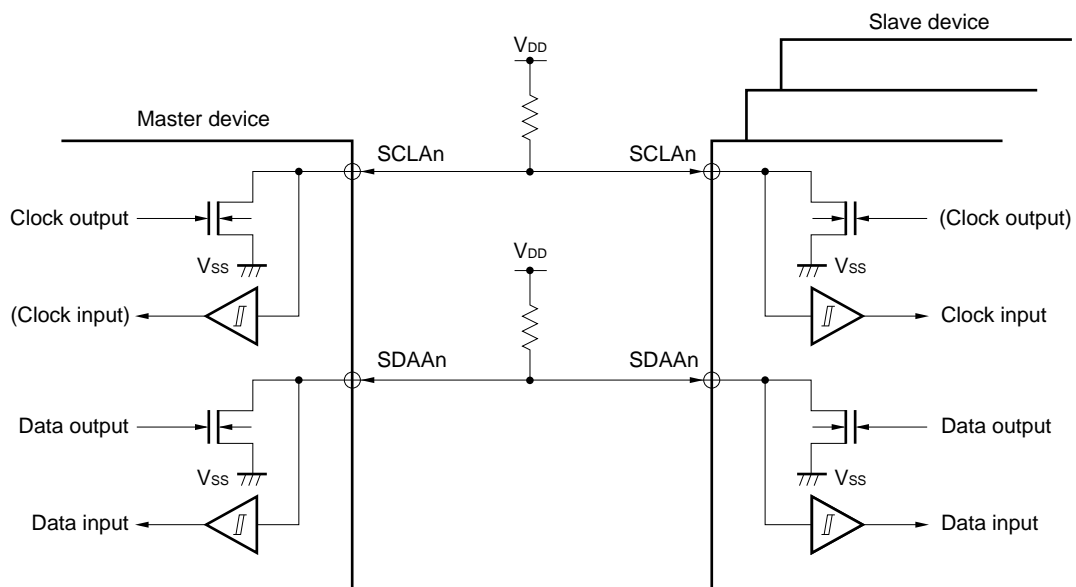
### 12.4.1 Pin configuration

The serial clock pin (SCLAn) and the serial data bus pin (SDAAn) are configured as follows.

- (1) SCLAn .... This pin is used for serial clock input and output.  
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.
- (2) SDAAn .... This pin is used for serial data input and output.  
This pin is an N-ch open-drain output for both master and slave devices. Input is Schmitt input.

Since outputs from the serial clock line and the serial data bus line are N-ch open-drain outputs, an external pull-up resistor is required.

Figure 12-13. Pin Configuration Diagram



**Remark** n = 0, 1

### 12.4.2 Setting transfer clock by using IICWLn and IICWHn registers

#### (1) Setting transfer clock on master side

$$\text{Transfer clock} = \frac{f_{\text{CLK}}}{\text{IICWL0} + \text{IICWH0} + f_{\text{CLK}}(t_{\text{R}} + t_{\text{F}})}$$

At this time, the optimal setting values of the IICWLn and IICWHn registers are as follows.  
(The fractional parts of all setting values are rounded up.)

- When the fast mode

$$\begin{aligned} \text{IICWLn} &= \frac{0.52}{\text{Transfer clock}} \times f_{\text{CLK}} \\ \text{IICWHn} &= \left( \frac{0.48}{\text{Transfer clock}} - t_{\text{R}} - t_{\text{F}} \right) \times f_{\text{CLK}} \end{aligned}$$

- When the normal mode

$$\begin{aligned} \text{IICWLn} &= \frac{0.47}{\text{Transfer clock}} \times f_{\text{CLK}} \\ \text{IICWHn} &= \left( \frac{0.53}{\text{Transfer clock}} - t_{\text{R}} - t_{\text{F}} \right) \times f_{\text{CLK}} \end{aligned}$$

- When the fast mode plus

$$\begin{aligned} \text{IICWLn} &= \frac{0.50}{\text{Transfer clock}} \times f_{\text{CLK}} \\ \text{IICWHn} &= \left( \frac{0.50}{\text{Transfer clock}} - t_{\text{R}} - t_{\text{F}} \right) \times f_{\text{CLK}} \end{aligned}$$

#### (2) Setting IICWLn and IICWHn registers on slave side

(The fractional parts of all setting values are truncated.)

- When the fast mode

$$\begin{aligned} \text{IICWLn} &= 1.3 \mu\text{s} \times f_{\text{CLK}} \\ \text{IICWHn} &= (1.2 \mu\text{s} - t_{\text{R}} - t_{\text{F}}) \times f_{\text{CLK}} \end{aligned}$$

- When the normal mode

$$\begin{aligned} \text{IICWLn} &= 4.7 \mu\text{s} \times f_{\text{CLK}} \\ \text{IICWHn} &= (5.3 \mu\text{s} - t_{\text{R}} - t_{\text{F}}) \times f_{\text{CLK}} \end{aligned}$$

- When the fast mode plus

$$\begin{aligned} \text{IICWLn} &= 0.50 \mu\text{s} \times f_{\text{CLK}} \\ \text{IICWHn} &= (0.50 \mu\text{s} - t_{\text{R}} - t_{\text{F}}) \times f_{\text{CLK}} \end{aligned}$$

(**Caution** and **Remarks** are listed on the next page.)

**Caution** Note the minimum  $f_{CLK}$  operation frequency when setting the transfer clock. The minimum  $f_{CLK}$  operation frequency for serial interface IICA is determined according to the mode.

**Fast mode:**  $f_{CLK} = 3.5 \text{ MHz (MIN.)}$

**Fast mode plus:**  $f_{CLK} = 10 \text{ MHz (MIN.)}$

**Normal mode:**  $f_{CLK} = 1 \text{ MHz (MIN.)}$

In addition, the fastest operation frequency of the operation clock of the serial interface IICA is 20 MHz (Max.). If the  $f_{CLK}$  exceeds 20 MHz, set the clock to  $f_{CLK}/2$  by setting the PRSn bit of IICCTLn1 register to 1.

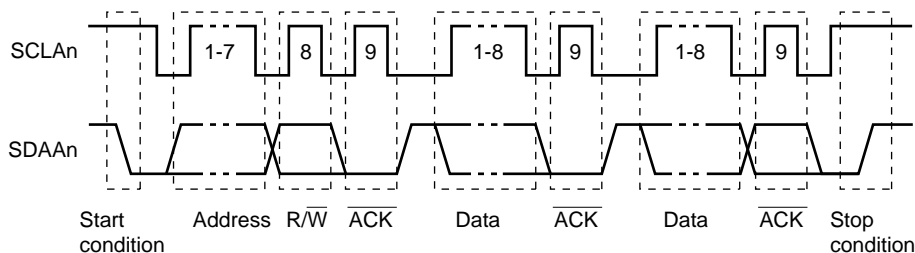
**Remarks 1.** Calculate the rise time ( $t_R$ ) and fall time ( $t_F$ ) of the SDAAn and SCLAn signals separately, because they differ depending on the pull-up resistance and wire load.

2. IICWLn: IICA low-level width setting register n  
IICWHn: IICA high-level width setting register n  
 $t_F$ : SDAAn and SCLAn signal falling times  
 $t_R$ : SDAAn and SCLAn signal rising times  
 $f_{CLK}$ : CPU/peripheral hardware clock frequency
3.  $n = 0, 1$

## 12.5 I<sup>2</sup>C Bus Definitions and Control Methods

The following section describes the I<sup>2</sup>C bus's serial data communication format and the signals used by the I<sup>2</sup>C bus. Figure 12-14 shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I<sup>2</sup>C bus's serial data bus.

Figure 12-14. I<sup>2</sup>C Bus Serial Data Transfer Timing



The master device generates the start condition, slave address, and stop condition.

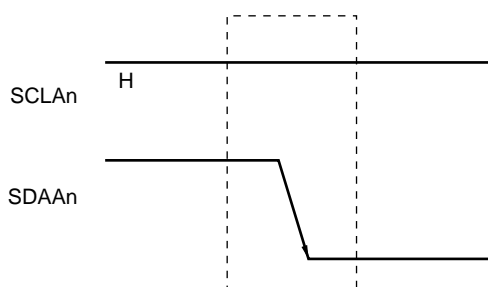
The acknowledge ( $\overline{\text{ACK}}$ ) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCLAn) is continuously output by the master device. However, in the slave device, the SCLAn pin low level period can be extended and a wait can be inserted.

### 12.5.1 Start conditions

A start condition is met when the SCLAn pin is at high level and the SDAAn pin changes from high level to low level. The start conditions for the SCLAn pin and SDAAn pin are signals that the master device generates to the slave device when starting a serial transfer. When the device is used as a slave, start conditions can be detected.

Figure 12-15. Start Conditions



A start condition is output when bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set (1) after a stop condition has been detected (SPDn: Bit 0 of the IICA status register n (IICSn) = 1). When a start condition is detected, bit 1 (STDn) of the IICSn register is set (1).

**Remark** n = 0, 1

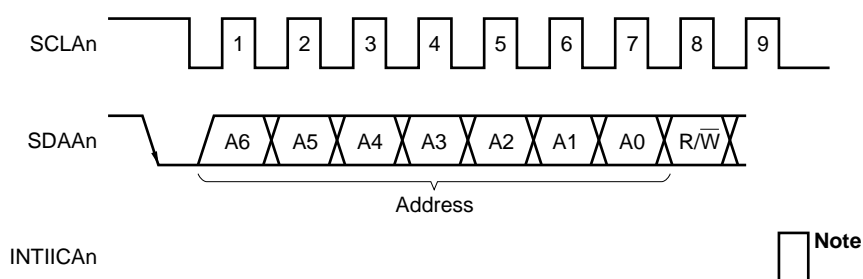
### 12.5.2 Addresses

The address is defined by the 7 bits of data that follow the start condition.

An address is a 7-bit data segment that is output in order to select one of the slave devices that are connected to the master device via the bus lines. Therefore, each slave device connected via the bus lines must have a unique address.

The slave devices include hardware that detects the start condition and checks whether or not the 7-bit address data matches the data values stored in the slave address register  $n$  (SVAn). If the address data matches the SVAn register values, the slave device is selected and communicates with the master device until the master device generates a start condition or stop condition.

Figure 12-16. Address



**Note** INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

Addresses are output when a total of 8 bits consisting of the slave address and the transfer direction described in **12.5.3 Transfer direction specification** are written to the IICA shift register  $n$  (IICAn). The received addresses are written to the IICAn register.

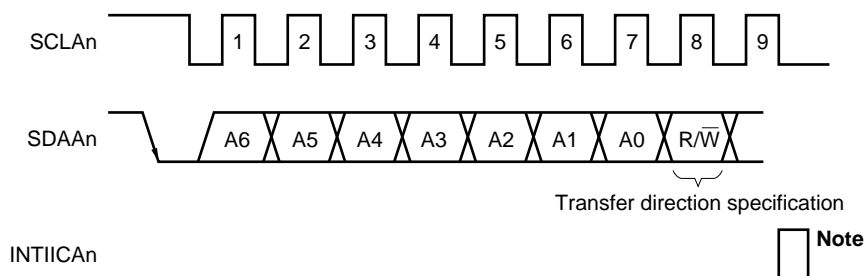
The slave address is assigned to the higher 7 bits of the IICAn register.

### 12.5.3 Transfer direction specification

In addition to the 7-bit address data, the master device sends 1 bit that specifies the transfer direction.

When this transfer direction specification bit has a value of "0", it indicates that the master device is transmitting data to a slave device. When the transfer direction specification bit has a value of "1", it indicates that the master device is receiving data from a slave device.

Figure 12-17. Transfer Direction Specification



**Note** INTIICAn is not issued if data other than a local address or extension code is received during slave device operation.

**Remark**  $n = 0, 1$



### 12.5.4 Acknowledge ( $\overline{\text{ACK}}$ )

$\overline{\text{ACK}}$  is used to check the status of serial data at the transmission and reception sides.

The reception side returns  $\overline{\text{ACK}}$  each time it has received 8-bit data.

The transmission side usually receives  $\overline{\text{ACK}}$  after transmitting 8-bit data. When  $\overline{\text{ACK}}$  is returned from the reception side, it is assumed that reception has been correctly performed and processing is continued. Whether  $\overline{\text{ACK}}$  has been detected can be checked by using bit 2 (ACKDn) of the IICA status register n (IICSn).

When the master receives the last data item, it does not return  $\overline{\text{ACK}}$  and instead generates a stop condition. If a slave does not return  $\overline{\text{ACK}}$  after receiving data, the master outputs a stop condition or restart condition and stops transmission. If  $\overline{\text{ACK}}$  is not returned, the possible causes are as follows.

- <1> Reception was not performed normally.
- <2> The final data item was received.
- <3> The reception side specified by the address does not exist.

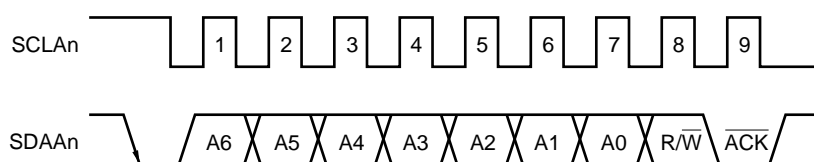
To generate  $\overline{\text{ACK}}$ , the reception side makes the SDAAn line low at the ninth clock (indicating normal reception).

Automatic generation of  $\overline{\text{ACK}}$  is enabled by setting bit 2 (ACKEn) of IICA control register n0 (IICCTLn0) to 1. Bit 3 (TRCn) of the IICSn register is set by the data of the eighth bit that follows 7-bit address information. Usually, set the ACKEn bit to 1 for reception (TRCn = 0).

If a slave can receive no more data during reception (TRCn = 0) or does not require the next data item, then the slave must inform the master, by clearing the ACKEn bit to 0, that it will not receive any more data.

When the master does not require the next data item during reception (TRCn = 0), it must clear the ACKEn bit to 0 so that  $\overline{\text{ACK}}$  is not generated. In this way, the master informs a slave at the transmission side that it does not require any more data (transmission will be stopped).

Figure 12-18.  $\overline{\text{ACK}}$



When the local address is received,  $\overline{\text{ACK}}$  is automatically generated, regardless of the value of the ACKEn bit. When an address other than that of the local address is received,  $\overline{\text{ACK}}$  is not generated (NACK).

When an extension code is received,  $\overline{\text{ACK}}$  is generated if the ACKEn bit is set to 1 in advance.

How  $\overline{\text{ACK}}$  is generated when data is received differs as follows depending on the setting of the wait timing.

- When 8-clock wait state is selected (bit 3 (WTIMn) of IICCTLn0 register = 0):  
By setting the ACKEn bit to 1 before releasing the wait state,  $\overline{\text{ACK}}$  is generated at the falling edge of the eighth clock of the SCLAn pin.
- When 9-clock wait state is selected (bit 3 (WTIMn) of IICCTLn0 register = 1):  
 $\overline{\text{ACK}}$  is generated by setting the ACKEn bit to 1 in advance.

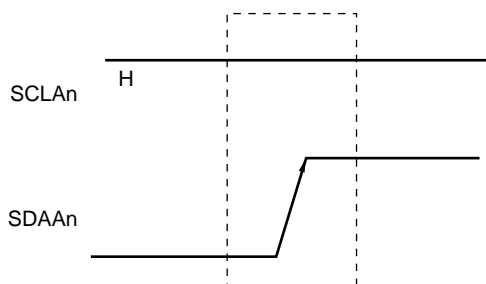
**Remark** n = 0, 1

### 12.5.5 Stop condition

When the SCLAn pin is at high level, changing the SDAAn pin from low level to high level generates a stop condition.

A stop condition is a signal that the master device generates to the slave device when serial transfer has been completed. When the device is used as a slave, stop conditions can be detected.

Figure 12-19. Stop Condition



A stop condition is generated when bit 0 (SPTn) of IICA control register n0 (IICCTLn0) is set to 1. When the stop condition is detected, bit 0 (SPDn) of the IICA status register n (IICSn) is set to 1 and INTIICAn is generated when bit 4 (SPIEn) of the IICCTLn0 register is set to 1.

**Remark** n = 0, 1

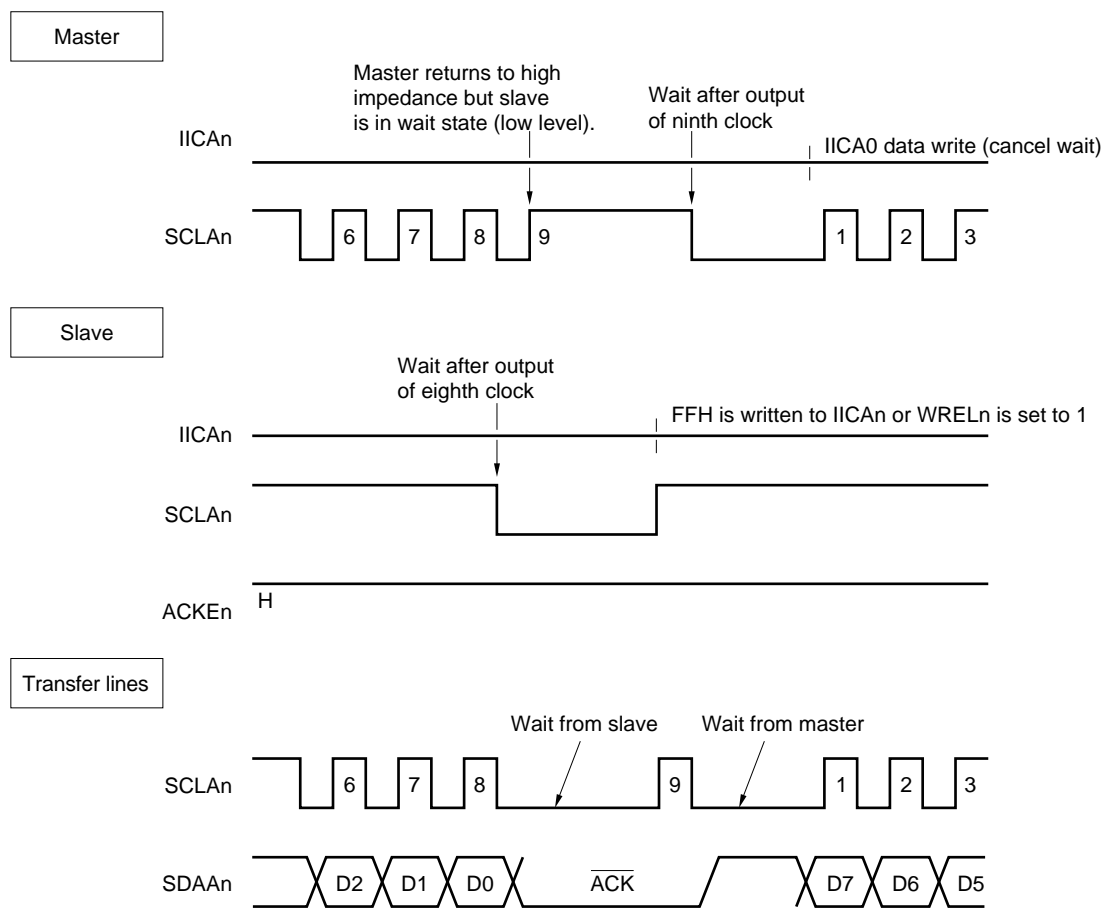
12.5.6 Wait

The wait is used to notify the communication partner that a device (master or slave) is preparing to transmit or receive data (i.e., is in a wait state).

Setting the SCLAn pin to low level notifies the communication partner of the wait state. When wait state has been canceled for both the master and slave devices, the next data transfer can begin.

Figure 12-20. Wait (1/2)

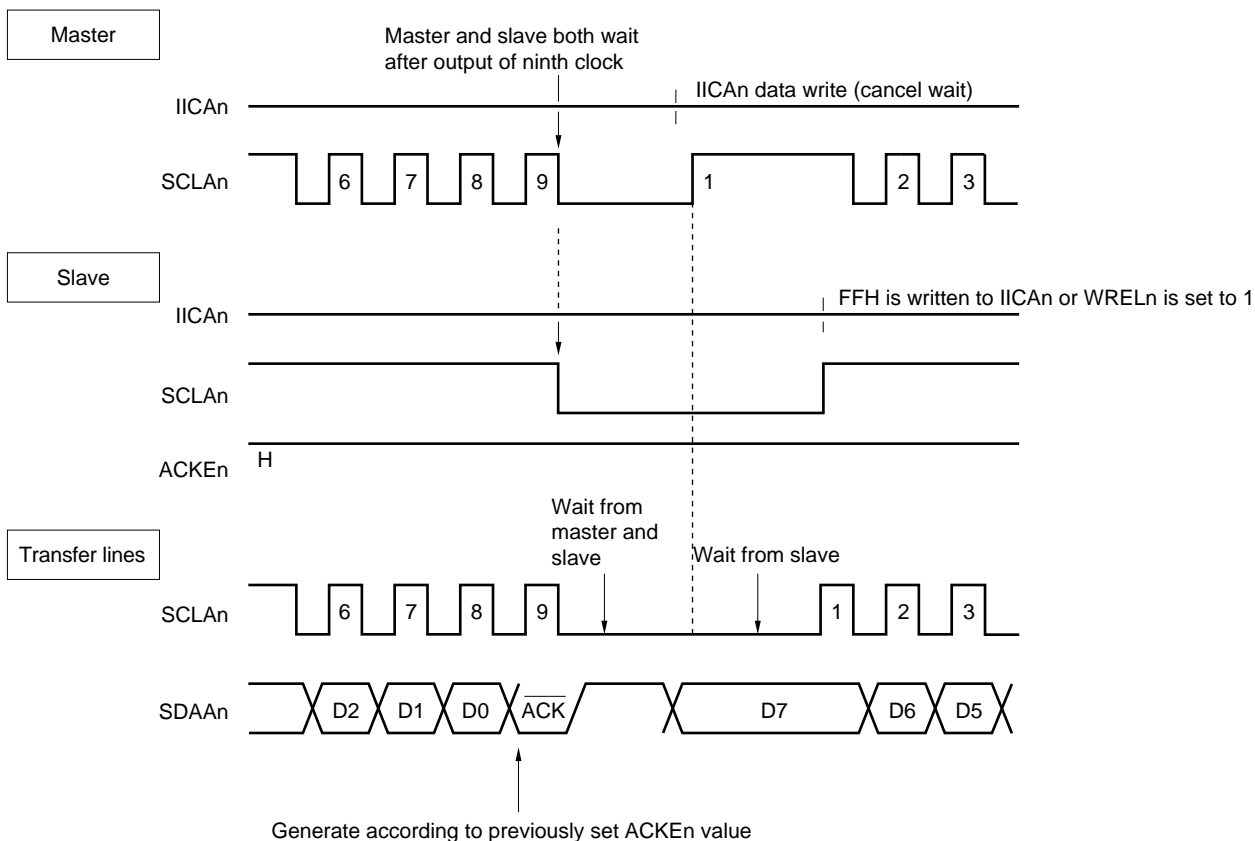
(1) When master device has a nine-clock wait and slave device has an eight-clock wait (master transmits, slave receives, and ACKEn = 1)



Remark n = 0, 1

Figure 12-20. Wait (2/2)

(2) When master and slave devices both have a nine-clock wait  
(master transmits, slave receives, and ACKEn = 1)



**Remark** ACKEn: Bit 2 of IICA control register n0 (IICCTLn0)  
WRELn: Bit 5 of IICA control register n0 (IICCTLn0)

A wait may be automatically generated depending on the setting of bit 3 (WTIMn) of IICA control register n0 (IICCTLn0). Normally, the receiving side cancels the wait state when bit 5 (WRELn) of the IICCTLn0 register is set to 1 or when FFH is written to the IICA shift register n (IICAn), and the transmitting side cancels the wait state when data is written to the IICAn register.

The master device can also cancel the wait state via either of the following methods.

- By setting bit 1 (STTn) of the IICCTLn0 register to 1
- By setting bit 0 (SPTn) of the IICCTLn0 register to 1

**Remark** n = 0, 1

### 12.5.7 Canceling wait

The I<sup>2</sup>C usually cancels a wait state by the following processing.

- Writing data to the IICA shift register n (IICAn)
- Setting bit 5 (WRELn) of IICA control register n0 (IICCTLn0) (canceling wait)
- Setting bit 1 (STTn) of the IICCTLn0 register (generating start condition)<sup>Note</sup>
- Setting bit 0 (SPTn) of the IICCTLn0 register (generating stop condition)<sup>Note</sup>

**Note** Master only

When the above wait canceling processing is executed, the I<sup>2</sup>C cancels the wait state and communication is resumed.

To cancel a wait state and transmit data (including addresses), write the data to the IICAn register.

To receive data after canceling a wait state, or to complete data transmission, set bit 5 (WRELn) of the IICCTLn0 register to 1.

To generate a restart condition after canceling a wait state, set bit 1 (STTn) of the IICCTLn0 register to 1.

To generate a stop condition after canceling a wait state, set bit n (SPTn) of the IICCTLn0 register to 1.

Execute the canceling processing only once for one wait state.

If, for example, data is written to the IICAn register after canceling a wait state by setting the WRELn bit to 1, an incorrect value may be output to SDAAn line because the timing for changing the SDAAn line conflicts with the timing for writing the IICAn register.

In addition to the above, communication is stopped if the IICEn bit is cleared to 0 when communication has been aborted, so that the wait state can be canceled.

If the I<sup>2</sup>C bus has deadlocked due to noise, processing is saved from communication by setting bit 6 (LRELn) of the IICCTLn0 register, so that the wait state can be canceled.

**Caution** If a processing to cancel a wait state is executed when WUPn = 1, the wait state will not be canceled.

**Remark** n = 0, 1

### 12.5.8 Interrupt request (INTIICAn) generation timing and wait control

The setting of bit 3 (WTIMn) of IICA control register n0 (IICCTLn0) determines the timing by which INTIICAn is generated and the corresponding wait control, as shown in Table 12-2.

**Table 12-2. INTIICAn Generation Timing and Wait Control**

WTIMn	During Slave Device Operation			During Master Device Operation		
	Address	Data Reception	Data Transmission	Address	Data Reception	Data Transmission
0	g <sup>Notes 1, 2</sup>	g <sup>Note 2</sup>	g <sup>Note 2</sup>	9	8	8
1	g <sup>Notes 1, 2</sup>	g <sup>Note 2</sup>	g <sup>Note 2</sup>	9	9	9

**Notes 1.** The slave device's INTIICAn signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to the slave address register n (SVAn).

At this point,  $\overline{ACK}$  is generated regardless of the value set to the IICCTLn0 register's bit 2 (ACKEn). For a slave device that has received an extension code, INTIICAn occurs at the falling edge of the eighth clock.

However, if the address does not match after restart, INTIICAn is generated at the falling edge of the 9th clock, but wait does not occur.

- 2.** If the received address does not match the contents of the slave address register n (SVAn) and extension code is not received, neither INTIICAn nor a wait occurs.

**Remark** The numbers in the table indicate the number of the serial clock's clock signals. Interrupt requests and wait control are both synchronized with the falling edge of these clock signals.

#### (1) During address transmission/reception

- Slave device operation: Interrupt and wait timing are determined depending on the conditions described in Notes 1 and 2 above, regardless of the WTIMn bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIMn bit.

#### (2) During data reception

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

#### (3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIMn bit.

#### (4) Wait cancellation method

The four wait cancellation methods are as follows.

- Writing data to the IICA shift register n (IICAn)
- Setting bit 5 (WRELn) of IICA control register n0 (IICCTLn0) (canceling wait)
- Setting bit 1 (STTn) of IICCTLn0 register (generating start condition)<sup>Note</sup>
- Setting bit 0 (SPTn) of IICCTLn0 register (generating stop condition)<sup>Note</sup>

**Note** Master only.

When an 8-clock wait has been selected (WTIMn = 0), the presence/absence of  $\overline{ACK}$  generation must be determined prior to wait cancellation.

#### (5) Stop condition detection

INTIICAn is generated when a stop condition is detected (only when SPIEn = 1).

**Remark** n = 0, 1

### 12.5.9 Address match detection method

In I<sup>2</sup>C bus mode, the master device can select a particular slave device by transmitting the corresponding slave address.

Address match can be detected automatically by hardware. An interrupt request (INTIICAn) occurs when the address set to the slave address register n (SVAn) matches the slave address sent by the master device, or when an extension code has been received.

### 12.5.10 Error detection

In I<sup>2</sup>C bus mode, the status of the serial data bus (SDAAn) during data transmission is captured by the IICA shift register n (IICAn) of the transmitting device, so the IICA data prior to transmission can be compared with the transmitted IICA data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

### 12.5.11 Extension code

- (1) When the higher 4 bits of the receive address are either "0000" or "1111", the extension code reception flag (EXCn) is set to 1 for extension code reception and an interrupt request (INTIICAn) is issued at the falling edge of the eighth clock. The local address stored in the slave address register n (SVAn) is not affected.
- (2) The settings below are specified if 11110xx0 is transferred from the master by using a 10-bit address transfer when the SVAn register is set to 11110xx0. Note that INTIICAn occurs at the falling edge of the eighth clock.
  - Higher four bits of data match: EXCn = 1
  - Seven bits of data match: COIn = 1

**Remark** EXCn: Bit 5 of IICA status register n (IICSn)  
COIn: Bit 4 of IICA status register n (IICSn)

- (3) Since the processing after the interrupt request occurs differs according to the data that follows the extension code, such processing is performed by software.  
If the extension code is received while a slave device is operating, then the slave device is participating in communication even if its address does not match.  
For example, after the extension code is received, if you do not wish to operate the target device as a slave device, set bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 to set the standby mode for the next communication operation.

**Table 12-3. Bit Definitions of Major Extension Codes**

Slave Address	R/W Bit	Description
0 0 0 0 0 0 0	0	General call address
1 1 1 1 0 x x	0	10-bit slave address specification (during address authentication)
1 1 1 1 0 x x	1	10-bit slave address specification (after address match, when read command is issued)

- Remarks**
1. See the I<sup>2</sup>C bus specifications issued by NXP Semiconductors for details of extension codes other than those described above.
  2. n = 0, 1

### 12.5.12 Arbitration

When several master devices simultaneously generate a start condition (when the STTn bit is set to 1 before the STDn bit is set to 1), communication among the master devices is performed as the number of clocks are adjusted until the data differs. This kind of operation is called arbitration.

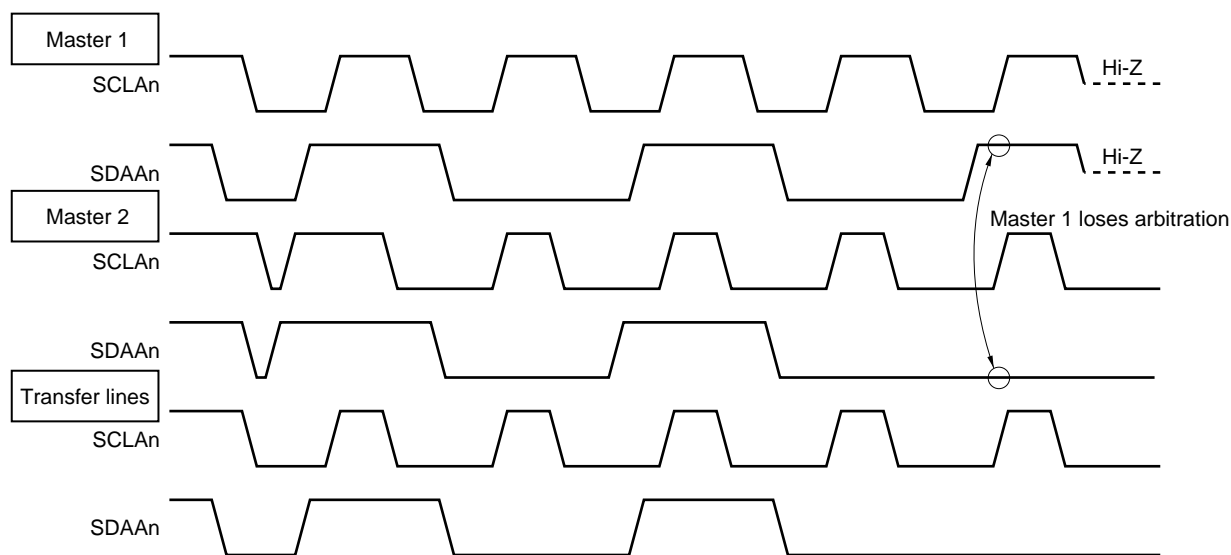
When one of the master devices loses in arbitration, an arbitration loss flag (ALDn) in the IICA status register n (IICSn) is set (1) via the timing by which the arbitration loss occurred, and the SCLAn and SDAAn lines are both set to high impedance, which releases the bus.

The arbitration loss is detected based on the timing of the next interrupt request (the eighth or ninth clock, when a stop condition is detected, etc.) and the ALDn = 1 setting that has been made by software.

For details of interrupt request timing, see **12.5.8 Interrupt request (INTIICAn) generation timing and wait control**.

**Remark** STDn: Bit 1 of IICA status register n (IICSn)  
STTn: Bit 1 of IICA control register n0 (IICCTLn0)

**Figure 12-21. Arbitration Timing Example**



**Remark** n = 0, 1



**Table 12-4. Status During Arbitration and Interrupt Request Generation Timing**

Status During Arbitration	Interrupt Request Generation Timing
During address transmission	At falling edge of eighth or ninth clock following byte transfer <sup>Note 1</sup>
Read/write data after address transmission	
During extension code transmission	
Read/write data after extension code transmission	
During data transmission	
During $\overline{\text{ACK}}$ transfer period after data transmission	
When restart condition is detected during data transfer	
When stop condition is detected during data transfer	When stop condition is generated (when SPIEn = 1) <sup>Note 2</sup>
When data is at low level while attempting to generate a restart condition	At falling edge of eighth or ninth clock following byte transfer <sup>Note 1</sup>
When stop condition is detected while attempting to generate a restart condition	When stop condition is generated (when SPIEn = 1) <sup>Note 2</sup>
When data is at low level while attempting to generate a stop condition	At falling edge of eighth or ninth clock following byte transfer <sup>Note 1</sup>
When SCLAn is at low level while attempting to generate a restart condition	

**Notes 1.** When the WTIMn bit (bit 3 of IICA control register n0 (IICCTLn0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIMn = 0 and the extension code's slave address is received, an interrupt request occurs at the falling edge of the eighth clock.

**2.** When there is a chance that arbitration will occur, set SPIEn = 1 for master device operation.

**Remarks 1.** SPIEn: Bit 4 of IICA control register n0 (IICCTLn0)

**2.** n = 0, 1

### 12.5.13 Wakeup function

The I<sup>2</sup>C bus slave function is a function that generates an interrupt request signal (INTIICAn) when a local address and extension code have been received.

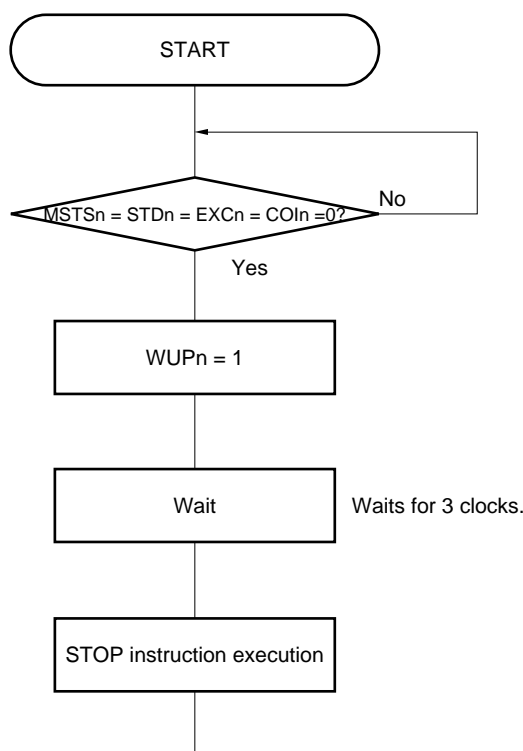
This function makes processing more efficient by preventing unnecessary INTIICAn signal from occurring when addresses do not match.

When a start condition is detected, wakeup standby mode is set. This wakeup standby mode is in effect while addresses are transmitted due to the possibility that an arbitration loss may change the master device (which has generated a start condition) to a slave device.

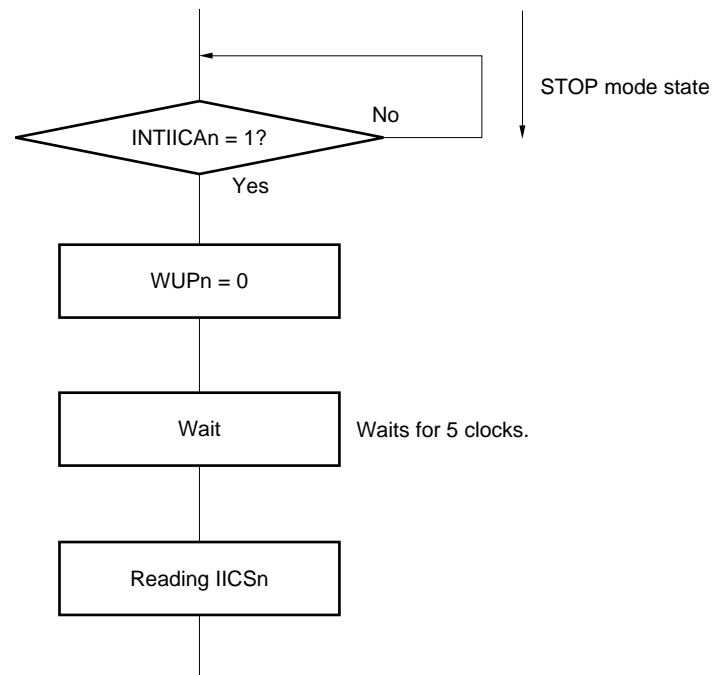
To use the wakeup function in the STOP mode, set the WUPn bit to 1. Addresses can be received regardless of the operation clock. An interrupt request signal (INTIICAn) is also generated when a local address and extension code have been received. Operation returns to normal operation by using an instruction to clear (0) the WUPn bit after this interrupt has been generated.

Figure 12-22 shows the flow for setting WUPn = 1 and Figure 12-23 shows the flow for setting WUPn = 0 upon an address match.

**Figure 12-22. Flow When Setting WUPn = 1**



**Remark** n = 0, 1

**Figure 12-23. Flow When Setting WUPn = 0 upon Address Match (Including Extension Code Reception)**

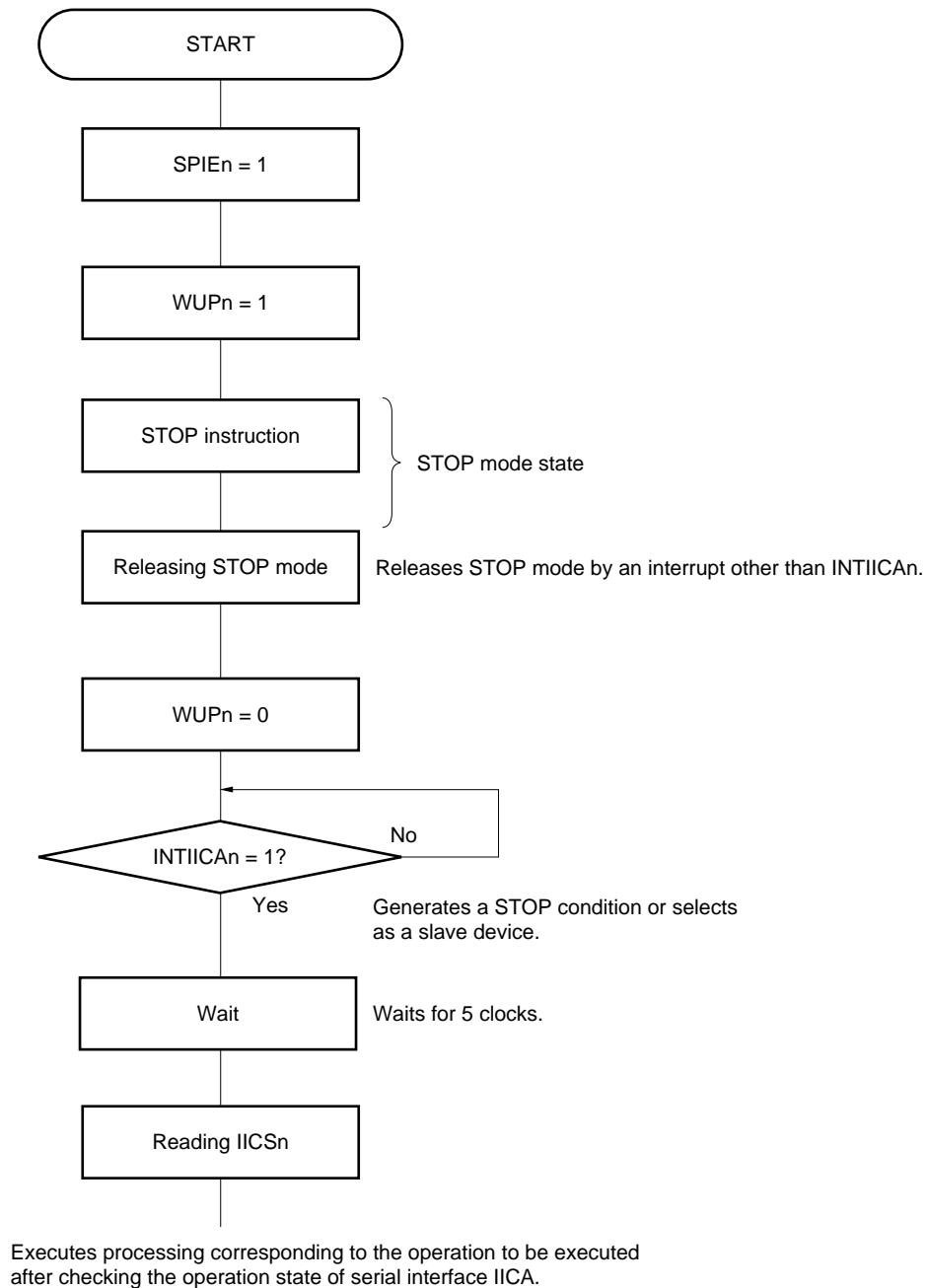
Executes processing corresponding to the operation to be executed after checking the operation state of serial interface IICA.

Use the following flows to perform the processing to release the STOP mode other than by an interrupt request (INTIICAn) generated from serial interface IICA.

- Master device operation: Flow shown in Figure 12-24
- Slave device operation: Same as the flow in Figure 12-23

**Remark** n = 0, 1

Figure 12-24. When Operating as Master Device after Releasing STOP Mode other than by INTIICAn



**Remark** n = 0, 1

### 12.5.14 Communication reservation

#### (1) When communication reservation function is enabled (bit n (IICRSVn) of IICA flag register n (IICFn) = 0)

To start master device communications when not currently using a bus, a communication reservation can be made to enable transmission of a start condition when the bus is released. There are two modes under which the bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ( $\overline{ACK}$  is not returned and the bus was released by setting bit 6 (LRELn) of IICA control register n0 (IICCTLn0) to 1 and saving communication).

If bit 1 (STTn) of the IICCTLn0 register is set to 1 while the bus is not used (after a stop condition is detected), a start condition is automatically generated and wait state is set.

If an address is written to the IICA shift register n (IICAn) after bit 4 (SPIEn) of the IICCTLn0 register was set to 1, and it was detected by generation of an interrupt request signal (INTIICAn) that the bus was released (detection of the stop condition), then the device automatically starts communication as the master. Data written to the IICAn register before the stop condition is detected is invalid.

When the STTn bit has been set to 1, the operation mode (as start condition or as communication reservation) is determined according to the bus status.

- If the bus has been released ..... a start condition is generated
- If the bus has not been released (standby mode)..... communication reservation

Check whether the communication reservation operates or not by using the MSTSn bit (bit 7 of the IICA status register n (IICSn)) after the STTn bit is set to 1 and the wait time elapses.

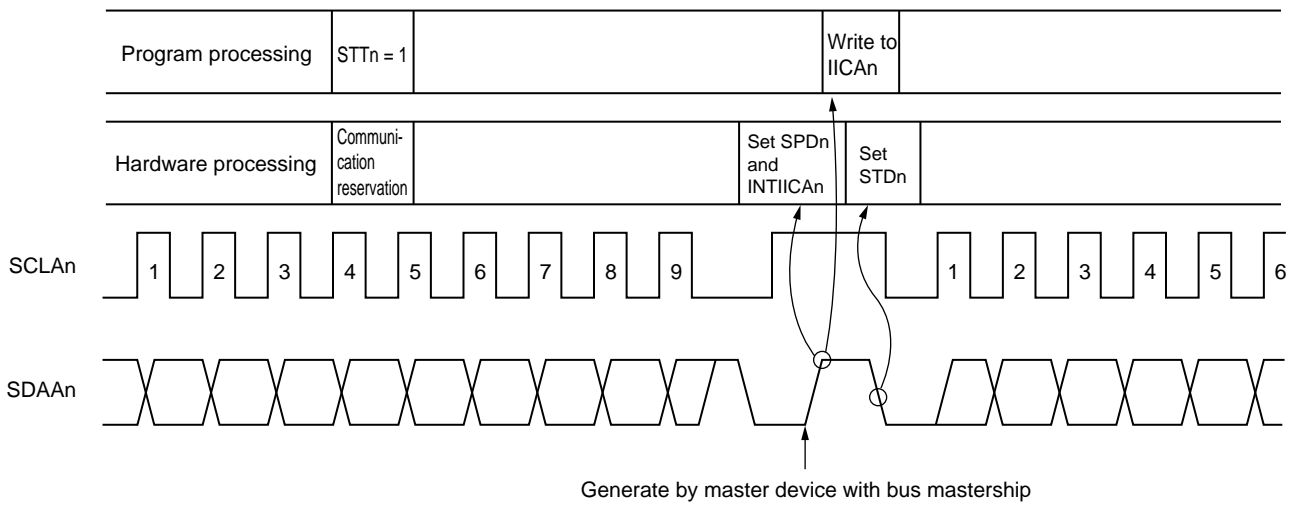
Use software to secure the wait time calculated by the following expression.

Wait time from setting STTn = 1 to checking the MSTSn flag:  
 $(IICWL_n \text{ setting value} + IICWH_n \text{ setting value} + 4) + t_F \times 2 \times f_{CLK} \text{ [clocks]}$

- Remarks**
1. IICWL<sub>n</sub>: IICA low-level width setting register n  
 IICWH<sub>n</sub>: IICA high-level width setting register n  
 t<sub>F</sub>: SDAAn and SCLAn signal falling times  
 f<sub>CLK</sub>: CPU/peripheral hardware clock frequency
  2. n = 0, 1

Figure 12-25 shows the communication reservation timing.

**Figure 12-25. Communication Reservation Timing**



- Remark**
- IICAn: IICA shift register n
  - STTn: Bit 1 of IICA control register n0 (IICCTLn0)
  - STDn: Bit 1 of IICA status register n (IICSn)
  - SPDn: Bit 0 of IICA status register n (IICSn)

Communication reservations are accepted via the timing shown in Figure 12-26. After bit 1 (STDn) of the IICA status register n (IICSn) is set to 1, a communication reservation can be made by setting bit 1 (STTn) of IICA control register n0 (IICCTLn0) to 1 before a stop condition is detected.

**Figure 12-26. Timing for Accepting Communication Reservations**

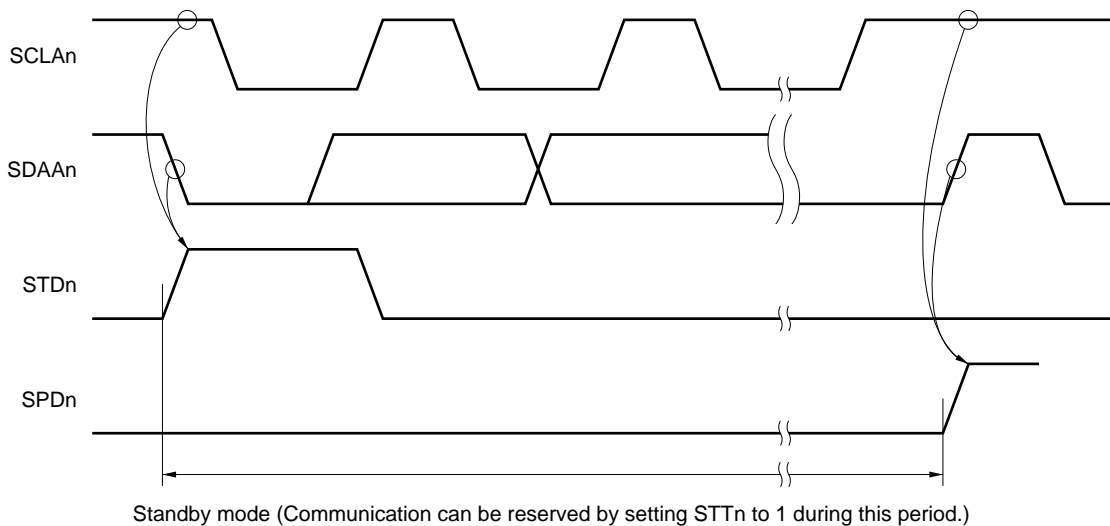
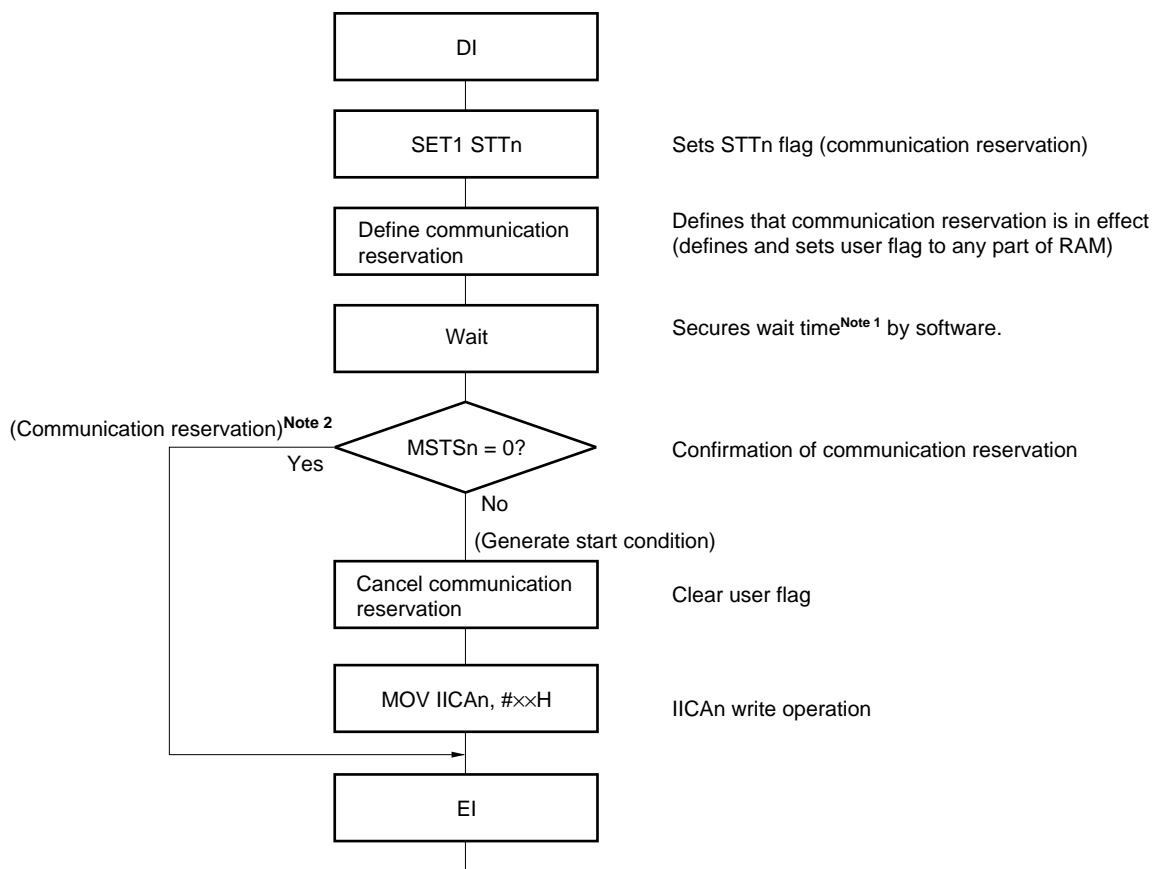


Figure 12-27 shows the communication reservation protocol.

- Remark** n = 0, 1

Figure 12-27. Communication Reservation Protocol



**Notes 1.** The wait time is calculated as follows.

$$(IICWL_n \text{ setting value} + IICWH_n \text{ setting value} + 4) + t_F \times 2 \times f_{CLK} [\text{clocks}]$$

- 2.** The communication reservation operation executes a write to the IICA shift register n (IICAn) when a stop condition interrupt request occurs.

**Remarks 1.** STTn: Bit 1 of IICA control register n0 (IICCTLn0)

MSTS<sub>n</sub>: Bit 7 of IICA status register n (IICS<sub>n</sub>)

IICAn: IICA shift register n

IICWL<sub>n</sub>: IICA low-level width setting register n

IICWH<sub>n</sub>: IICA high-level width setting register n

t<sub>F</sub>: SDAAn and SCLAn signal falling times

f<sub>CLK</sub>: CPU/peripheral hardware clock frequency

- 2.** n = 0, 1

**(2) When communication reservation function is disabled (bit 0 (IICRSVn) of IICA flag register n (IICFn) = 1)**

When bit 1 (STTn) of IICA control register n0 (IICCTLn0) is set to 1 when the bus is not used in a communication during bus communication, this request is rejected and a start condition is not generated. The following two statuses are included in the status where bus is not used.

- When arbitration results in neither master nor slave operation
- When an extension code is received and slave operation is disabled ( $\overline{ACK}$  is not returned and the bus was released by setting bit 6 (LRELn) of the IICCTLn0 register to 1 and saving communication)

To confirm whether the start condition was generated or request was rejected, check STCFn (bit 7 of the IICFn register). It takes up to 5 clocks until the STCFn bit is set to 1 after setting STTn = 1. Therefore, secure the time by software.

**Remark** n = 0, 1



### 12.5.15 Cautions

(1) When  $STCENn = 0$

Immediately after I<sup>2</sup>C operation is enabled ( $IICEn = 1$ ), the bus communication status ( $IICBSYn = 1$ ) is recognized regardless of the actual bus status. When changing from a mode in which no stop condition has been detected to a master device communication mode, first generate a stop condition to release the bus, then perform master device communication.

When using multiple masters, it is not possible to perform master device communication when the bus has not been released (when a stop condition has not been detected).

Use the following sequence for generating a stop condition.

- <1> Set IICA control register n1 ( $IICCTLn1$ ).
- <2> Set bit 7 ( $IICEn$ ) of IICA control register n0 ( $IICCTLn0$ ) to 1.
- <3> Set bit 0 ( $SPTn$ ) of the  $IICCTLn0$  register to 1.

(2) When  $STCENn = 1$

Immediately after I<sup>2</sup>C operation is enabled ( $IICEn = 1$ ), the bus released status ( $IICBSYn = 0$ ) is recognized regardless of the actual bus status. To generate the first start condition ( $STTn = 1$ ), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

(3) If other I<sup>2</sup>C communications are already in progress

If I<sup>2</sup>C operation is enabled and the device participates in communication already in progress when the SDAAn pin is low and the SCLAn pin is high, the macro of I<sup>2</sup>C recognizes that the SDAAn pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code,  $\overline{ACK}$  is returned, but this interferes with other I<sup>2</sup>C communications. To avoid this, start I<sup>2</sup>C in the following sequence.

- <1> Clear bit 4 ( $SPIEn$ ) of the  $IICCTLn0$  register to 0 to disable generation of an interrupt request signal ( $INTIICAn$ ) when the stop condition is detected.
- <2> Set bit 7 ( $IICEn$ ) of the  $IICCTLn0$  register to 1 to enable the operation of I<sup>2</sup>C.
- <3> Wait for detection of the start condition.
- <4> Set bit 6 ( $LRELn$ ) of the  $IICCTLn0$  register to 1 before  $\overline{ACK}$  is returned (4 to 80 clocks after setting the  $IICEn$  bit to 1), to forcibly disable detection.

(4) Setting the  $STTn$  and  $SPTn$  bits (bits 1 and 0 of the  $IICCTLn0$  register) again after they are set and before they are cleared to 0 is prohibited.

(5) When transmission is reserved, set the  $SPIEn$  bit (bit 4 of the  $IICCTLn0$  register) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to the IICA shift register n ( $IICAn$ ) after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set the  $SPIEn$  bit to 1 when the  $MSTS$  bit (bit 7 of the IICA status register n ( $IICSn$ )) is detected by software.

**Remark** n = 0, 1

### 12.5.16 Communication operations

The following shows three operation procedures with the flowchart.

#### (1) Master operation in single master system

The flowchart when using the R7F0C01072DNP and R7F0C010B2DFP-C as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

#### (2) Master operation in multimaster system

In the I<sup>2</sup>C bus multimaster system, whether the bus is released or used cannot be judged by the I<sup>2</sup>C bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the R7F0C01072DNP and R7F0C010B2DFP-C take part in a communication with bus released state.

This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the R7F0C01072DNP and R7F0C010B2DFP-C loose in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

#### (3) Slave operation

An example of when the R7F0C01072DNP and R7F0C010B2DFP-C is used as the I<sup>2</sup>C bus slave is shown below.

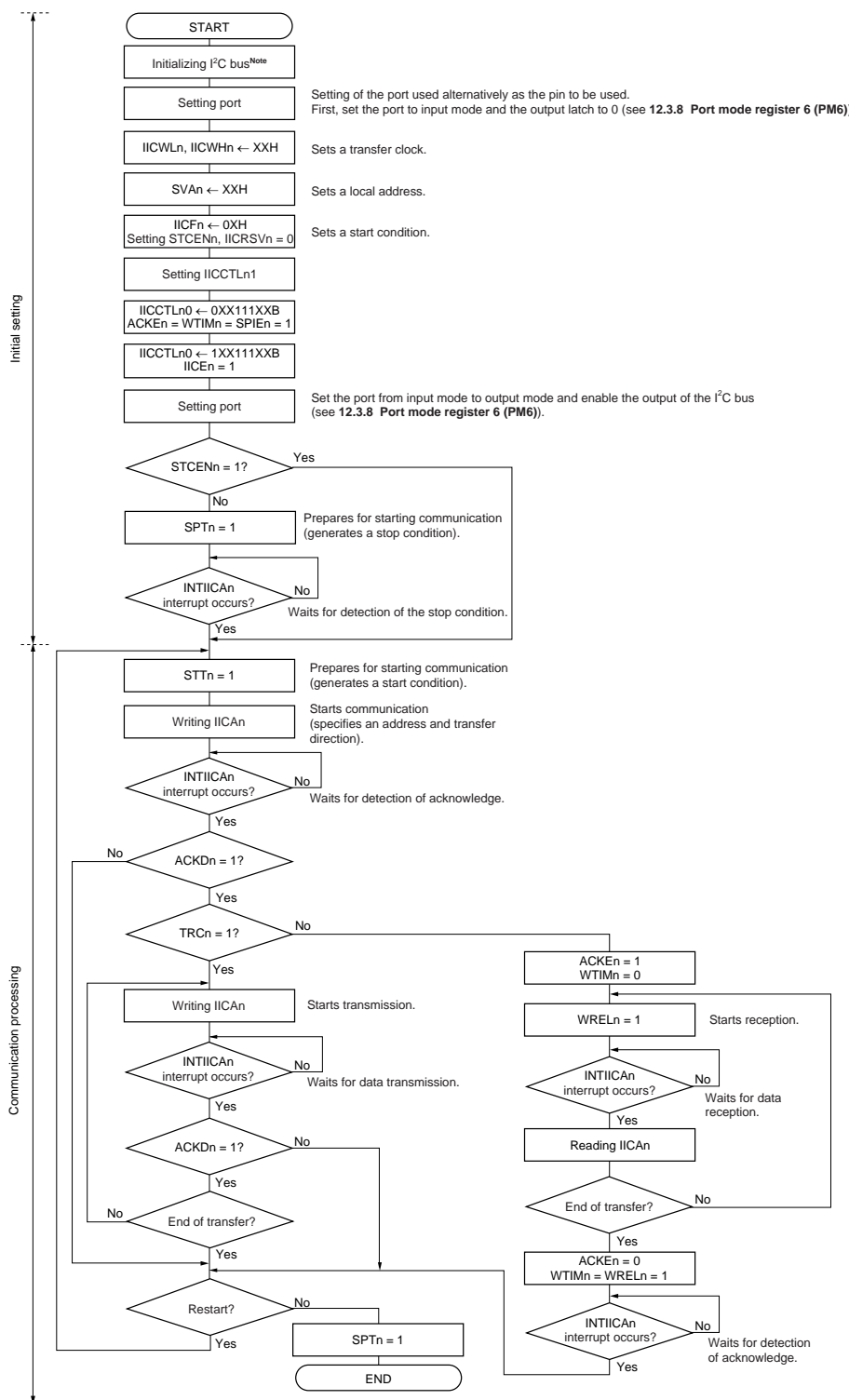
When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIICAn interrupt occurrence (communication waiting). When an INTIICAn interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

By checking the flags, necessary communication processing is performed.

**Remark** n = 0, 1

(1) Master operation in single-master system

Figure 12-28. Master Operation in Single-Master System



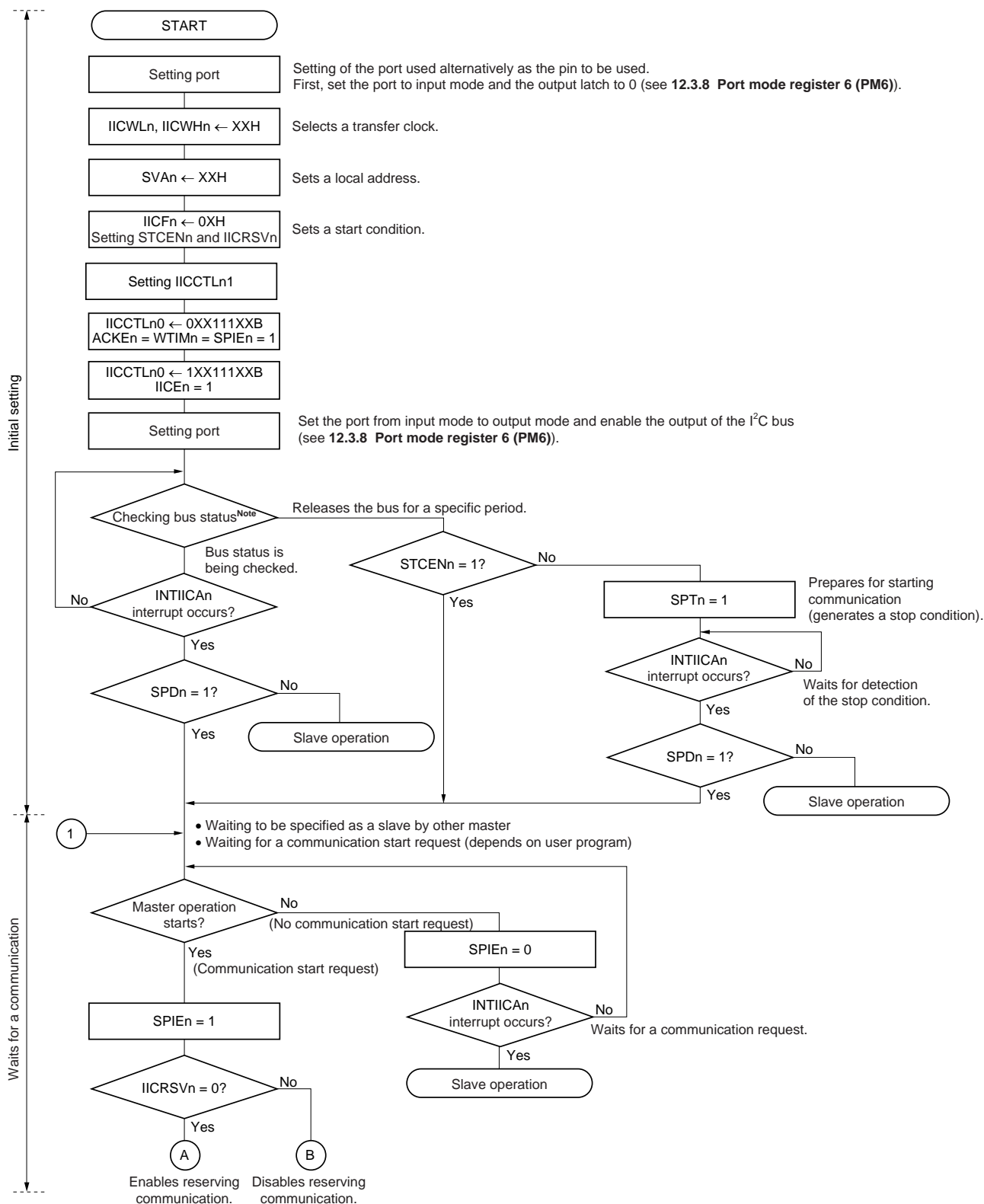
**Note** Release (SCLAn and SDAAn pins = high level) the I<sup>2</sup>C bus in conformance with the specifications of the product that is communicating. If EEPROM is outputting a low level to the SDAAn pin, for example, set the SCLAn pin in the output port mode, and output a clock pulse from the output port until the SDAAn pin is constantly at high level.

**Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

**2.** n = 0, 1

(2) Master operation in multi-master system

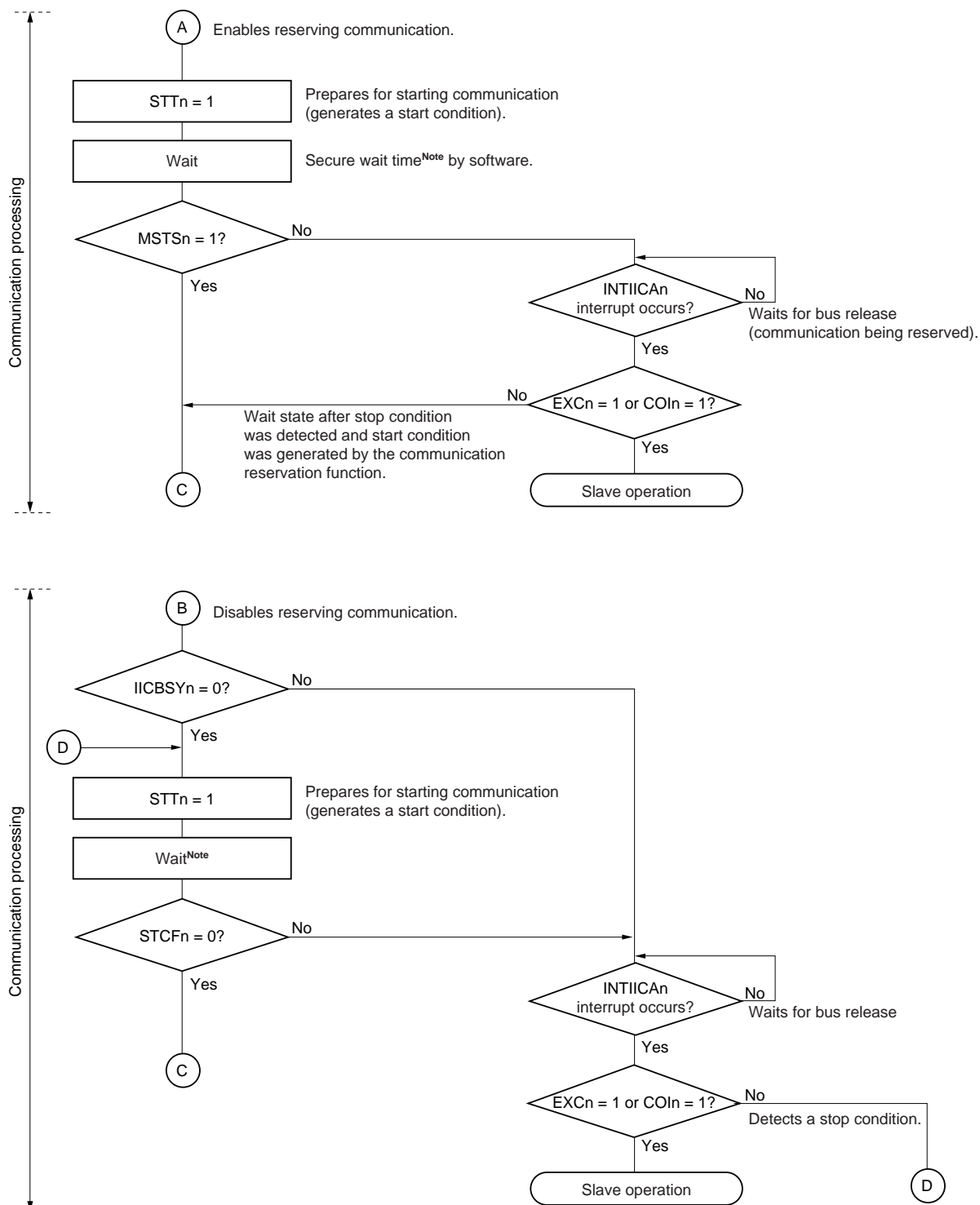
Figure 12-29. Master Operation in Multi-Master System (1/3)



**Note** Confirm that the bus is released (CLDn bit = 1, DADn bit = 1) for a specific period (for example, for a period of one frame). If the SDAAn pin is constantly at low level, decide whether to release the I<sup>2</sup>C bus (SCLAn and SDAAn pins = high level) in conformance with the specifications of the product that is communicating.

**Remark** n = 0, 1

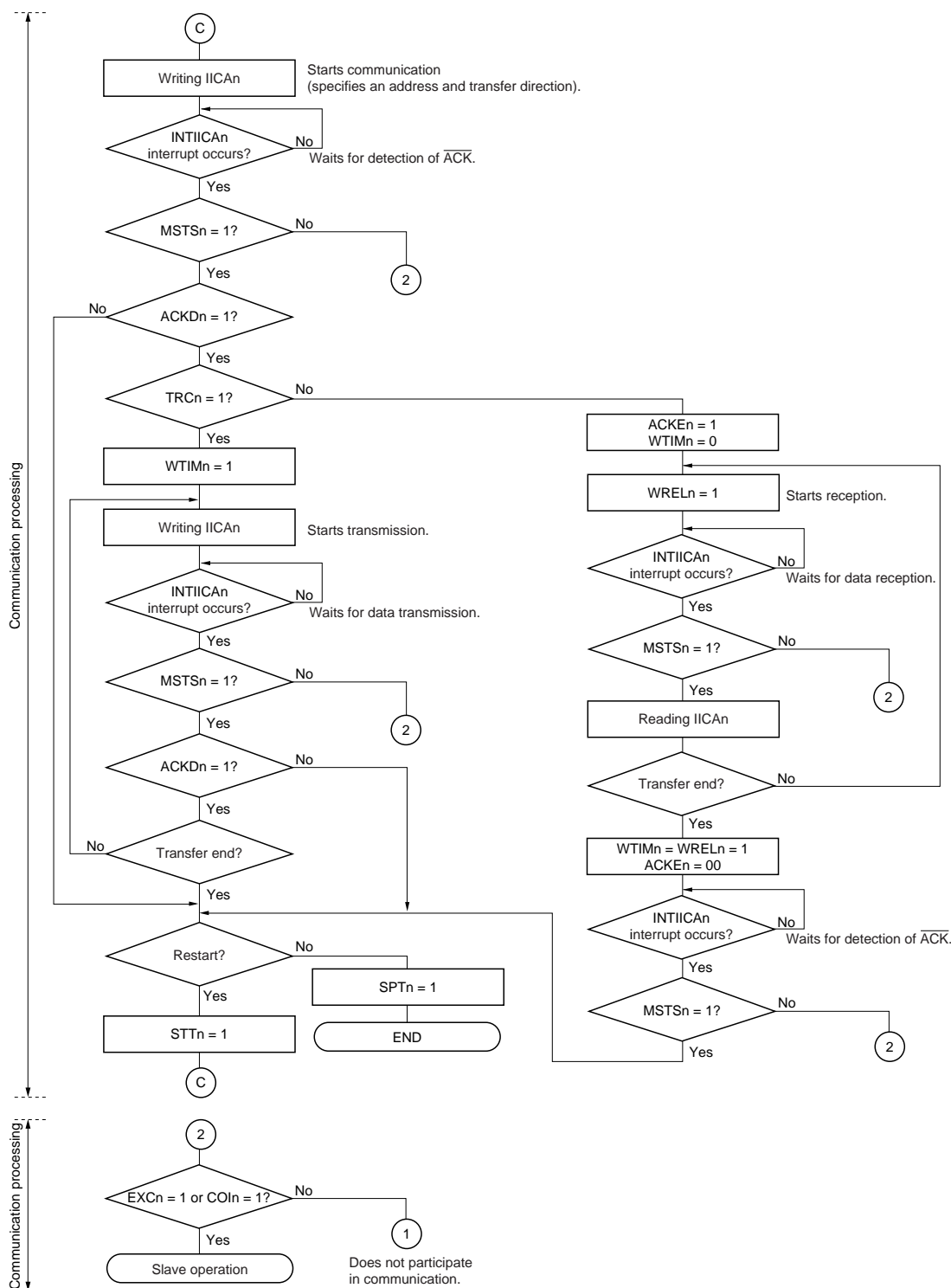
Figure 12-29. Master Operation in Multi-Master System (2/3)



**Note** The wait time is calculated as follows.  
 $(IICWLn \text{ setting value} + IICWHn \text{ setting value} + 4) \times f_{CLK} + t_F \times 2$  [clocks]

- Remarks 1.**
- IICWL<sub>n</sub>: IICA low-level width setting register n
  - IICWH<sub>n</sub>: IICA high-level width setting register n
  - t<sub>F</sub>: SDA<sub>n</sub> and SCL<sub>n</sub> signal falling times
  - f<sub>CLK</sub>: CPU/peripheral hardware clock frequency
- 2.** n = 0, 1

Figure 12-29. Master Operation in Multi-Master System (3/3)



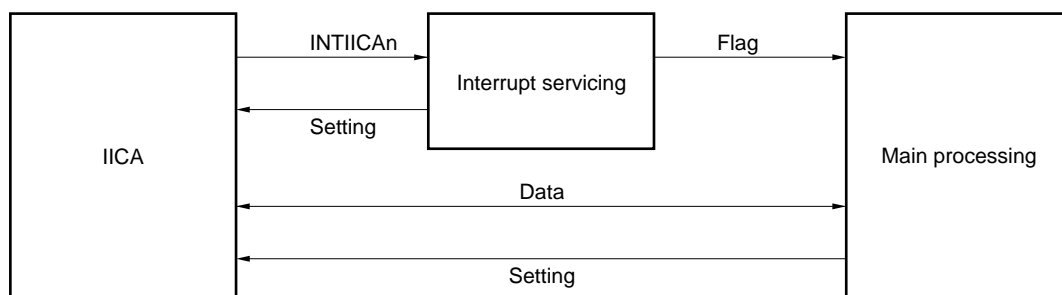
- Remarks**
1. Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.
  2. To use the device as a master in a multi-master system, read the MSTSn bit each time interrupt INTIICAn has occurred to check the arbitration result.
  3. To use the device as a slave in a multi-master system, check the status by using the IICA status register n (IICSn) and IICA flag register n (IICFn) each time interrupt INTIICAn has occurred, and determine the processing to be performed next.
  4. n = 0, 1

**(3) Slave operation**

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIICAn interrupt (processing that must substantially change the operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICAn interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.



Therefore, data communication processing is performed by preparing the following three flags and passing them to the main processing instead of INTIICAn.

**<1> Communication mode flag**

This flag indicates the following two communication statuses.

- Clear mode: Status in which data communication is not performed
- Communication mode: Status in which data communication is performed (from valid address detection to stop condition detection, no detection of  $\overline{\text{ACK}}$  from master, address mismatch)

**<2> Ready flag**

This flag indicates that data communication is enabled. Its function is the same as the INTIICAn interrupt for ordinary data communication. This flag is set by interrupt servicing and cleared by the main processing. Clear this flag by interrupt servicing when communication is started. However, the ready flag is not set by interrupt servicing when the first data is transmitted. Therefore, the first data is transmitted without the flag being cleared (an address match is interpreted as a request for the next data).

**<3> Communication direction flag**

This flag indicates the direction of communication. Its value is the same as the TRCn bit.

**Remark** n = 0, 1

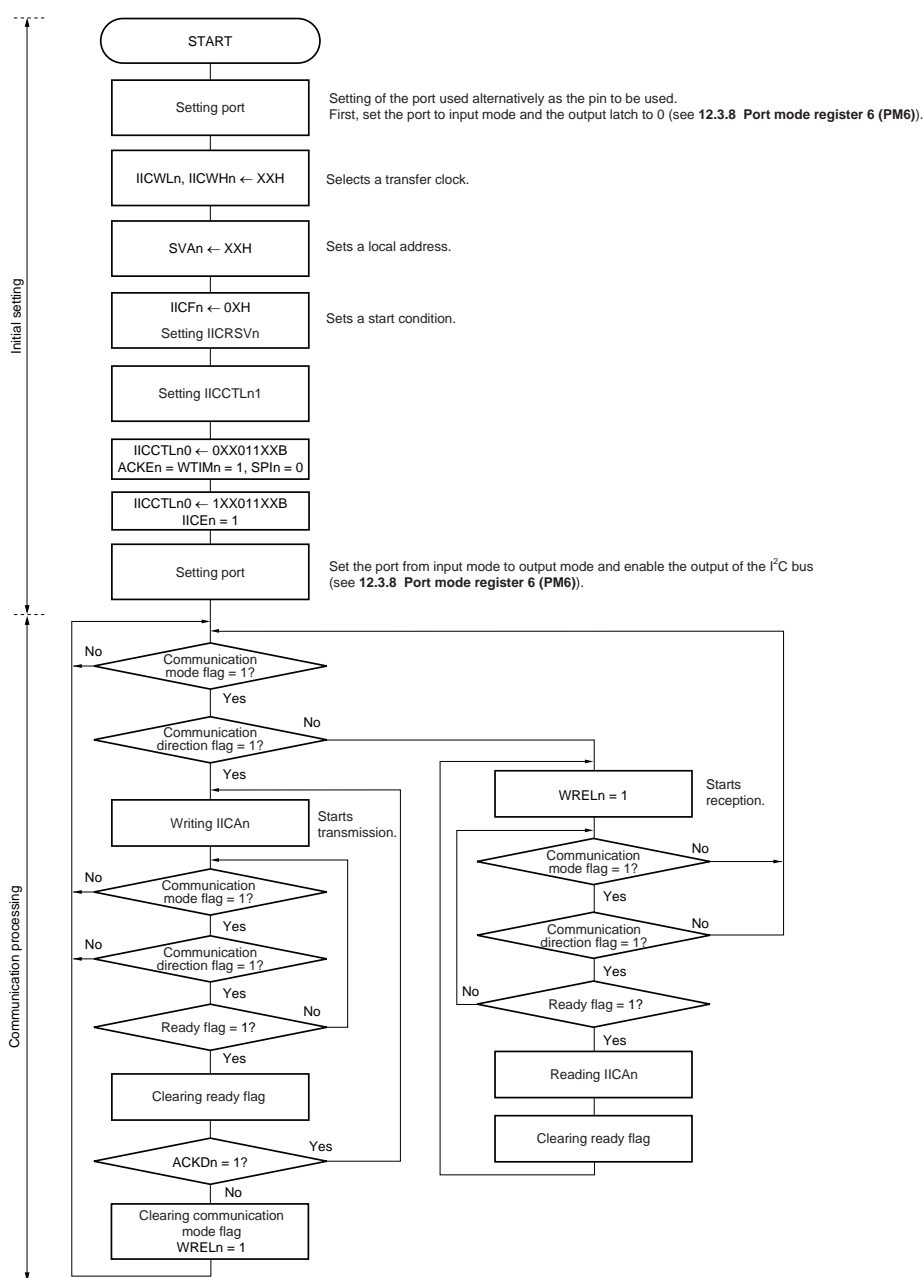
The main processing of the slave operation is explained next.

Start serial interface IICA and wait until communication is enabled. When communication is enabled, execute communication by using the communication mode flag and ready flag (processing of the stop condition and start condition is performed by an interrupt. Here, check the status by using the flags).

The transmission operation is repeated until the master no longer returns  $\overline{ACK}$ . If  $\overline{ACK}$  is not returned from the master, communication is completed.

For reception, the necessary amount of data is received. When communication is completed,  $\overline{ACK}$  is not returned as the next data. After that, the master generates a stop condition or restart condition. Exit from the communication status occurs in this way.

Figure 12-30. Slave Operation Flowchart (1)



- Remarks 1.** Conform to the specifications of the product that is in communication, regarding the transmission and reception formats.
- 2.** n = 0, 1

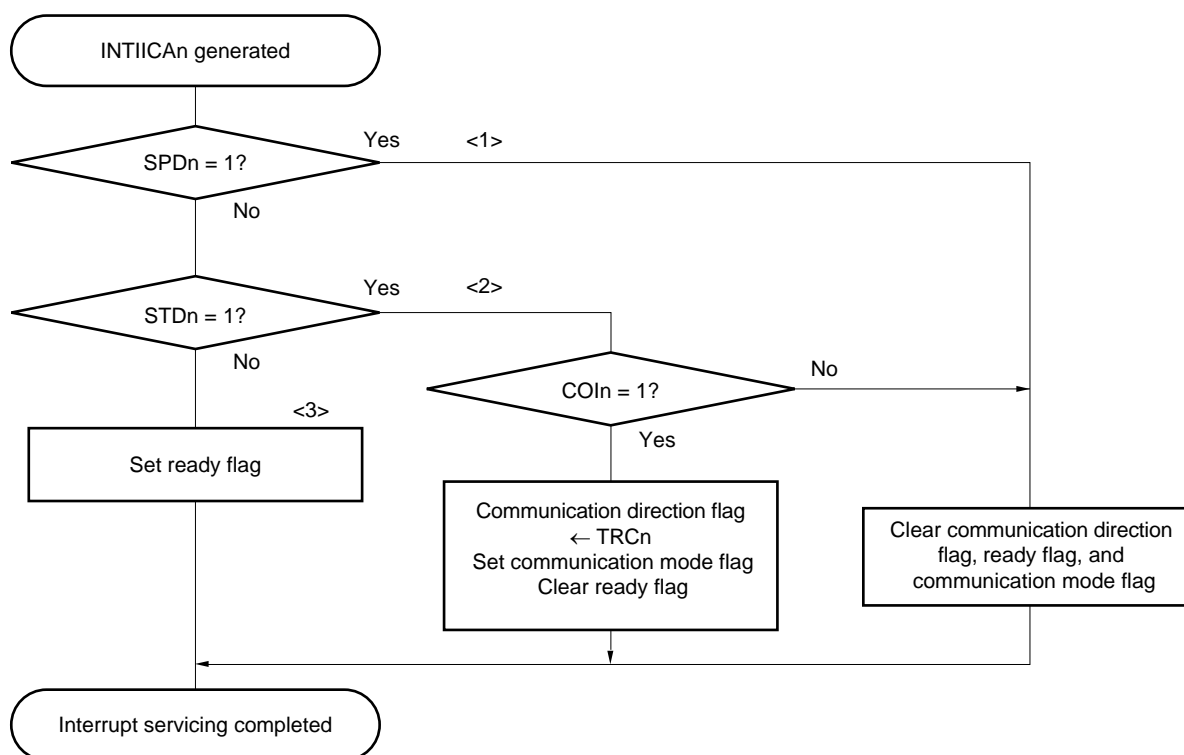


An example of the processing procedure of the slave with the INTIICAn interrupt is explained below (processing is performed assuming that no extension code is used). The INTIICAn interrupt checks the status, and the following operations are performed.

- <1> Communication is stopped if the stop condition is issued.
- <2> If the start condition is issued, the address is checked and communication is completed if the address does not match. If the address matches, the communication mode is set, wait is cancelled, and processing returns from the interrupt (the ready flag is cleared).
- <3> For data transmit/receive, only the ready flag is set. Processing returns from the interrupt with the I<sup>2</sup>C bus remaining in the wait state.

**Remark** <1> to <3> above correspond to <1> to <3> in Figure 12-31 Slave Operation Flowchart (2).

**Figure 12-31. Slave Operation Flowchart (2)**



### 12.5.17 Timing of I<sup>2</sup>C interrupt request (INTIICAn) occurrence

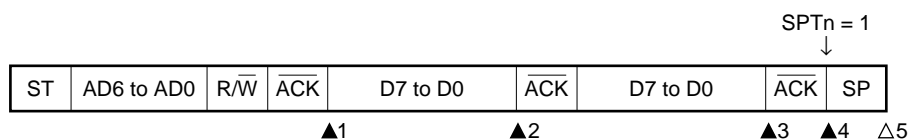
The timing of transmitting or receiving data and generation of interrupt request signal INTIICAn, and the value of the IICA status register n (IICSn) when the INTIICAn signal is generated are shown below.

- Remarks 1.**
- ST: Start condition
  - AD6 to AD0: Address
  - R/W: Transfer direction specification
  - ACK: Acknowledge
  - D7 to D0: Data
  - SP: Stop condition
- 2.** n = 0, 1

## (1) Master device operation

## (a) Start ~ Address ~ Data ~ Data ~ Stop (transmission/reception)

## (i) When WTIMn = 0



▲1: IICSn = 1000x110B

▲2: IICSn = 1000x000B

▲3: IICSn = 1000x000B (Sets the WTIMn bit to 1)<sup>Note</sup>

▲4: IICSn = 1000xx00B (Sets the SPTn bit to 1)

Δ5: IICSn = 00000001B

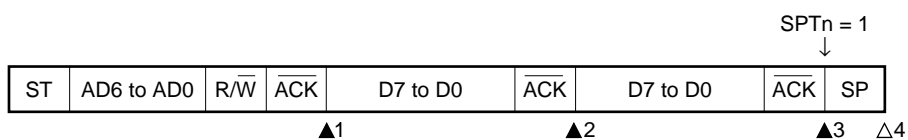
**Note** To generate a stop condition, set the WTIMn bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

**Remark** ▲: Always generated

Δ: Generated only when SPIEn = 1

x: Don't care

## (ii) When WTIMn = 1



▲1: IICSn = 1000x110B

▲2: IICSn = 1000x100B

▲3: IICSn = 1000xx00B (Sets the SPTn bit to 1)

Δ4: IICSn = 00000001B

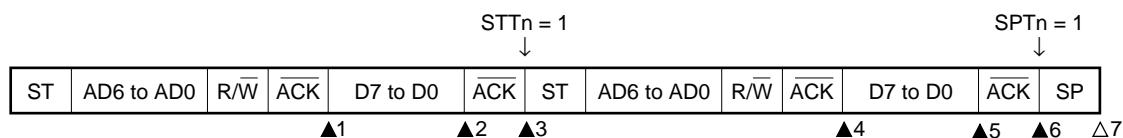
**Remark** ▲: Always generated

Δ: Generated only when SPIEn = 1

x: Don't care

**Remark** n = 0, 1

## (b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

(i) When  $WTIMn = 0$ 

▲1: IICSn = 1000x110B

▲2: IICSn = 1000x000B (Sets the  $WTIMn$  bit to 1)<sup>Note 1</sup>▲3: IICSn = 1000xx00B (Clears the  $WTIMn$  bit to 0<sup>Note 2</sup>, sets the  $STTn$  bit to 1)

▲4: IICSn = 1000x110B

▲5: IICSn = 1000x000B (Sets the  $WTIMn$  bit to 1)<sup>Note 3</sup>▲6: IICSn = 1000xx00B (Sets the  $SPTn$  bit to 1)

△7: IICSn = 00000001B

**Notes 1.** To generate a start condition, set the  $WTIMn$  bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

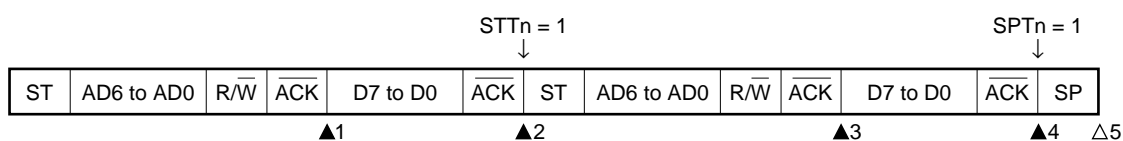
**2.** Clear the  $WTIMn$  bit to 0 to restore the original setting.

**3.** To generate a stop condition, set the  $WTIMn$  bit to 1 and change the timing for generating the INTIICAn interrupt request signal.

**Remark** ▲: Always generated

△: Generated only when  $SPIEn = 1$

x: Don't care

(ii) When  $WTIMn = 1$ 

▲1: IICSn = 1000x110B

▲2: IICSn = 1000xx00B (Sets the  $STTn$  bit to 1)

▲3: IICSn = 1000x110B

▲4: IICSn = 1000xx00B (Sets the  $SPTn$  bit to 1)

△5: IICSn = 00000001B

**Remark** ▲: Always generated

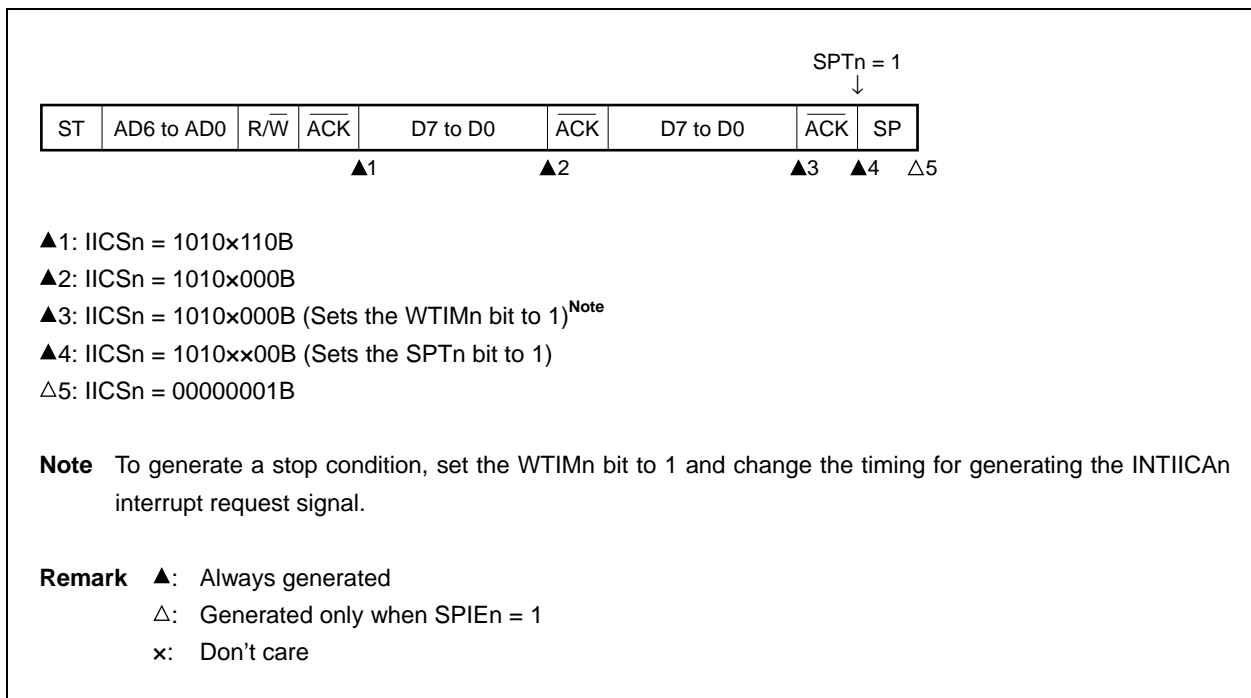
△: Generated only when  $SPIEn = 1$

x: Don't care

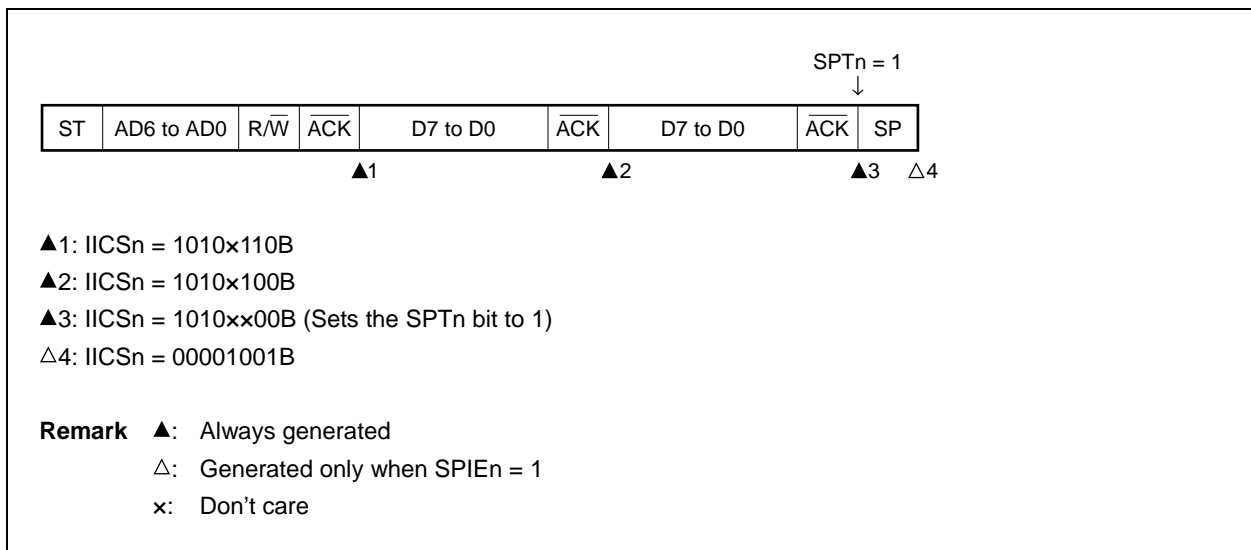
**Remark** n = 0, 1

(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

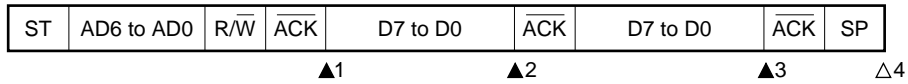
(i) When WTIMn = 0



(ii) When WTIMn = 1



**Remark** n = 0, 1

**(2) Slave device operation (slave address data reception)****(a) Start ~ Address ~ Data ~ Data ~ Stop****(i) When WTIMn = 0**

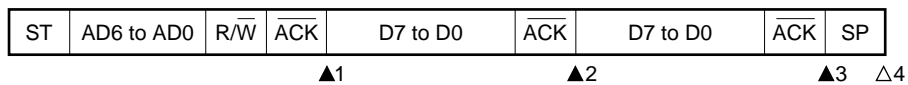
▲1: IICSn = 0001x110B

▲2: IICSn = 0001x000B

▲3: IICSn = 0001x000B

△4: IICSn = 00000001B

**Remark** ▲: Always generated  
 △: Generated only when SPIEn = 1  
 x: Don't care

**(ii) When WTIMn = 1**

▲1: IICSn = 0001x110B

▲2: IICSn = 0001x100B

▲3: IICSn = 0001xx00B

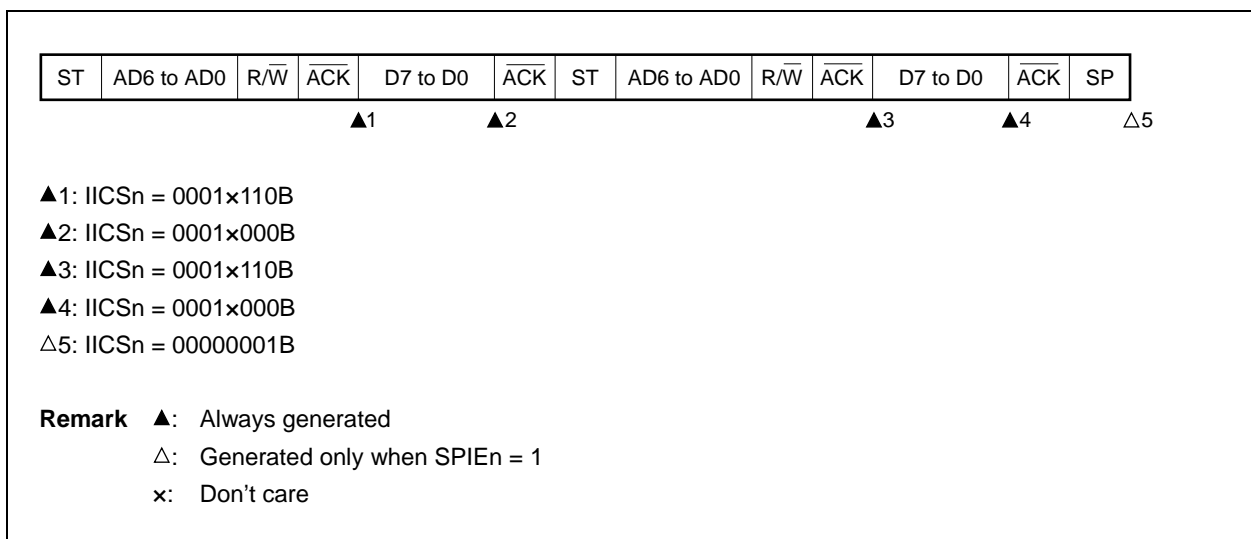
△4: IICSn = 00000001B

**Remark** ▲: Always generated  
 △: Generated only when SPIEn = 1  
 x: Don't care

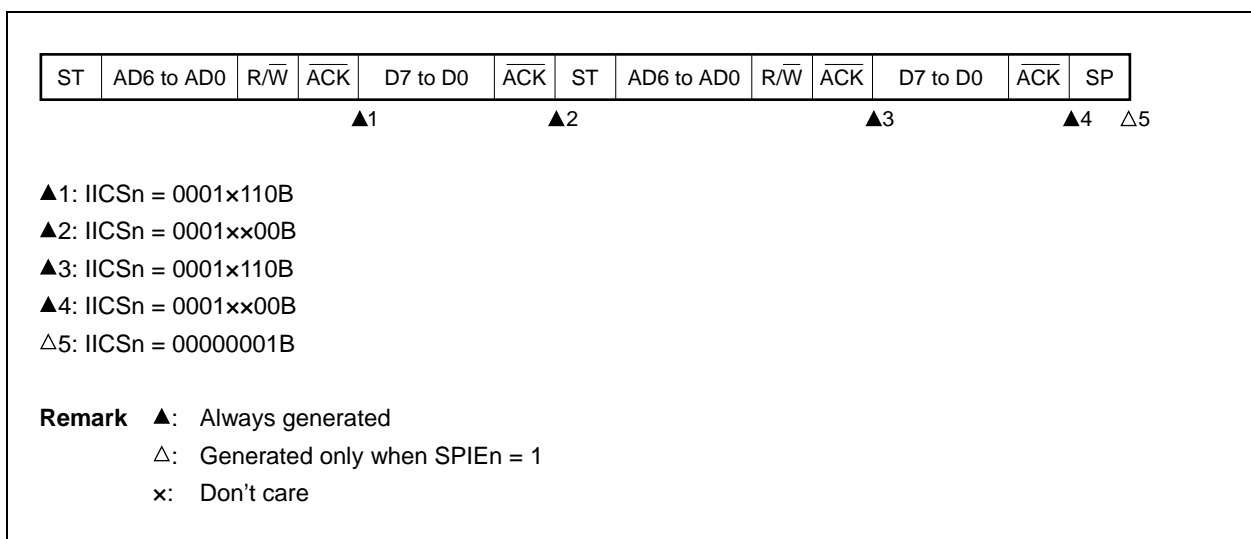
**Remark** n = 0, 1

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(i) When WTIMn = 0 (after restart, matches with SVAn)

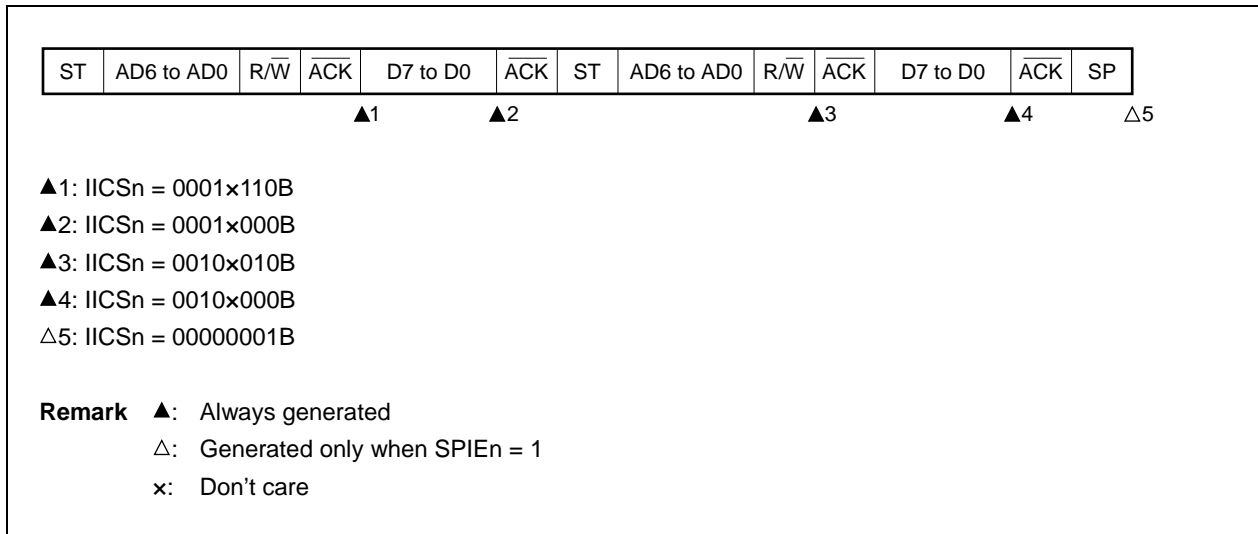
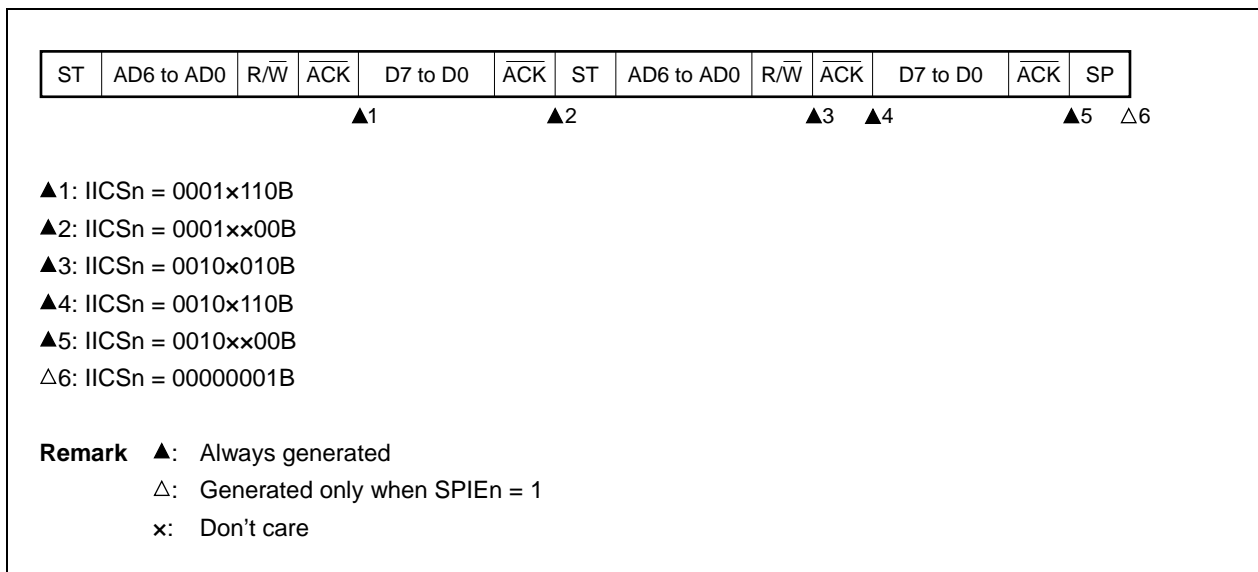


(ii) When WTIMn = 1 (after restart, matches with SVAn)



**Remark** n = 0, 1

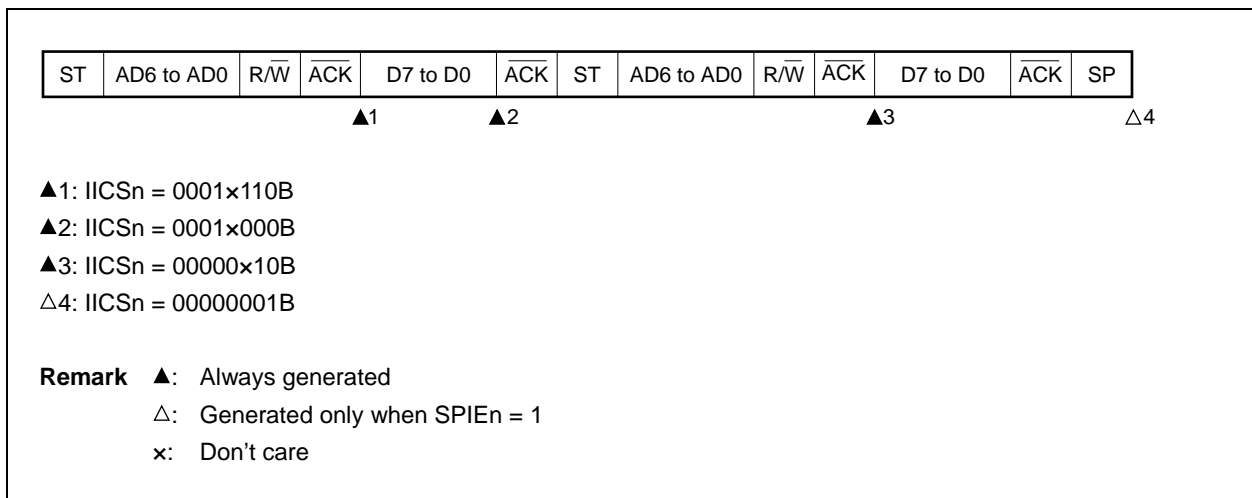
## (c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

(i) When  $WTIMn = 0$  (after restart, does not match address (= extension code))(ii) When  $WTIMn = 1$  (after restart, does not match address (= extension code))

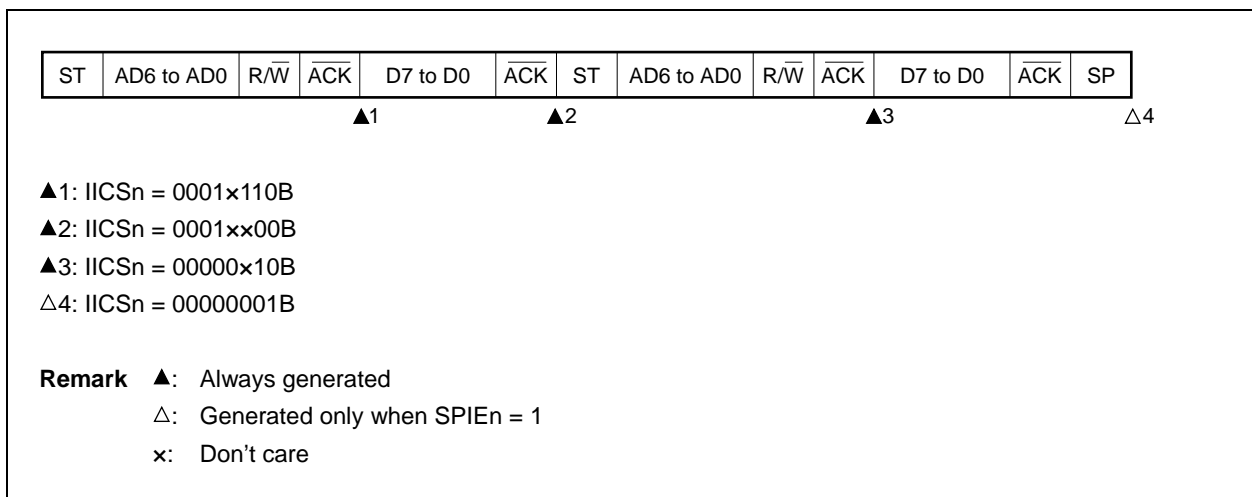
**Remark** n = 0, 1

(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop

(i) When WTIMn = 0 (after restart, does not match address (= not extension code))



(ii) When WTIMn = 1 (after restart, does not match address (= not extension code))

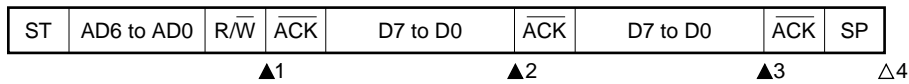


**Remark** n = 0, 1



**(3) Slave device operation (when receiving extension code)**

The device is always participating in communication when it receives an extension code.

**(a) Start ~ Code ~ Data ~ Data ~ Stop****(i) When  $WTIMn = 0$** 

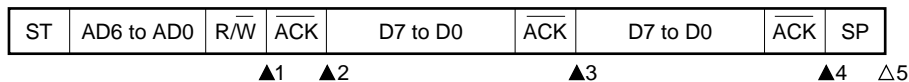
▲1: IICSn = 0010x010B

▲2: IICSn = 0010x000B

▲3: IICSn = 0010x000B

△4: IICSn = 00000001B

**Remark** ▲: Always generated  
 △: Generated only when SPIEn = 1  
 x: Don't care

**(ii) When  $WTIMn = 1$** 

▲1: IICSn = 0010x010B

▲2: IICSn = 0010x110B

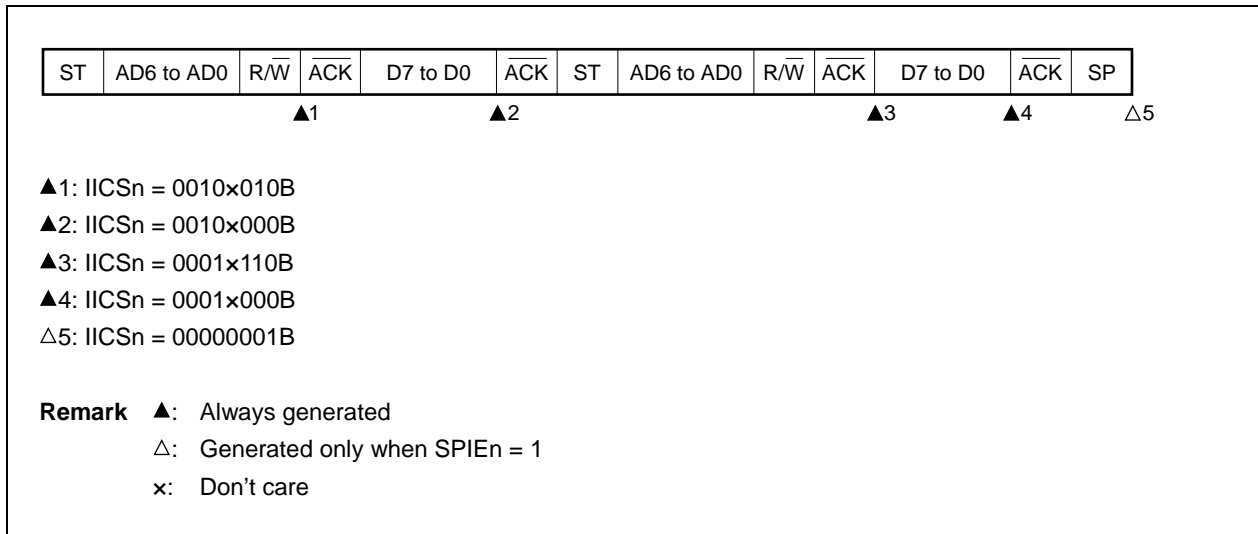
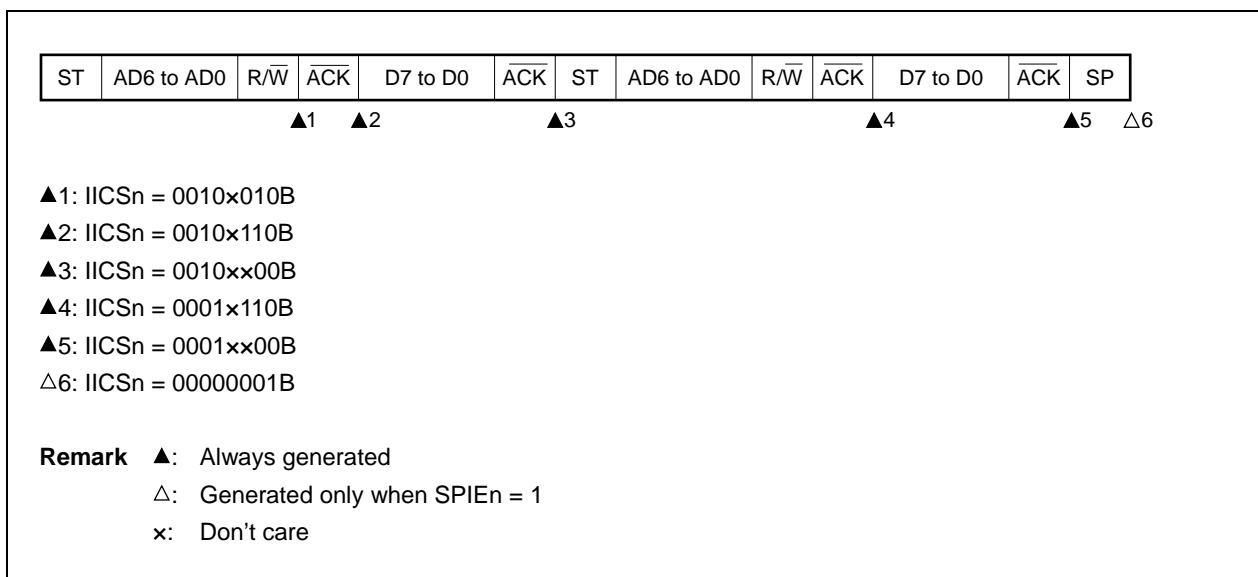
▲3: IICSn = 0010x100B

▲4: IICSn = 0010xx00B

△5: IICSn = 00000001B

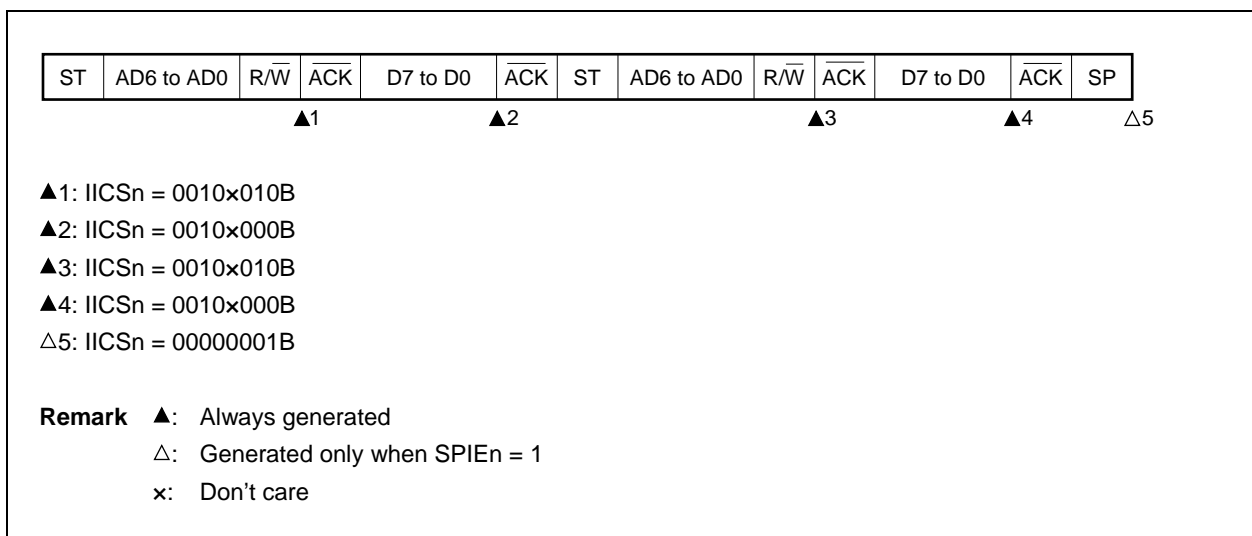
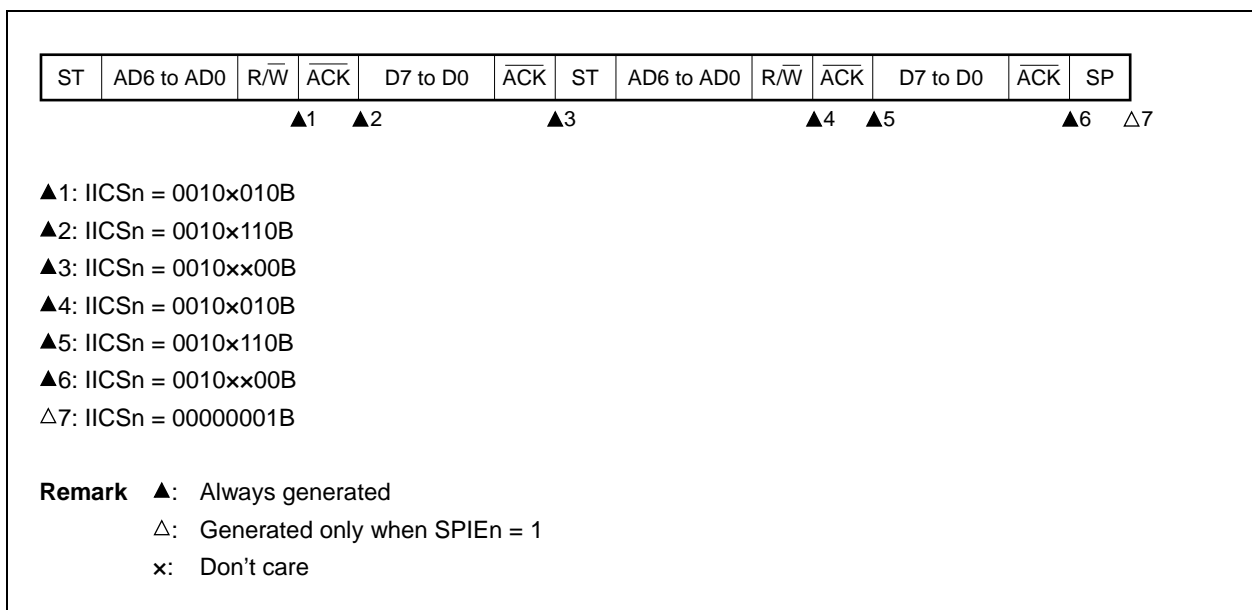
**Remark** ▲: Always generated  
 △: Generated only when SPIEn = 1  
 x: Don't care

**Remark** n = 0, 1

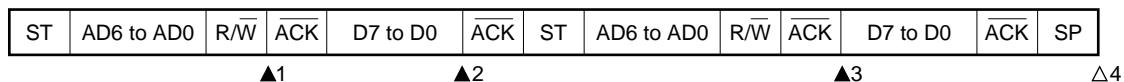
**(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop****(i) When WTIMn = 0 (after restart, matches SVAn)****(ii) When WTIMn = 1 (after restart, matches SVAn)**

**Remark** n = 0, 1

## (c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

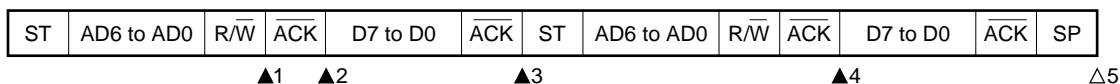
(i) When  $WTIMn = 0$  (after restart, extension code reception)(ii) When  $WTIMn = 1$  (after restart, extension code reception)

**Remark** n = 0, 1

**(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop****(i) When WTIMn = 0 (after restart, does not match address (= not extension code))**▲1: IICS<sub>n</sub> = 0010x010B▲2: IICS<sub>n</sub> = 0010x000B▲3: IICS<sub>n</sub> = 00000x10B▲4: IICS<sub>n</sub> = 00000001B**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

x: Don't care

**(ii) When WTIMn = 1 (after restart, does not match address (= not extension code))**▲1: IICS<sub>n</sub> = 0010x010B▲2: IICS<sub>n</sub> = 0010x110B▲3: IICS<sub>n</sub> = 0010xx00B▲4: IICS<sub>n</sub> = 00000x10B▲5: IICS<sub>n</sub> = 00000001B**Remark** ▲: Always generated

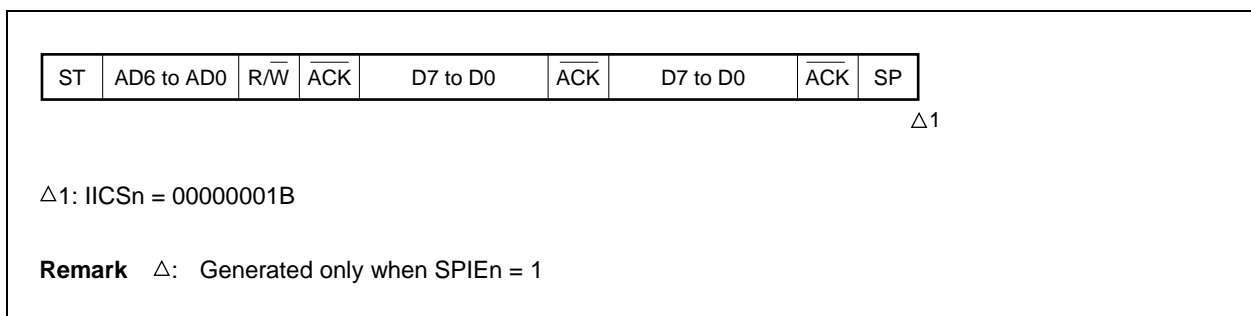
△: Generated only when SPIEn = 1

x: Don't care

**Remark** n = 0, 1

**(4) Operation without communication**

**(a) Start ~ Code ~ Data ~ Data ~ Stop**

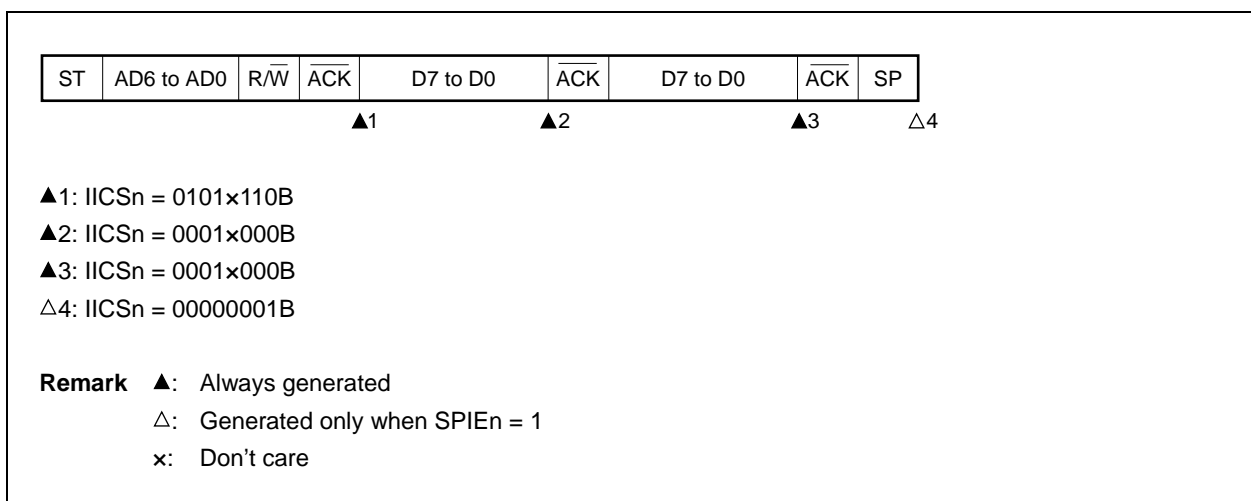


**(5) Arbitration loss operation (operation as slave after arbitration loss)**

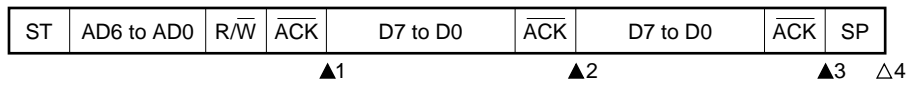
When the device is used as a master in a multi-master system, read the MSTSn bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

**(a) When arbitration loss occurs during transmission of slave address data**

**(i) When WTIMn = 0**



**Remark** n = 0, 1

(ii) When  $WTIMn = 1$ 

▲1: IICSn = 0101x110B

▲2: IICSn = 0001x100B

▲3: IICSn = 0001xx00B

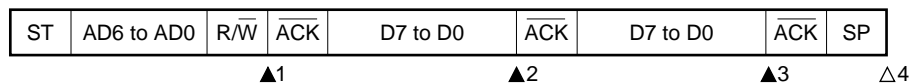
△4: IICSn = 00000001B

**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

x: Don't care

## (b) When arbitration loss occurs during transmission of extension code

(i) When  $WTIMn = 0$ 

▲1: IICSn = 0110x010B

▲2: IICSn = 0010x000B

▲3: IICSn = 0010x000B

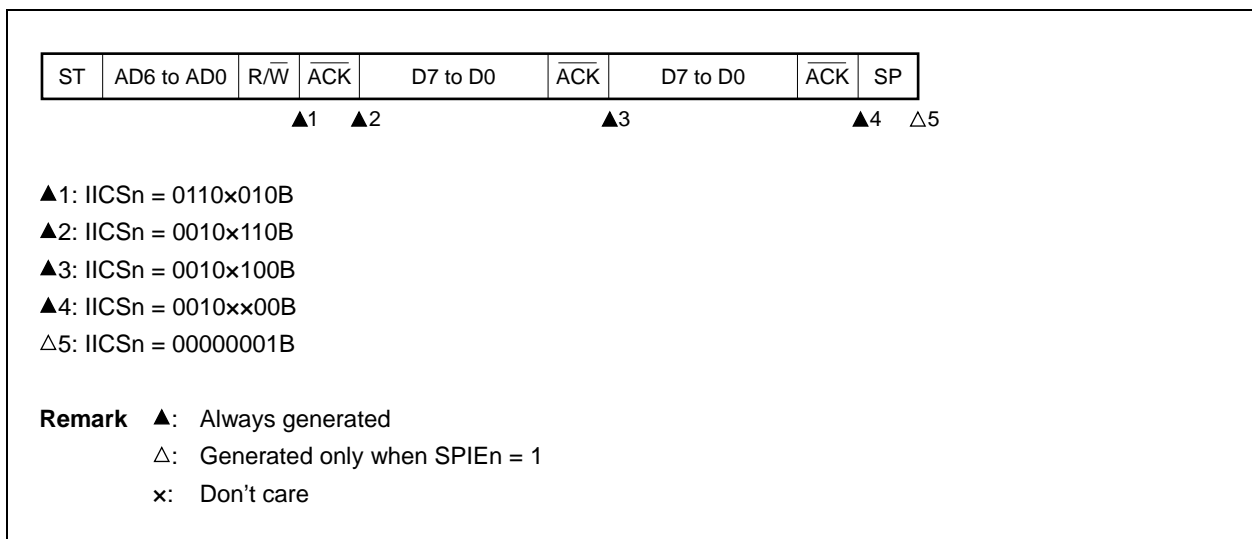
△4: IICSn = 00000001B

**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

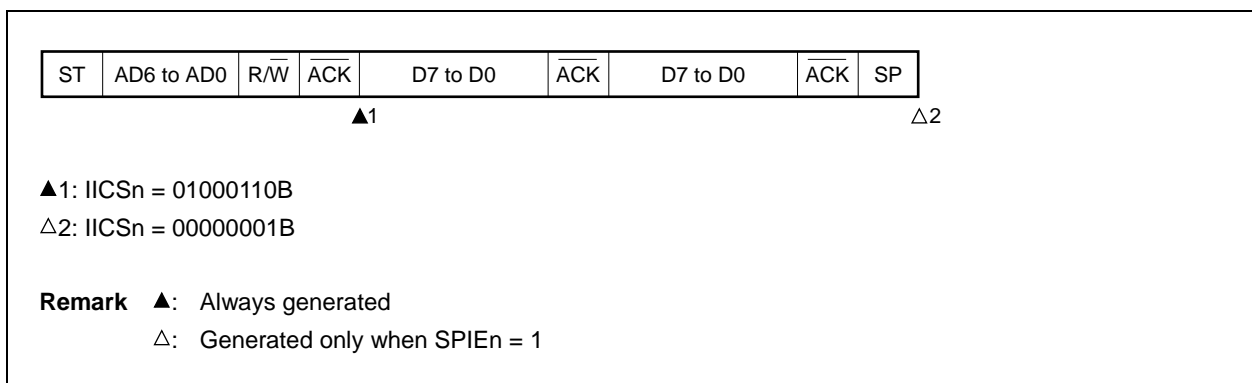
x: Don't care

**Remark** n = 0, 1

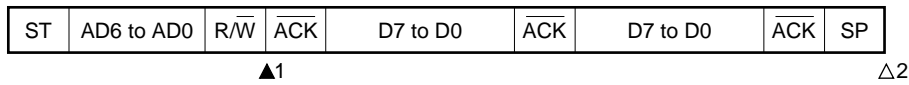
(ii) When  $WTIMn = 1$ 

## (6) Operation when arbitration loss occurs (no communication after arbitration loss)

When the device is used as a master in a multi-master system, read the MSTS<sub>n</sub> bit each time interrupt request signal INTIICAn has occurred to check the arbitration result.

(a) When arbitration loss occurs during transmission of slave address data (when  $WTIMn = 1$ )

**Remark** n = 0, 1

**(b) When arbitration loss occurs during transmission of extension code**

▲1: IICSn = 0110x010B

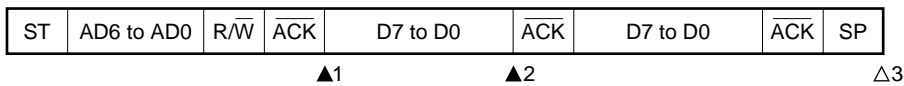
Sets LRELn = 1 by software

△2: IICSn = 00000001B

**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

x: Don't care

**(c) When arbitration loss occurs during transmission of data****(i) When WTIMn = 0**

▲1: IICSn = 10001110B

▲2: IICSn = 01000000B

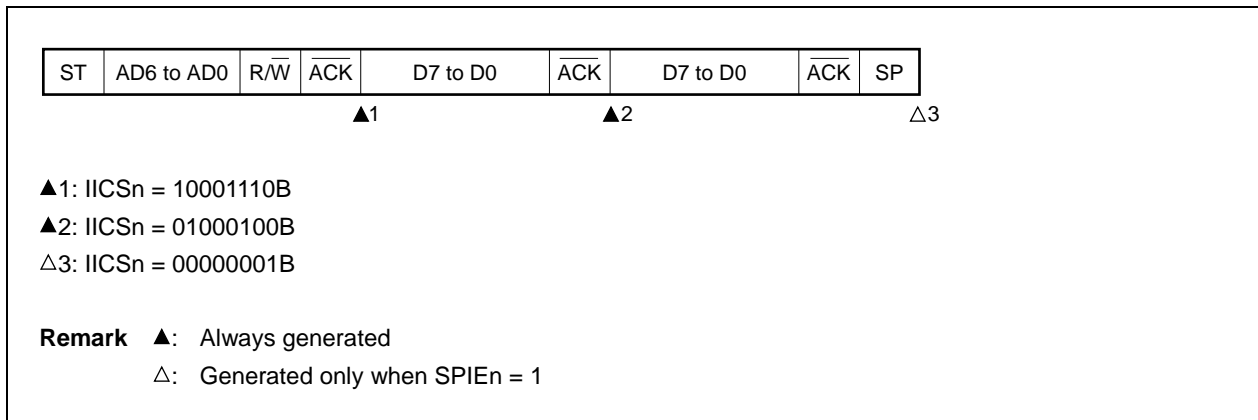
△3: IICSn = 00000001B

**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

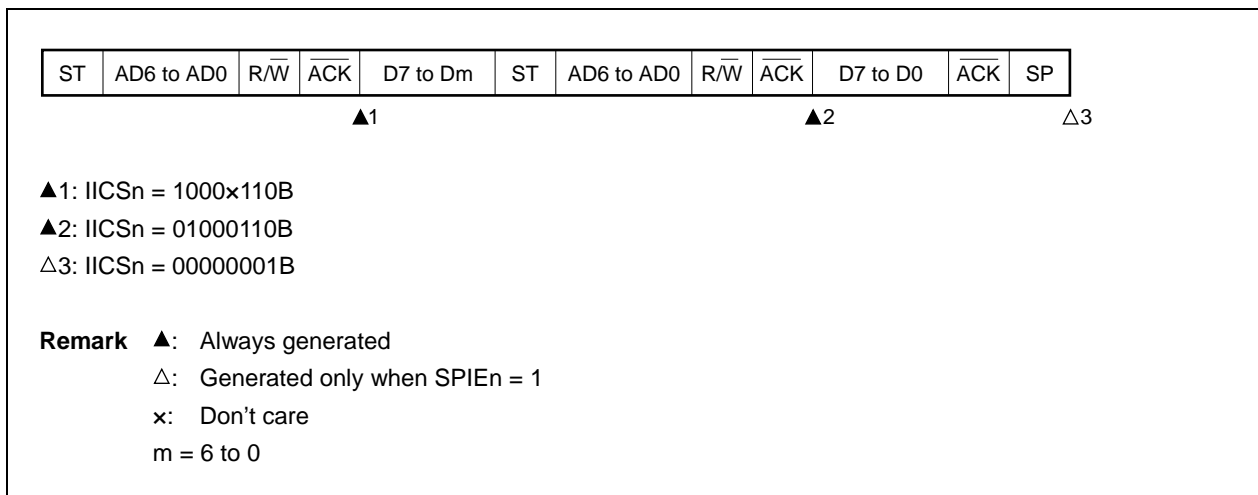
**Remark** n = 0, 1



(ii) When  $WTIMn = 1$ 

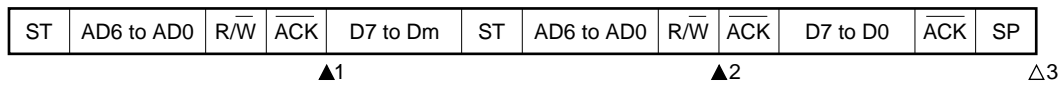
## (d) When loss occurs due to restart condition during data transfer

## (i) Not extension code (Example: unmatched with SVAn)



**Remark** n = 0, 1

## (ii) Extension code



▲1: IICSn = 1000x110B

▲2: IICSn = 01100010B

Sets LRELn = 1 by software

△3: IICSn = 00000001B

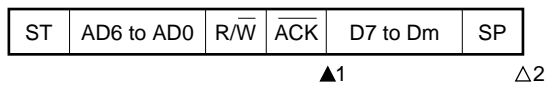
**Remark** ▲: Always generated

△: Generated only when SPIEn = 1

x: Don't care

m = 6 to 0

## (e) When loss occurs due to stop condition during data transfer



▲1: IICSn = 10000110B

△2: IICSn = 01000001B

**Remark** ▲: Always generated

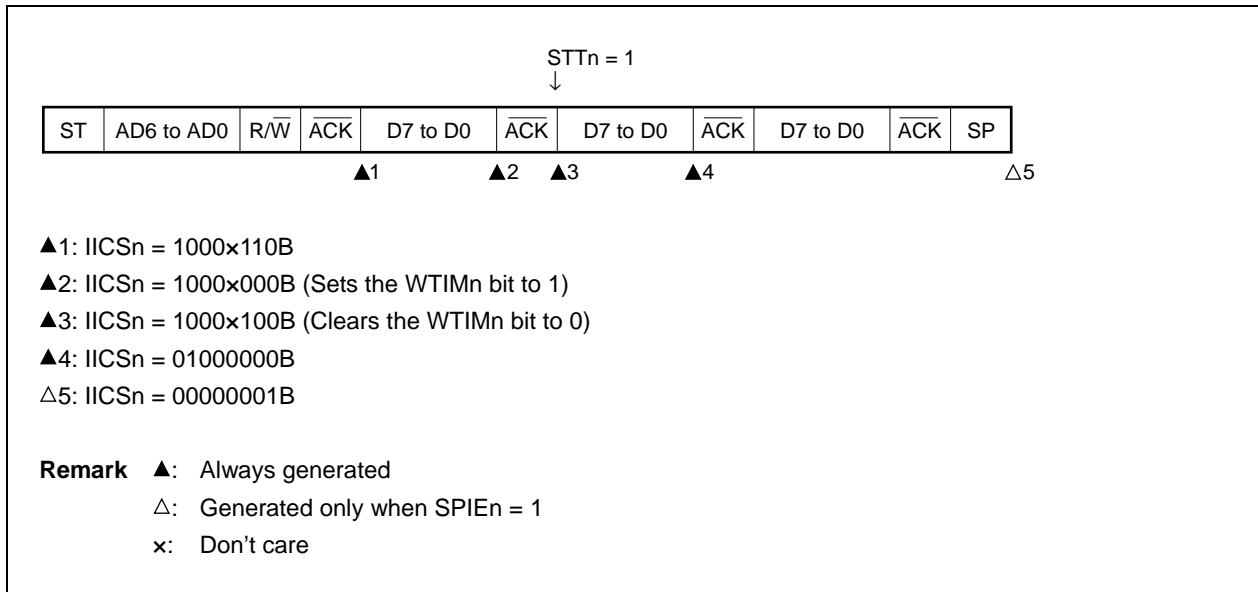
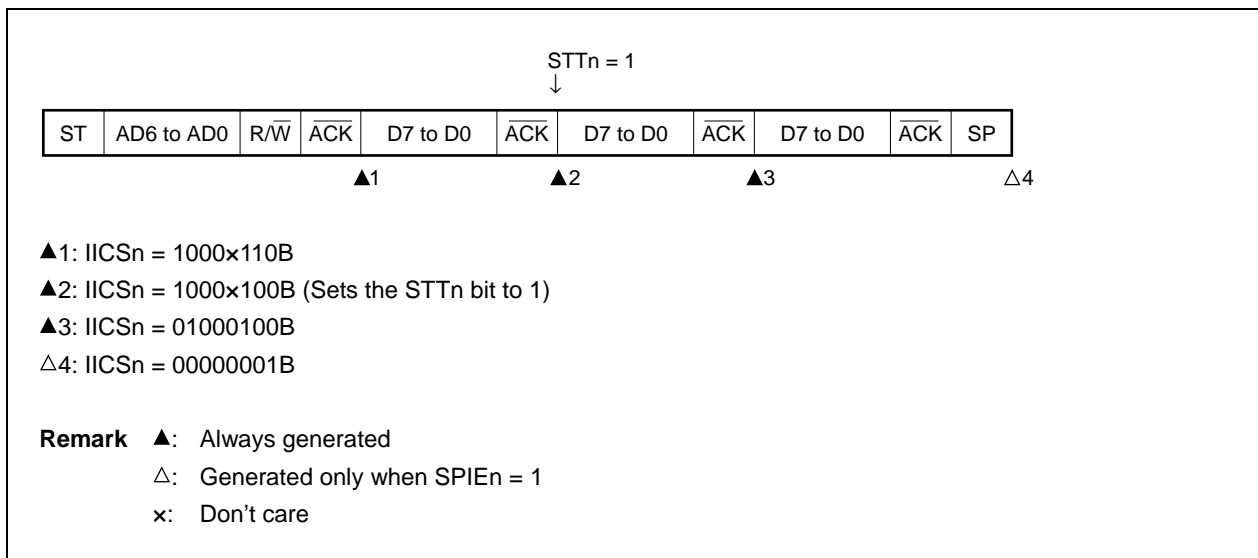
△: Generated only when SPIEn = 1

x: Don't care

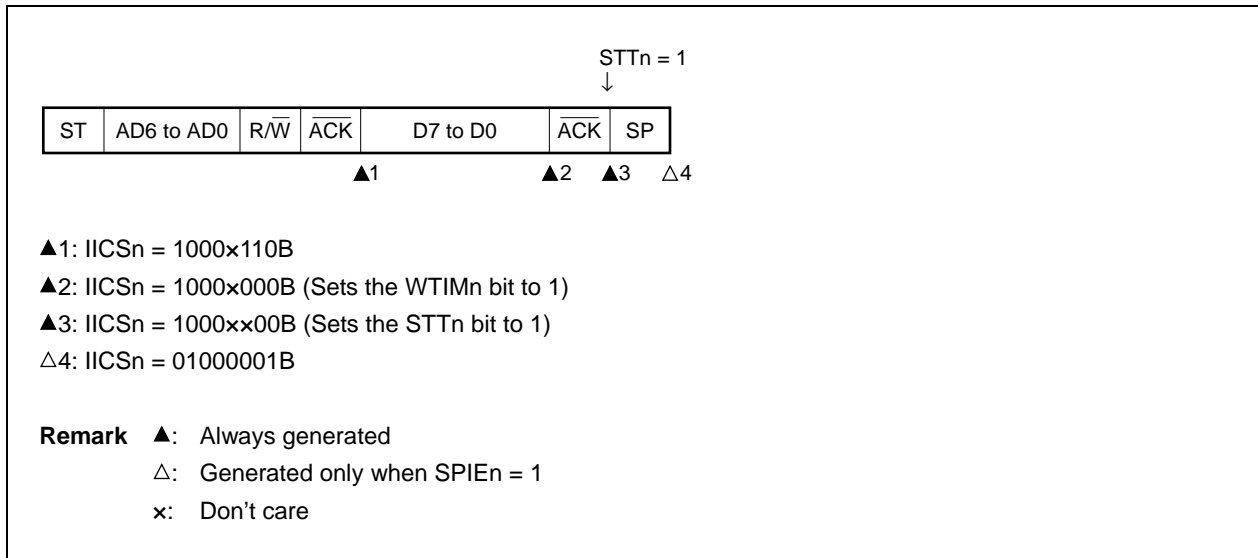
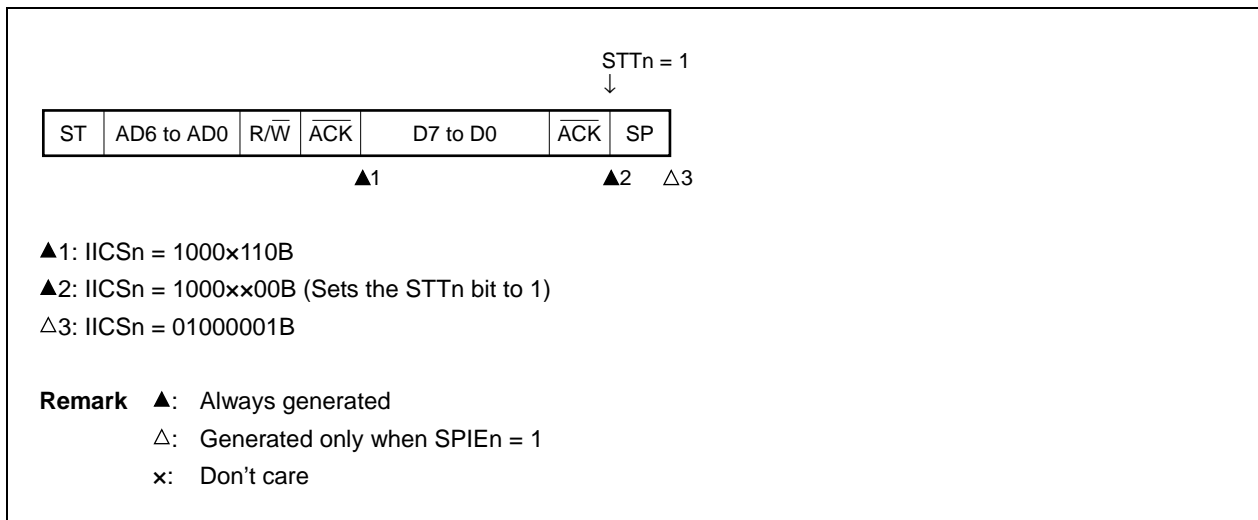
m = 6 to 0

**Remark** n = 0, 1

## (f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

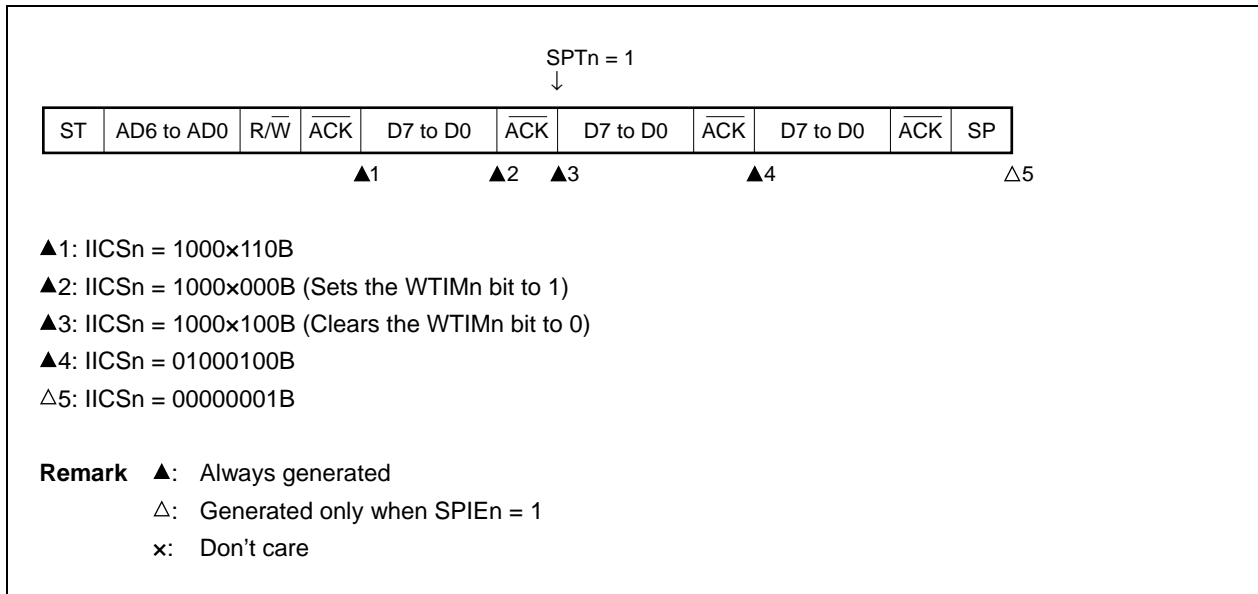
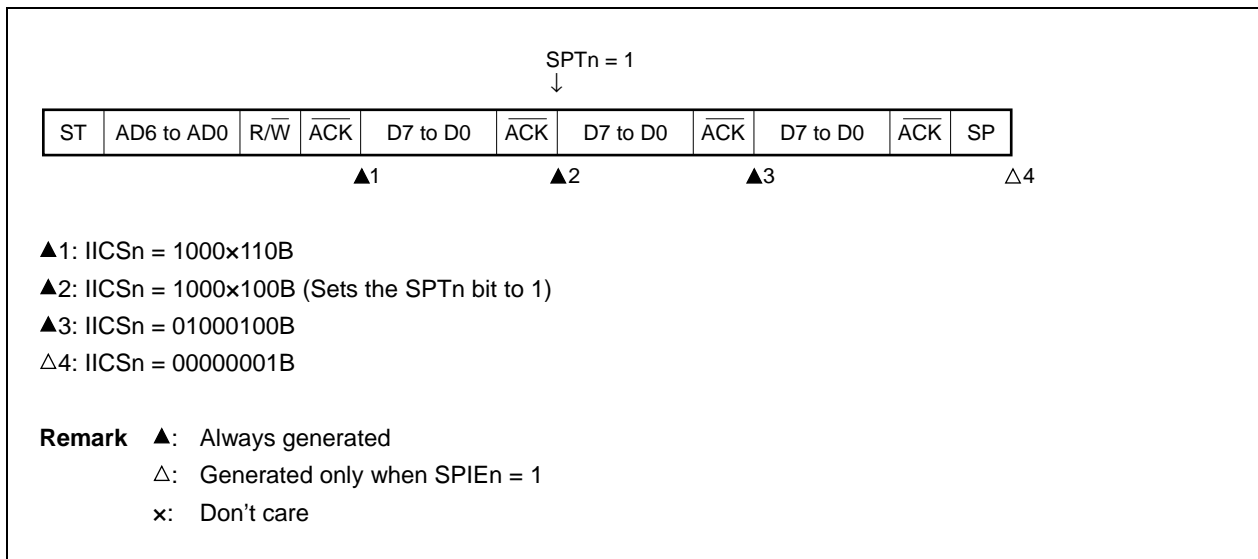
(i) When  $WTIMn = 0$ (ii) When  $WTIMn = 1$ 

**Remark** n = 0, 1

**(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition****(i) When  $WTIMn = 0$** **(ii) When  $WTIMn = 1$** 

**Remark** n = 0, 1

## (h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

(i) When  $WTIMn = 0$ (ii) When  $WTIMn = 1$ 

**Remark** n = 0, 1

## 12.6 Timing Charts

When using the I<sup>2</sup>C bus mode, the master device outputs an address via the serial bus to select one of several slave devices as its communication partner.

After outputting the slave address, the master device transmits the TRCn bit (bit 3 of the IICA status register n (IICSn)), which specifies the data transfer direction, and then starts serial communication with the slave device.

Figures 12-32 and 12-33 show timing charts of the data communication.

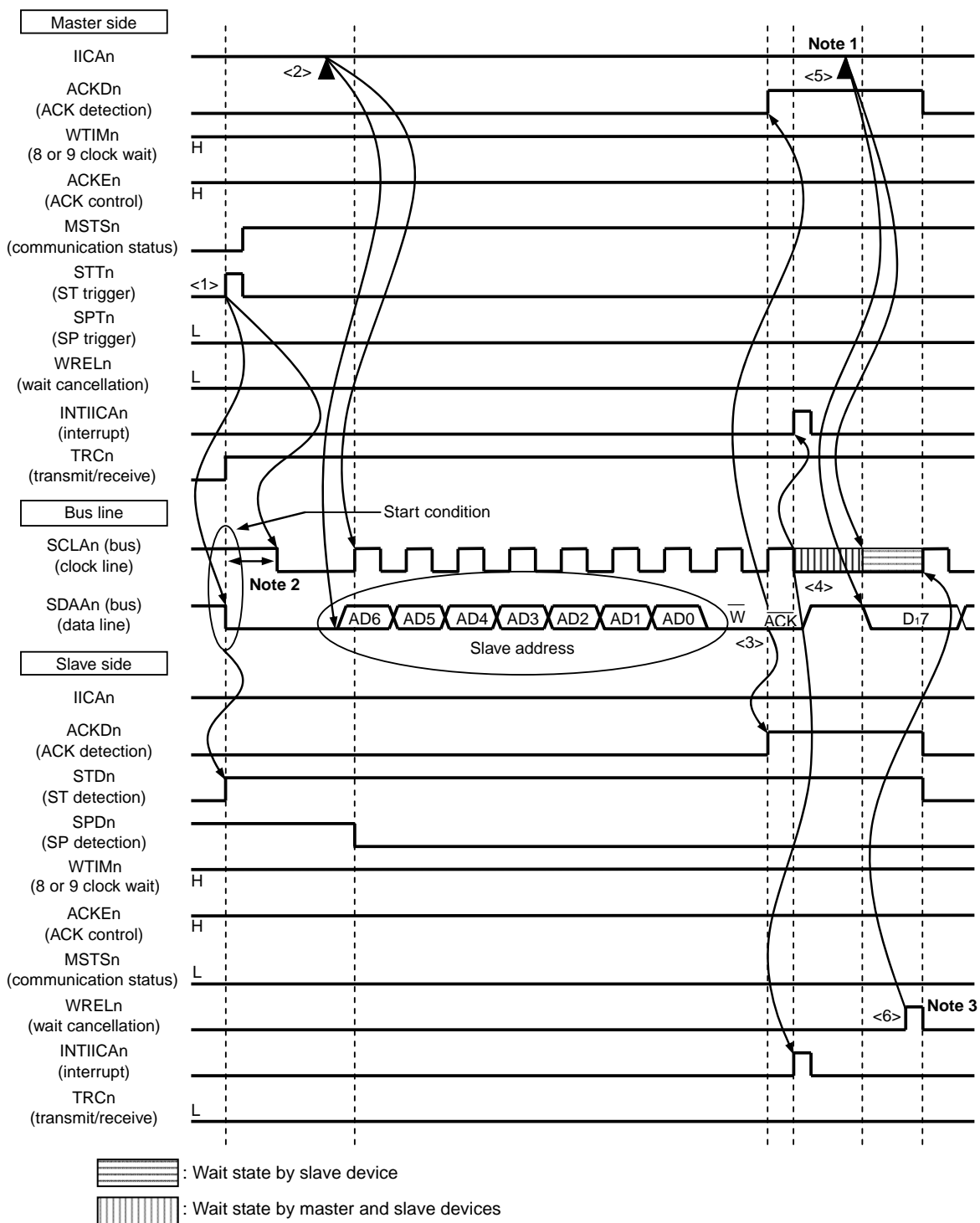
The IICA shift register n (IICAn)'s shift operation is synchronized with the falling edge of the serial clock (SCLAn). The transmit data is transferred to the SO latch and is output (MSB first) via the SDAAn pin.

Data input via the SDAAn pin is captured into IICAn at the rising edge of SCLAn.

**Remark** n = 0, 1

**Figure 12-32. Example of Master to Slave Communication (9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/4)**

**(1) Start condition ~ address ~ data**



- Notes**
  - Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.
  - Make sure that the time between the fall of the SDAAn pin signal and the fall of the SCLAn pin signal is at least 4.0  $\mu$ s when specifying standard mode and at least 0.6  $\mu$ s when specifying fast mode.
  - For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

**Remark** n = 0, 1

The meanings of <1> to <6> in (1) Start condition ~ address ~ data in Figure 12-32 are explained below.

- <1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn = 1 changes SDAAn from 1 to 0) is generated once the bus data line goes low (SDAAn). When the start condition is subsequently detected, the master device enters the master device communication status (MSTS<sub>n</sub> = 1). The master device is ready to communicate once the bus clock line goes low (SCLAn = 0) after the hold time has elapsed.
- <2> The master device writes the address + W (transmission) to the IICA shift register n (IICAn) and transmits the slave address.
- <3> In the slave device if the address received matches the address (SVAn value) of a slave device<sup>Note</sup>, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)<sup>Note</sup>.
- <5> The master device writes the data to transmit to the IICAn register and releases the wait status that it set by the master device.
- <6> If the slave device releases the wait status (WRELn = 1), the master device starts transferring data to the slave device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remarks 1.** <1> to <15> in Figure 12-32 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

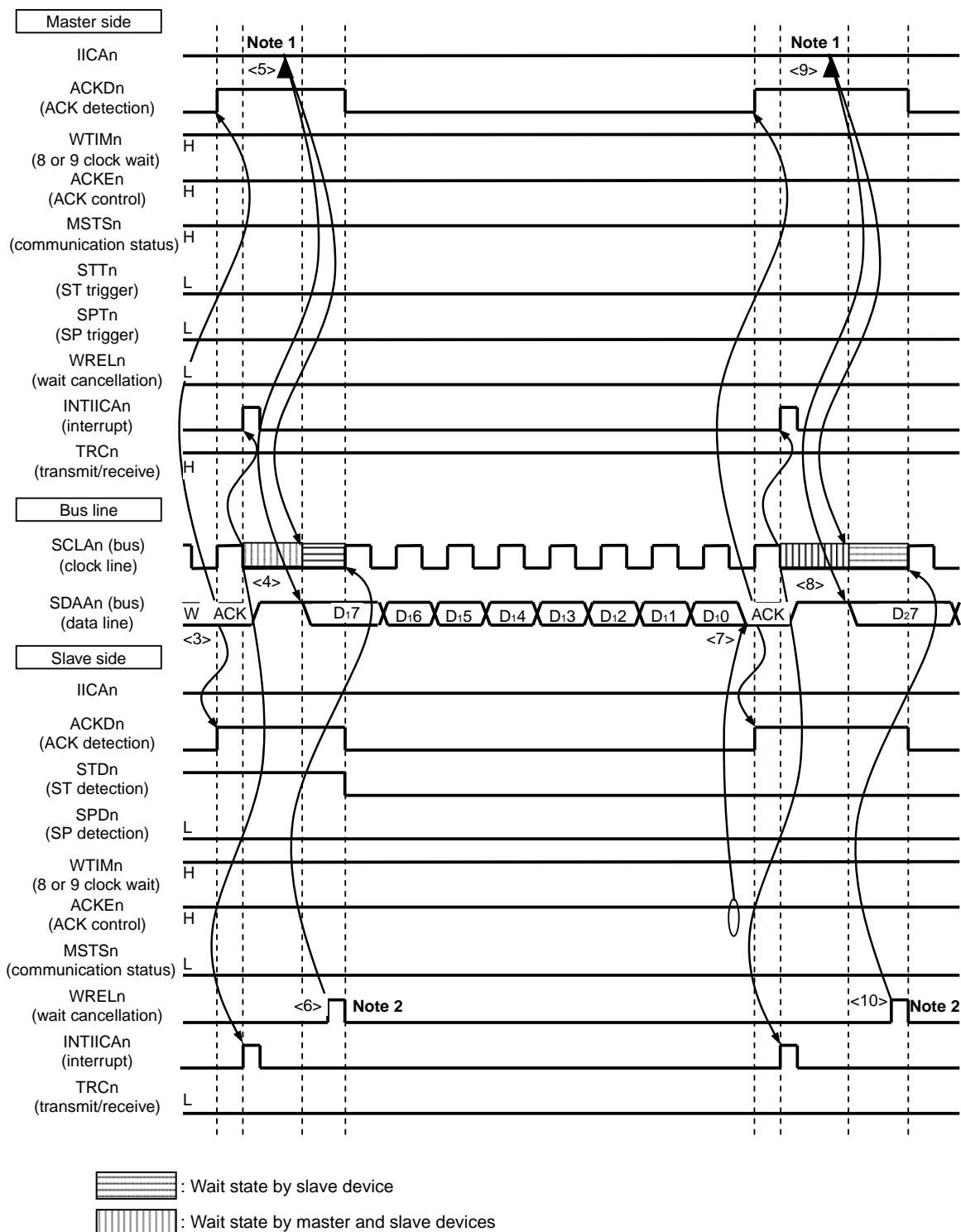
Figure 12-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 12-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 12-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

- 2. n = 0, 1



**Figure 12-32. Example of Master to Slave Communication  
(9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/4)**

**(2) Address ~ data ~ data**



- Notes**
1. Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.
  2. For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

**Remark** n = 0, 1

The meanings of <3> to <10> in (2) Address ~ data ~ data in Figure 12-32 are explained below.

- <3> In the slave device if the address received matches the address (SVAn value) of a slave device<sup>Note</sup>, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)<sup>Note</sup>.
- <5> The master device writes the data to transmit to the IICA shift register n (IICAn) and releases the wait status that it set by the master device.
- <6> If the slave device releases the wait status (WRELn = 1), the master device starts transferring data to the slave device.
- <7> After data transfer is completed, because of ACKEn = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <8> The master device and slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).
- <9> The master device writes the data to transmit to the IICAn register and releases the wait status that it set by the master device.
- <10> The slave device reads the received data and releases the wait status (WRELn = 1). The master device then starts transferring data to the slave device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

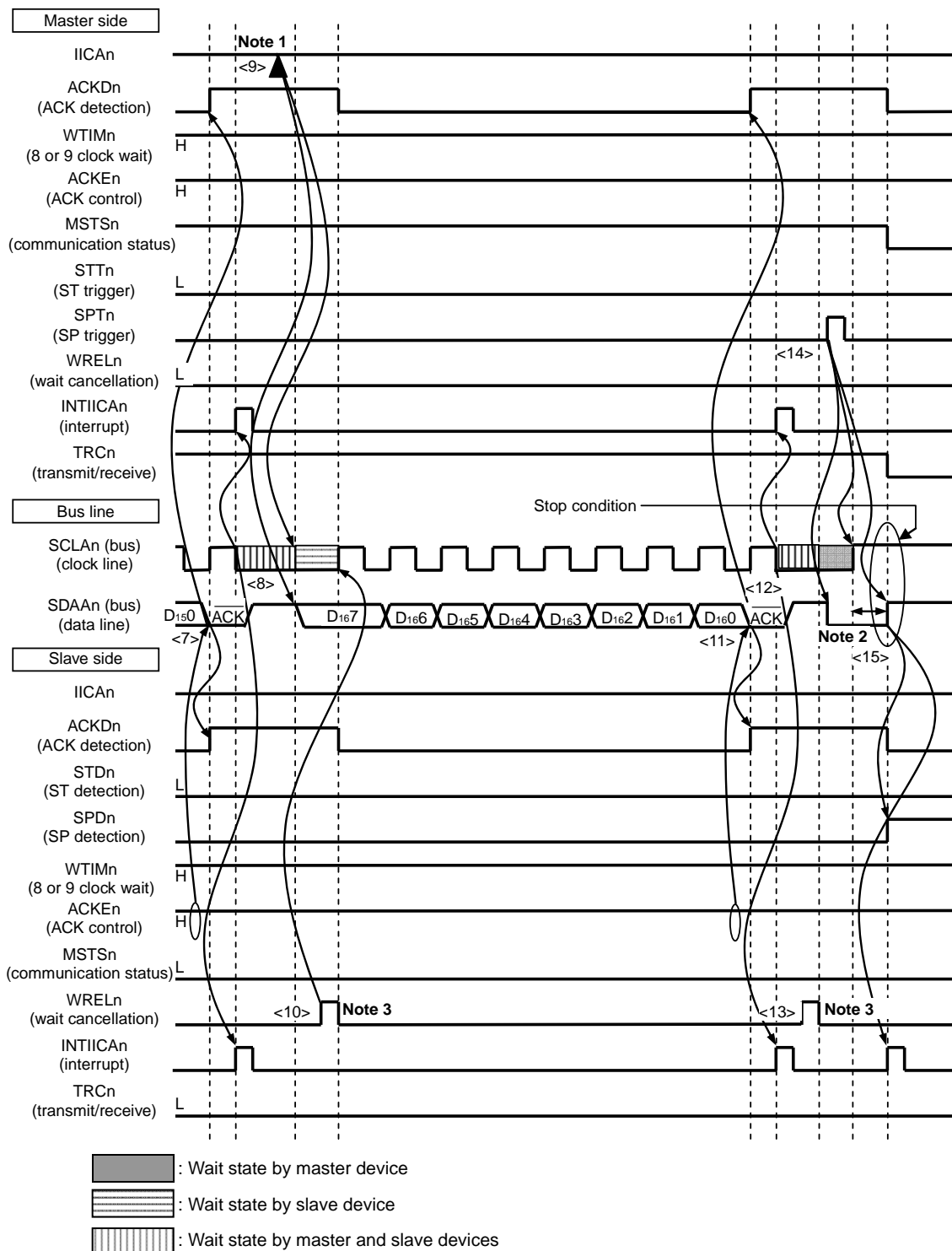
**Remarks 1.** <1> to <15> in Figure 12-32 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

Figure 12-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 12-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 12-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**2.** n = 0, 1

**Figure 12-32. Example of Master to Slave Communication**  
**(9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/4)**

**(3) Data ~ data ~ Stop condition**



- Notes**
1. Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a master device.
  2. Make sure that the time between the rise of the SCLAn pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0  $\mu$ s when specifying standard mode and at least 0.6  $\mu$ s when specifying fast mode.
  3. For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

**Remark** n = 0, 1

The meanings of <7> to <15> in (3) Data ~ data ~ stop condition in Figure 12-32 are explained below.

- <7> After data transfer is completed, because of ACKEn = 1, the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <8> The master device and slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).
- <9> The master device writes the data to transmit to the IICA shift register n (IICAn) and releases the wait status that it set by the master device.
- <10> The slave device reads the received data and releases the wait status (WRELn = 1). The master device then starts transferring data to the slave device.
- <11> When data transfer is complete, the slave device (ACKEn = 1) sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <12> The master device and slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).
- <13> The slave device reads the received data and releases the wait status (WRELn = 1).
- <14> By the master device setting a stop condition trigger (SPTn = 1), the bus data line is cleared (SDAAn = 0) and the bus clock line is set (SCLAn = 1). After the stop condition setup time has elapsed, by setting the bus data line (SDAAn = 1), the stop condition is then generated (i.e. SCLAn = 1 changes SDAAn from 0 to 1).
- <15> When a stop condition is generated, the slave device detects the stop condition and issues an interrupt (INTIICAn: stop condition).

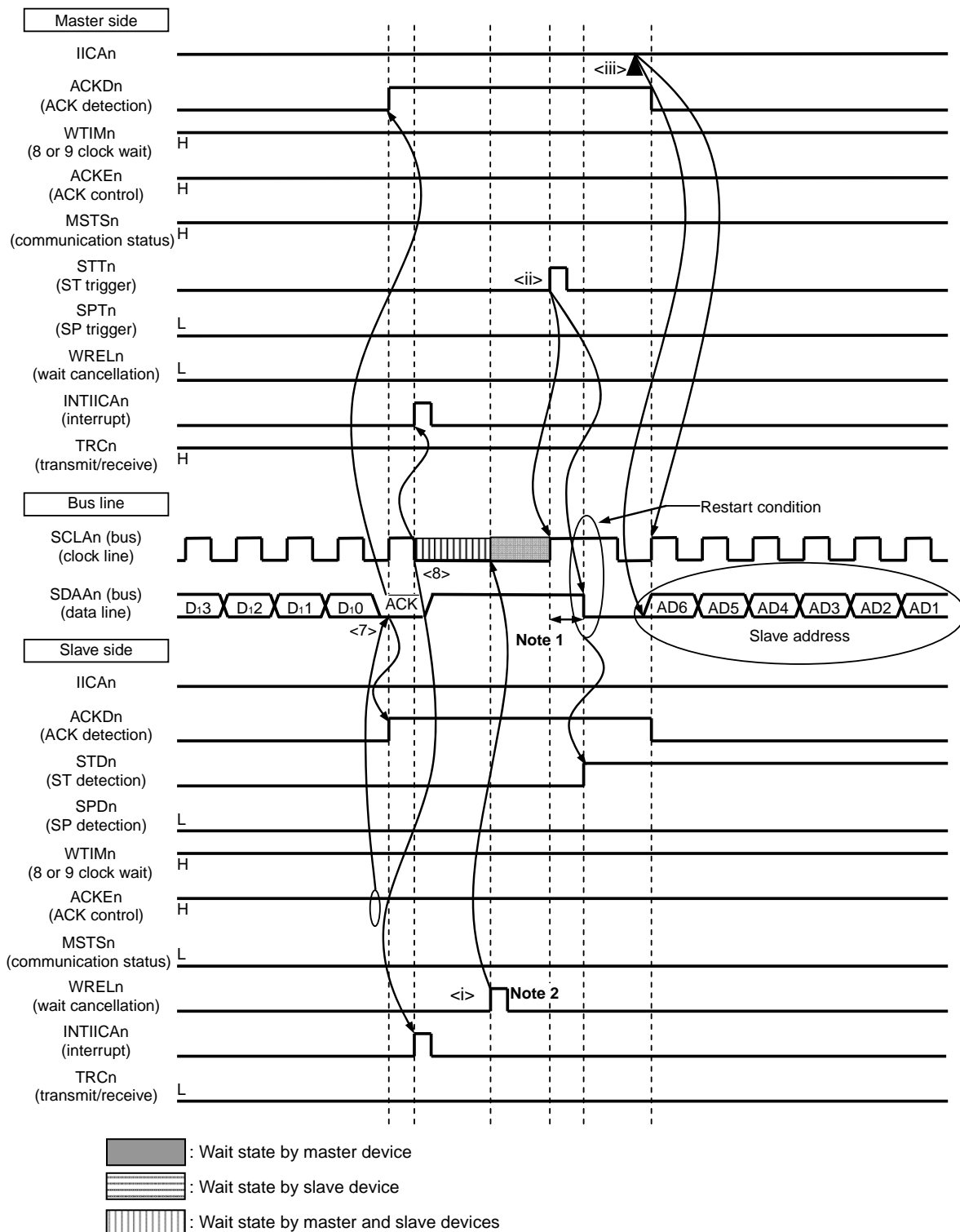
**Remarks 1.** <1> to <15> in Figure 12-32 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

Figure 12-32 (1) Start condition ~ address ~ data shows the processing from <1> to <6>, Figure 12-32 (2) Address ~ data ~ data shows the processing from <3> to <10>, and Figure 12-32 (3) Data ~ data ~ stop condition shows the processing from <7> to <15>.

**2.** n = 0, 1

Figure 12-32. Example of Master to Slave Communication  
(9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (4/4)

(4) Data ~ restart condition ~ address



- Notes**
1. Make sure that the time between the rise of the SCLAn pin signal and the generation of the start condition after a restart condition has been issued is at least 4.7  $\mu$ s when specifying standard mode and at least 0.6  $\mu$ s when specifying fast mode.
  2. For releasing wait state during reception of a slave device, write "FFH" to IICAn or set the WRELn bit.

**Remark** n = 0, 1

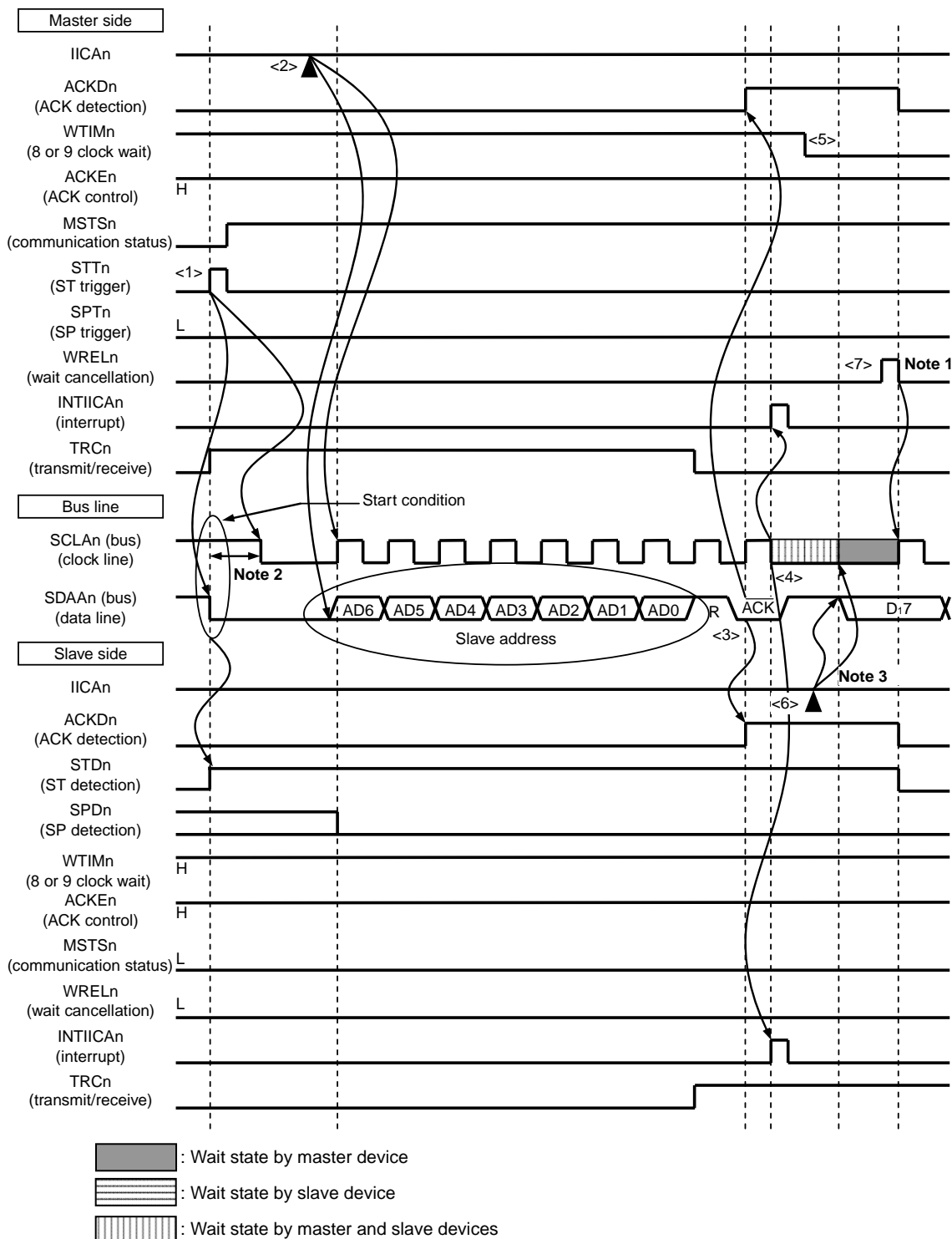
The following describes the operations in Figure 12-32 (4) Data ~ restart condition ~ address. After the operations in steps <7> and <8>, the operations in steps <i> to <iii> are performed. These steps return the processing to step <iii>, the data transmission step.

- <7> After data transfer is completed, because of  $ACKEn = 1$ , the slave device sends an ACK by hardware to the master device. The ACK is detected by the master device ( $ACKDn = 1$ ) at the rising edge of the 9th clock.
- <8> The master device and slave device set a wait status ( $SCLAn = 0$ ) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt ( $INTIICAn$ : end of transfer).
- <i> The slave device reads the received data and releases the wait status ( $WRELn = 1$ ).
- <ii> The start condition trigger is set again by the master device ( $STTn = 1$ ) and a start condition (i.e.  $SCLAn = 1$  changes  $SDAAn$  from 1 to 0) is generated once the bus clock line goes high ( $SCLAn = 1$ ) and the bus data line goes low ( $SDAAn = 0$ ) after the restart condition setup time has elapsed. When the start condition is subsequently detected, the master device is ready to communicate once the bus clock line goes low ( $SCLAn = 0$ ) after the hold time has elapsed.
- <iii> The master device writing the address + R/W (transmission) to the IICA shift register ( $IICAn$ ) enables the slave address to be transmitted.

**Remark** n = 0, 1

**Figure 12-33. Example of Slave to Master Communication**  
**(8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/3)**

**(1) Start condition ~ address ~ data**



**Remark** n = 0, 1

The meanings of <1> to <7> in (1) Start condition ~ address ~ data in Figure 12-33 are explained below.

- <1> The start condition trigger is set by the master device (STTn = 1) and a start condition (i.e. SCLAn = 1 changes SDAAn from 1 to 0) is generated once the bus data line goes low (SDAAn). When the start condition is subsequently detected, the master device enters the master device communication status (MSTSn = 1). The master device is ready to communicate once the bus clock line goes low (SCLAn = 0) after the hold time has elapsed.
- <2> The master device writes the address + R (reception) to the IICA shift register n (IICAn) and transmits the slave address.
- <3> In the slave device if the address received matches the address (SVAn value) of a slave device<sup>Note</sup>, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)<sup>Note</sup>.
- <5> The timing at which the master device sets the wait status changes to the 8th clock (WTIMn = 0).
- <6> The slave device writes the data to transmit to the IICAn register and releases the wait status that it set by the slave device.
- <7> The master device releases the wait status (WRELn = 1) and starts transferring data from the slave device to the master device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

**Remarks 1.** <1> to <19> in Figure 12-33 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

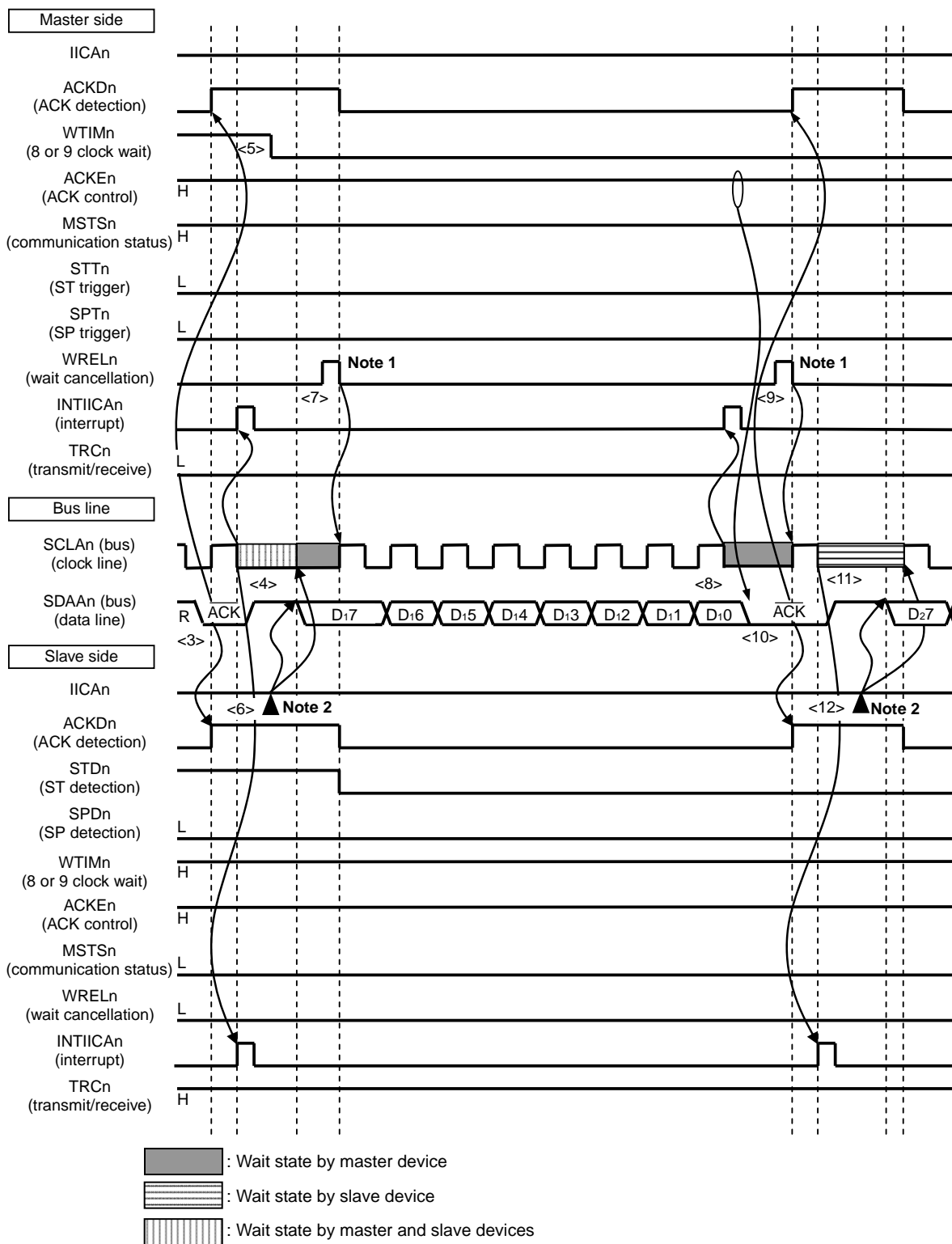
Figure 12-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 12-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 12-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

- 2. n = 0, 1



**Figure 12-33. Example of Slave to Master Communication (8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/3)**

**(2) Address ~ data ~ data**



- Notes**
- For releasing wait state during reception of a master device, write "FFH" to IICAn or set the WRELn bit.
  - Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a slave device.

**Remark** n = 0, 1

The meanings of <3> to <12> in (2) Address ~ data ~ data in Figure 12-33 are explained below.

- <3> In the slave device if the address received matches the address (SVAn value) of a slave device<sup>Note</sup>, that slave device sends an ACK by hardware to the master device. The ACK is detected by the master device (ACKDn = 1) at the rising edge of the 9th clock.
- <4> The master device issues an interrupt (INTIICAn: end of address transmission) at the falling edge of the 9th clock. The slave device whose address matched the transmitted slave address sets a wait status (SCLAn = 0) and issues an interrupt (INTIICAn: address match)<sup>Note</sup>.
- <5> The master device changes the timing of the wait status to the 8th clock (WTIMn = 0).
- <6> The slave device writes the data to transmit to the IICA shift register n (IICAn) and releases the wait status that it set by the slave device.
- <7> The master device releases the wait status (WRELn = 1) and starts transferring data from the slave device to the master device.
- <8> The master device sets a wait status (SCLAn = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICAn: end of transfer). Because of ACKEn = 1 in the master device, the master device then sends an ACK by hardware to the slave device.
- <9> The master device reads the received data and releases the wait status (WRELn = 1).
- <10> The ACK is detected by the slave device (ACKDn = 1) at the rising edge of the 9th clock.
- <11> The slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICAn: end of transfer).
- <12> By the slave device writing the data to transmit to the IICAn register, the wait status set by the slave device is released. The slave device then starts transferring data to the master device.

**Note** If the transmitted address does not match the address of the slave device, the slave device does not return an ACK to the master device (NACK: SDAAn = 1). The slave device also does not issue the INTIICAn interrupt (address match) and does not set a wait status. The master device, however, issues the INTIICAn interrupt (end of address transmission) regardless of whether it receives an ACK or NACK.

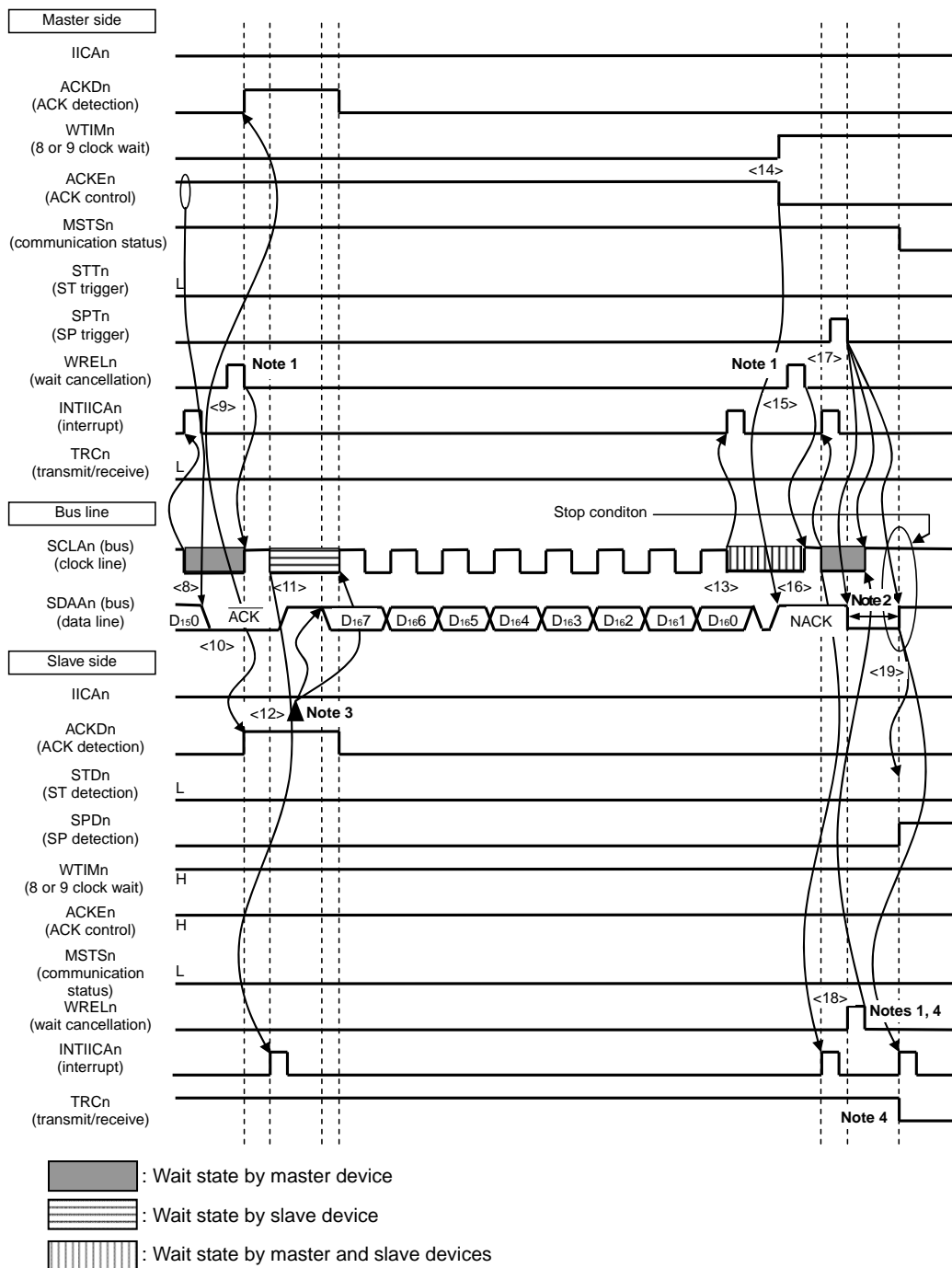
**Remarks 1.** <1> to <19> in Figure 12-33 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

Figure 12-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 12-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 12-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

2. n = 0, 1

**Figure 12-33. Example of Slave to Master Communication**  
**(8-Clock and 9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/3)**

**(3) Data ~ data ~ stop condition**



- Notes**
1. To cancel a wait state, write "FFH" to IICAn or set the WRELn bit.
  2. Make sure that the time between the rise of the SCLAn pin signal and the generation of the stop condition after a stop condition has been issued is at least 4.0  $\mu$ s when specifying standard mode and at least 0.6  $\mu$ s when specifying fast mode.
  3. Write data to IICAn, not setting the WRELn bit, in order to cancel a wait state during transmission by a slave device.
  4. If a wait state during transmission by a slave device is canceled by setting the WRELn bit, the TRCn bit will be cleared.

**Remark** n = 0, 1

The meanings of <8> to <19> in (3) Data ~ data ~ stop condition in Figure 12-33 are explained below.

- <8> The master device sets a wait status (SCLAn = 0) at the falling edge of the 8th clock, and issues an interrupt (INTIICAn: end of transfer). Because of ACKEn = 0 in the master device, the master device then sends an ACK by hardware to the slave device.
- <9> The master device reads the received data and releases the wait status (WRELn = 1).
- <10> The ACK is detected by the slave device (ACKDn = 1) at the rising edge of the 9th clock.
- <11> The slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and the slave device issue an interrupt (INTIICAn: end of transfer).
- <12> By the slave device writing the data to transmit to the IICA register, the wait status set by the slave device is released. The slave device then starts transferring data to the master device.
- <13> The master device issues an interrupt (INTIICAn: end of transfer) at the falling edge of the 8th clock, and sets a wait status (SCLAn = 0). Because ACK control (ACKEn = 1) is performed, the bus data line is at the low level (SDAAn = 0) at this stage.
- <14> The master device sets NACK as the response (ACKEn = 0) and changes the timing at which it sets the wait status to the 9th clock (WTIMn = 1).
- <15> If the master device releases the wait status (WRELn = 1), the slave device detects the NACK (ACK = 0) at the rising edge of the 9th clock.
- <16> The master device and slave device set a wait status (SCLAn = 0) at the falling edge of the 9th clock, and both the master device and slave device issue an interrupt (INTIICAn: end of transfer).
- <17> When the master device issues a stop condition (SPTn = 1), the bus data line is cleared (SDAAn = 0) and the master device releases the wait status. The master device then waits until the bus clock line is set (SCLAn = 1).
- <18> The slave device acknowledges the NACK, halts transmission, and releases the wait status (WRELn = 1) to end communication. Once the slave device releases the wait status, the bus clock line is set (SCLAn = 1).
- <19> Once the master device recognizes that the bus clock line is set (SCLAn = 1) and after the stop condition setup time has elapsed, the master device sets the bus data line (SDAAn = 1) and issues a stop condition (i.e. SCLAn =1 changes SDAAn from 0 to 1). The slave device detects the generated stop condition and slave device issue an interrupt (INTIICAn: stop condition).

**Remarks 1.** <1> to <19> in Figure 12-33 represent the entire procedure for communicating data using the I<sup>2</sup>C bus.

Figure 12-33 (1) Start condition ~ address ~ data shows the processing from <1> to <7>, Figure 12-33 (2) Address ~ data ~ data shows the processing from <3> to <12>, and Figure 12-33 (3) Data ~ data ~ stop condition shows the processing from <8> to <19>.

**2.** n = 0, 1

## CHAPTER 13 DMA CONTROLLER

The R7F0C010 have an internal DMA (Direct Memory Access) controller.

Data can be automatically transferred between the peripheral hardware supporting DMA, SFRs, and internal RAM without via CPU.

As a result, the normal internal operation of the CPU and data transfer can be executed in parallel with transfer between the SFR and internal RAM, and therefore, a large capacity of data can be processed. In addition, real-time control using communication, timer, and A/D can also be realized.

### 13.1 Functions of DMA Controller

- Number of DMA channels: 2 channels
- Transfer unit: 8 or 16 bits
- Maximum transfer unit: 1024 times
- Transfer type: 2-cycle transfer (One transfer is processed in 2 clocks and the CPU stops during that processing.)
- Transfer mode: Single-transfer mode
- Transfer request: Selectable from the following peripheral hardware interrupts
  - A/D converter
  - Serial interface (CSI00, UART0)
  - Timer (channel 0, 1, 2, or 3)
- Transfer target: Between SFR and internal RAM

Here are examples of functions using DMA.

- Successive transfer of serial interface
- Batch transfer of analog data
- Capturing A/D conversion result at fixed interval
- Capturing port value at fixed interval

## 13.2 Configuration of DMA Controller

The DMA controller includes the following hardware.

**Table 13-1. Configuration of DMA Controller**

Item	Configuration
Address registers	<ul style="list-style-type: none"> <li>• DMA SFR address registers 0, 1 (DSA0, DSA1)</li> <li>• DMA RAM address registers 0, 1 (DRA0, DRA1)</li> </ul>
Count register	<ul style="list-style-type: none"> <li>• DMA byte count registers 0, 1 (DBC0, DBC1)</li> </ul>
Control registers	<ul style="list-style-type: none"> <li>• DMA mode control registers 0, 1 (DMC0, DMC1)</li> <li>• DMA operation control register 0, 1 (DRC0, DRC1)</li> </ul>

### 13.2.1 DMA SFR address register n (DSAn)

This is an 8-bit register that is used to set an SFR address that is the transfer source or destination of DMA channel n. Set the lower 8 bits of the SFR addresses FFF00H to FFFFH.

This register is not automatically incremented but fixed to a specific value.

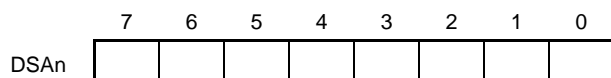
In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

The DSAn register can be read or written in 8-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 00H.

**Figure 13-1. Format of DMA SFR Address Register n (DSAn)**

Address: FFFB0H (DSA0), FFFB1H (DSA1) After reset: 00H R/W



**Remark** n: DMA channel number (n = 0, 1)

**13.2.2 DMA RAM address register n (DRAn)**

This is a 16-bit register that is used to set a RAM address that is the transfer source or destination of DMA channel n.

Addresses of the internal RAM area other than the general-purpose registers (see **Table 13-2**) can be set to this register.

Set the lower 16 bits of the RAM address.

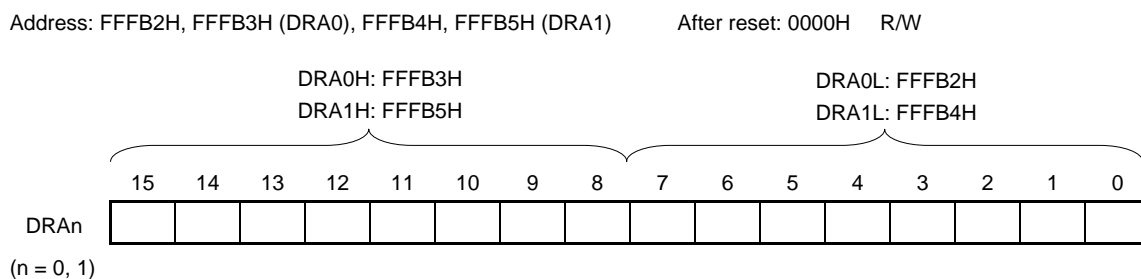
This register is automatically incremented when DMA transfer has been started. It is incremented by +1 in the 8-bit transfer mode and by +2 in the 16-bit transfer mode. DMA transfer is started from the address set to this DRAn register. When the data of the last address has been transferred, the DRAn register stops with the value of the last address +1 in the 8-bit transfer mode, and the last address +2 in the 16-bit transfer mode.

In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

The DRAn register can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 0000H.

**Figure 13-2. Format of DMA RAM Address Register n (DRAn)**



**Table 13-2. Internal RAM Area Other than the General-purpose Registers**

Part Number	Internal RAM Area Other than the General-purpose Registers
R7F0C010	FF900H to FFEDFH

**Remark** n: DMA channel number (n = 0, 1)

### 13.2.3 DMA byte count register n (DBCn)

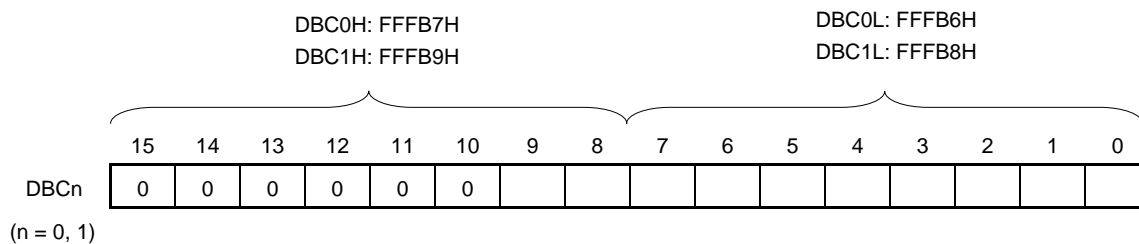
This is a 10-bit register that is used to set the number of times DMA channel n executes transfer. Be sure to set the number of times of transfer to this DBCn register before executing DMA transfer (up to 1024 times).

Each time DMA transfer has been executed, this register is automatically decremented. By reading this DBCn register during DMA transfer, the remaining number of times of transfer can be learned.

The DBCn register can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer. Reset signal generation clears this register to 0000H.

**Figure 13-3. Format of DMA Byte Count Register n (DBCn)**

Address: FFFB6H, FFFB7H (DBC0), FFFB8H, FFFB9H (DBC1) After reset: 0000H R/W



DBCn[9:0]	Number of times of transfer (when DBCn is written)	Remaining number of times of transfer (when DBCn is read)
000H	1024	Completion of transfer or waiting for 1024 times of DMA transfer
001H	1	Waiting for remaining one time of DMA transfer
002H	2	Waiting for remaining two times of DMA transfer
003H	3	Waiting for remaining three times of DMA transfer
• • •	• • •	• • •
3FEH	1022	Waiting for remaining 1022 times of DMA transfer
3FFH	1023	Waiting for remaining 1023 times of DMA transfer

- Cautions**
1. Be sure to clear bits 15 to 10 to “0”.
  2. If the general-purpose register is specified or the internal RAM space is exceeded as a result of continuous transfer, the general-purpose register or SFR space are written or read, resulting in loss of data in these spaces. Be sure to set the number of times of transfer that is within the internal RAM space.

**Remark** n: DMA channel number (n = 0, 1)



### 13.3 Registers Controlling DMA Controller

DMA controller is controlled by the following registers.

- DMA mode control register n (DMCn)
- DMA operation control register n (DRCn)

**Remark** n: DMA channel number (n = 0, 1)

### 13.3.1 DMA mode control register n (DMCn)

The DMCn register is a register that is used to set a transfer mode of DMA channel n. It is used to select a transfer direction, data size, setting of pending, and start source. Bit 7 (STGn) is a software trigger that starts DMA.

Rewriting bits 6, 5, and 3 to 0 of the DMCn register is prohibited during operation (when DSTn = 1).

The DMCn register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 13-4. Format of DMA Mode Control Register n (DMCn) (1/2)**

Address: FFFBAH (DMC0), FFFBBH (DMC1) After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	3	2	1	0
DMCn	STGn	DRSn	DSn	DWAITn	0	IFCn2	IFCn1	IFCn0
STGn <sup>Note 1</sup>	DMA transfer start software trigger							
0	No trigger operation							
1	DMA transfer is started when DMA operation is enabled (DENn = 1).							
DMA transfer is performed once by writing 1 to the STGn bit when DMA operation is enabled (DENn = 1). When this bit is read, 0 is always read.								
DRSn	Selection of DMA transfer direction							
0	SFR to internal RAM							
1	Internal RAM to SFR							
DSn	Specification of transfer data size for DMA transfer							
0	8 bits							
1	16 bits							
DWAITn <sup>Note 2</sup>	Pending of DMA transfer							
0	Executes DMA transfer upon DMA start request (not held pending).							
1	Holds DMA start request pending if any.							
DMA transfer that has been held pending can be started by clearing the value of the DWAITn bit to 0. It takes 2 clocks to actually hold DMA transfer pending when the value of the DWAITn bit is set to 1.								

- Notes**
1. The software trigger (STGn) can be used regardless of the IFCn2 to IFCn0 bits values.
  2. When DMA transfer is held pending while using two or more DMA channels, be sure to hold the DMA transfer pending for all channels (by setting the DWAIT0 and DWAIT1 bits to 1).

**Remark** n: DMA channel number (n = 0, 1)

Figure 13-4. Format of DMA Mode Control Register n (DMCn) (2/2)

Address: FFFBAH (DMC0), FFFBBH (DMC1) After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	3	2	1	0
DMCn	STGn	DRSn	DSn	DWAITn	0	IFCn2	IFCn1	IFCn0

IFCn2	IFCn1	IFCn0	Selection of DMA start source <sup>Note</sup>	
			Trigger signal	Trigger contents
0	0	0	–	Disables DMA transfer by interrupt. (Only software trigger is enabled.)
0	0	1	INTAD	A/D conversion end interrupt
0	1	0	INTTM00	End of timer channel 00 count or capture end interrupt
0	1	1	INTTM01	End of timer channel 01 count or capture end interrupt
1	0	0	INTTM02	End of timer channel 02 count or capture end interrupt
1	0	1	INTTM03	End of timer channel 03 count or capture end interrupt
1	1	0	INTSR0/INTCSI01	UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt
1	1	1	INTST0	UART0 reception transfer end interrupt
Other than above			Setting prohibited	

**Note** The software trigger (STGn) can be used regardless of the IFCn2 to IFCn0 bits values.

**Remark** n: DMA channel number (n = 0, 1)

### 13.3.2 DMA operation control register n (DRCn)

The DRCn register is a register that is used to enable or disable transfer of DMA channel n.

Rewriting bit 7 (DENn) of this register is prohibited during operation (when DSTn = 1).

The DRCn register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 13-5. Format of DMA Operation Control Register n (DRCn)**

Address: FFFBCH (DRC0), FFFBDH (DRC1) After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	<0>
DRCn	DENn	0	0	0	0	0	0	DSTn

DENn	DMA operation enable flag
0	Disables operation of DMA channel n (stops operating clock of DMA).
1	Enables operation of DMA channel n.
DMAC waits for a DMA trigger when DSTn = 1 after DMA operation is enabled (DENn = 1).	

DSTn	DMA transfer mode flag
0	DMA transfer of DMA channel n is completed.
1	DMA transfer of DMA channel n is not completed (still under execution).
DMAC waits for a DMA trigger when DSTn = 1 after DMA operation is enabled (DENn = 1). When a software trigger (STGn) or the start source trigger set by the IFCn2 to IFCn0 bits is input, DMA transfer is started. When DMA transfer is completed after that, this bit is automatically cleared to 0. Write 0 to this bit to forcibly terminate DMA transfer under execution.	

**Caution** The DSTn flag is automatically cleared to 0 when a DMA transfer is completed.

Writing the DENn flag is enabled only when DSTn = 0. When a DMA transfer is terminated without waiting for generation of the interrupt (INTDMA<sub>n</sub>) of DMA<sub>n</sub>, therefore, set the DSTn bit to 0 and then the DENn bit to 0 (for details, see 13.5.5 Forced termination by software).

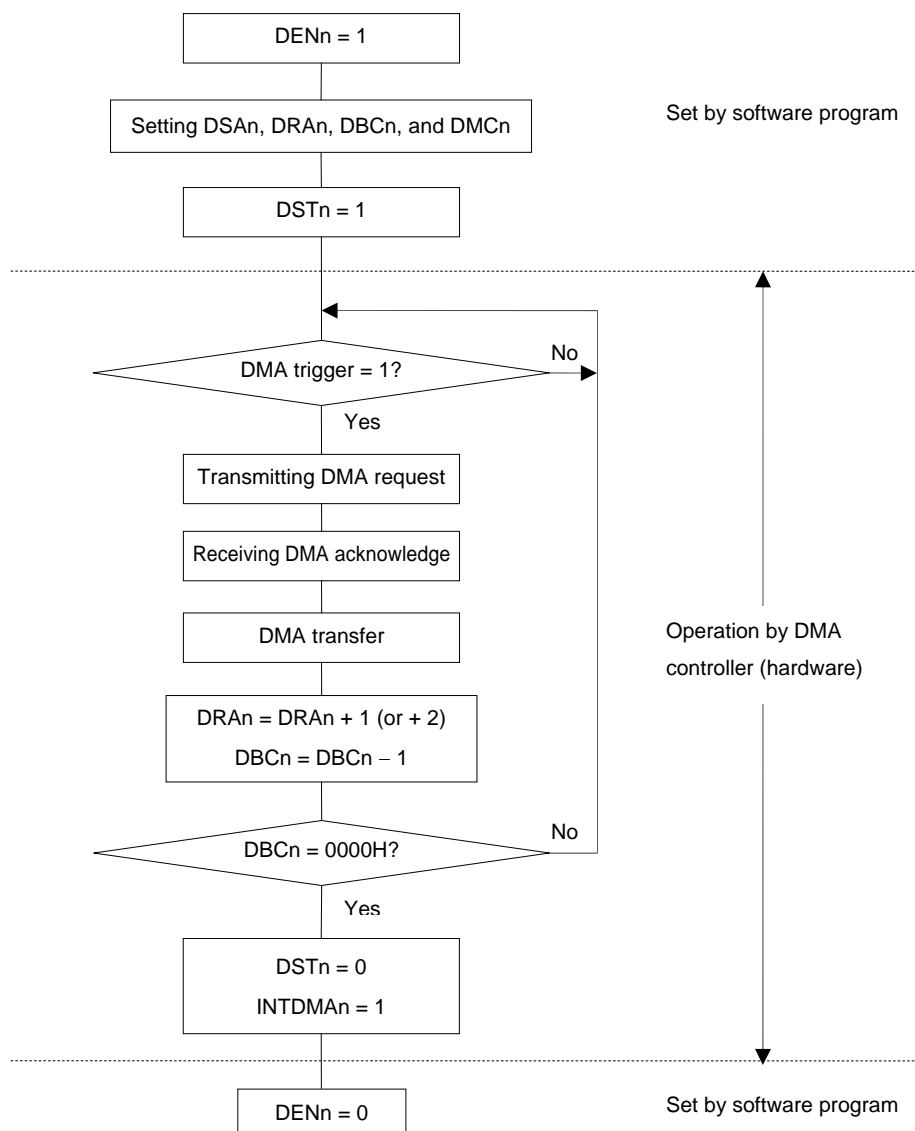
**Remark** n: DMA channel number (n = 0, 1)

## 13.4 Operation of DMA Controller

### 13.4.1 Operation procedure

- <1> The DMA controller is enabled to operate when  $DEN_n = 1$ . Before writing the other registers, be sure to set the  $DEN_n$  bit to 1. Use 80H to write with an 8-bit manipulation instruction.
- <2> Set an SFR address, a RAM address, the number of times of transfer, and a transfer mode of DMA transfer to DMA SFR address register n ( $DSAn$ ), DMA RAM address register n ( $DRAn$ ), DMA byte count register n ( $DBCn$ ), and DMA mode control register n ( $DMCn$ ).
- <3> The DMA controller waits for a DMA trigger when  $DST_n = 1$ . Use 81H to write with an 8-bit manipulation instruction.
- <4> When a software trigger ( $STG_n$ ) or a start source trigger specified by the  $IFCn2$  to  $IFCn0$  bits is input, a DMA transfer is started.
- <5> Transfer is completed when the number of times of transfer set by the  $DBCn$  register reaches 0, and transfer is automatically terminated by occurrence of an interrupt ( $INTDMA_n$ ).
- <6> Stop the operation of the DMA controller by clearing the  $DEN_n$  bit to 0 when the DMA controller is not used.

Figure 13-6. Operation Procedure



**Remark** n: DMA channel number (n = 0, 1)

### 13.4.2 Transfer mode

The following four modes can be selected for DMA transfer by using bits 6 and 5 (DRSn and DS<sub>n</sub>) of DMA mode control register n (DMCn).

DRSn	DS <sub>n</sub>	DMA Transfer Mode
0	0	Transfer from SFR of 1-byte data (fixed address) to RAM (address is incremented by +1)
0	1	Transfer from SFR of 2-byte data (fixed address) to RAM (address is incremented by +2)
1	0	Transfer from RAM of 1-byte data (address is incremented by +1) to SFR (fixed address)
1	1	Transfer from RAM of 2-byte data (address is incremented by +2) to SFR (fixed address)

By using these transfer modes, up to 1024 bytes of data can be consecutively transferred by using the serial interface, data resulting from A/D conversion can be consecutively transferred, and port data can be scanned at fixed time intervals by using a timer.

### 13.4.3 Termination of DMA transfer

When DBC<sub>n</sub> = 00H and DMA transfer is completed, the DST<sub>n</sub> bit is automatically cleared to 0. An interrupt request (INTDMA<sub>n</sub>) is generated and transfer is terminated.

When the DST<sub>n</sub> bit is cleared to 0 to forcibly terminate DMA transfer, DMA byte count register n (DBC<sub>n</sub>) and DMA RAM address register n (DRAn) hold the value when transfer is terminated.

The interrupt request (INTDMA<sub>n</sub>) is not generated if transfer is forcibly terminated.

**Remark** n: DMA channel number (n = 0, 1)

## 13.5 Example of Setting of DMA Controller

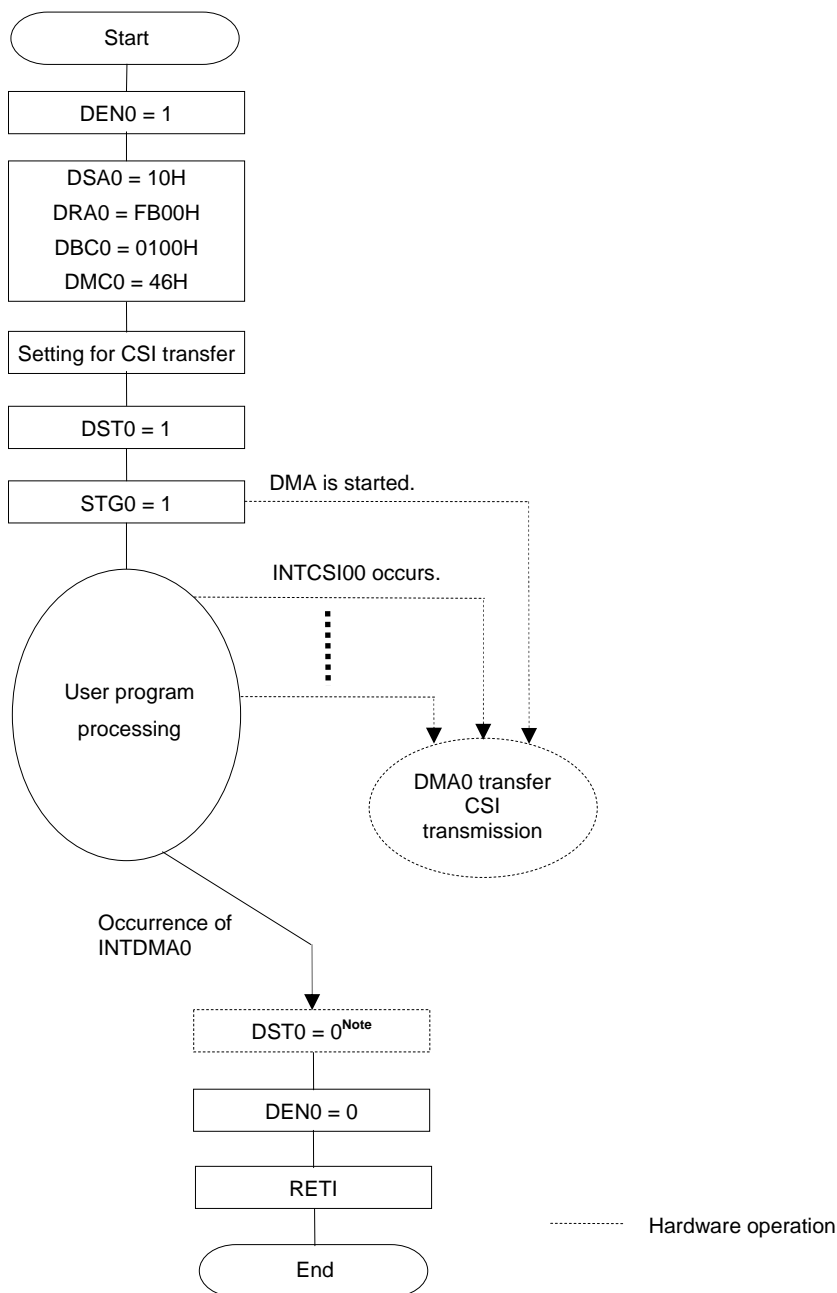
### 13.5.1 CSI consecutive transmission

A flowchart showing an example of setting for CSI consecutive transmission is shown below.

- Consecutive transmission of CSI00 (256 bytes)
- DMA channel 0 is used for DMA transfer.
- DMA start source: INTCSI00 (software trigger (STG0) only for the first start source)
- Interrupt of CSI00 is specified by IFC02 to IFC00 = 0110B.
- Transfers FFB00H to FFBFFH (256 bytes) of RAM to FFF10H of the data register (SIO00) of CSI.

**Remark** IFC02 to IFC00: Bits 2 to 0 of DMA mode control registers 0 (DMC0)

Figure 13-7. Example of Setting for CSI Consecutive Transmission



**Note** The DST0 flag is automatically cleared to 0 when a DMA transfer is completed. Writing the DEN0 flag is enabled only when DST0 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA0 (INTDMA0), set the DST0 bit to 0 and then the DEN0 bit to 0 (for details, see 13.5.5 Forced termination by software).

The first trigger for consecutive transmission is not started by the interrupt of CSI. In this example, it starts by a software trigger.

CSI transmission of the second time and onward is automatically executed.

A DMA interrupt (INTDMA0) occurs when the last transmit data has been written to the data register.



### 13.5.2 Consecutive capturing of A/D conversion results

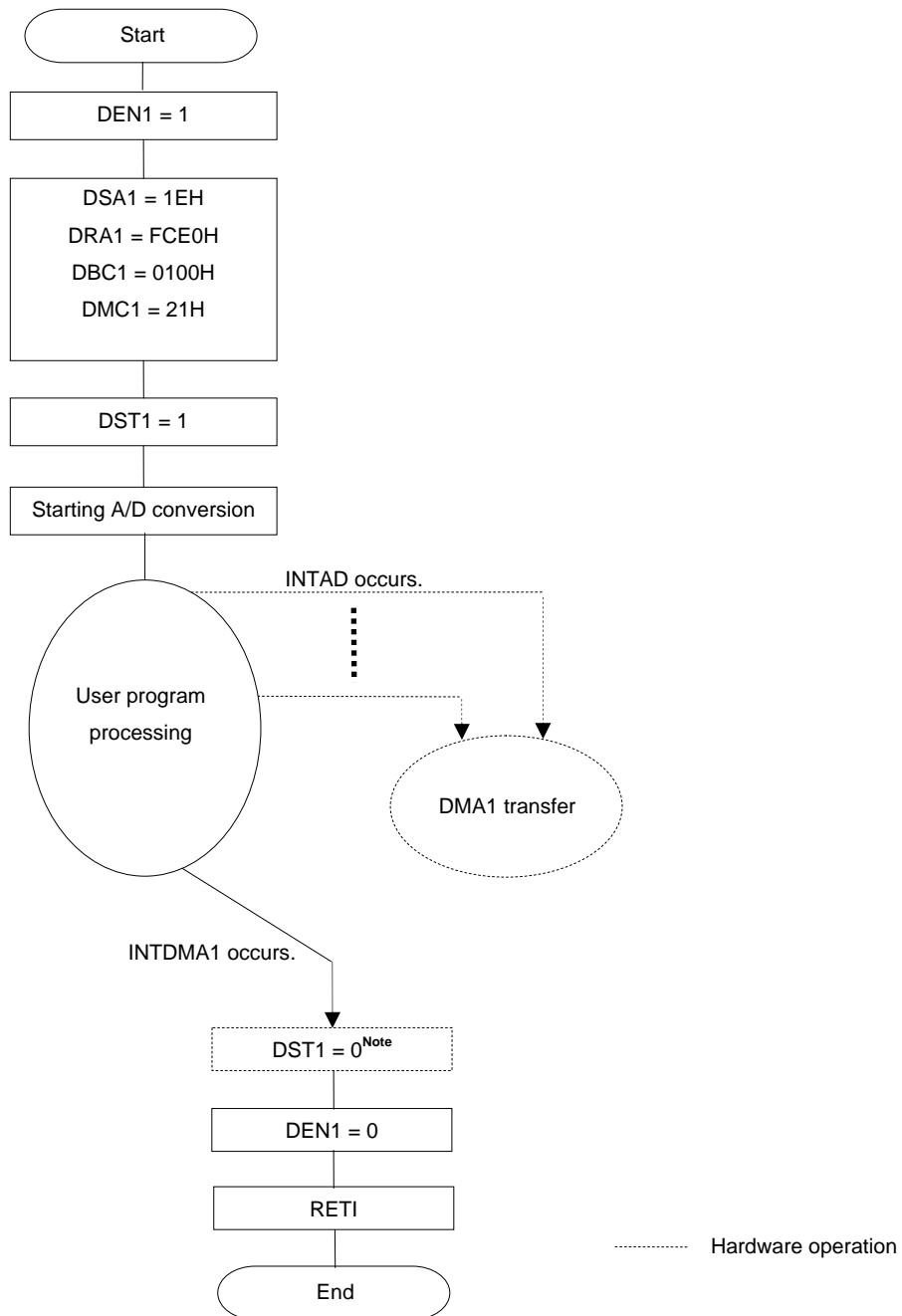
A flowchart of an example of setting for consecutively capturing A/D conversion results is shown below.

- Consecutive capturing of A/D conversion results.
- DMA channel 1 is used for DMA transfer.
- DMA start source: INTAD
- Interrupt of A/D is specified by IFC12 to IFC10 = 0001B.
- Transfers FFF1EH and FFF1FH (2 bytes) of the 12-bit A/D conversion result register (ADCR) to 512 bytes of FFCE0H to FFEDFH of RAM.

<R>

**Remark** IFC12 to IFC10: Bits 2 to 0 of DMA mode control registers 1 (DMC1)

Figure 13-8. Example of Setting of Consecutively Capturing A/D Conversion Results



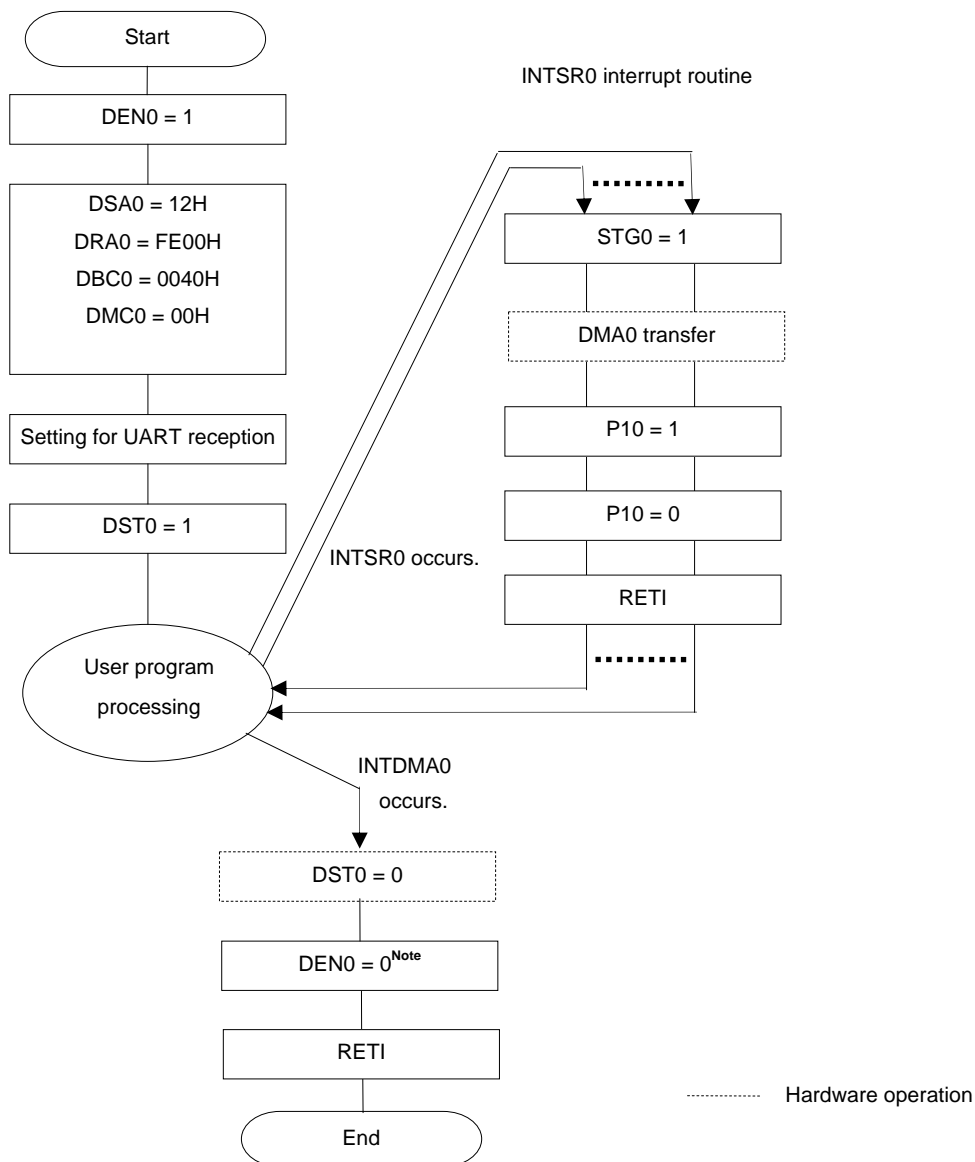
**Note** The DST1 flag is automatically cleared to 0 when a DMA transfer is completed. Writing the DEN1 flag is enabled only when DST1 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA1 (INTDMA1), set the DST1 bit to 0 and then the DEN1 bit to 0 (for details, see 13.5.5 Forced termination by software).

### 13.5.3 UART consecutive reception + ACK transmission

A flowchart illustrating an example of setting for UART consecutive reception + ACK transmission is shown below.

- Consecutively receives data from UART0 and outputs ACK to P10 on completion of reception.
- DMA channel 0 is used for DMA transfer.
- DMA start source: Software trigger (DMA transfer on occurrence of an interrupt is disabled.)
- Transfers FFF12H of UART receive data register 0 (RXD0) to 64 bytes of FFE00H to FFE3FH of RAM.

Figure 13-9. Example of Setting for UART Consecutive Reception + ACK Transmission



**Note** The DST0 flag is automatically cleared to 0 when a DMA transfer is completed. Writing the DEN0 flag is enabled only when DST0 = 0. To terminate a DMA transfer without waiting for occurrence of the interrupt of DMA0 (INTDMA0), set the DST0 bit to 0 and then the DEN0 bit to 0 (for details, see 13.5.5 Forced termination by software).

**Remark** This is an example where a software trigger is used as a DMA start source. If ACK is not transmitted and if only data is consecutively received from UART, the UART reception end interrupt (INTSR0) can be used to start DMA for data reception.

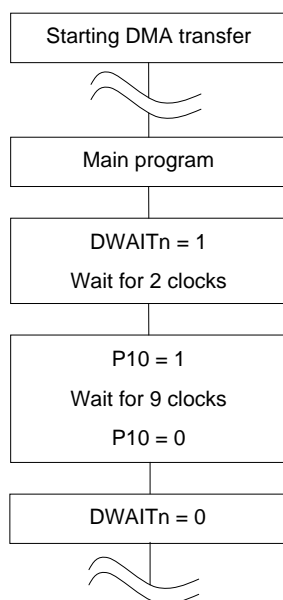
### 13.5.4 Holding DMA transfer pending by DWAITn bit

When DMA transfer is started, transfer is performed while an instruction is executed. At this time, the operation of the CPU is stopped and delayed for the duration of 2 clocks. If this poses a problem to the operation of the set system, a DMA transfer can be held pending by setting the DWAITn bit to 1. The DMA transfer for a transfer trigger that occurred while DMA transfer was held pending is executed after the pending status is canceled. However, because only one transfer trigger can be held pending for each channel, even if multiple transfer triggers occur for one channel during the pending status, only one DMA transfer is executed after the pending status is canceled.

To output a pulse with a width of 10 clocks of the operating frequency from the P10 pin, for example, the clock width increases to 12 if a DMA transfer is started midway. In this case, the DMA transfer can be held pending by setting the DWAITn bit to 1.

After setting the DWAITn bit to 1, it takes two clocks until a DMA transfer is held pending.

Figure 13-10. Example of Setting for Holding DMA Transfer Pending by DWAITn Bit



**Caution** When DMA transfer is held pending while using two DMA channels, be sure to held the DMA transfer pending for all channels (by setting DWAIT0 and DWAIT1 to 1). If the DMA transfer of one channel is executed while that of the other channel is held pending, DMA transfer might not be held pending for the latter channel.

- Remarks**
1. n: DMA channel number (n = 0, 1)
  2. 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

### 13.5.5 Forced termination by software

After the DSTn bit is set to 0 by software, it takes up to 2 clocks until a DMA transfer is actually stopped and the DSTn bit is set to 0. To forcibly terminate a DMA transfer by software without waiting for occurrence of the interrupt (INTDMA<sub>n</sub>) of DMA<sub>n</sub>, therefore, perform either of the following processes.

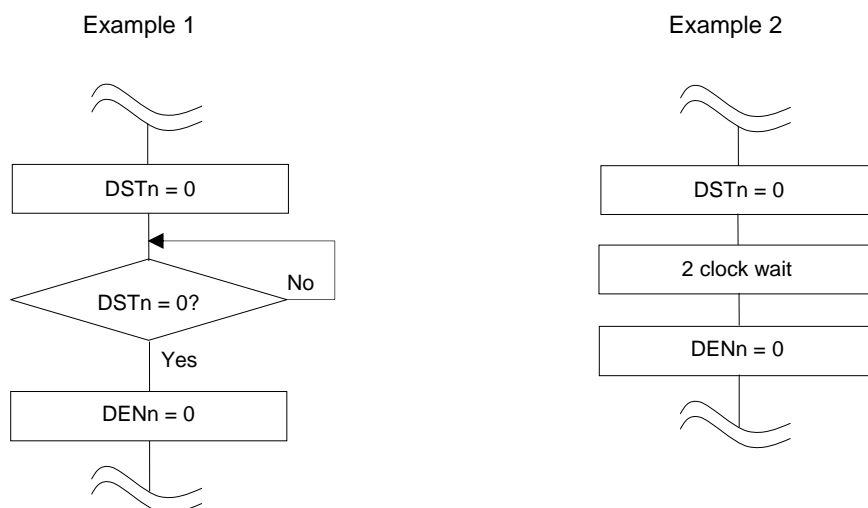
<When using one DMA channel>

- Set the DSTn bit to 0 (use DRCn = 80H to write with an 8-bit manipulation instruction) by software, confirm by polling that the DSTn bit has actually been cleared to 0, and then set the DENn bit to 0 (use DRCn = 00H to write with an 8-bit manipulation instruction).
- Set the DSTn bit to 0 (use DRCn = 80H to write with an 8-bit manipulation instruction) by software and then set the DENn bit to 0 (use DRCn = 00H to write with an 8-bit manipulation instruction) two or more clocks after.

<When using two DMA channels>

- To forcibly terminate DMA transfer by software when using two or more DMA channels (by setting DSTn to 0), clear the DSTn bit to 0 after the DMA transfer is held pending by setting the DWAITn bits of all using channels to 1. Next, clear the DWAITn bits of all using channels to 0 to cancel the pending status, and then clear the DENn bit to 0.

**Figure 13-11. Forced Termination of DMA Transfer (1/2)**



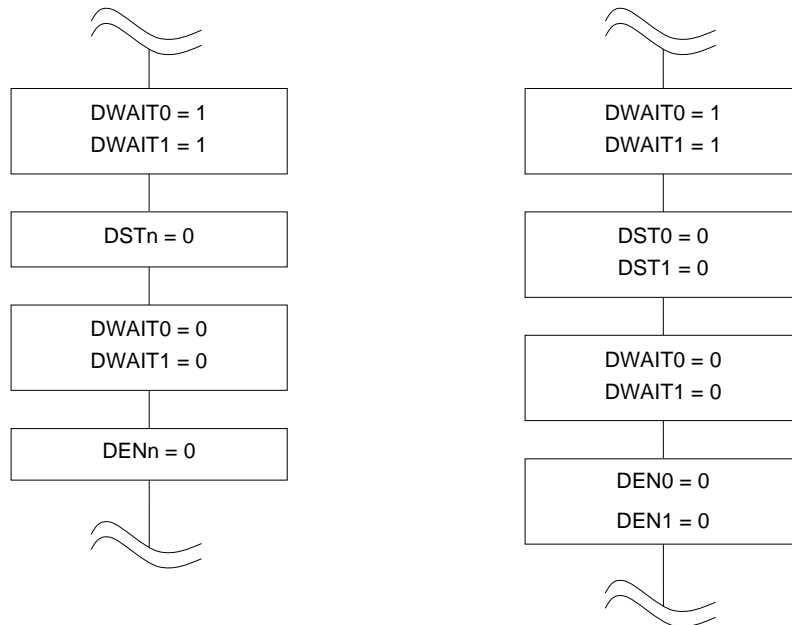
- Remarks**
1. n: DMA channel number (n = 0, 1)
  2. 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

Figure 13-11. Forced Termination of DMA Transfer (2/2)

## Example 3

- Procedure for forcibly terminating the DMA transfer for one channel if both channels are used

- Procedure for forcibly terminating the DMA transfer for both channels if both channels are used



**Caution** In example 3, the system is not required to wait two clock cycles after the  $DWAITn$  bit is set to 1. In addition, the system does not have to wait two clock cycles after clearing the  $DSTn$  bit to 0, because more than two clock cycles elapse from when the  $DSTn$  bit is cleared to 0 to when the  $DENn$  bit is cleared to 0.

- Remarks**
1.  $n$ : DMA channel number ( $n = 0, 1$ )
  2. 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

### 13.6 Cautions on Using DMA Controller

#### (1) Priority of DMA

During DMA transfer, a request from the other DMA channel is held pending even if generated. The pending DMA transfer is started after the ongoing DMA transfer is completed. If two or more DMA requests are generated at the same time, however, their priority are DMA channel 0 > DMA channel 1.

If a DMA request and an interrupt request are generated at the same time, the DMA transfer takes precedence, and then interrupt servicing is executed.

#### (2) DMA response time

The response time of DMA transfer is as follows.

**Table 13-3. Response Time of DMA Transfer**

	Minimum Time	Maximum Time
Response time	3 clocks	10 clocks <sup>Note</sup>

**Note** The maximum time necessary to execute an instruction from internal RAM is 16 clock cycles.

**Cautions 1.** The above response time does not include the two clock cycles required for a DMA transfer.

**2.** When executing a DMA pending instruction (see 13.6 (4)), the maximum response time is extended by the execution time of that instruction to be held pending.

**3.** Do not specify successive transfer triggers for a channel within a period equal to the maximum response time plus one clock cycle, because they might be ignored.

**Remark** 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

#### (3) Operation in standby mode

The DMA controller operates as follows in the standby mode.

**Table 13-4. DMA Operation in Standby Mode**

Status	DMA Operation
HALT mode	Normal operation
STOP mode	Stops operation. If DMA transfer and STOP instruction execution contend, DMA transfer may be damaged. Therefore, stop DMA before executing the STOP instruction.



**(4) DMA pending instruction**

Even if a DMA request is generated, DMA transfer is held pending immediately after the following instructions.

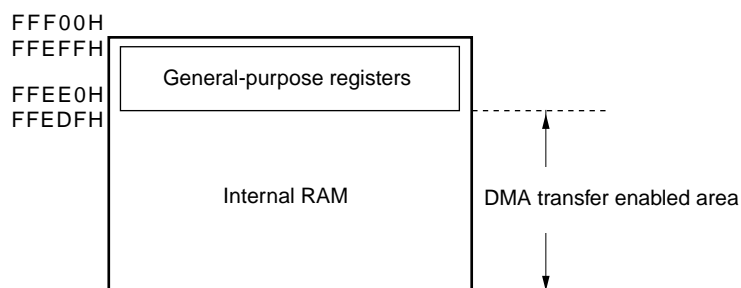
- CALL !addr16
- CALL \$!addr20
- CALL !!addr20
- CALL rp
- CALLT [addr5]
- BRK
- Bit manipulation instructions for registers IF0L, IF0H, IF1L, MK0L, MK0H, MK1L, PR00L, PR00H, PR01L, PR10L, PR10H, PR11L, and PSW each.
- Instruction for accessing the data flash memory

**(5) Operation if address in general-purpose register area or other than those of internal RAM area is specified**

The address indicated by DMA RAM address register n (DRAn) is incremented during DMA transfer. If the address is incremented to an address in the general-purpose register area or exceeds the area of the internal RAM, the following operation is performed.

- In mode of transfer from SFR to RAM  
The data of that address is lost.
- In mode of transfer from RAM to SFR  
Undefined data is transferred to SFR.

In either case, malfunctioning may occur or damage may be done to the system. Therefore, make sure that the address is within the internal RAM area other than the general-purpose register area.

**(6) Operation if instructions for accessing the data flash area**

- Because DMA transfer is suspended to access to the data flash area, be sure to add the DMA pending instruction.

If the data flash area is accessed after an next instruction execution from start of DMA transfer, a 3-clock wait will be inserted to the next instruction.

Instruction 1

DMA transfer

Instruction 2 ← The wait of three clock cycles occurs.

MOV A, !DataFlash area

## CHAPTER 14 EVENT LINK CONTROLLER (ELC)

### <R> 14.1 Functions of ELC

The event link controller (ELC) mutually connects (links) events output from each peripheral function. By linking events, it becomes possible to coordinate operation between peripheral functions directly without going through the CPU.

The ELC has the following functions.

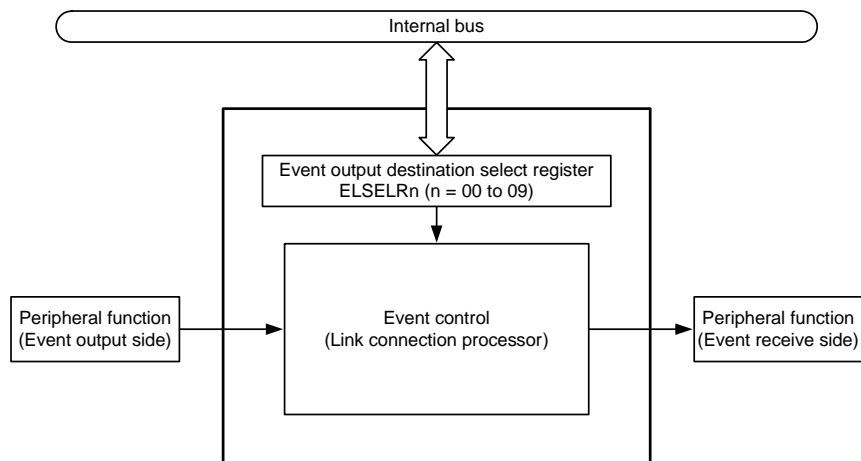
- Capable of directly linking event signals from 10 types of peripheral functions to specified peripheral functions
- Event signals can be used as activation sources for operating any one of three types of peripheral functions

### 14.2 Configuration of ELC

Figure 14-1 shows the Event Link Controller Block Diagram.

<R>

Figure 14-1. ELC Block Diagram



**<R> 14.3 Registers Controlling ELC**

Table 14-1 lists the Registers Controlling ELC.

**Table 14-1. Registers Controlling ELC**

Item	Configuration
Control register	Event Output Destination Select Register n (ELSELRn)

**14.3.1 Event output destination select register n (ELSELRn) (n = 00 to 09)**

An ELSELRn register links each event signal to an operation of an event-receiving peripheral function (link destination peripheral function) after reception.

Do not set multiple event inputs to the same event output destination (event receive side). The operation of the event-receiving peripheral function will become undefined, and event signals may not be received correctly. In addition, do not set the event link generation source and the event link output destination to the same function.

Set an ELSELRn register during a period when no event output peripheral functions are generating event signals.

Table 14-2 lists the correspondence between ELSELRn (n = 00 to 09) registers and peripheral functions, and Table 14-3 lists the correspondence between values set to ELSELRn (n = 00 to 09) registers and operation of link destination peripheral functions at reception.

**<R> Figure 14-2. Format of Event Output Destination Select Register n (ELSELRn)**

Address: F0300H (ELSELR00) to F0309H (ELSELR09) After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ELSELRn	0	0	0	0	0	0	ELSELRn1	ELSELRn0

ELSELRn1	ELSELRn0	Event Link Selection
0	0	Event link disabled
0	1	Select operation of peripheral function to link <sup>Note</sup>
1	0	Select operation of peripheral function to link <sup>Note</sup>
1	1	Select operation of peripheral function to link <sup>Note</sup>

**Note** See Table 14-3 Correspondence Between Values Set to ELSELRn (n = 00 to 09) Registers and Operation of Link Destination Peripheral Functions at Reception.

**Table 14-2. Correspondence Between ELSELRn (n = 00 to 09) Registers and Peripheral Functions**

Register Name	Event Generator (Output Origin of Event Input n)	Event Description
ELSELR00	External interrupt edge detection 0	INTP0
ELSELR01	External interrupt edge detection 1	INTP1
ELSELR02	External interrupt edge detection 2	INTP2
ELSELR03	External interrupt edge detection 3	INTP3
ELSELR04	External interrupt edge detection 4	INTP4
ELSELR05	External interrupt edge detection 5	INTP5
ELSELR06	TAU channel 00 Count end/Capture end	INTTM00
ELSELR07	TAU channel 01 Count end/Capture end	INTTM01
ELSELR08	TAU channel 02 Count end/Capture end	INTTM02
ELSELR09	TAU channel 03 Count end/Capture end	INTTM03

**Table 14-3. Correspondence Between Values Set to ELSELRn (n = 00 to 09) Registers and Operation of Link Destination Peripheral Functions at Reception**

Bits ELSELR1 and ELSELR0 in ELSELRn Register	Link Destination Peripheral Function	Operation When Receiving Event
0000B	None	Event link stops
0001B	A/D converter	A/D conversion starts
0010B	DA0	Real-time output
0011B	DA1	Real-time output

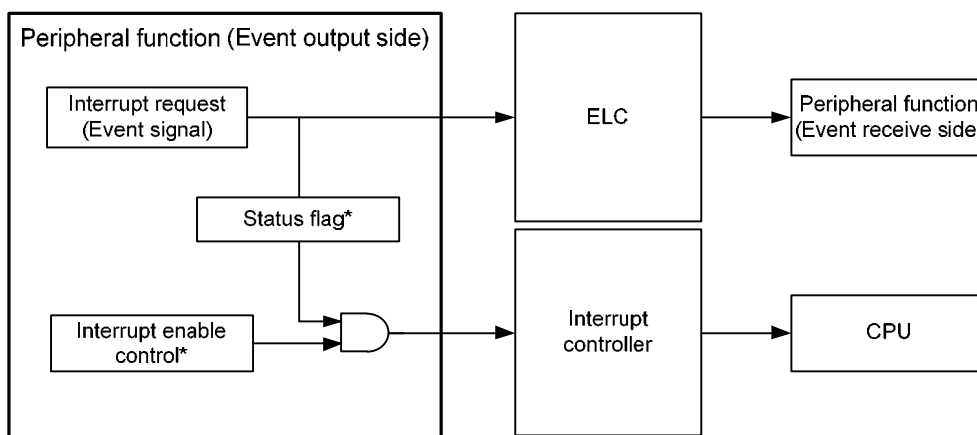
<R> **14.4 Operation**

The path for using an event signal generated by a peripheral function as an interrupt request to the interrupt control circuit is independent from the path for using it as an ELC event. Therefore, each event signal can be used as an event signal for operation of an event-receiving peripheral function, regardless of interrupt control.

<R> Figure 14-2 shows the relationship between interrupt handling and ELC. The figure show an example of an interrupt request status flag and a peripheral function possessing the enable bits that control enabling/disabling of such interrupts.

A peripheral function which receives an event from the ELC will perform the operation corresponding to the event-receiving peripheral function after reception of an event (see **Table14-3 Correspondence Between Values Set to ELSELRn (n = 00 to 09) Registers and Operation of Link Destination Peripheral Functions at Reception**).

**Figure 14-3. Relationship Between Interrupt Handling and ELC**



<R>

<R> **Note** Not available depending on the peripheral function.

Table 14 - 4 lists the Response of Peripheral Functions That Receive Events.

**Table 14-4. Response of Peripheral Functions That Receive Events**

Event Receiver No.	Event Link Destination Function	Operation after Event Reception	Response
1	A/D converter	A/D conversion	An event from the ELC is directly used as a hardware trigger of A/D conversion.
2	D/A converter of channel 0	Real time output (channel 0)	D/A conversion of channel 0 is started after 2 or 3 cycles of $f_{CLK}$ after an ELC event is generated.
3	D/A converter of channel 1	Real time output (channel 1)	D/A conversion of channel 0 is started after 2 or 3 cycles of $f_{CLK}$ after an ELC event is generated.

<R>

## CHAPTER 15 INTERRUPT FUNCTIONS

<R> The interrupt function switches the program execution to other processing. When the branch processing is finished, the program returns to the interrupted processing.

### 15.1 Interrupt Function Types

The following two types of interrupt functions are used.

#### (1) Maskable interrupts

These interrupts undergo mask control. Maskable interrupts can be divided into two priority groups by setting the priority specification flag registers (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L).

Multiple interrupt servicing can be applied to low-priority interrupts when high-priority interrupts are generated. If two or more interrupt requests, each having the same priority, are simultaneously generated, then they are processed according to the default priority of vectored interrupt servicing. Default priority, see **Table 15-1**.

A standby release signal is generated and STOP, HALT, and SNOOZE modes are released.

External interrupt requests and internal interrupt requests are provided as maskable interrupts.

External: 6, Internal: 12

#### (2) Software interrupt

This is a vectored interrupt generated by executing the BRK instruction. It is acknowledged even when interrupts are disabled. The software interrupt does not undergo interrupt priority control.

### 15.2 Interrupt Sources and Configuration

Interrupt sources include maskable interrupts and software interrupts. In addition, they also have up to seven reset sources (see **Table 15-1**). The vector codes that store the program start address when branching due to the generation of a reset or various interrupt requests are two bytes each, so interrupts jump to a 64 K address of 00000H to 0FFFFH.

Table 15-1. Interrupt Source List (1/2)

Interrupt Type	Default Priority <sup>Note 1</sup>	Interrupt Source		Internal/External	Vector Table Address	Basic Configuration Type <sup>Note 2</sup>
		Name	Trigger			
Maskable	0	INTWDTI	Watchdog timer interval <sup>Note 3</sup> (75% of overflow time+1/2f <sub>IL</sub> )	Internal	0004H	(A)
	1	INTLVI	Voltage detection <sup>Note 4</sup>		0006H	
	2	INTP0	Pin input edge detection	External	0008H	(B)
	3	INTP1			000AH	
	4	INTP2			000CH	
	5	INTP3			000EH	
	6	INTP4			0010H	
	7	INTP5			0012H	
	8	INTAD			End of A/D conversion	
	9	INTIICA0	End of IICA0 communication	0016H		
	10	INTFL	Reserved <sup>Note 5</sup>	0018H		
	11	INTDMA0	End of DMA0 transfer	001AH		
	12	INTDMA1	End of DMA1 transfer	001CH		
	13	INTST0/ INTCSI00	UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt	001EH		
	14	INTSR0	UART0 reception transfer end	0020H		
	15	INTSRE0	UART0 reception communication error occurrence	0022H		
		INTTM01H	End of timer channel 01 count or capture (at higher 8-bit timer operation)			
	16	INTTM03H	End of timer channel 03 count or capture (at higher 8-bit timer operation)	0028H		
	17	INTIICA1	End of IICA1 communication	002AH		
	18	INTTM00	End of timer channel 00 count or capture	002CH		
	19	INTTM01	End of timer channel 01 count or capture (at 16-bit/lower 8-bit timer operation)	002EH		
20	INTTM02	End of timer channel 02 count or capture	0030H			
21	INTTM03	End of timer channel 03 count or capture (at 16-bit/lower 8-bit timer operation)	0032H			

- Notes**
1. The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 21 indicates the lowest priority.
  2. Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 15-1.
  3. When bit 7 (WDTINT) of the option byte (000C0H) is set to 1.
  4. When bit 7 (LVIMD) of the voltage detection level register (LVIS) is cleared to 0.
  5. Be used at the flash self programming library.

Table 15-1. Interrupt Source List (2/2)

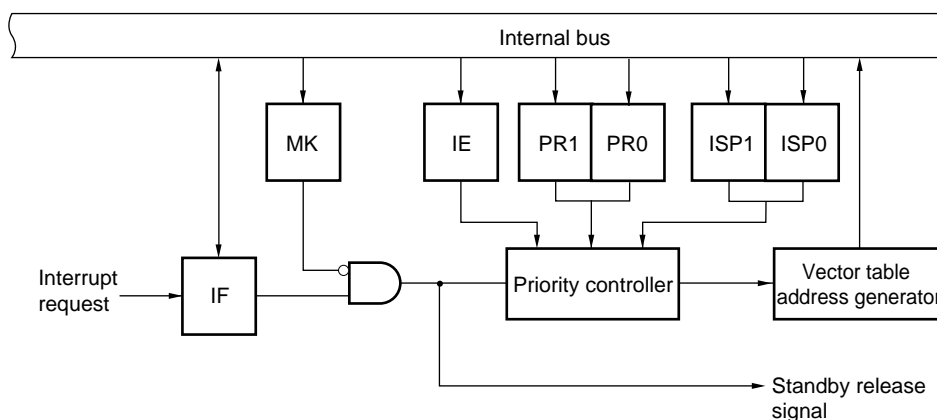
Interrupt Type	Default Priority <sup>Note 1</sup>	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuration Type <sup>Note 2</sup>
Software	–	BRK	Execution of BRK instruction	–	007EH	(C)
Reset	–	RESET	RESET pin input	–	0000H	–
		POR	Power-on-reset			
		LVD	Voltage detection <sup>Note 3</sup>			
		WDT	Overflow of watchdog timer			
		TRAP	Execution of illegal instruction <sup>Note 4</sup>			
		IAW	Illegal-memory access			
		RPE	RAM parity error			

- Notes**
- The default priority determines the sequence of interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 21 indicates the lowest priority.
  - Basic configuration types (A) to (C) correspond to (A) to (C) in Figure 15-1.
  - When bit 7 (LVIMD) of the voltage detection level register (LVIS) is set to 1.
  - When the instruction code in FFH is executed.  
Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

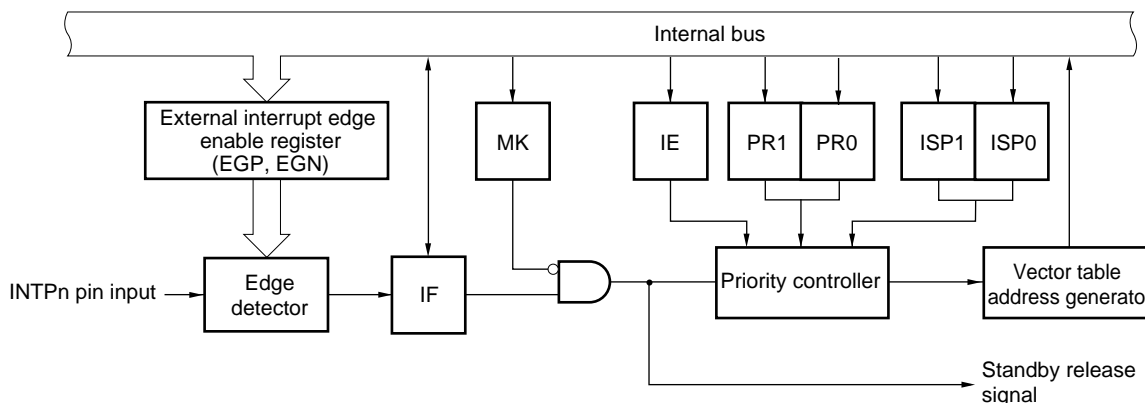


Figure 15-1. Basic Configuration of Interrupt Function

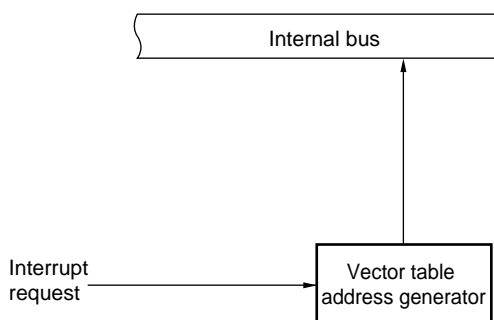
(A) Internal maskable interrupt



(B) External maskable interrupt (INTPn)



(C) Software interrupt



- IF: Interrupt request flag
- IE: Interrupt enable flag
- ISP0: In-service priority flag 0
- ISP1: In-service priority flag 1
- MK: Interrupt mask flag
- PR0: Priority specification flag 0
- PR1: Priority specification flag 1

### 15.3 Registers Controlling Interrupt Functions

The following 6 types of registers are used to control the interrupt functions.

- Interrupt request flag registers (IF0L, IF0H, IF1L)
- Interrupt mask flag registers (MK0L, MK0H, MK1L)
- Priority specification flag registers (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L)
- External interrupt rising edge enable registers (EGP0)
- External interrupt falling edge enable registers (EGN0)
- Program status word (PSW)

Table 15-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

**Table 15-2. Flags Corresponding to Interrupt Request Sources**

Interrupt Source	Interrupt Request Flag		Interrupt Mask Flag		Priority Specification Flag	
	Register	Register	Register	Register	Register	Register
INTWDTI	WDTIIF	IF0L	WDTIMK	MK0L	WDTIPR0, WDTIPR1	PR00L, PR10L
INTLVI	LVIIIF		LVIMK		LVIPR0, LVIPR1	
INTP0	PIF0		PMK0		PPR00, PPR10	
INTP1	PIF1		PMK1		PPR01, PPR11	
INTP2	PIF2		PMK2		PPR02, PPR12	
INTP3	PIF3		PMK3		PPR03, PPR13	
INTP4	PIF4		PMK4		PPR04, PPR14	
INTP5	PIF5		PMK5		PPR05, PPR15	
INTAD	ADIF	IF0H	ADMK	MK0H	ADPR0, ADPR1	PR00H, PR10H
INTIICA0	IICAIF0		IICAMK0		IICAPR00, IICAPR10	
INTFL	FLIF		FLMK		FLPR0, FLPR1	
INTDMA0	DMAIF0		DMAMK0		DMAPR00, DMAPR10	
INTDMA1	DMAIF1		DMAMK1		DMAPR01, DMAPR11	
INTST0 <sup>Note 1</sup>	STIF0 <sup>Note 1</sup>		STMK0 <sup>Note 1</sup>		STPR00, STPR10 <sup>Note 1</sup>	
INTCSI00 <sup>Note 1</sup>	CSIIF00 <sup>Note 1</sup>		CSIMK00 <sup>Note 1</sup>		CSIPR000, CSIPR100 <sup>Note 1</sup>	
INTSR0	SRIF0		SRMK0		SRPR00, SRPR10	
INTSRE0 <sup>Note 2</sup>	SREIF0 <sup>Note 2</sup>		SREMK0 <sup>Note 2</sup>		SREPR00, SREPR10 <sup>Note 2</sup>	
INTTM01H <sup>Note 2</sup>	TMIF01H <sup>Note 2</sup>		TMMK01H <sup>Note 2</sup>		TMPR001H, TMPR101H <sup>Note 2</sup>	
INTTM03H	TMIF03H	IF1L	TMMK03H	MK1L	TMPR003H, TMPR103H	PR01L, PR11L
INTIICA1	IICAIF1		IICAMK1		IICAPR01, IICAPR11	
INTTM00	TMIF00		TMMK00		TMPR000, TMPR100	
INTTM01	TMIF01		TMMK01		TMPR001, TMPR101	
INTTM02	TMIF02		TMMK02		TMPR002, TMPR102	
INTTM03	TMIF03		TMMK03		TMPR003, TMPR103	

**Notes 1.** If one of the interrupt sources INTST0 and INTCSI00 is generated, bit 5 of the IF0H register is set to 1. Bit 5 of the MK0H, PR00H, and PR10H registers supports these three interrupt sources.

- 2.** Do not use a UART0 reception error interrupt and an interrupt of channel 1 of TAU0 (at higher 8-bit timer operation) at the same time because they share flags for the interrupt request sources. When the UART0 reception error interrupt is not used (EOC01 = 0), UART0 and channel 1 of TAU0 (at higher 8-bit timer operation) can be used at the same time. If one of the interrupt sources INTSRE0 and INTTM01H is

<R>

generated, bit 7 of the IF0H register is set to 1. Bit 7 of the MK0H, PR00H, and PR10H registers supports these two interrupt sources.

### 15.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L)

The interrupt request flags are set to 1 when the corresponding interrupt request is generated or an instruction is executed. They are cleared to 0 when an instruction is executed upon acknowledgment of an interrupt request or upon reset signal generation.

When an interrupt is acknowledged, the interrupt request flag is automatically cleared and then the interrupt routine is entered.

The IF0L, IF0H, and IF1L registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the IF0L and IF0H registers are combined to form 16-bit register IF0, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 15-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L)**

Address: FFFE0H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0L	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0	LVIIIF	WDTIIF

Address: FFFE1H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
IF0H	SREIF0 TMIF01H	SRIF0	STIF0 CSIIF00	DMAIF1	DMAIF0	FLIF	IICAIF0	ADIF

Address: FFFE2H After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	1	0
IF1L	TMIF03	TMIF02	TMIF01	TMIF00	IICAIF1	TMIF03H	0	0

XXIFX	Interrupt request flag
0	No interrupt request signal is generated
1	Interrupt request is generated, interrupt request status

- <R> **Cautions**
1. Be sure to set bits that are not available to the initial value.
  2. When manipulating a flag of the interrupt request flag register, use a 1-bit memory manipulation instruction (CLR1). When describing in C language, use a bit manipulation instruction such as "IF0L.0 = 0;" or "\_asm("clr1 IF0L, 0");" because the compiled assembler must be a 1-bit memory manipulation instruction (CLR1).

If a program is described in C language using an 8-bit memory manipulation instruction such as "IF0L &= 0xfe;" and compiled, it becomes the assembler of three instructions.

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

In this case, even if the request flag of the another bit of the same interrupt request flag register (IF0L) is set to 1 at the timing between "mov a, IF0L" and "mov IF0L, a", the flag is cleared to 0 at "mov IF0L, a". Therefore, care must be exercised when using an 8-bit memory manipulation instruction in C language.

### 15.3.2 Interrupt mask flag registers (MK0L, MK0H, MK1L)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing.

The MK0L, MK0H, and MK1L registers can be set by a 1-bit or 8-bit memory manipulation instruction. When the MK0L and MK0H registers are combined to form 16-bit registers MK0, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 15-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L)**

Address: FFFE4H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0L	PMK5	PMK4	PMK3	PMK2	PMK1	PMK0	LVIMK	WDTIMK

Address: FFFE5H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
MK0H	SREMK0 TMMK01H	SRMK0	STMK0 CSIMK00	DMAMK1	DMAMK0	FLMK	IICAMK0	ADMK

Address: FFFE6H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	1	0
MK1L	TMMK03	TMMK02	TMMK01	TMMK00	IICAMK1	TMMK03H	1	1

XXMKX	Interrupt servicing control
0	Interrupt servicing enabled
1	Interrupt servicing disabled

<R> **Caution** Be sure to set bits that are not available to the initial value.

### 15.3.3 Priority specification flag registers (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L)

The priority specification flag registers are used to set the corresponding maskable interrupt priority level.

A priority level is set by using the PR0xy and PR1xy registers in combination (xy = 0L, 0H, 1L, 1H, 2L, or 2H).

The PR00L, PR00H, PR01L, PR10L, PR10H, and PR11L registers can be set by a 1-bit or 8-bit memory manipulation instruction. If the PR00L and PR00H registers, and the PR10L and PR10H registers are combined to form 16-bit registers PR00 and PR10, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 15-4. Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L)**

Address: FFFE8H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR00L	PPR05	PPR04	PPR03	PPR02	PPR01	PPR00	LVIPR0	WDTIPR0

Address: FFFECH After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR10L	PPR15	PPR14	PPR13	PPR12	PPR11	PPR10	LVIPR1	WDTIPR1

Address: FFFE9H After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR00H	SREPR00 TMPR001H	SRPR00	STPR00 CSIPR000	DMAPR01	DMAPR00	FLPR0	IICAPR00	ADPR0

Address: FFFEDH After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	<1>	<0>
PR10H	SREPR10 TMPR101H	SRPR10	STPR10 CSIPR100	DMAPR11	DMAPR10	FLPR1	IICAPR10	ADPR1

Address: FFFEAH After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	1	0
PR01L	TMPR003	TMPR002	TMPR001	TMPR000	IICAPR01	TMPR003H	1	1

Address: FFFEEH After reset: FFH R/W

Symbol	<7>	<6>	<5>	<4>	<3>	<2>	1	0
PR11L	TMPR103	TMPR102	TMPR101	TMPR100	IICAPR11	TMPR103H	1	1

XXPR1X	XXPR0X	Priority level selection
0	0	Specify level 0 (high priority level)
0	1	Specify level 1
1	0	Specify level 2
1	1	Specify level 3 (low priority level)

<R> **Caution** Be sure to set bits that are not available to the initial value.

### 15.3.4 External interrupt rising edge enable register (EGP0), external interrupt falling edge enable register (EGN0)

These registers specify the valid edge for INTP0 to INTP5.

The EGP0 and EGN0 registers can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

**Figure 15-5. Format of External Interrupt Rising Edge Enable Register (EGP0) and External Interrupt Falling Edge Enable Register (EGN0)**

Address: FFF38H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGP0	0	0	EGP5	EGP4	EGP3	EGP2	EGP1	EGP0

Address: FFF39H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
EGN0	0	0	EGN5	EGN4	EGN3	EGN2	EGN1	EGN0

EGPn	EGNn	INTPn pin valid edge selection (n = 0 to 5)
0	0	Edge detection disabled
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

Table 15-3 shows the ports corresponding to the EGPn and EGNn bits.

**Table 15-3. Ports Corresponding to EGPn and EGNn bits**

Detection Enable Bit		Interrupt Request Signal
EGP0	EGN0	INTP0
EGP1	EGN1	INTP1
EGP2	EGN2	INTP2
EGP3	EGN3	INTP3
EGP4	EGN4	INTP4
EGP5	EGN5	INTP5

<R>

**Caution** When the input port pins used for the external interrupt functions are switched to the output mode, the INTPn interrupt might be generated upon detection of a valid edge.

When switching the input port pins to the output mode, set the port mode register (PMxx) to 0 after disabling the edge detection (by setting EGPn and EGNn to 0).

**Remarks** 1. For details on the edge detection ports, see 2.1 Pin Function List.

2. n = 0 to 5

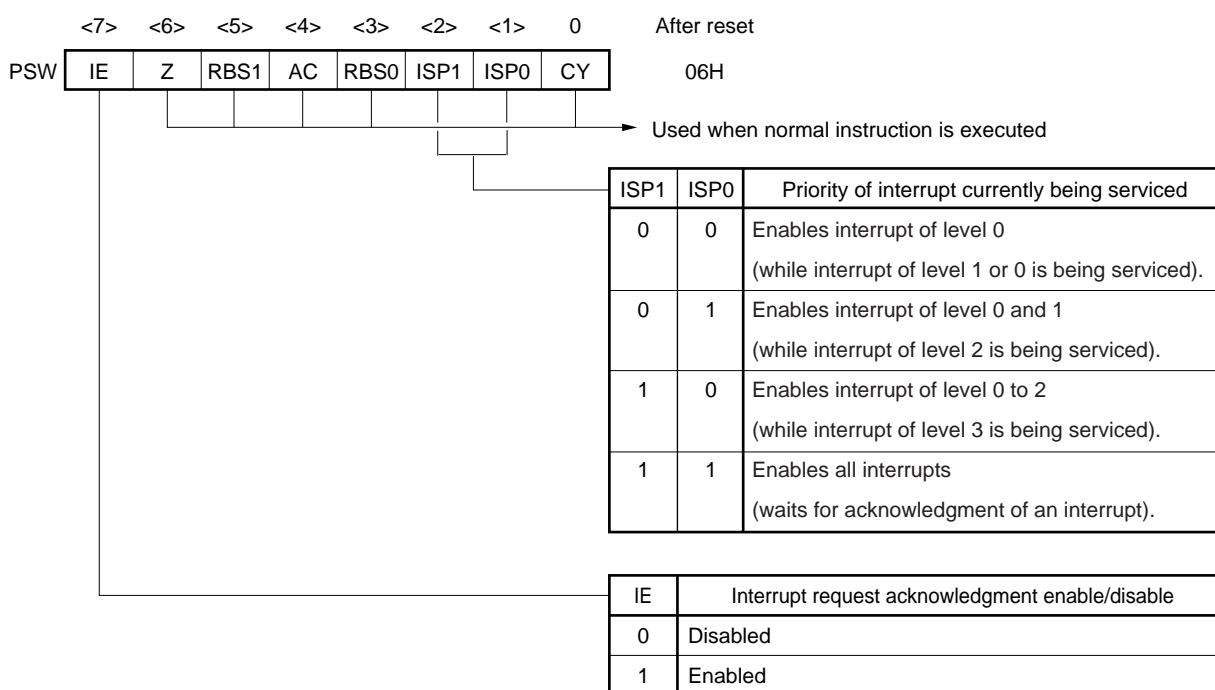
### 15.3.5 Program status word (PSW)

The program status word is a register used to hold the instruction execution result and the current status for an interrupt request. The IE flag that sets maskable interrupt enable/disable and the ISP0 and ISP1 flags that controls multiple interrupt servicing are mapped to the PSW.

Besides 8-bit read/write, this register can carry out operations using bit manipulation instructions and dedicated instructions (EI and DI). When a vectored interrupt request is acknowledged, if the BRK instruction is executed, the contents of the PSW are automatically saved into a stack and the IE flag is reset to 0. Upon acknowledgment of a maskable interrupt request, if the value of the priority specification flag register of the acknowledged interrupt is not 00, its value minus 1 is transferred to the ISP0 and ISP1 flags. The PSW contents are also saved into the stack with the PUSH PSW instruction. They are restored from the stack with the RETI, RETB, and POP PSW instructions.

Reset signal generation sets PSW to 06H.

Figure 15-6. Configuration of Program Status Word





## 15.4 Interrupt Servicing Operations

### 15.4.1 Maskable interrupt request acknowledgment

A maskable interrupt request becomes acknowledgeable when the interrupt request flag is set to 1 and the mask (MK) flag corresponding to that interrupt request is cleared to 0. A vectored interrupt request is acknowledged if interrupts are in the interrupt enabled state (when the IE flag is set to 1). However, a low-priority interrupt request is not acknowledged during servicing of a higher priority interrupt request.

The times from generation of a maskable interrupt request until vectored interrupt servicing is performed are listed in Table 15-4 below.

For the interrupt request acknowledgment timing, see **Figures 15-8 and 15-9**.

**Table 15-4. Time from Generation of Maskable Interrupt Until Servicing**

	Minimum Time	Maximum Time <sup>Note</sup>
Servicing time	9 clocks	16 clocks

**Note** Maximum time does not apply when an instruction from the internal RAM area is executed.

**Remark** 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

If two or more maskable interrupt requests are generated simultaneously, the request with a higher priority level specified in the priority specification flag is acknowledged first. If two or more interrupts requests have the same priority level, the request with the highest default priority is acknowledged first.

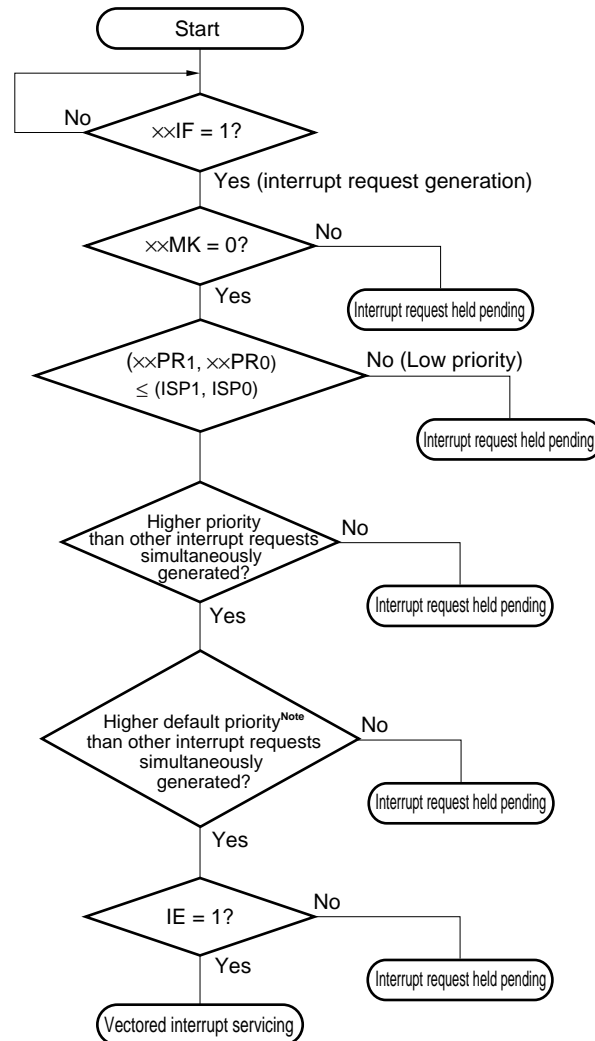
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 15-7 shows the interrupt request acknowledgment algorithm.

If a maskable interrupt request is acknowledged, the contents are saved into the stacks in the order of PSW, then PC, the IE flag is reset (0), and the contents of the priority specification flag corresponding to the acknowledged interrupt are transferred to the ISP1 and ISP0 flags. The vector table data determined for each interrupt request is the loaded into the PC and branched.

Restoring from an interrupt is possible by using the RETI instruction.

Figure 15-7. Interrupt Request Acknowledgment Processing Algorithm



xxIF: Interrupt request flag

xxMK: Interrupt mask flag

xxPR0: Priority specification flag 0

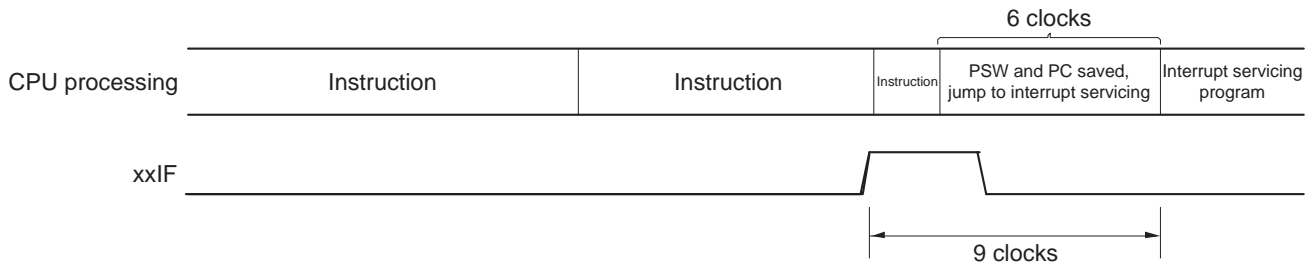
xxPR1: Priority specification flag 1

IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)

ISP0, ISP1: Flag that indicates the priority level of the interrupt currently being serviced (see Figure 15-6)

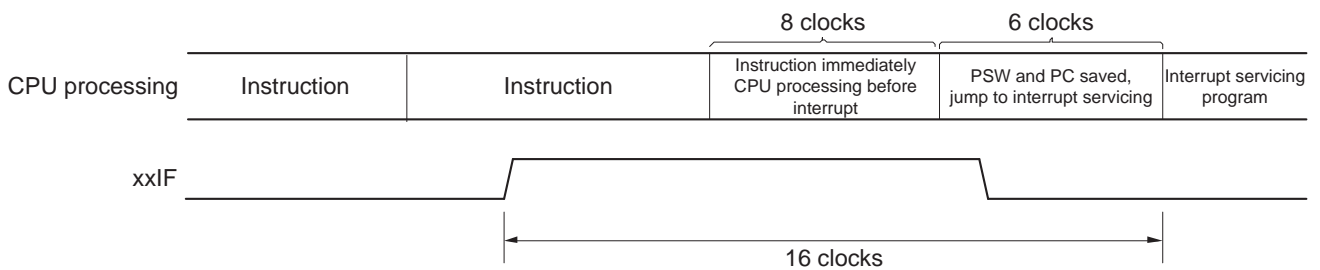
**Note** For the default priority, see Table 15-1 Interrupt Source List.

<R> **Figure 15-8. Interrupt Request Acknowledgment Timing (Minimum Time)**



**Remark** 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

<R> **Figure 15-9. Interrupt Request Acknowledgment Timing (Maximum Time)**



**Remark** 1 clock:  $1/f_{CLK}$  ( $f_{CLK}$ : CPU clock)

### 15.4.2 Software interrupt request acknowledgment

A software interrupt request is acknowledged by BRK instruction execution. Software interrupts cannot be disabled.

If a software interrupt request is acknowledged, the contents are saved into the stacks in the order of the program status word (PSW), then program counter (PC), the IE flag is reset (0), and the contents of the vector table (0007EH, 0007FH) are loaded into the PC and branched.

Restoring from a software interrupt is possible by using the RETB instruction.

**Caution** Can not use the RETI instruction for restoring from the software interrupt.

### 15.4.3 Multiple interrupt servicing

Multiple interrupt servicing occurs when another interrupt request is acknowledged during execution of an interrupt.

Multiple interrupt servicing does not occur unless the interrupt request acknowledgment enabled state is selected (IE = 1). When an interrupt request is acknowledged, interrupt request acknowledgment becomes disabled (IE = 0). Therefore, to enable multiple interrupt servicing, it is necessary to set (1) the IE flag with the EI instruction during interrupt servicing to enable interrupt acknowledgment.

Moreover, even if interrupts are enabled, multiple interrupt servicing may not be enabled, this being subject to interrupt priority control. Two types of priority control are available: default priority control and programmable priority control. Programmable priority control is used for multiple interrupt servicing.

In the interrupt enabled state, if an interrupt request with a priority higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority equal to or lower than that of the interrupt currently being serviced is generated during interrupt servicing, it is not acknowledged for multiple interrupt servicing. Interrupt requests that are not enabled because interrupts are in the interrupt disabled state or because they have a lower priority are held pending. When servicing of the current interrupt ends, the pending interrupt request is acknowledged following execution of at least one main processing instruction execution.

Table 15-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 15-10 shows multiple interrupt servicing examples.

**Table 15-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing**

Multiple Interrupt Request Interrupt Being Serviced		Maskable Interrupt Request								Software Interrupt Request
		Priority Level 0 (PR = 00)		Priority Level 1 (PR = 01)		Priority Level 2 (PR = 10)		Priority Level 3 (PR = 11)		
		IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	IE = 1	IE = 0	
Maskable interrupt	ISP1 = 0 ISP0 = 0	○	×	×	×	×	×	×	×	○
	ISP1 = 0 ISP0 = 1	○	×	○	×	×	×	×	×	○
	ISP1 = 1 ISP0 = 0	○	×	○	×	○	×	×	×	○
Software interrupt		○	×	○	×	○	×	○	×	○

**Remarks 1.** ○: Multiple interrupt servicing enabled

**2.** ×: Multiple interrupt servicing disabled

**3.** ISP0, ISP1, and IE are flags contained in the PSW.

ISP1 = 0, ISP0 = 0: An interrupt of level 1 or level 0 is being serviced.

ISP1 = 0, ISP0 = 1: An interrupt of level 2 is being serviced.

ISP1 = 1, ISP0 = 0: An interrupt of level 3 is being serviced.

ISP1 = 1, ISP0 = 1: Wait for An interrupt acknowledgment (all interrupts enabled).

IE = 0: Interrupt request acknowledgment is disabled.

IE = 1: Interrupt request acknowledgment is enabled.

**4.** PR is a flag contained in the PR00L, PR00H, PR01L, PR10L, PR10H, and PR11L registers.

PR = 00: Specify level 0 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 0$  (higher priority level)

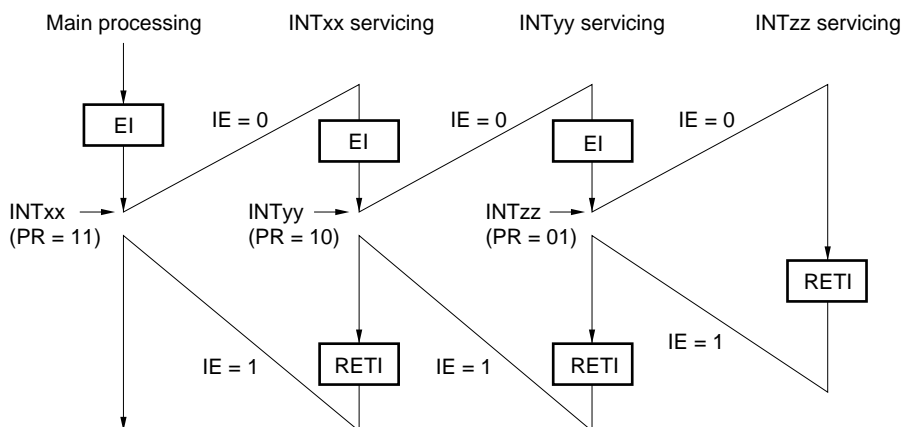
PR = 01: Specify level 1 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 1$

PR = 10: Specify level 2 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 0$

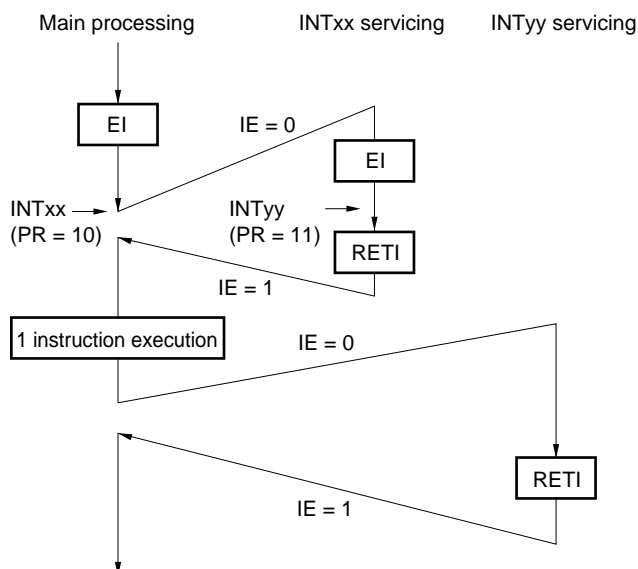
PR = 11: Specify level 3 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 1$  (lower priority level)

<R>

Figure 15-10. Examples of Multiple Interrupt Servicing (1/2)

**Example 1. Multiple interrupt servicing occurs twice**

During servicing of interrupt INTxx, two interrupt requests, INTyy and INTzz, are acknowledged, and multiple interrupt servicing takes place. Before each interrupt request is acknowledged, the EI instruction must always be issued to enable interrupt request acknowledgment.

**Example 2. Multiple interrupt servicing does not occur due to priority control**

Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 0$  (higher priority level)

PR = 01: Specify level 1 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 1$

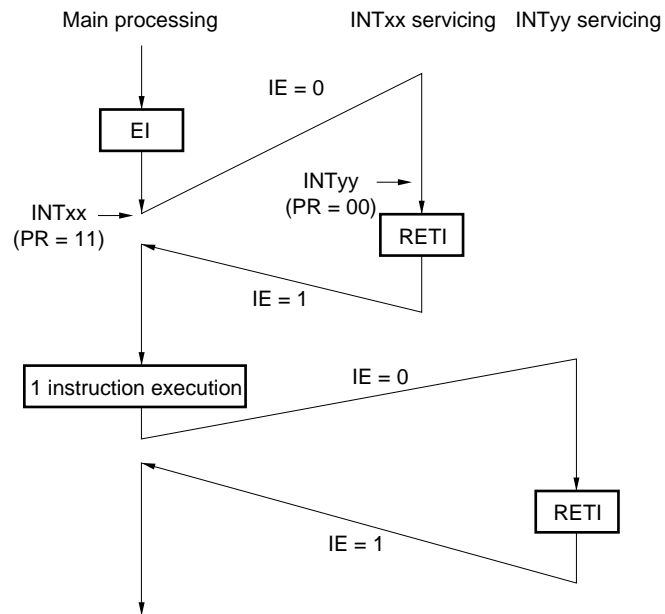
PR = 10: Specify level 2 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 0$

PR = 11: Specify level 3 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 1$  (lower priority level)

IE = 0: Interrupt request acknowledgment is disabled

IE = 1: Interrupt request acknowledgment is enabled.

Figure 15-10. Examples of Multiple Interrupt Servicing (2/2)

**Example 3. Multiple interrupt servicing does not occur because interrupts are not enabled**

Interrupts are not enabled during servicing of interrupt INTxx (EI instruction is not issued), therefore, interrupt request INTyy is not acknowledged and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 00: Specify level 0 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 0$  (higher priority level)

PR = 01: Specify level 1 with  $\times\times PR1\times = 0$ ,  $\times\times PR0\times = 1$

PR = 10: Specify level 2 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 0$

PR = 11: Specify level 3 with  $\times\times PR1\times = 1$ ,  $\times\times PR0\times = 1$  (lower priority level)

IE = 0: Interrupt request acknowledgment is disabled

IE = 1: Interrupt request acknowledgment is enabled.

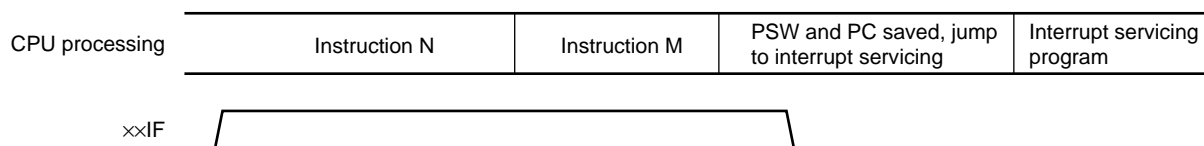
#### 15.4.4 Interrupt request hold

There are instructions where, even if an interrupt request is issued while the instructions are being executed, interrupt request acknowledgment is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- MOV PSW, #byte
  - MOV PSW, A
  - MOV1 PSW. bit, CY
  - SET1 PSW. bit
  - CLR1 PSW. bit
  - RETB
  - RETI
  - POP PSW
  - BTCLR PSW. bit, \$addr20
  - EI
  - DI
  - SKC
  - SKNC
  - SKZ
  - SKNZ
  - SKH
  - SKNH
- <R> • Write instructions for the IF0L, IF0H, IF1L, IF2L, MK0L, MK0H, MK1L, MK2L, PR00L, PR00H, PR01L, PR02L, PR10L, PR10H, PR11L, and PR12L registers

Figure 15-11 shows the timing at which interrupt requests are held pending.

**Figure 15-11. Interrupt Request Hold**



- Remarks**
1. Instruction N: Interrupt request hold instruction
  2. Instruction M: Instruction other than interrupt request hold instruction



## CHAPTER 16 STANDBY FUNCTION

### 16.1 Standby Function

The standby function reduces the operating current of the system, and the following three modes are available.

#### (1) HALT mode

HALT instruction execution sets the HALT mode. In the HALT mode, the CPU operation clock is stopped. If the high-speed system clock oscillator or high-speed on-chip oscillator is operating before the HALT mode is set, oscillation of each clock continues. In this mode, the operating current is not decreased as much as in the STOP mode, but the HALT mode is effective for restarting operation immediately upon interrupt request generation and carrying out intermittent operations frequently.

#### (2) STOP mode

STOP instruction execution sets the STOP mode. In the STOP mode, the high-speed system clock oscillator and high-speed on-chip oscillator stop, stopping the whole system, thereby considerably reducing the CPU operating current.

Because this mode can be cleared by an interrupt request, it enables intermittent operations to be carried out. However, because a wait time is required to secure the oscillation stabilization time after the STOP mode is released when the X1 clock is selected, select the HALT mode if it is necessary to start processing immediately upon interrupt request generation.

#### (3) SNOOZE mode

In the case of CSIp or A/D conversion request by UARTq data reception, the STOP mode is exited, the CSIp or UARTq data is received without operating the CPU, and A/D conversion is performed. This can only be specified when the high-speed on-chip oscillator is selected for the CPU/peripheral hardware clock (f<sub>CLK</sub>).

In either of these two modes, all the contents of registers, flags and data memory just before the standby mode is set are held. The I/O port output latches and output buffer statuses are also held.

- Cautions**
1. When shifting to the STOP mode, be sure to stop the peripheral hardware operation operating with main system clock before executing STOP instruction (except SNOOZE mode setting unit).
  2. When using CSIp, UARTq, or the A/D converter in the SNOOZE mode, set up serial standby control register m (SSCm) and A/D converter mode register 2 (ADM2) before switching to the STOP mode. For details, see 9.3 Registers Controlling A/D Converter and 11.3 Registers Controlling Serial Array Unit.
  3. The following sequence is recommended for power consumption reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS) and bit 0 (ADCE) of A/D converter mode register 0 (ADM0) to 0 to stop the A/D conversion operation, and then execute the STOP instruction.
  4. It can be selected by the option byte whether the low-speed on-chip oscillator continues oscillating or stops in the HALT or STOP mode. For details, see CHAPTER 22 OPTION BYTE.

**Remark** p = 00; q = 0; m = 0

## 16.2 Registers Controlling Standby Function

The standby function is controlled by the following two registers.

- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

<R>

**Remark** For the registers that start, stop, or select the clock, see **CHAPTER 5 CLOCK GENERATOR**. For registers which control the SNOOZE mode, **CHAPTER 9 A/D CONVERTER** and **CHAPTER 11 SERIAL ARRAY UNIT**.

### 16.2.1 Oscillation stabilization time counter status register (OSTC)

This is the register that indicates the count status of the X1 clock oscillation stabilization time counter.

The X1 clock oscillation stabilization time can be checked in the following case.

- If the X1 clock starts oscillation while the high-speed on-chip oscillator clock is being used as the CPU clock.
- If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock with the X1 clock oscillating.

The OSTC register can be read by a 1-bit or 8-bit memory manipulation instruction.

When reset is released (reset by  $\overline{\text{RESET}}$  input, POR, LVD, WDT, and executing an illegal instruction), the STOP instruction and MSTOP bit (bit 7 of clock operation status control register (CSC)) = 1 clear this register to 00H.

**Figure 16-1. Format of Oscillation Stabilization Time Counter Status Register (OSTC)**

Address: FFFA2H After reset: 00H R

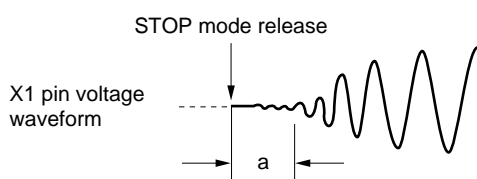
Symbol	7	6	5	4	3	2	1	0
OSTC	MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18

MOST 8	MOST 9	MOST 10	MOST 11	MOST 13	MOST 15	MOST 17	MOST 18	Oscillation stabilization time status		
									$f_x = 10 \text{ MHz}$	$f_x = 20 \text{ MHz}$
0	0	0	0	0	0	0	0	$2^8/f_x \text{ max.}$	25.6 $\mu\text{s max.}$	12.8 $\mu\text{s max.}$
1	0	0	0	0	0	0	0	$2^9/f_x \text{ min.}$	25.6 $\mu\text{s min.}$	12.8 $\mu\text{s min.}$
1	1	0	0	0	0	0	0	$2^9/f_x \text{ min.}$	51.2 $\mu\text{s min.}$	25.6 $\mu\text{s min.}$
1	1	1	0	0	0	0	0	$2^{10}/f_x \text{ min.}$	102 $\mu\text{s min.}$	51.2 $\mu\text{s min.}$
1	1	1	1	0	0	0	0	$2^{11}/f_x \text{ min.}$	204 $\mu\text{s min.}$	102 $\mu\text{s min.}$
1	1	1	1	1	0	0	0	$2^{13}/f_x \text{ min.}$	819 $\mu\text{s min.}$	409 $\mu\text{s min.}$
1	1	1	1	1	1	0	0	$2^{15}/f_x \text{ min.}$	3.27 ms min.	1.63 ms min.
1	1	1	1	1	1	1	0	$2^{17}/f_x \text{ min.}$	13.1 ms min.	6.55 ms min.
1	1	1	1	1	1	1	1	$2^{18}/f_x \text{ min.}$	26.2 ms min.	13.1 ms min.

<R>

- Cautions**
1. After the above time has elapsed, the bits are set to 1 in order from the MOST8 bit and remain 1.
  2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by the oscillation stabilization time select register (OSTS). If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.
    - Desired OSTC register oscillation stabilization time  $\leq$  Oscillation stabilization time set by OSTs register

Note, therefore, that only the status up to the oscillation stabilization time set by the OSTs register is set to the OSTC register after STOP mode is released.
  3. The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts ("a" below).



**Remark**  $f_x$ : X1 clock oscillation frequency

### 16.2.2 Oscillation stabilization time select register (OSTS)

This register is used to select the X1 clock oscillation stabilization wait time when the STOP mode is released.

When the X1 clock is selected as the CPU clock, the operation waits for the time set using the OSTS register after the STOP mode is released.

When the high-speed on-chip oscillator clock is selected as the CPU clock, confirm with the oscillation stabilization time counter status register (OSTC) that the desired oscillation stabilization time has elapsed after the STOP mode is released. The oscillation stabilization time can be checked up to the time set using the OSTC register.

The OSTS register can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 07H.

**Figure 16-2. Format of Oscillation Stabilization Time Select Register (OSTS)**

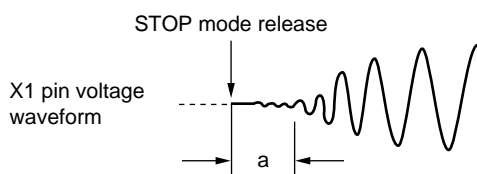
Address: FFFA3H After reset: 07H R/W

Symbol	7	6	5	4	3	2	1	0
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0

OSTS2	OSTS1	OSTS0		Oscillation stabilization time selection	
				$f_x = 10 \text{ MHz}$	$f_x = 20 \text{ MHz}$
0	0	0	$2^8/f_x$	25.6 $\mu\text{s}$	12.8 $\mu\text{s}$
0	0	1	$2^9/f_x$	51.2 $\mu\text{s}$	25.6 $\mu\text{s}$
0	1	0	$2^{10}/f_x$	102 $\mu\text{s}$	51.2 $\mu\text{s}$
0	1	1	$2^{11}/f_x$	204 $\mu\text{s}$	102 $\mu\text{s}$
1	0	0	$2^{13}/f_x$	819 $\mu\text{s}$	409 $\mu\text{s}$
1	0	1	$2^{15}/f_x$	3.27 ms	1.63 ms
1	1	0	$2^{17}/f_x$	13.1 ms	6.55 ms
1	1	1	$2^{18}/f_x$	26.2 ms	13.1 ms

- <R>
- Cautions**
- To set the STOP mode when the X1 clock is used as the CPU clock, set the OSTS register before executing the STOP instruction.
  - Before changing the setting of the OSTS register, confirm that the count operation of the OSTC register is completed.
  - Do not change the value of the OSTS register during the X1 clock oscillation stabilization time.
  - The oscillation stabilization time counter counts up to the oscillation stabilization time set by the OSTS register. If the STOP mode is entered and then released while the high-speed on-chip oscillator clock is being used as the CPU clock, set the oscillation stabilization time as follows.
    - Desired OSTC register oscillation stabilization time  $\leq$  Oscillation stabilization time set by OSTS register

Note, therefore, that only the status up to the oscillation stabilization time set by the OSTS register is set to the OSTC register after STOP mode is released.
  - The X1 clock oscillation stabilization wait time does not include the time until clock oscillation starts ("a" below).



**Remark**  $f_x$ : X1 clock oscillation frequency

## 16.3 Standby Function Operation

### 16.3.1 HALT mode

#### (1) HALT mode

The HALT mode is set by executing the HALT instruction. HALT mode can be set regardless of whether the CPU clock before the setting was the high-speed system clock or high-speed on-chip oscillator clock.

The operating statuses in the HALT mode are shown below.

<R> **Caution** Because the interrupt request signal is used to clear the HALT mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the HALT mode is not entered even if the HALT instruction is executed in such a situation..

Table 16-1. Operating Statuses in HALT Mode

HALT Mode Setting		When HALT Instruction Is Executed While CPU Is Operating on Main System Clock					
		When CPU Is Operating on High-speed On-chip Oscillator Clock ( $f_{IH}$ )	When CPU Is Operating on X1 Clock ( $f_x$ )	When CPU Is Operating on External Main System Clock ( $f_{EX}$ )			
Item							
System clock		Clock supply to the CPU is stopped					
Main system clock	$f_{IH}$	Operation continues (cannot be stopped)	Operation disabled				
	$f_x$	Operation disabled	Operation continues (cannot be stopped)	Cannot operate			
	$f_{EX}$		Cannot operate	Operation continues (cannot be stopped)			
$f_{IL}$		Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H)					
CPU		Operation stopped					
Code flash memory		Operation stopped					
Data flash memory							
RAM							
Port (latch)							
Timer array unit		Operable					
Watchdog timer		See <b>CHAPTER 8 WATCHDOG TIMER</b>					
Clock output/buzzer output		Operable					
A/D converter							
D/A converter							
Serial array unit (SAU)							
Serial interface (IICA)							
DMA controller							
<R>	Event link controller (ELC)				Operable function blocks can be linked		
Power-on-reset function					Operable		
Voltage detection function							
External interrupt							
CRC operation function	High-speed CRC	In the calculation of the RAM area, operable when DMA is executed only					
	General-purpose CRC						
RAM parity error detection function		Operable when DMA is executed only					
RAM guard function							
SFR guard function							
Illegal-memory access detection function							

**Remark** Operation stopped: Operation is automatically stopped before switching to the HALT mode.

Operation disabled: Operation is stopped before switching to the HALT mode.

$f_{IH}$ : High-speed on-chip oscillator clock       $f_{EX}$ : External main system clock

$f_{IL}$ : Low-speed on-chip oscillator clock       $f_x$ : X1 clock

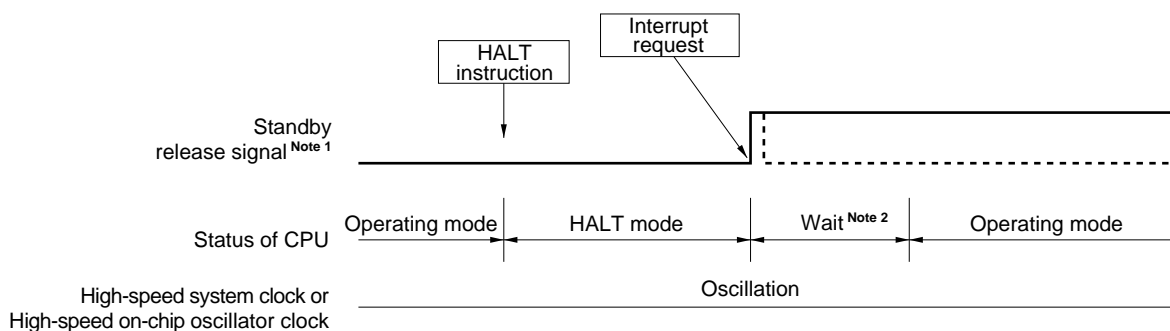
**(2) HALT mode release**

The HALT mode can be released by the following two sources.

**(a) Release by unmasked interrupt request**

When an unmasked interrupt request is generated, the HALT mode is released. If interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 16-3. HALT Mode Release by Interrupt Request Generation**



<R>

**Notes 1.** Refer to **Table 15-1. Interrupt Source List (1/2)**.

**2.** Wait time for HALT mode release

- When vectored interrupt servicing is carried out  
Main system clock: 15 to 16 clock
- When vectored interrupt servicing is not carried out  
Main system clock: 9 to 10 clock

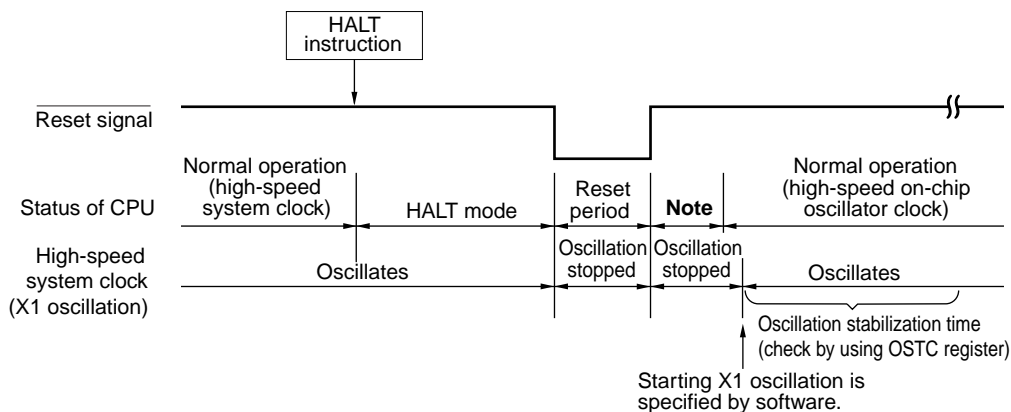
**Remark** The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

**(b) Release by reset signal generation**

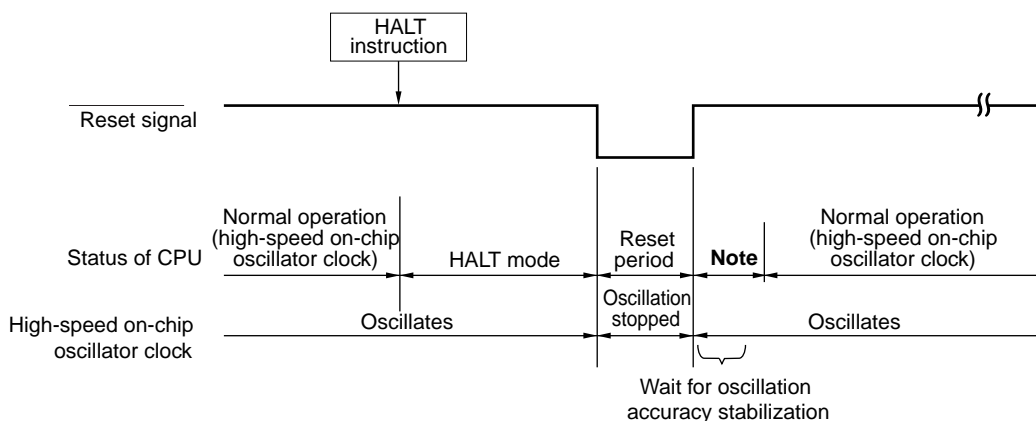
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 16-4. HALT Mode Release by Reset**

**(1) When high-speed system clock is used as CPU clock**



**(2) When high-speed on-chip oscillator clock is used as CPU clock**



**Note** For the reset processing time, see **CHAPTER 17 RESET FUNCTION**.  
 For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 18 POWER-ON-RESET CIRCUIT**.



### 16.3.2 STOP mode

#### (1) STOP mode setting and operating statuses

The STOP mode is set by executing the STOP instruction, and it can be set only when the CPU clock before the setting was the high-speed on-chip oscillator clock, X1 clock, or external main system clock.

- <R> **Cautions**
- 1. Because the interrupt request signal is used to clear the STOP mode, if the interrupt mask flag is 0 (the interrupt processing is enabled) and the interrupt request flag is 1 (the interrupt request signal is generated), the STOP mode is immediately cleared if set when the STOP instruction is executed in such a situation. Accordingly, once the STOP instruction is executed, the system returns to its normal operating mode after the elapse of release time from the STOP mode..**
  - 2. When using CSIp, UARTq, or the A/D converter in the SNOOZE mode, set up serial standby control register m (SSCm) and A/D converter mode register 2 (ADM2) before switching to the STOP mode. For details, see 9.3 Registers Controlling A/D Converter and 11.3 Registers Controlling Serial Array Unit.**

**Remark** p = 00; q = 0; m = 0

The operating statuses in the STOP mode are shown below.

Table 16-2. Operating Statuses in STOP Mode

STOP Mode Setting		When STOP Instruction Is Executed While CPU Is Operating on Main System Clock		
		When CPU Is Operating on High-speed On-chip Oscillator Clock ( $f_{IH}$ )	When CPU Is Operating on X1 Clock ( $f_x$ )	When CPU Is Operating on External Main System Clock ( $f_{EX}$ )
Item				
System clock		Clock supply to the CPU is stopped		
Main system clock	$f_{IH}$	Stopped		
	$f_x$			
	$f_{EX}$			
$f_{IL}$		Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H)		
CPU		Operation stopped		
Code flash memory				
Data flash memory		Operation stopped		
RAM		Operation stopped		
Port (latch)		Status before STOP mode was set is retained		
Timer array unit		Operation disabled		
Watchdog timer		See <b>CHAPTER 8 WATCHDOG TIMER</b>		
Clock output/buzzer output		Operation stopped		
A/D converter		Wakeup operation is enabled (switching to the SNOOZE mode)		
D/A converter		Operable (status before STOP mode was set is retained)		
Serial array unit (SAU)		Wakeup operation is enabled only for CSIp and UARTq (switching to the SNOOZE mode) Operation is disabled for anything other than CSIp and UARTq		
Serial interface (IICA)		Wakeup by address match operable		
DMA controller		Operation disabled		
Event link controller (ELC)		Operable function blocks can be linked		
Power-on-reset function		Operable		
Voltage detection function				
External interrupt				
CRC operation function	High-speed CRC	Operation stopped		
	General-purpose CRC			
RAM parity error detection function				
RAM guard function				
SFR guard function				
Illegal-memory access detection function				

&lt;R&gt;

&lt;R&gt;

(Remarks are listed on the next page.)

- Cautions**
- To use the peripheral hardware that stops operation in the STOP mode, and the peripheral hardware for which the clock that stops oscillating in the STOP mode after the STOP mode is released, restart the peripheral hardware.
  - To stop the low-speed on-chip oscillator clock in the STOP mode, must previously be set an option byte to stop the watchdog timer operation in the HALT/STOP mode (bit 0 (WDSTBYON) of 000C0H = 0).
  - To shorten oscillation stabilization time after the STOP mode is released when the CPU operates with the high-speed system clock (X1 oscillation), temporarily switch the CPU clock to the high-

speed on-chip oscillator clock before the execution of the STOP instruction. Before changing the CPU clock from the high-speed on-chip oscillator clock to the high-speed system clock (X1 oscillation) after the STOP mode is released, check the oscillation stabilization time with the oscillation stabilization time counter status register (OSTC).

**Remarks 1.** Operation stopped: Operation is automatically stopped before switching to the STOP mode.

Operation disabled: Operation is stopped before switching to the STOP mode.

f<sub>IH</sub>: High-speed on-chip oscillator clock      f<sub>IL</sub>: Low-speed on-chip oscillator clock

f<sub>x</sub>: X1 clock

f<sub>EX</sub>: External main system clock

2. p = 00; q = 0

## (2) STOP mode release

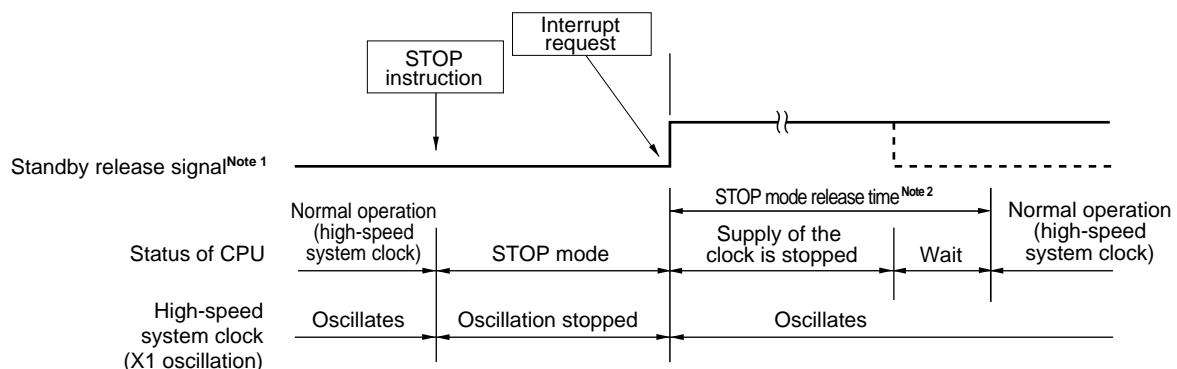
The STOP mode can be released by the following two sources.

### (a) Release by unmasked interrupt request

When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

**Figure 16-5. STOP Mode Release by Interrupt Request Generation (1/2)**

#### (1) When high-speed system clock (X1 oscillation) is used as CPU clock



**Notes 1.** For details of the standby release signal, see **Figure 15-1**.

#### 2. STOP mode release time

Supply of the clock is stopped:

- 18  $\mu$ s to "whichever is longer 65  $\mu$ s or the oscillation stabilization time (set by OSTC)"

Wait:

- When vectored interrupt servicing is carried out: 10 to 11 clocks
- When vectored interrupt servicing is not carried out: 4 to 5 clocks

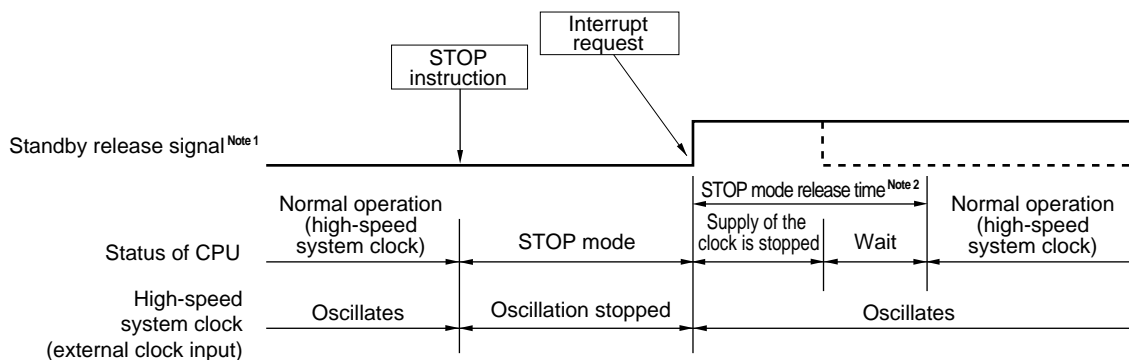
<R>

**Caution** To reduce the oscillation stabilization time after release from the STOP mode while CPU operates based on the high-speed system clock (X1 oscillation), switch the clock to the high-speed on-chip oscillator clock temporarily before executing the STOP instruction.

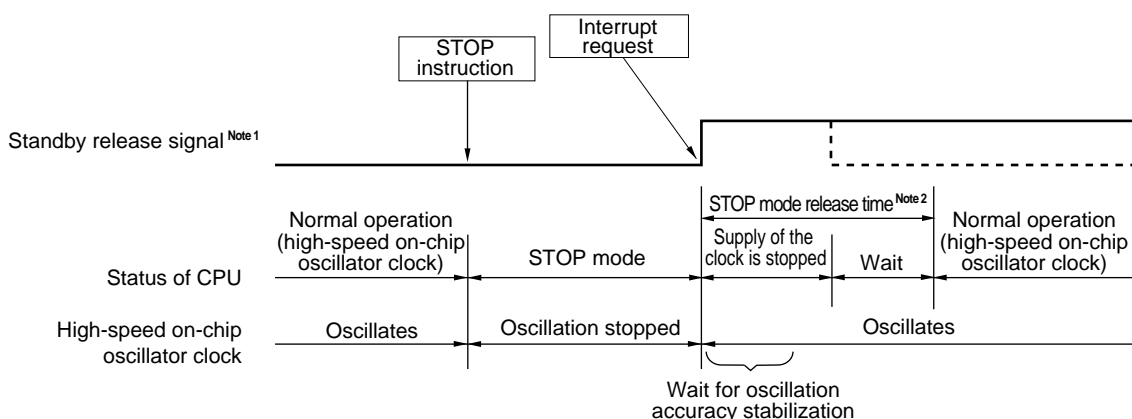
- Remarks**
1. The clock supply stop time varies depending on the temperature conditions and STOP mode period.
  2. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

**Figure 16-5. STOP Mode Release by Interrupt Request Generation (2/2)**

**(2) When high-speed system clock (external clock input) is used as CPU clock**



**(3) When high-speed on-chip oscillator clock is used as CPU clock**



**Notes** 1. For details of the standby release signal, see **Figure 15-1**.

2. STOP mode release time

Supply of the clock is stopped:

- 18  $\mu$ s to 65  $\mu$ s

Wait:

- When vectored interrupt servicing is carried out: 7 clocks
- When vectored interrupt servicing is not carried out: 1 clock

- Remarks**
1. The clock supply stop time varies depending on the temperature conditions and STOP mode period.
  2. The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

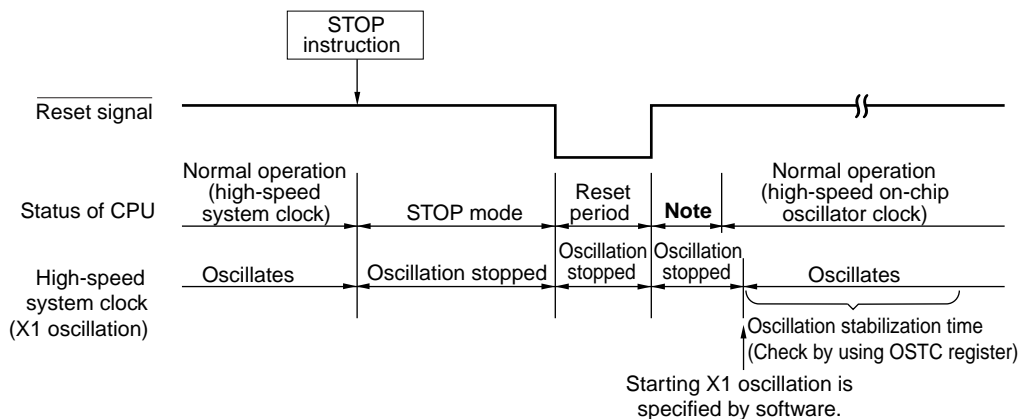
<R>

**(b) Release by reset signal generation**

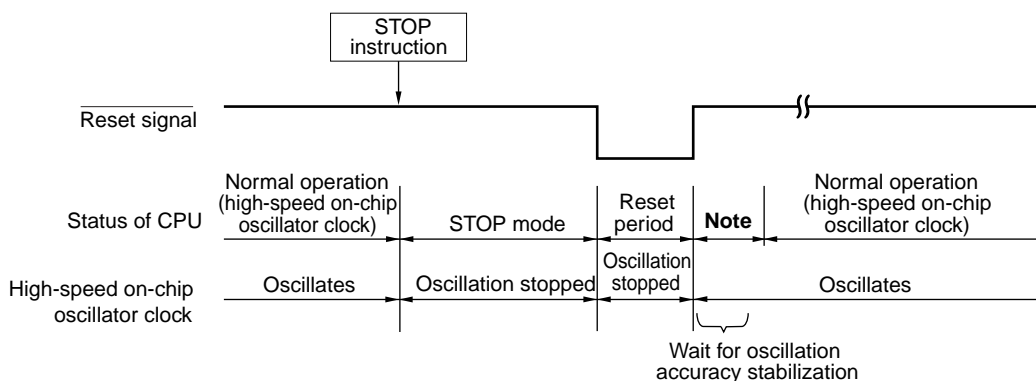
When the reset signal is generated, STOP mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

**Figure 16-6. STOP Mode Release by Reset**

**(1) When high-speed system clock is used as CPU clock**



**(2) When high-speed on-chip oscillator clock is used as CPU clock**



**Note** For the reset processing time, see **CHAPTER 17 RESET FUNCTION**.  
 For the reset processing time of the power-on-reset circuit (POR) and voltage detector (LVD), see **CHAPTER 18 POWER-ON-RESET CIRCUIT**.

### 16.3.3 SNOOZE mode

#### (1) SNOOZE mode setting and operating statuses

The SNOOZE mode can only be specified for CSIp, UARTq, or the A/D converter. Note that this mode can only be specified if the CPU clock is the high-speed on-chip oscillator clock.

When using CSIp or UARTq in the SNOOZE mode, set the SWCm bit of serial standby control register m (SSCm) to 1 immediately before switching to the STOP mode. For details, see **11.3 Registers Controlling Serial Array Unit**.

When using the A/D converter in the SNOOZE mode, set the AWC bit of A/D converter mode register 2 (ADM2) to 1 immediately before switching to the STOP mode. For details, see **9.3 Registers Controlling A/D Converter**.

**Remark** p = 00; q = 0; m = 0

In SNOOZE mode transition, wait status to be only following time.

<R> Transition time from STOP mode to SNOOZE mode:

- 18.96 to 28.95  $\mu$ s

From SNOOZE to normal operation

- When vectored interrupt servicing is carried out:
  - HS (High-speed main) mode: 6.79 to 12.4  $\mu$ s + 7 clocks
  - LS (Low-speed main) mode: 2.58 to 7.8  $\mu$ s + 7 clocks
- When vectored interrupt servicing is not carried out:
  - HS (High-speed main) mode: 6.79 to 12.4  $\mu$ s + 1 clock
  - LS (Low-speed main) mode: 2.58 to 7.8  $\mu$ s + 1 clock

The operating statuses in the SNOOZE mode are shown below.

Table 16-3. Operating Statuses in SNOOZE Mode

SNOOZE Mode Setting Item		When Inputting CSIp/UARTq Data Reception Signal or A/D Converter Timer Trigger Signal While in STOP Mode	
		When CPU Is Operating on High-speed On-chip Oscillator Clock ( $f_{IH}$ )	
System clock		Clock supply to the CPU is stopped	
Main system clock	$f_{IH}$	Operation started	
	$f_X$	Stopped	
	$f_{EX}$		
$f_{IL}$		Set by bits 0 (WDSTBYON) and 4 (WDTON) of option byte (000C0H)	
CPU		Operation stopped	
Code flash memory			
Data flash memory			
RAM			
Port (latch)		Use of the status while in the STOP mode continues	
Timer array unit		Operation disabled	
Watchdog timer		See <b>CHAPTER 8 WATCHDOG TIMER</b>	
Clock output/buzzer output		Operation stopped	
A/D converter		Operable	
D/A converter		Operable (status before SNOOZE mode was set is retained)	
Serial array unit (SAU)		Operable only CSIp and UARTq. Operation disabled other than CSIp and UARTq.	
Serial interface (IICA)		Operation disabled	
DMA controller			
Event link controller (ELC)			
Power-on-reset function		Operable	
Voltage detection function			
External interrupt			
CRC operation function		Operation disabled	
RAM parity error detection function			
RAM guard function			
SFR guard function			
Illegal-memory access detection function			

&lt;R&gt;

**Remarks 1.** Operation stopped: Operation is automatically stopped before switching to the SNOOZE mode.

Operation disabled: Operation is stopped before switching to the SNOOZE mode.

$f_{IH}$ : High-speed on-chip oscillator clock       $f_{IL}$ : Low-speed on-chip oscillator clock

$f_X$ : X1 clock

$f_{EX}$ : External main system clock

**2.** p = 00; q = 0

## CHAPTER 17 RESET FUNCTION

The following seven operations are available to generate a reset signal.

- (1) External reset input via  $\overline{\text{RESET}}$  pin
- (2) Internal reset by watchdog timer program loop detection
- (3) Internal reset by comparison of supply voltage and detection voltage of power-on-reset (POR) circuit
- (4) Internal reset by comparison of supply voltage of the voltage detector (LVD) and detection voltage
- (5) Internal reset by execution of illegal instruction<sup>Note</sup>
- (6) Internal reset by RAM parity error
- (7) Internal reset by illegal-memory access

External and internal resets start program execution from the address stored at 0000H and 0001H when the reset signal is generated.

A reset is effected when a low level is input to the  $\overline{\text{RESET}}$  pin, the watchdog timer overflows, or by POR and LVD circuit voltage detection, execution of illegal instruction<sup>Note</sup>, RAM parity error or illegal-memory access, and each item of hardware is set to the status shown in Tables 17-1.

When a low level is input to the  $\overline{\text{RESET}}$  pin, the device is reset. It is released from the reset status when a high level is input to the  $\overline{\text{RESET}}$  pin and program execution is started with the high-speed on-chip oscillator clock after reset processing. A reset by the watchdog timer overflow, execution of illegal instruction, detection of RAM parity error, or detection of illegal memory access is automatically released, and program execution starts using the high-speed onchip oscillator clock (see **Figures 17-2** and **17-3**) after reset processing. Reset by POR and LVD circuit supply voltage detection is automatically released when  $V_{DD} \geq V_{POR}$  or  $V_{DD} \geq V_{LVD}$  after the reset, and program execution starts using the high-speed on-chip oscillator clock (see **CHAPTER 18 POWER-ON-RESET CIRCUIT** and **CHAPTER 19 VOLTAGE DETECTOR**) after reset processing.

**Note** The illegal instruction is generated when instruction code FFH is executed.

Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Cautions** 1. For an external reset, input a low level for 10  $\mu\text{s}$  or more to the  $\overline{\text{RESET}}$  pin.

(To perform an external reset upon power application, a low level of at least 10  $\mu\text{s}$  must be continued during the period in which the supply voltage is within the operating range.)

The operating voltage range depends on the setting of the user option byte (000C2H).

The following shows the operating voltage range.

HS (high-speed main) mode:  $V_{DD} = 2.7$  to  $3.6$  V@1 MHz to 24 MHz

LS (low-speed main) mode:  $V_{DD} = 2.7$  to  $3.6$  V@1 MHz to 8 MHz

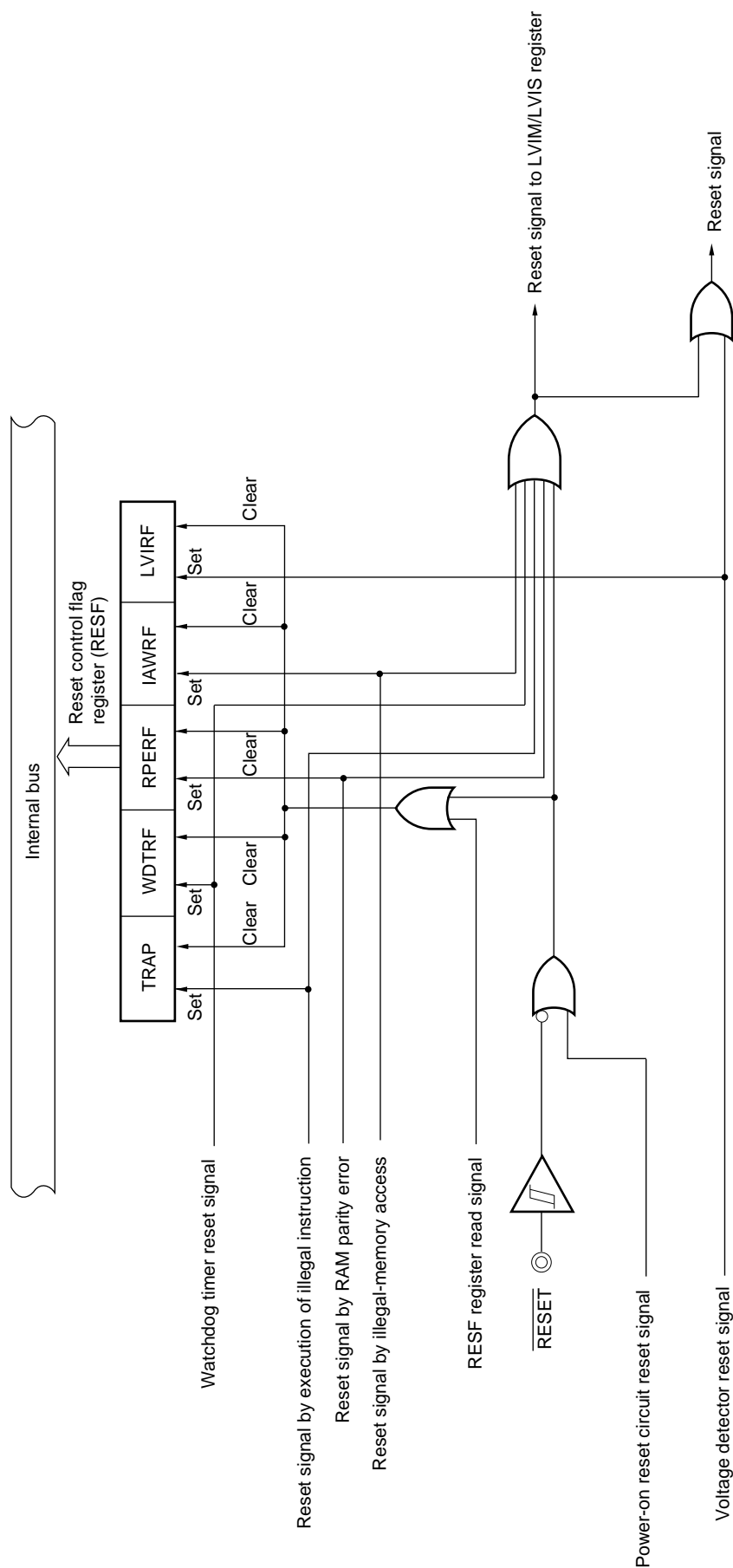
2. During reset input, the X1 clock, high-speed on-chip oscillator clock, and low-speed on-chip oscillator clock stop oscillating. External main system clock input becomes invalid.
3. Each of the SFRs and 2nd SFRs are initialized when a reset is applied, so P40 becomes high-impedance (in the case of an external reset or POR reset) or is pulled-up (in the case of other types of reset), and the other port pins become high-impedance.

**Remark**  $V_{POR}$ : POR power supply rise detection voltage

$V_{LVD}$ : LVD detection voltage



Figure 17-1. Block Diagram of Reset Function



**Caution** An LVD circuit internal reset does not reset the LVD circuit.

**Remarks 1.** LVIM: Voltage detection register

**2.** LVIS: Voltage detection level register

&lt;R&gt;

### 17.1 Timing of Reset Operation

This LSI is reset by input of the low level on the  $\overline{\text{RESET}}$  pin and released from the reset state by input of the high level on the  $\overline{\text{RESET}}$  pin. After reset processing, execution of the program with the high-speed on-chip oscillator clock as the operating clock starts.

Figure 17-2. Timing of Reset by  $\overline{\text{RESET}}$  Input

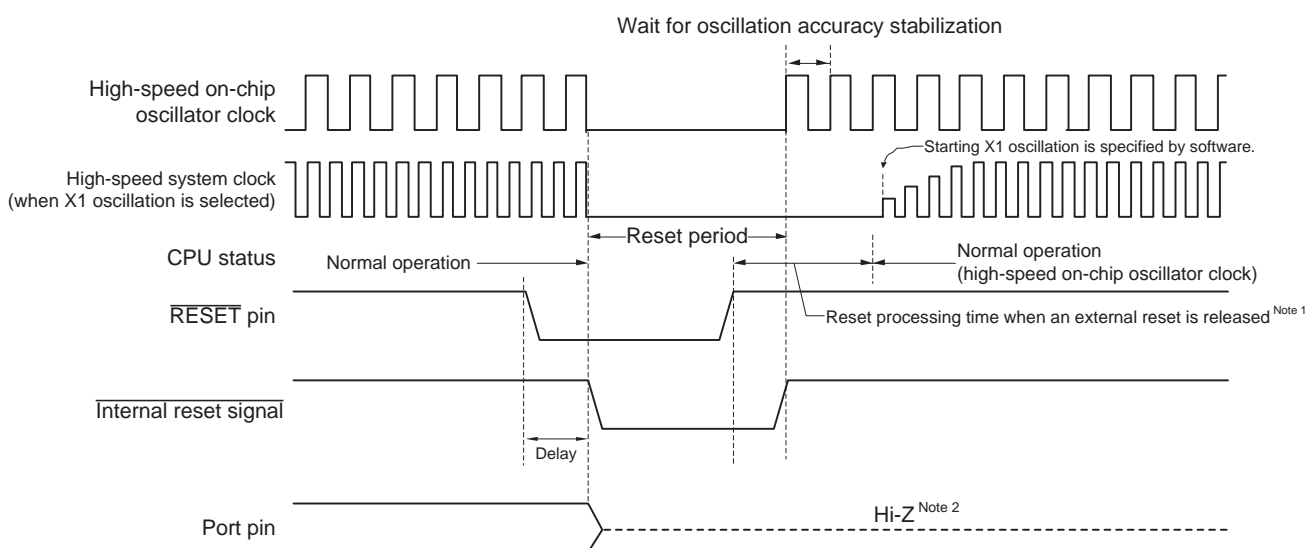
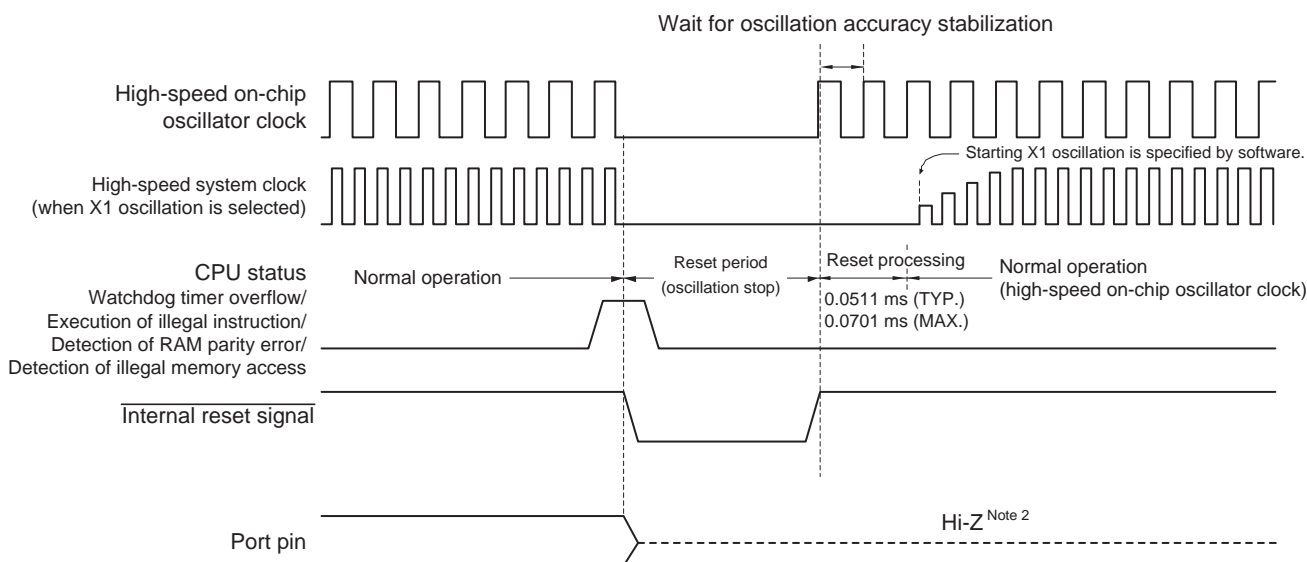


Figure 17-3. Timing of Reset Due to Execution of Illegal Instruction, Watchdog Timer Overflow, RAM Parity Error, or Illegal Memory Access



**Notes 1.** Reset times (times for release from the external reset state)

After the first release of the POR: 0.672 ms (typ.), 0.832 ms (max.) when the LVD is in use.

0.399 ms (typ.), 0.519 ms (max.) when the LVD is off.

After the second release of the POR: 0.531 ms (typ.), 0.675 ms (max.) when the LVD is in use.

0.259 ms (typ.), 0.362 ms (max.) when the LVD is off.

After power is supplied, a voltage stabilization waiting time of about 0.99 ms (typ.) and up to 2.30 ms (max.) is required before reset processing starts after release of the external reset.

2. The state of P40 is as follows.

- High-impedance during the external reset period or reset period by the POR.
- High level during other types of reset or after receiving a reset signal (connected to the on-chip pull-up resistance).

**Caution** A watchdog timer internal reset resets the watchdog timer.

**Remark** For the reset timing of the power-on-reset circuit and voltage detector, see **CHAPTER 18 POWER-ON-RESET CIRCUIT** and **CHAPTER 19 VOLTAGE DETECTOR**.

Table 17-1. Operation Statuses During Reset Period

Item	During Reset Period			
System clock	Clock supply to the CPU is stopped.			
Main system clock	$f_{IH}$	Operation stopped		
	$f_X$	Operation stopped (the X1 and X2 pins are input port mode)		
	$f_{EX}$	Clock input invalid (the pin is input port mode)		
$f_{IL}$	Operation stopped			
CPU				
Code flash memory	Operation stopped			
Data flash memory	Operation stopped			
RAM	Operation stopped			
Port (latch)	P40 becomes high impedance (external reset or POR reset). P40 is pulled-up (resets other than external reset and POR reset). The port pins except for P40 become high impedance.			
Timer array unit	Operation stopped			
Watchdog timer				
Clock output/buzzer output				
A/D converter				
D/A converter				
Serial array unit (SAU)				
Serial interface (IICA)				
DMA controller				
Event link controller (ELC)				
PWM option unit				
Power-on-reset function			Detection operation possible	
Voltage detection function			Operation is possible in the case of an LVD reset and stopped in the case of other types of reset.	
External interrupt			Operation stopped	
CRC operation function	High-speed CRC			
	General-purpose CRC			
RAM parity error detection function				
RAM guard function				
SFR guard function				
Illegal-memory access detection function				

&lt;R&gt;

**Remark**  $f_{IH}$ : High-speed on-chip oscillator clock  
 $f_X$ : X1 oscillation clock  
 $f_{EX}$ : External main system clock  
 $f_{IL}$ : Low-speed on-chip oscillator clock

Table 17-2. Hardware Statuses After Reset Acknowledgment (1/3)

Hardware		After Reset Acknowledgment <sup>Note 1</sup>
Program counter (PC)		The contents of the reset vector table (0000H, 0001H) are set.
Stack pointer (SP)		Undefined
Program status word (PSW)		06H
RAM	Data memory	Undefined
	General-purpose registers	Undefined
Processor mode control register (PMC)		00H
Port registers (P1 to P4, P6, P12, P13) (output latches)		00H
Port mode registers (PM1 to PM4, PM6)		FFH
Port mode control register 1 (PMC1)		FFH
Pull-up resistor option registers (PU1, PU3, PU4)		00H (PU4 is 01H)
Clock operation mode control register (CMC)		00H
Clock operation status control register (CSC)		C0H
System clock control register (CKC)		00H
Oscillation stabilization time counter status register (OSTC)		00H
Oscillation stabilization time select register (OSTS)		07H
Noise filter enable registers 0, 1 (NFEN0, NFEN1)		00H
Peripheral enable register 0 (PER0)		00H
Peripheral enable register 1 (PER1)		00H
High-speed on-chip oscillator frequency select register (HOCODIV)		Undefined
High-speed on-chip oscillator trimming register (HIOTRM)		Undefined <sup>Note 2</sup>
Timer array unit	Timer data registers 00 to 03 (TDR00 to TDR03)	0000H
	Timer mode registers 00 to 03 (TMR00 to TMR03)	0000H
	Timer status registers 00 to 03 (TSR00 to TSR03)	0000H
	Timer input select register 0 (TIS0)	00H
	Timer counter registers 00 to 03 (TCR00 to TCR03)	FFFFH
	Timer channel enable status register 0 (TE0)	0000H
	Timer channel start register 0 (TS0)	0000H
	Timer channel stop register 0 (TT0)	0000H
	Timer clock select register 0 (TPS0)	0000H
	Timer output register 0 (TO0)	0000H
	Timer output enable register 0 (TOE0)	0000H
	Timer output level register 0 (TOL0)	0000H
	Timer output mode register 0 (TOM0)	0000H

- Notes**
1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.
  2. The reset value differs for each chip.

Table 17-2. Hardware Statuses After Reset Acknowledgment (2/3)

Hardware		Status After Reset Acknowledgment <sup>Note 1</sup>
Clock output/buzzer output controller	Clock output select registers 0, 1 (CKS0, CKS1)	00H
Watchdog timer	Enable register (WDTE)	1AH/9AH <sup>Note 2</sup>
<R> A/D converter	12-bit A/D conversion result register (ADCR)	0000H
	8-bit A/D conversion result register (ADCRH)	00H
	Mode registers 0 to 2 (ADM0 to ADM2)	00H
	Conversion result comparison upper limit setting register (ADUL)	FFH
	Conversion result comparison lower limit setting register (ADLL)	00H
	A/D test register (ADTES)	00H
	Analog input channel specification register (ADS)	00H
	A/D port configuration register (ADPC)	00H
D/A converter	D/A conversion value setting registers 0, 1 (DACS0, DACS1)	00H
	D/A converter mode register (DAM)	00H
Serial array unit (SAU)	Serial data registers 00, 01 (SDR00, SDR01)	0000H
	Serial status registers 00, 01 (SSR00, SSR01)	0000H
	Serial flag clear trigger registers 00, 01 (SIR00, SIR01)	0000H
	Serial mode registers 00, 01 (SMR00, SMR01)	0020H
	Serial communication operation setting registers 00, 01 (SCR00, SCR01)	0087H
	Serial channel enable status register 0 (SE0)	0000H
	Serial channel start register 0 (SS0)	0000H
	Serial channel stop register 0 (ST0)	0000H
	Serial clock select register 0 (SPS0)	0000H
	Serial output register 0 (SO0)	0F0FH
	Serial output enable register 0 (SOE0)	0000H
	Serial output level register 0 (SOL0)	0000H
	Serial standby control register 0 (SSC0)	0000H
	Input switch control register (ISC)	00H
Serial interface IICA	IICA shift register 0, 1 (IICA0, IICA1)	00H
	IICA status register 0, 1 (IICS0, IICS1)	00H
	IICA flag register 0, 1 (IICF0, IICF1)	00H
	IICA control register 00, 10 (IICCTL00, IICCTL10)	00H
	IICA control register 01, 11 (IICCTL01, IICCTL11)	00H
	IICA low-level width setting register 0, 1 (IICWL0, IICWL1)	FFH
	IICA high-level width setting register 0, 1 (IICWH0, IICWH1)	FFH
	Slave address register 0, 1 (SVA0, SVA1)	00H

**Notes** 1. During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

2. The reset value of WDTE is determined by the option byte setting.

Table 17-2. Hardware Statuses After Reset Acknowledgment (3/3)

Hardware		Status After Reset Acknowledgment <sup>Note 1</sup>
Reset function	Reset control flag register (RESF)	Undefined <sup>Note 2</sup>
Voltage detector (LVD)	Voltage detection register (LVIM)	00H <sup>Note 2</sup>
	Voltage detection level register (LVIS)	00H/01H/81H <sup>Notes 2, 3</sup>
DMA controller	SFR address registers 0, 1 (DSA0, DSA1)	00H
	RAM address registers 0, 1 (DRA0, DRA1)	0000H
	Byte count registers 0, 1 (DBC0, DBC1)	0000H
	Mode control registers 0, 1 (DMC0, DMC1)	00H
	Operation control registers 0, 1 (DRC0, DRC1)	00H
ELC	Event output select registers 00 to 09 (ELSELR00 to ELSELR09)	00H
Interrupt	Request flag registers 0L, 0H, 1L (IF0L, IF0H, IF1L)	00H
	Mask flag registers 0L, 0H, 1L (MK0L, MK0H, MK1L)	FFH
	Priority specification flag registers 00L, 00H, 01L, 10L, 10H, 11L (PR00L, PR00H, PR01L, PR10L, PR10H, PR11L)	FFH
	External interrupt rising edge enable register 0 (EGP0)	00H
	External interrupt falling edge enable register 0 (EGN0)	00H
Safety functions	Flash memory CRC control register (CRC0CTL)	00H
	Flash memory CRC operation result register (PGCRCL)	0000H
	CRC input register (CCRIN)	00H
	CRC data register (CRCD)	0000H
	Invalid memory access detection control register (IAWCTL)	00H
	RAM parity error control register (RPECTL)	00H
Flash memory	Data flash control register (DFLCTL)	00H
BCD correction circuit	BCD correction result register (BCDAJ)	Undefined

**Notes 1.** During reset signal generation or oscillation stabilization time wait, only the PC contents among the hardware statuses become undefined. All other hardware statuses remain unchanged after reset.

**2.** These values vary depending on the reset source.

Reset Source		RESET Input	Reset by POR	Reset by Execution of Illegal Instruction	Reset by WDT	Reset by RAM Parity Error	Reset by Illegal-memory Access	Reset by LVD	
RESF	TRAP bit	Cleared (0)		Set (1)	Held			Held	
	WDTRF bit			Held	Set (1)	Held			
	RPERF bit			Held		Set (1)	Held		
	IAWRF bit			Held			Set (1)		
	LVIRF bit			Held			Set (1)		
LVIM	LVISEN bit	Cleared (0)						Held	
	LVIOMSK bit	Held							
	LVIF bit								
LVIS		Cleared (00H/01H/81H)							

**3.** The generation of reset signal other than an LVD reset sets as follows.

- When option byte LVIMDS1, LVIMDS0 = 1, 0: 00H
- When option byte LVIMDS1, LVIMDS0 = 1, 1: 81H
- When option byte LVIMDS1, LVIMDS0 = 0, 1: 01H

## 17.2 Register for Confirming Reset Source

### 17.2.1 Reset control flag register (RESF)

Many internal reset generation sources exist in the R7F0C010. The reset control flag register (RESF) is used to store which source has generated the reset request.

The RESF register can be read by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$  input, reset by power-on-reset (POR) circuit, and reading the RESF register clear TRAP, WDTRF, RPERF, IAWRF, and LVIRF flags.

**Figure 17-4. Format of Reset Control Flag Register (RESF)**

Address: FFFA8H After reset: 00H<sup>Note 1</sup> R

Symbol	7	6	5	4	3	2	1	0
RESF	TRAP	0	0	WDTRF	0	RPERF	IAWRF	LVIRF

TRAP	Internal reset request by execution of illegal instruction <sup>Note 2</sup>
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

WDTRF	Internal reset request by watchdog timer (WDT)
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

RPERF	Internal reset request by RAM parity
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

IAWRF	Internal reset request by illegal-memory access
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

LVIRF	Internal reset request by voltage detector (LVD)
0	Internal reset request is not generated, or the RESF register is cleared.
1	Internal reset request is generated.

<R>

**Notes 1.** The value after reset varies depending on the reset source. See **Table 17 - 3**.

**2.** The illegal instruction is generated when instruction code FFH is executed.

Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

**Cautions 1.** Do not read data by a 1-bit memory manipulation instruction.

<R>

**2.** When enabling RAM parity error resets (RPERDIS = 0), be sure to initialize the used RAM area at data access or the used RAM area + 10 bytes at execution of instruction from the RAM area. Reset generation enables RAM parity error resets (RPERDIS = 0). For details, see 20.5 RAM parity error detection function.



The status of the RESF register when a reset request is generated is shown in Table 17-3.

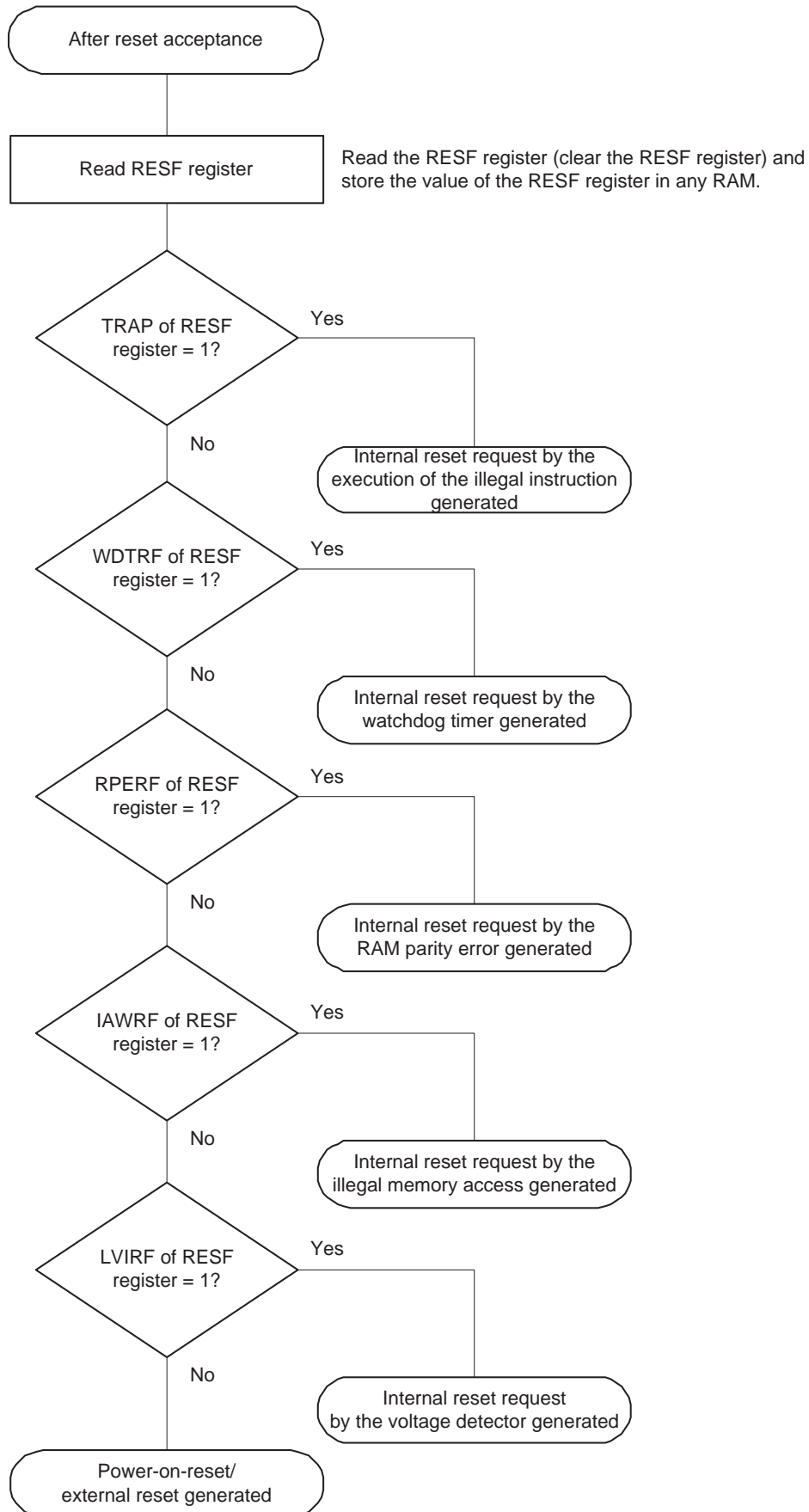
**Table 17-3. RESF Register Status When Reset Request Is Generated**

Reset Source Flag	$\overline{\text{RESET}}$ Input	Reset by POR	Reset by Execution of Illegal Instruction	Reset by WDT	Reset by RAM Parity Error	Reset by Illegal- memory Access	Reset by LVD
TRAP bit	Cleared (0)	Cleared (0)	Set (1)	Held	Held	Held	Held
WDTRF bit			Held	Set (1)			
RPERF bit			Held	Set (1)			
IAWRF bit			Held	Set (1)			
LVIRF bit			Held	Set (1)			

Figure 17 - 5 shows the Procedure for Checking Reset Source.

<R>

Figure 17-5. Procedure for Checking Reset Source



## CHAPTER 18 POWER-ON-RESET CIRCUIT

### 18.1 Functions of Power-on-reset Circuit

The power-on-reset circuit (POR) has the following functions.

- Generates internal reset signal at power on.  
<R> The reset is released when the supply voltage ( $V_{DD}$ ) exceeds the detection voltage ( $V_{POR}$ ). However, keep the reset status using the voltage detection function or external reset pin until the voltage reaches the operating voltage range shown in **27.4 AC Characteristics**.
- Compares supply voltage ( $V_{DD}$ ) and detection voltage ( $V_{PDR}$ ), generates internal reset signal when  $V_{DD} < V_{PDR}$ .  
<R> However, when the operating voltage falls, enter STOP mode, or enable the reset status using the voltage detection function or external reset pin before the voltage falls below the operating voltage range shown in **27.4 AC Characteristics**. When restarting operation, confirm that the supply voltage has returned to the operating voltage range.

**Caution** If an internal reset signal is generated in the POR circuit, TRAP, WDTRF, RPERF, IAWRF, and LVIRF flags of the reset control flag register (RESF) is cleared.

**Remarks 1.** This product incorporates multiple hardware functions that generate an internal reset signal. A flag that indicates the reset source is located in the reset control flag register (RESF) for when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access. The RESF register is not cleared to 00H and the flag is set to 1 when an internal reset signal is generated by the watchdog timer (WDT), voltage-detector (LVD), illegal instruction execution, RAM parity error, or illegal-memory access. For details of the RESF register, see **CHAPTER 17 RESET FUNCTION**.

ADPC: A/D port configuration register

2.  $V_{POR}$ : POR power supply rise detection voltage

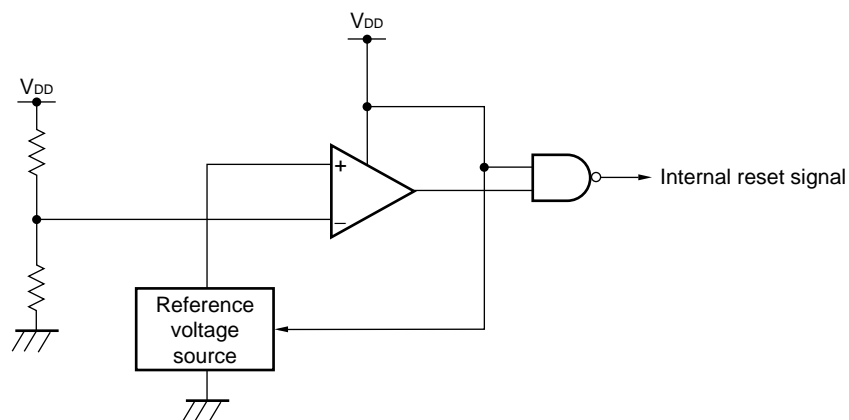
$V_{PDR}$ : POR power supply fall detection voltage

For details, see **27.6.4 POR circuit characteristics**.

## 18.2 Configuration of Power-on-reset Circuit

The block diagram of the power-on-reset circuit is shown in Figure 18-1.

**Figure 18-1. Block Diagram of Power-on-reset Circuit**

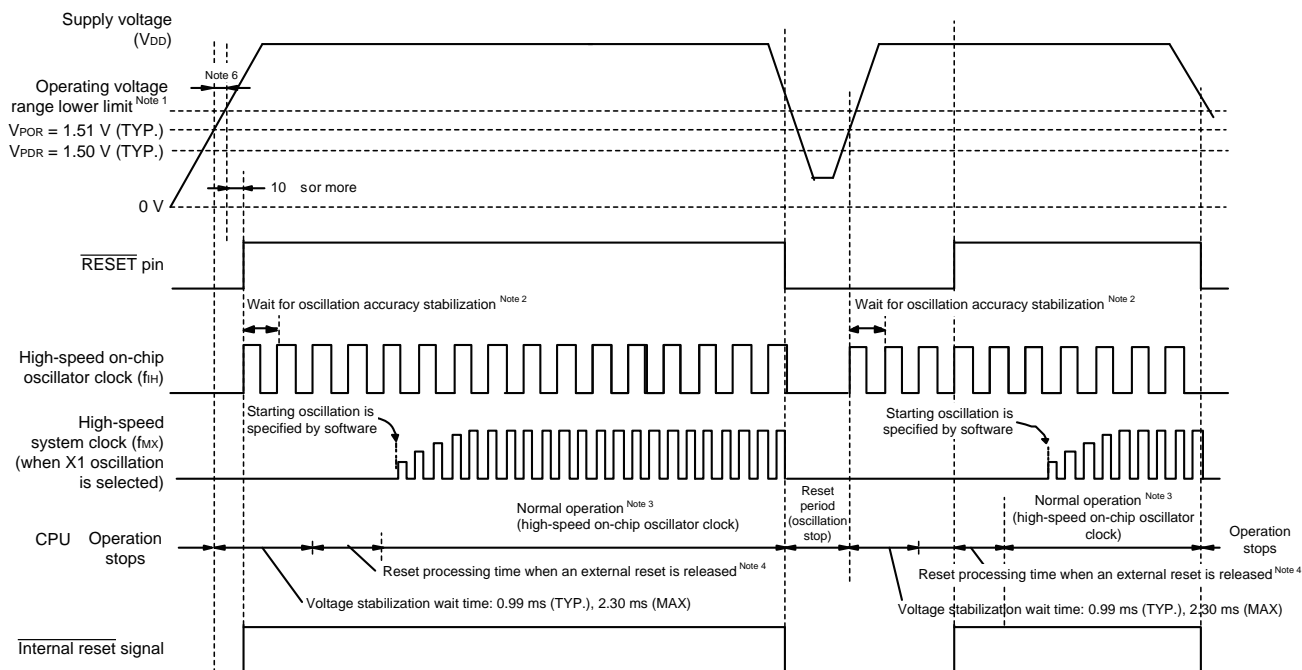


## 18.3 Operation of Power-on-reset Circuit

<R> The timing of generation of the internal reset signal by the power-on-reset circuit and voltage detector is shown below.

**Figure 18-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (1/3)**

**(1) Power-on reset circuit (when LVD is OFF) (option byte 000C1H: VPOC2 = 1)**



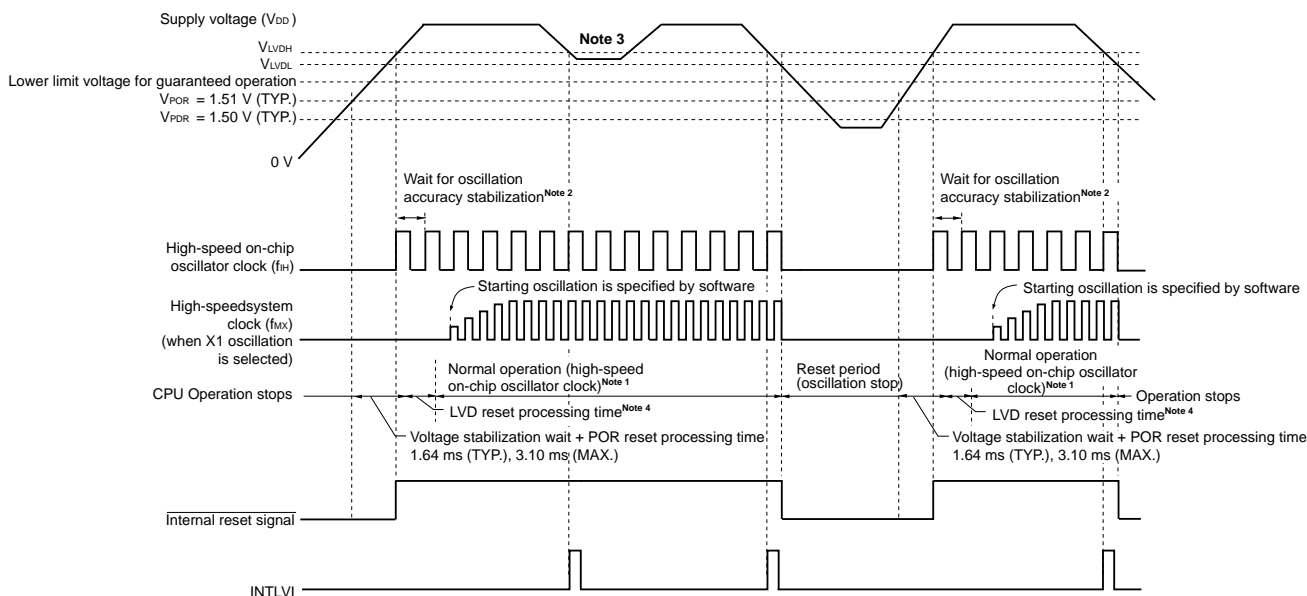
- Notes**
1. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.
  2. The high-speed on-chip oscillator clock and a high-speed system clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time.
  3. The time until normal operation starts includes the following reset processing time when the external reset is released (after the first release of POR) after the RESET signal is driven high (1) as well as the voltage stabilization wait time after V<sub>POR</sub> (1.51 V, TYP.) is reached.  
Reset processing time when the external reset is released is shown below.  
After the first release of POR: 0.672 ms (TYP.), 0.832 ms (MAX.) (when the LVD is in use) 0.399 ms (TYP.), 0.519 ms (MAX.) (when the LVD is off).
  4. Reset processing time when the external reset is released after the second release of POR is shown below.  
After the second release of POR: 0.531 ms (TYP.), 0.675 ms (MAX.) (when the LVD is in use) 0.259 ms (TYP.), 0.362 ms (MAX.) (when the LVD is off).
  5. After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **27.4 AC Characteristics**. This is done by controlling the externally input reset signal. After power supply is turned off, this LSI should be placed in the STOP mode, or in the reset state by utilizing the voltage detection circuit or externally input reset signal, before the voltage falls below the operating range. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Caution** For power-on reset, be sure to use the externally input reset signal on the RESET pin when the LVD is off. For details, see CHAPTER 19 VOLTAGE DETECTOR.

**Remark** V<sub>POR</sub>: POR power supply rise detection voltage  
V<sub>PDR</sub>: POR power supply fall detection voltage

**Figure 18-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (2/3)**

**(2) LVD interrupt & reset mode (option byte 000C1: LVIMDS1, LVIMDS0 = 1, 0)**

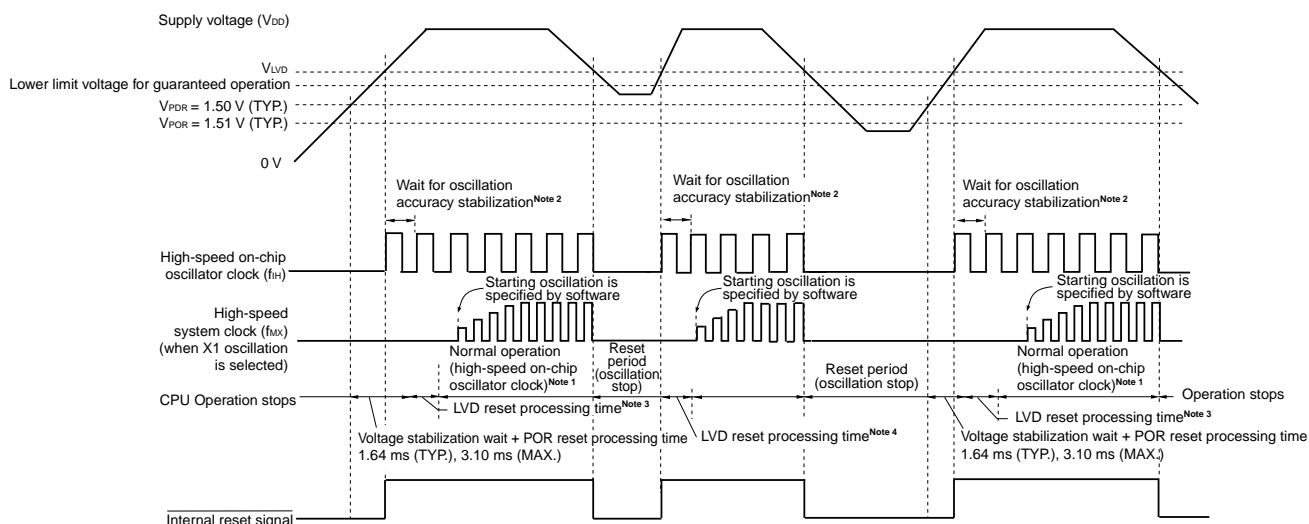


- Notes**
1. The high-speed on-chip oscillator clock and a high-speed system clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time.
  2. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.
  3. After the first interrupt request signal (INTLVI) is generated, the LVIL and LVIMD bits of the voltage detection level register (LVIS) are automatically set to 1. If the operating voltage returns to 2.7 V or higher without falling below the voltage detection level ( $V_{LVDL}$ ), after INTLVI is generated, perform the required backup processing, and then use software to specify the initial settings in order (see **Figure 19-8 Initial Setting of Interrupt and Reset Mode**).
  4. The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ( $V_{LVDH}$ ) is reached as well as the voltage stabilization wait + POR reset processing time after the  $V_{POR}$  (1.51 V, typ.) is reached.  
LVD reset processing time: 0 ms to 0.0701 ms (max.)

**Remark**  $V_{LVDH}$ ,  $V_{LVDL}$ : LVD detection voltage  
 $V_{POR}$ : POR power supply rise detection voltage  
 $V_{PDR}$ : POR power supply fall detection voltage

**Figure 18-2. Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (3/3)**

**(3) When LVD is reset mode (option byte 000C1H: LVIMDS1 = 1, LVIMDS0 = 1)**



- Notes**
1. The high-speed on-chip oscillator clock and a high-speed system clock can be selected as the CPU clock. To use the X1 clock, use the oscillation stabilization time counter status register (OSTC) to confirm the lapse of the oscillation stabilization time.
  2. The internal reset processing time includes the oscillation accuracy stabilization time of the high-speed on-chip oscillator clock.
  3. The time until normal operation starts includes the following LVD reset processing time after the LVD detection level ( $V_{LVD}$ ) is reached as well as the voltage stabilization wait + POR reset processing time after the  $V_{PDR}$  (1.51 V, typ.) is reached.  
LVD reset processing time: 0 ms to 0.0701 ms (max.)
  4. When the power supply voltage is below the lower limit for operation and the power supply voltage is then restored after an internal reset is generated only by the voltage detector (LVD), the following LVD reset processing time is required after the LVD detection level ( $V_{LVD}$ ) is reached.  
LVD reset processing time: 0.0511 ms (typ.), 0.0701 ms (max.)

<R> **Remarks 1.**  $V_{LVD}$ : LVD detection voltage  
 $V_{POR}$ : POR power supply rise detection voltage  
 $V_{PDR}$ : POR power supply fall detection voltage

2. When the LVD interrupt mode is selected (option byte 000C1H: LVIMD1 = 0, LVIMD0 = 1), the time until normal operation starts after power is turned on is the same as the time specified in Note 2 of **Figure 18 - 2 (3)**

### 18.4 Cautions for Power-on-reset Circuit

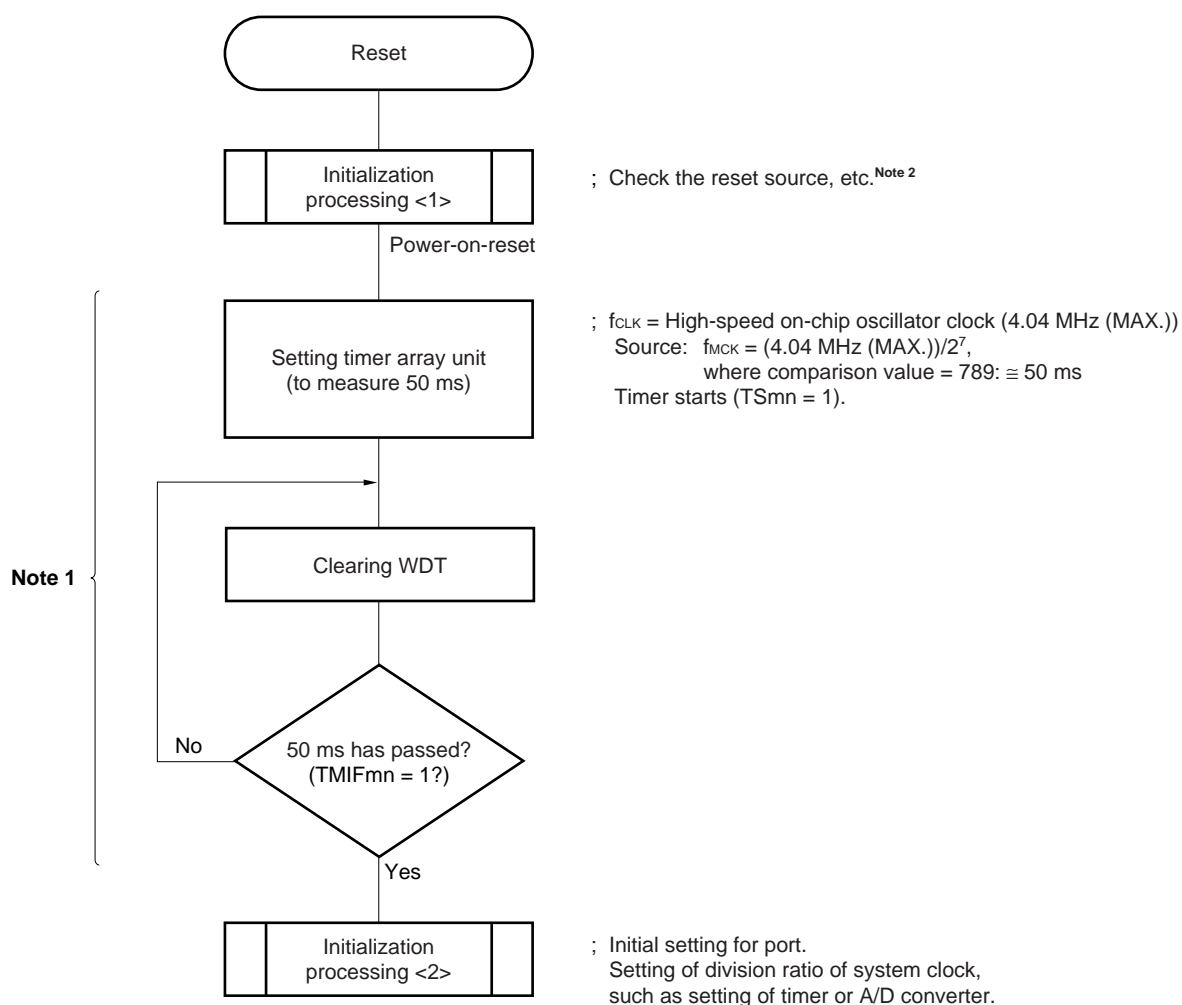
In a system where the supply voltage ( $V_{DD}$ ) fluctuates for a certain period in the vicinity of the POR detection voltage ( $V_{POR}$ ,  $V_{PDR}$ ), the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller can be arbitrarily set by taking the following action.

<Action>

After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

**Figure 18-3. Example of Software Processing After Reset Release (1/2)**

- If supply voltage fluctuation is 50 ms or less in vicinity of POR detection voltage



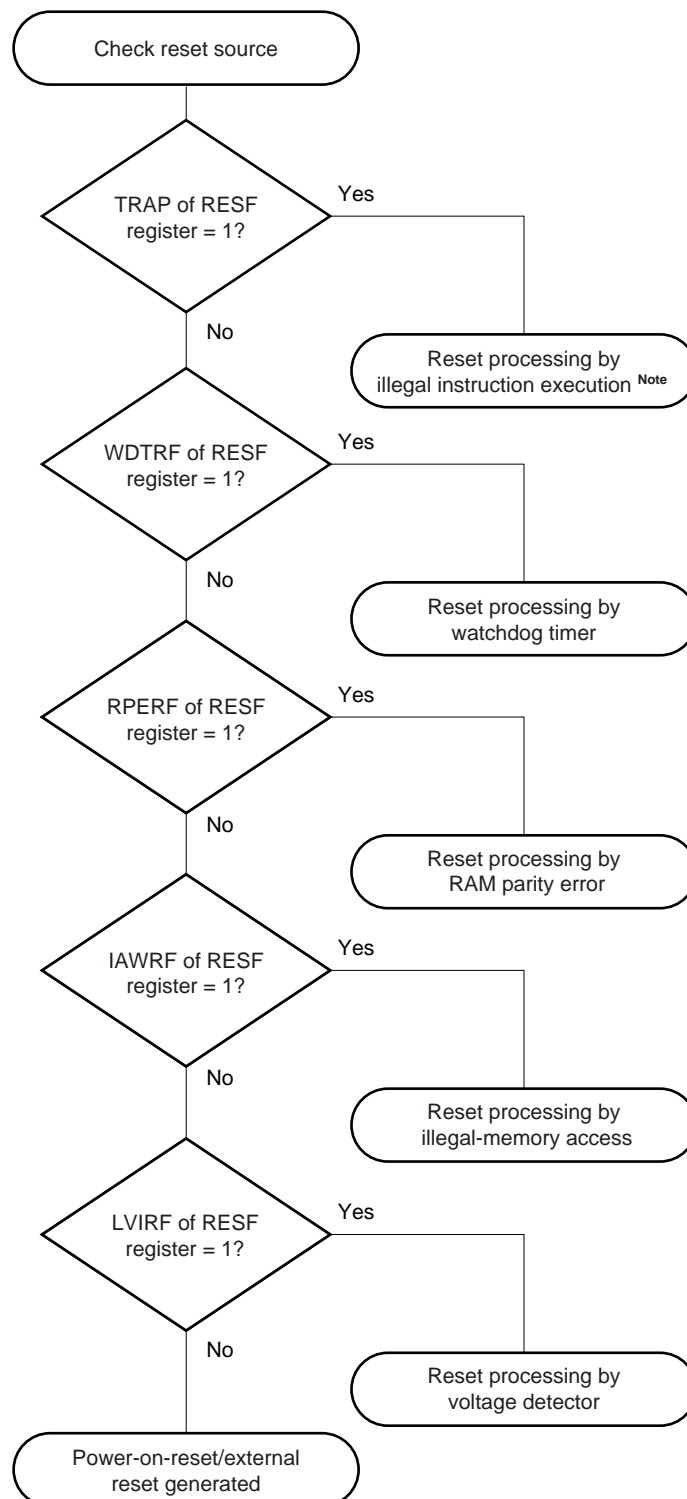
- Notes**
1. If reset is generated again during this period, initialization processing <2> is not started.
  2. A flowchart is shown on the next page.

**Remark** m = 0, n = 0 to 3



Figure 18-3. Example of Software Processing After Reset Release (2/2)

- Checking reset source



**Note** The illegal instruction is generated when instruction code FFH is executed.

Reset by the illegal instruction execution not issued by emulation with the in-circuit emulator or on-chip debug emulator.

## CHAPTER 19 VOLTAGE DETECTOR

## 19.1 Functions of Voltage Detector

The voltage detector (LVD) has the following functions.

- <R> • The LVD circuit compares the supply voltage ( $V_{DD}$ ) with the detection voltage ( $V_{LVDH}$ ,  $V_{LVDL}$ ,  $V_{LVD}$ ), and generates an internal reset or internal interrupt signal.
- The detection level for the power supply detection voltage ( $V_{LVDH}$ ,  $V_{LVDL}$ ) can be selected by using the option byte as one of four levels (for details, see **CHAPTER 22 OPTION BYTE**).
- Operable in STOP mode.
- <R> • After power is supplied, the reset state must be retained until the operating voltage becomes in the range defined in **27.4 AC Characteristics**. This is done by utilizing the voltage detection circuit or controlling the externally input reset signal. After the power supply is turned off, this LSI should be placed in the STOP mode, or placed in the reset state by utilizing the voltage detection circuit or controlling the externally input reset signal before the voltage falls below the operating range. The range of operating voltage varies with the setting of the user option byte (000C2H).
- The following three operation modes can be selected by using the option byte.

**(a) Interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0)**

For the two detection voltages selected by the option byte 000C1H, the high-voltage detection level ( $V_{LVDH}$ ) is used for generating interrupts and ending resets, and the low-voltage detection level ( $V_{LVDL}$ ) is used for triggering resets.

**(b) Reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1)**

The detection voltage ( $V_{LVD}$ ) selected by the option byte 000C1H is used for triggering and ending resets.

**(c) Interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1)**

The detection voltage ( $V_{LVD}$ ) selected by the option byte 000C1H is used for generating interrupts/reset release.

Two detection voltages ( $V_{LVDH}$ ,  $V_{LVDL}$ ) can be specified in the interrupt & reset mode, and one ( $V_{LVD}$ ) can be specified in the reset mode and interrupt mode.

The reset and interrupt signals are generated as follows according to the option byte (LVIMDS0, LVIMDS1) selection.

	Interrupt & Reset Mode (LVIMDS1, LVIMDS0 = 1, 0)	Reset Mode (LVIMDS1, LVIMDS0 = 1, 1)	Interrupt Mode (LVIMDS1, LVIMDS0 = 0, 1)
<R>	Generates an internal interrupt signal when $V_{DD} < V_{LVDH}$ , and an internal reset when $V_{DD} < V_{LVDL}$ . Releases the reset signal when $V_{DD} \geq V_{LVH}$ .	Generates an internal reset signal when $V_{DD} < V_{LVD}$ and releases the reset signal when $V_{DD} \geq V_{LVD}$ .	Releases an internal reset by detecting $V_{DD} \geq V_{LVD}$ at power on after the first release of the POR. Generates an interrupt request signal by detecting $V_{DD} < V_{LVD}$ or $V_{DD} \geq V_{LVD}$ at power on after the second release of the POR.

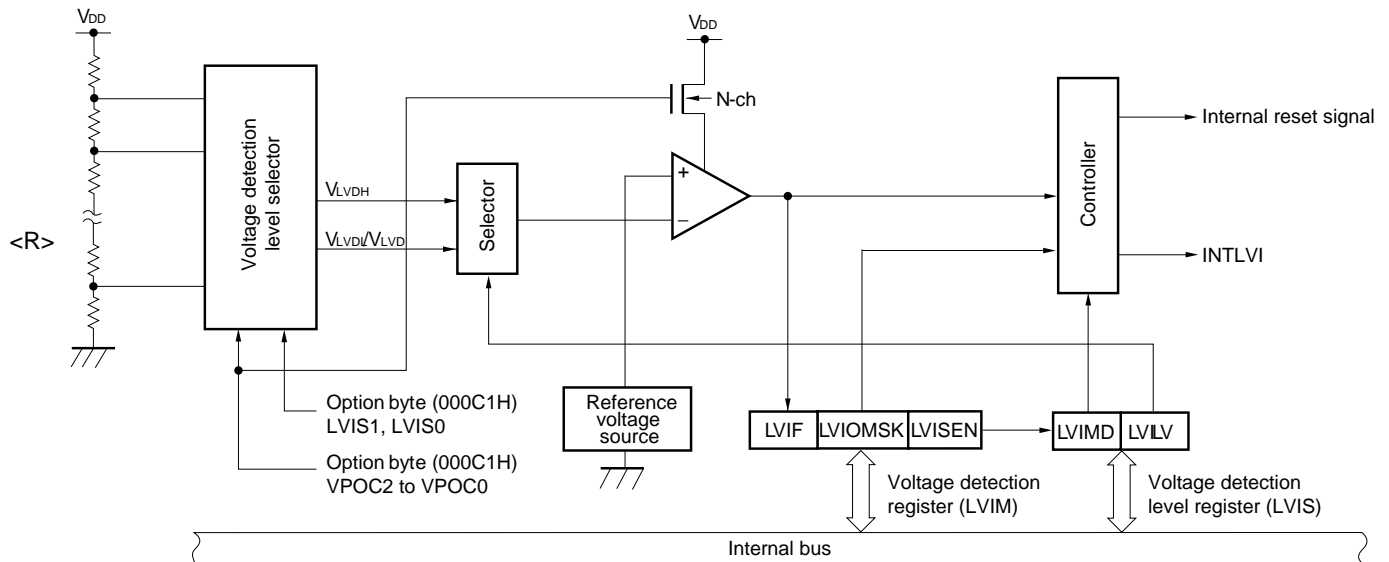
While the voltage detector is operating, whether the supply voltage is more than or less than the detection level can be checked by reading the voltage detection flag (LVIF: bit 0 of the voltage detection register (LVIM)).

Bit 0 (LVIRF) of the reset control flag register (RESF) is set to 1 if reset occurs. For details of the RESF register, see **CHAPTER 17 RESET FUNCTION**.

## 19.2 Configuration of Voltage Detector

The block diagram of the voltage detector is shown in Figure 19-1.

Figure 19-1. Block Diagram of Voltage Detector



## 19.3 Registers Controlling Voltage Detector

The voltage detector is controlled by the following registers.

- Voltage detection register (LVIM)
- Voltage detection level register (LVIS)

### 19.3.1 Voltage detection register (LVIM)

This register is used to specify whether to enable or disable rewriting the voltage detection level register (LVIS), as well as to check the LVD output mask status.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 19-2. Format of Voltage Detection Register (LVIM)**

Address: FFFA9H    After reset: 00H<sup>Note 1</sup>    R/W<sup>Note 2</sup>

Symbol	<7>	6	5	4	3	2	<1>	<0>
LVIM	LVISEN	0	0	0	0	0	LVIOMSK	LVIF
	LVISEN	Specification of whether to enable or disable rewriting the voltage detection level register (LVIS)						
<R>	0	Disabling of rewriting the LVIS register (LVIOMSK = 0 (Mask of LVD output is invalid))						
	1	Enabling of rewriting the LVIS register <sup>Note 3</sup> (LVIOMSK = 1 (Mask of LVD output is valid))						
	LVIOMSK	Mask status flag of LVD output						
<R>	0	Mask of LVD output is invalid						
	1	Mask of LVD output is valid <sup>Notes 3, 4</sup>						
	LVIF	Voltage detection flag						
	0	Supply voltage ( $V_{DD}$ ) $\geq$ detection voltage ( $V_{LVD}$ ), or when LVD operation is disabled						
	1	Supply voltage ( $V_{DD}$ ) $<$ detection voltage ( $V_{LVD}$ )						

**Notes** 1. The reset value changes depending on the reset source.

If the LVIS register is reset by LVD, it is not reset but holds the current value. In other reset, LVISEN is cleared to 0.

2. Bits 0 and 1 are read-only.

<R> 3. LVISEN and LVIOMSK can only be set in the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0).

Do not change the initial value in other modes.

<R> 4. LVIOMSK bit is automatically set to "1" when the interrupt & reset mode is selected (option byte LVIMDS1, LVIMDS0 = 1, 0) and reset or interrupt by LVD is masked.

- Period during LVISEN = 1

- Waiting period from the time when LVD interrupt is generated until LVD detection voltage becomes stable

- Waiting period from the time when the value of LVILV bit changes until LVD detection voltage becomes stable

### 19.3.2 Voltage detection level register (LVIS)

This register selects the voltage detection level.

This register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation input sets this register to 00H/01H/81H<sup>Note 1</sup>.

**Figure 19-3. Format of Voltage Detection Level Select Register (LVIS)**

Address: FFFAAH    After reset: 00H/01H/81H<sup>Note 1</sup>    R/W

Symbol	<7>	6	5	4	3	2	1	<0>
LVIS	LVIMD	0	0	0	0	0	0	LVILV

LVIMD <sup>Note 2</sup>	Operation mode of voltage detection
0	Interrupt mode
1	Reset mode

LVILV <sup>Note 2</sup>	LVD detection level
0	High-voltage detection level (V <sub>LVDH</sub> )
1	Low-voltage detection level (V <sub>LVDL</sub> or V <sub>LVDL</sub> )

- Notes 1.** The reset value changes depending on the reset source and the setting of the option byte. This register is not cleared (00H) by LVD reset. The generation of reset signal other than an LVD reset sets as follows.
- When option byte LVIMDS1, LVIMDS0 = 1, 0: 00H
  - When option byte LVIMDS1, LVIMDS0 = 1, 1: 81H
  - When option byte LVIMDS1, LVIMDS0 = 0, 1: 01H
- 2.** Writing “0” can only be allowed when LVIMDS1 and LVIMDS0 are set to 1 and 0 (interrupt and reset mode) by the option byte. In other cases, writing is not allowed and the value is switched automatically when reset or interrupt is generated.

- Cautions 1.** Only rewrite the value of the LVIS register after setting the LVISEN bit (bit 7 of the LVIM register) to 1.
- 2.** Specify the LVD operation mode and detection voltage (V<sub>LVDH</sub>, V<sub>LVDL</sub>, V<sub>LVLD</sub>) by using the option byte (000C1H). Table 19-1 shows the option byte (000C1H) settings. For details about the option byte, see CHAPTER 22 OPTION BYTE.

<R>

**Table 19-1. LVD Operation Mode and Detection Voltage Settings for User Option Byte (000C1H)**

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt & reset mode)

Detection voltage			Option byte setting value						
V <sub>LVDH</sub>		V <sub>LVDL</sub>	Mode setting		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0
Rising edge	Falling edge	Falling edge	LVIMDS1	LVIMDS0					
2.92 V	2.86 V	2.75 V	1	0	0	1	1	1	0
3.02 V	2.96 V							0	1
Other than above			Setting prohibited						

- LVD setting (reset mode)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0		
Rising edge	Falling edge							LVIMDS1	LVIMDS0
2.81 V	2.75 V	1	1	0	1	1	1	1	
2.92 V	2.86 V				1	1	1	0	
3.02 V	2.96 V				1	1	0	1	
3.13 V	3.06 V				0	1	0	0	
Other than above			Setting prohibited						

- LVD setting (interrupt mode)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0		
Rising edge	Falling edge							LVIMDS1	LVIMDS0
2.81 V	2.75 V	0	1	0	1	1	1	1	
2.92 V	2.86 V				1	1	1	0	
3.02 V	2.96 V				1	1	0	1	
3.13 V	3.06 V				0	1	0	0	
Other than above			Setting prohibited						

- LVD setting (LVDOFF)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting	VPOC2	VPOC1	VPOC0	LVIS1	LVIS0		
Rising edge	Falling edge							LVIMDS1	LVIMDS0
–	–	0/1	1	1	×	×	×	×	
Other than above			Setting prohibited						

<R> **Remark 1.** ×: don't care.

**Remark 2.** The detection voltage is a TYP. value. For details, see 27.6.5 LVD circuit characteristics.

## 19.4 Operation of Voltage Detector

### 19.4.1 When used as reset mode

- When starting operation

Start in the following initial setting state.

Specify the operation mode (the reset mode (LVIMDS1, LVIMDS0 = 1, 1)) and the detection voltage ( $V_{LVD}$ ) by using the option byte 000C1H.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to 0 (disable rewriting of voltage detection level register (LVIS))
- When the option byte LVIMDS1 and LVIMDS0 are set to 1, the initial value of the LVIS register is set to 81H.  
Bit 7 (LVIMD) is 1 (reset mode).  
Bit 0 (LVILV) is 1 (low-voltage detection level:  $V_{LVDL}$  or  $V_{LVD}$ ).

<R>

- Operation in LVD reset mode

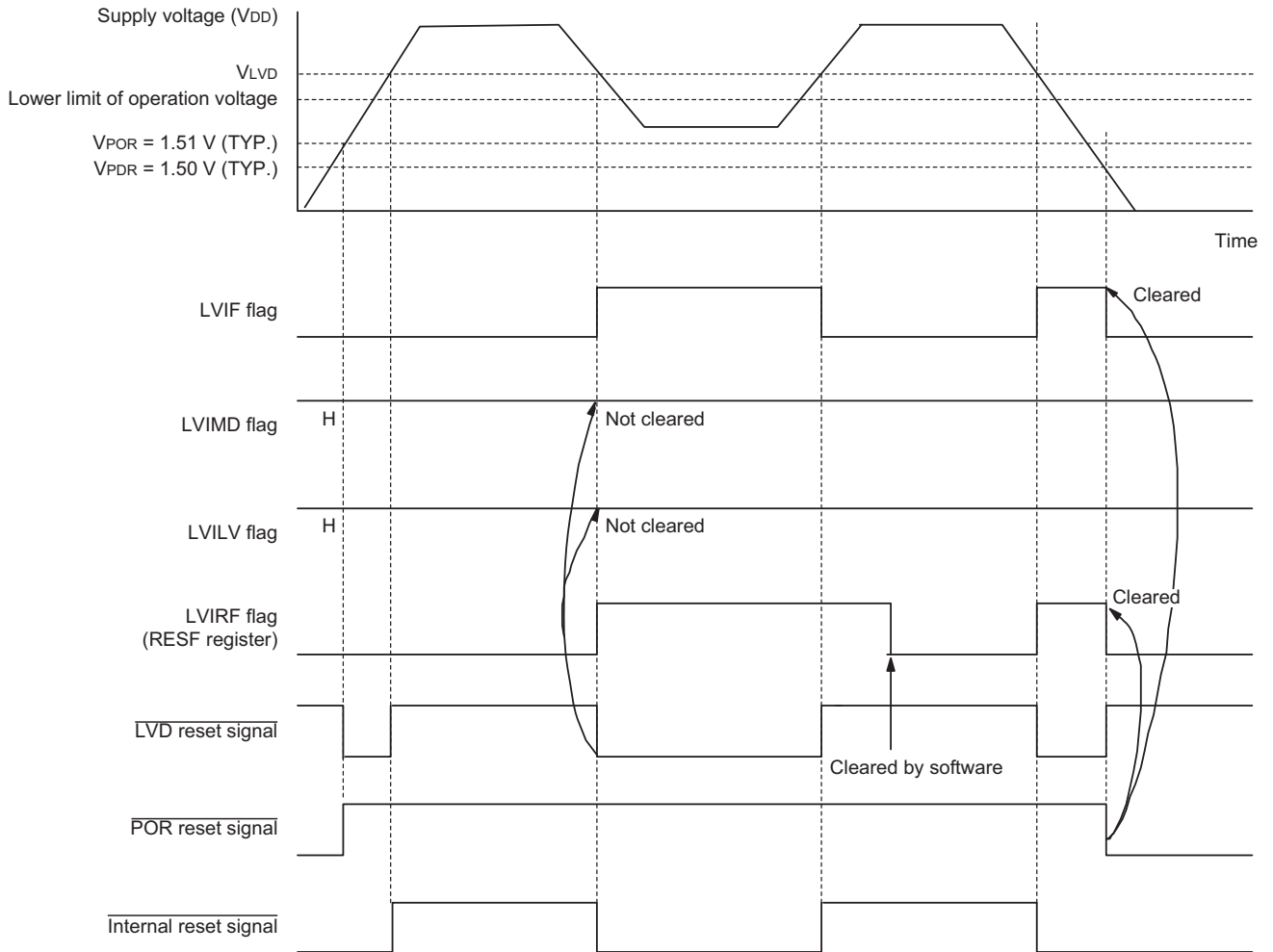
In the reset mode (option byte LVIMDS1, LVIMDS0 = 1, 1), the state of an internal reset by LVD is retained until the supply voltage ( $V_{DD}$ ) exceeds the voltage detection level ( $V_{LVD}$ ) after power is supplied. The internal reset is released when the supply voltage ( $V_{DD}$ ) exceeds the voltage detection level ( $V_{LVD}$ ).

At the fall of the operating voltage, an internal reset by LVD is generated when the supply voltage ( $V_{DD}$ ) falls below the voltage detection level ( $V_{LVD}$ ).

Figure 19-4 shows the timing of the internal reset signal generated by the voltage detector.

<R>

**Figure 19-4. Timing of Voltage Detector Internal Reset Signal Generation  
(Option Byte LVIMDS1, LVIMDS0 = 1, 1)**



**Remark** V<sub>POR</sub>: POR power supply rise detection voltage  
V<sub>PDR</sub>: POR power supply fall detection voltage



### 19.4.2 When used as interrupt mode

- When starting operation

Specify the operation mode (the interrupt mode (LVIMDS1, LVIMDS0 = 0, 1)) and the detection voltage ( $V_{LVD}$ ) by using the option byte 000C1H.

Start in the following initial setting state.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to 0 (disable rewriting of voltage detection level register (LVIS))
- When the option byte LVIMDS1 is clear to 0 and LVIMDS0 is set to 1, the initial value of the LVIS register is set to 00H.
  - Bit 7 (LVIMD) is 0 (interrupt mode).
  - Bit 0 (LVILV) is 1 (low-voltage detection level:  $V_{LVDL}$  or  $V_{LVD}$ ).

<R>

- Operation in LVD interrupt mode

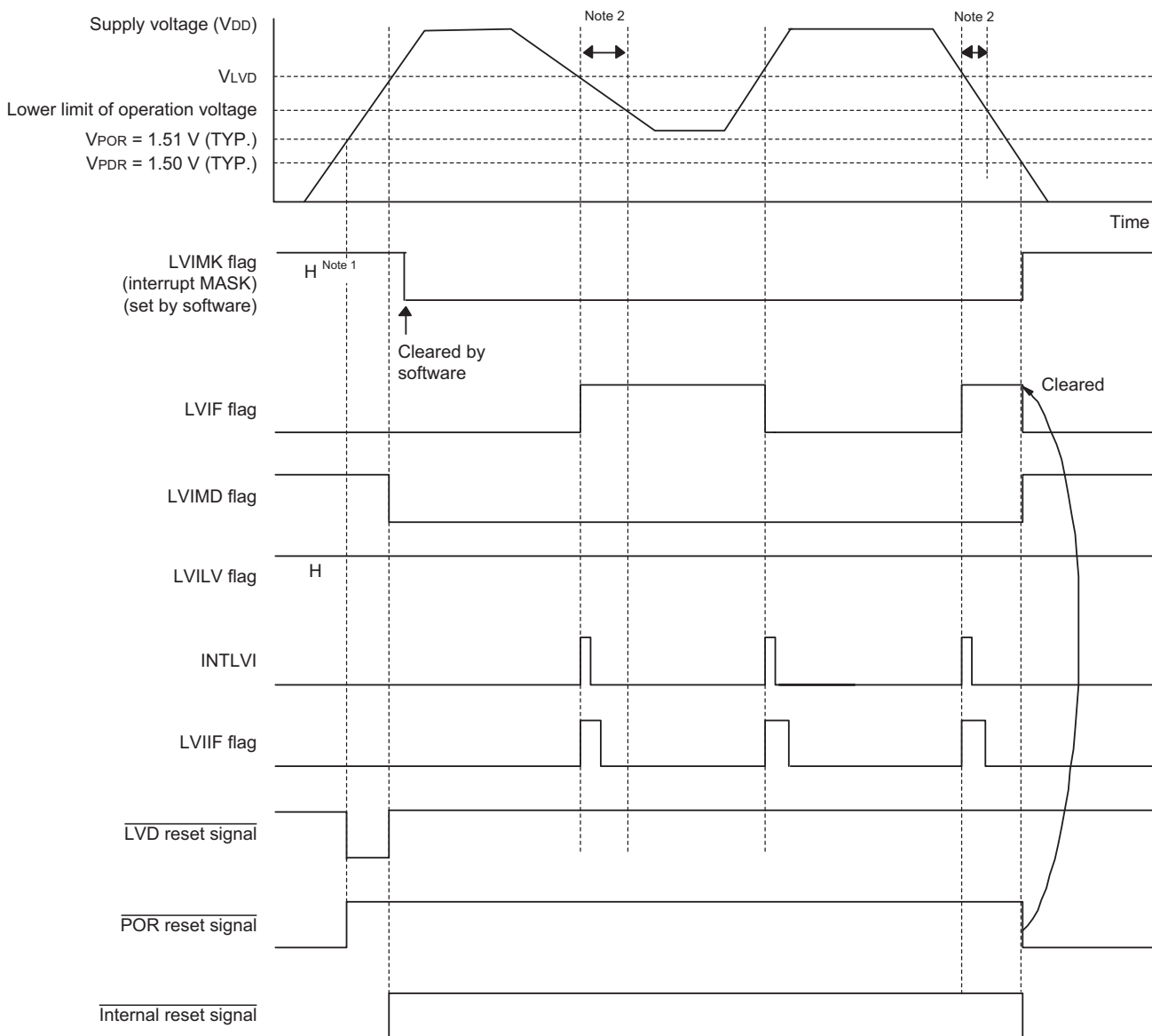
In the interrupt mode (option byte LVIMDS1, LVIMDS0 = 0, 1), the state of an internal reset by LVD is retained until the supply voltage ( $V_{DD}$ ) exceeds the voltage detection level ( $V_{LVD}$ ) after power is supplied (after the first release of the POR). The internal reset is released when the supply voltage ( $V_{DD}$ ) exceeds the voltage detection level ( $V_{LVD}$ ).

An interrupt request signal by LVD (INTLVI) is generated, when the supply voltage ( $V_{DD}$ ) falls below the voltage detection level ( $V_{LVD}$ ) or when the supply voltage ( $V_{DD}$ ) exceeds the voltage detection level ( $V_{LVD}$ ) after the second release of the POR. When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **27.4 AC Characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

Figure 19-5 shows the timing of the internal interrupt signal generated by the voltage detector.

<R>

**Figure 19-5. Timing of Voltage Detector Internal Interrupt Signal Generation  
(Option Byte LVIMDS1, LVIMDS0 = 0, 1)**



- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
  2. When the voltage falls, this LSI should be placed in the STOP mode, or placed in the reset state by controlling the externally input reset signal, before the voltage falls below the operating voltage range defined in **27.4 AC Characteristics**. When restarting the operation, make sure that the operation voltage has returned within the range of operation.

**Remark** V<sub>POR</sub>: POR power supply rise detection voltage  
V<sub>PDR</sub>: POR power supply fall detection voltage

### 19.4.3 When used as interrupt and reset mode

- When starting operation

Specify the operation mode (the interrupt and reset (LVIMDS1, LVIMDS0 = 1, 0)) and the detection voltage ( $V_{LVDH}$ ,  $V_{LVDL}$ ) by using the option byte 000C1H.

Start in the following initial setting state.

- Set bit 7 (LVISEN) of the voltage detection register (LVIM) to 0 (disable rewriting of voltage detection level register (LVIS))
- When the option byte LVIMDS1 is set to 1 and LVIMDS0 is clear to 0, the initial value of the LVIS register is set to 00H.
  - Bit 7 (LVIMD) is 0 (interrupt mode).
  - Bit 0 (LVILV) is 0 (high-voltage detection level:  $V_{LVDH}$ ).

<R>

- Operation in LVD interrupt & reset mode

In the interrupt & reset mode (option byte LVIMDS1, LVIMDS0 = 1, 0), the state of an internal reset by LVD is retained until the supply voltage ( $V_{DD}$ ) exceeds the high-voltage detection level ( $V_{LVDH}$ ) after power is supplied.

The internal reset is released when the supply voltage ( $V_{DD}$ ) exceeds the high-voltage detection level ( $V_{LVDH}$ ).

An interrupt request signal by LVD (INTLVI) is generated and arbitrary save processing is performed when the supply voltage ( $V_{DD}$ ) falls below the high-voltage detection level ( $V_{LVDH}$ ).

After that, an internal reset by LVD is generated when the supply voltage ( $V_{DD}$ ) falls below the low-voltage detection level ( $V_{LVDL}$ ).

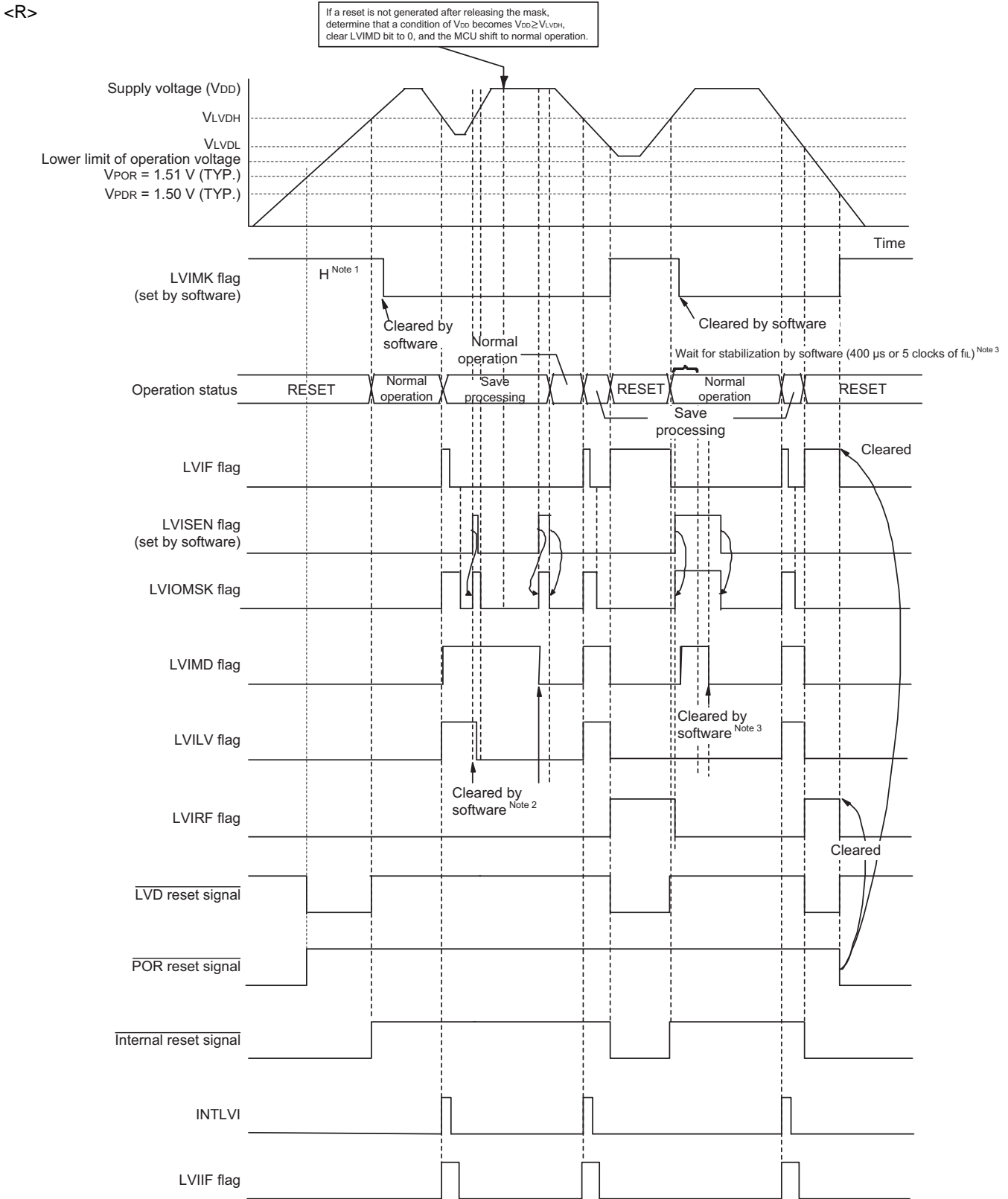
After INTLVI is generated, an interrupt request signal is not generated even if the supply voltage becomes equal to or higher than

the high-voltage detection voltage ( $V_{LVDH}$ ) without falling below the low-voltage detection voltage ( $V_{LVDL}$ ).

To use the LVD reset & interrupt mode, perform the processing according to **Figure 19 - 7 Setting Procedure for Initial Setting of Interrupt and Reset Mode** and **Figure 19 - 8 Example of Software Processing If Supply Voltage Fluctuation is 50  $\mu$ s or Less in Vicinity of LVD Detection Voltage**.

Figure 19-6 shows the timing of the internal reset signal and interrupt signal generated by the voltage detector.

**Figure 19-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (1/2)**



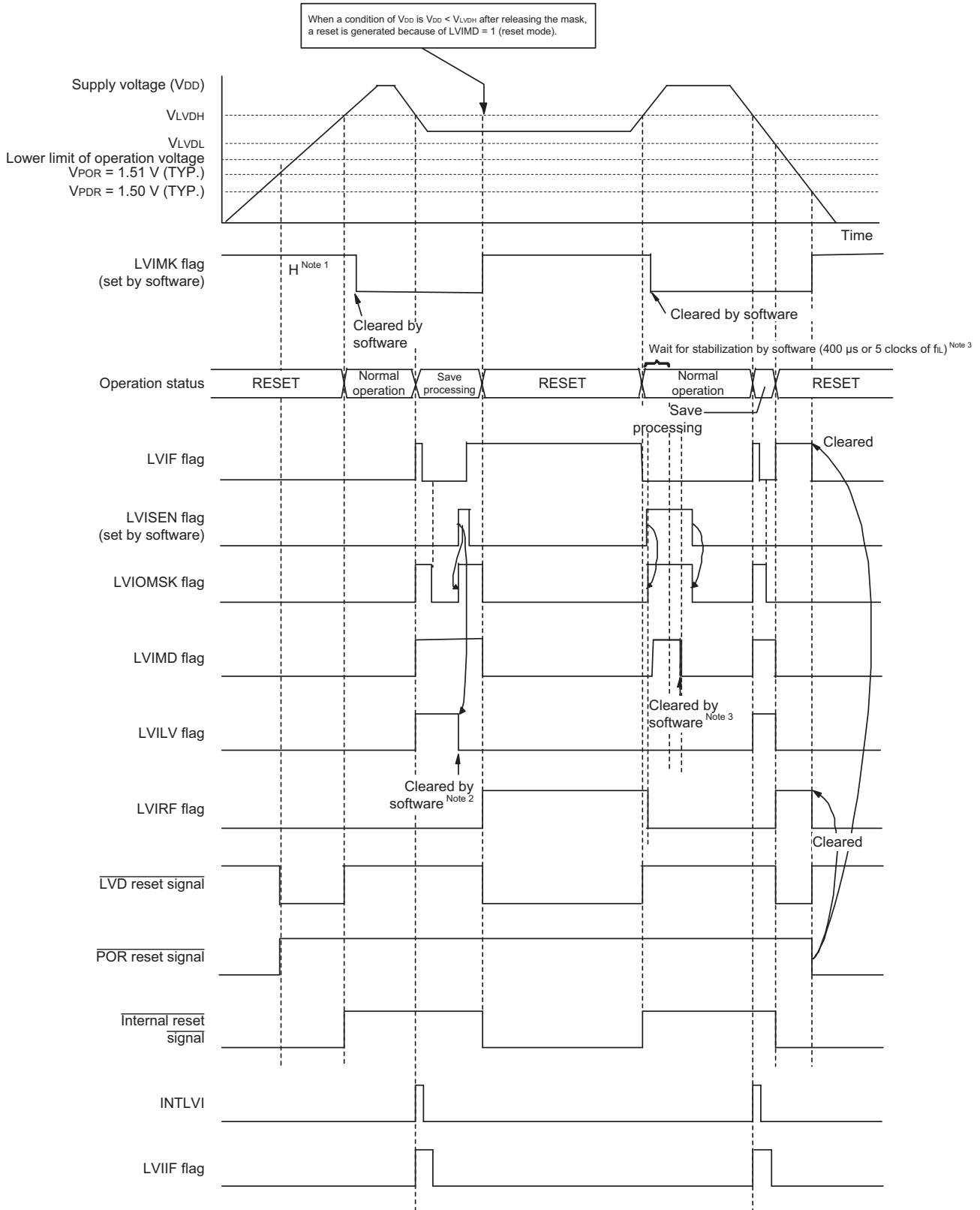
(Notes and Remark are listed on the next page.)

- <R> **Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
  2. After an interrupt is generated, perform the processing according to **Figure 19-7 Setting Procedure for Operating Voltage Check and Reset** in interrupt and reset mode.
  3. After a reset is released, perform the processing according to **Figure 19-8 Setting Procedure for Initial Setting of Interrupt and Reset Mode in interrupt and reset mode**.

**Remark**  $V_{POR}$ : POR power supply rise detection voltage  
 $V_{PDR}$ : POR power supply fall detection voltage

<R>

Figure 19-6. Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0) (2/2)

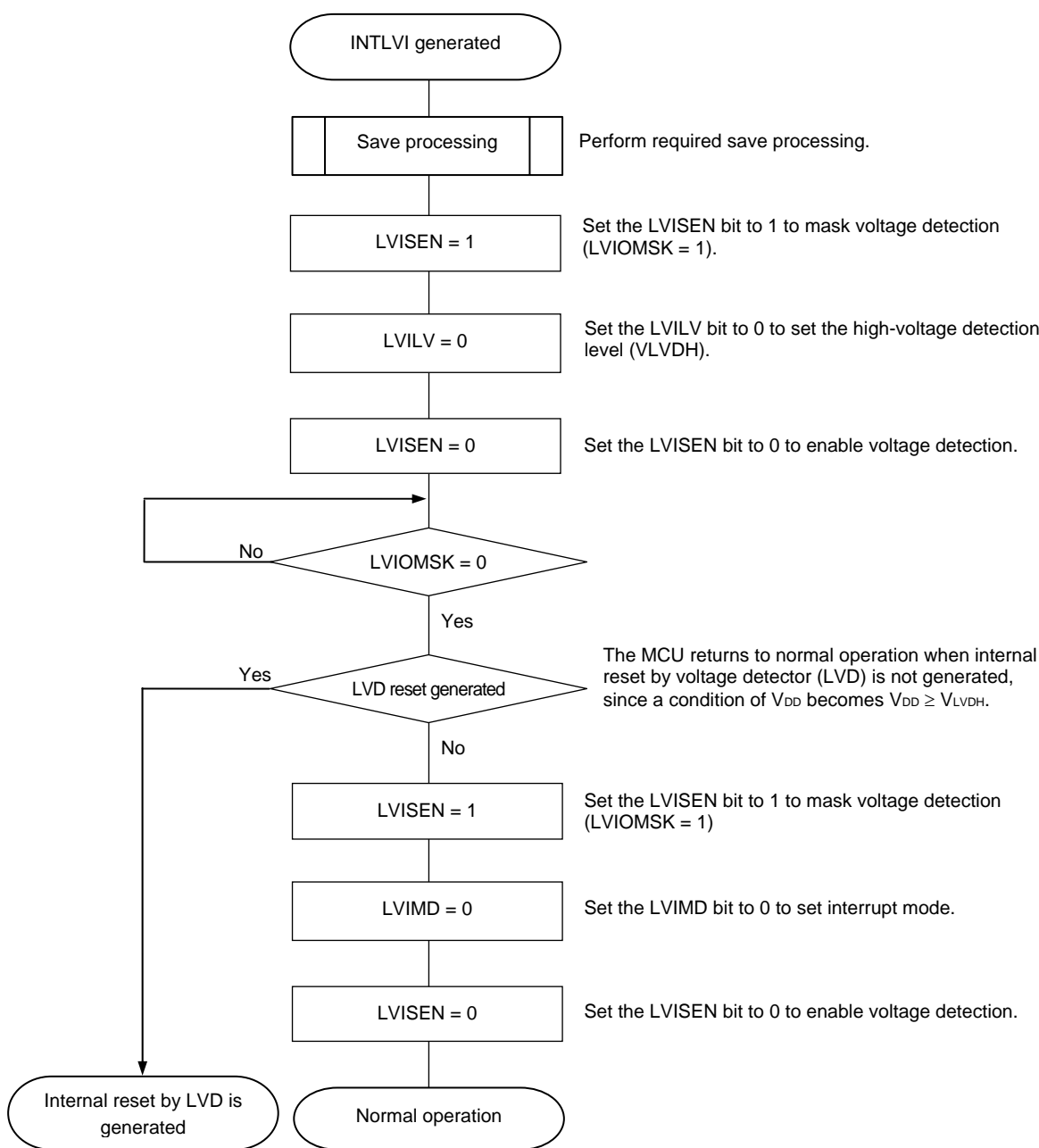


(Notes and Remark are listed on the next page.)

- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
  2. After an interrupt is generated, perform the processing according to **Figure 19-7 Setting Procedure for Operating Voltage Check and Reset** in interrupt and reset mode.
  3. After a reset is released, perform the processing according to **Figure 19-8 Setting Procedure for Initial Setting of Interrupt and Reset Mode** in interrupt and reset mode.

**Remark**  $V_{POR}$ : POR power supply rise detection voltage  
 $V_{PDR}$ : POR power supply fall detection voltage

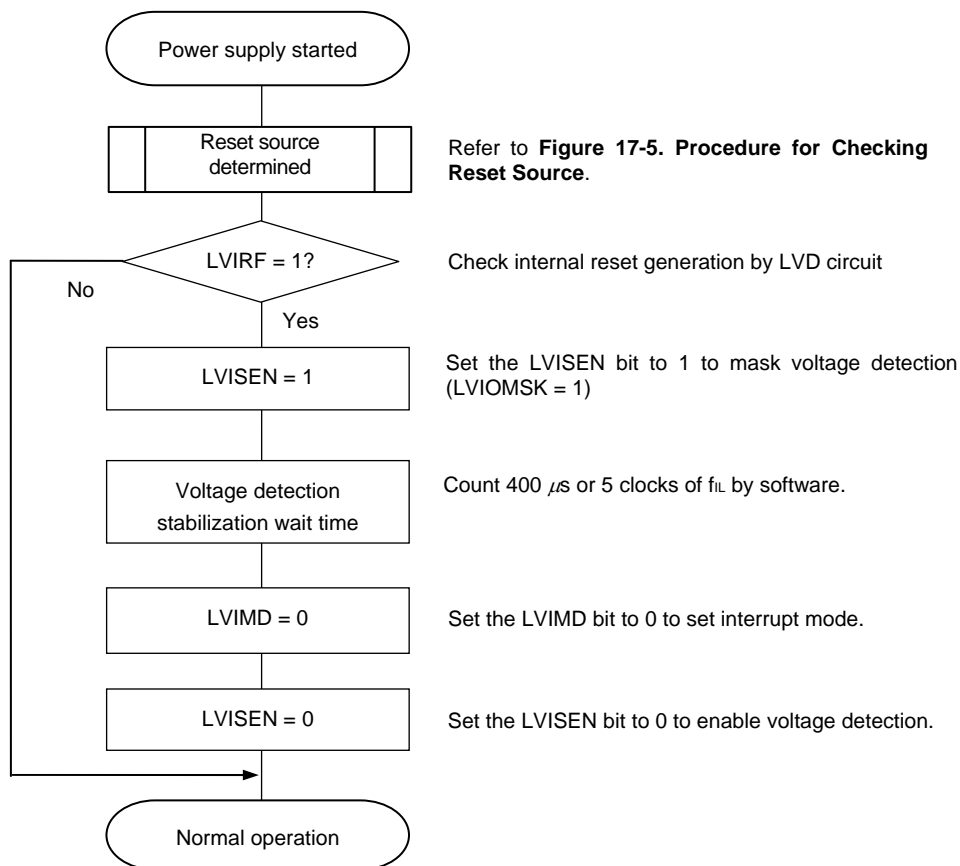
**Figure 19-7. Setting Procedure for Operating Voltage Check and Reset**



When setting an interrupt and reset mode (LVIMDS1, LVIMDS0 = 1, 0), voltage detection stabilization wait time for 400  $\mu$ s or 5 clocks of  $f_{IL}$  is necessary after LVD reset is released (LVIRF = 1). After waiting until voltage detection stabilizes, (0) clear the LVIMD bit for initialization. While voltage detection stabilization wait time is being counted and when the LVIMD bit is rewritten, set LVISEN to 1 to mask a reset or interrupt generation by LVD.

<R> Figure 19-8 shows the procedure for Setting Procedure for Initial Setting of Interrupt and Reset Mode.

<R> **Figure 19-8. Setting Procedure for Initial Setting of Interrupt and Reset Mode**



**Remark**  $f_{IL}$ : Low-speed on-chip oscillator clock frequency



19.5 Cautions for Voltage Detector

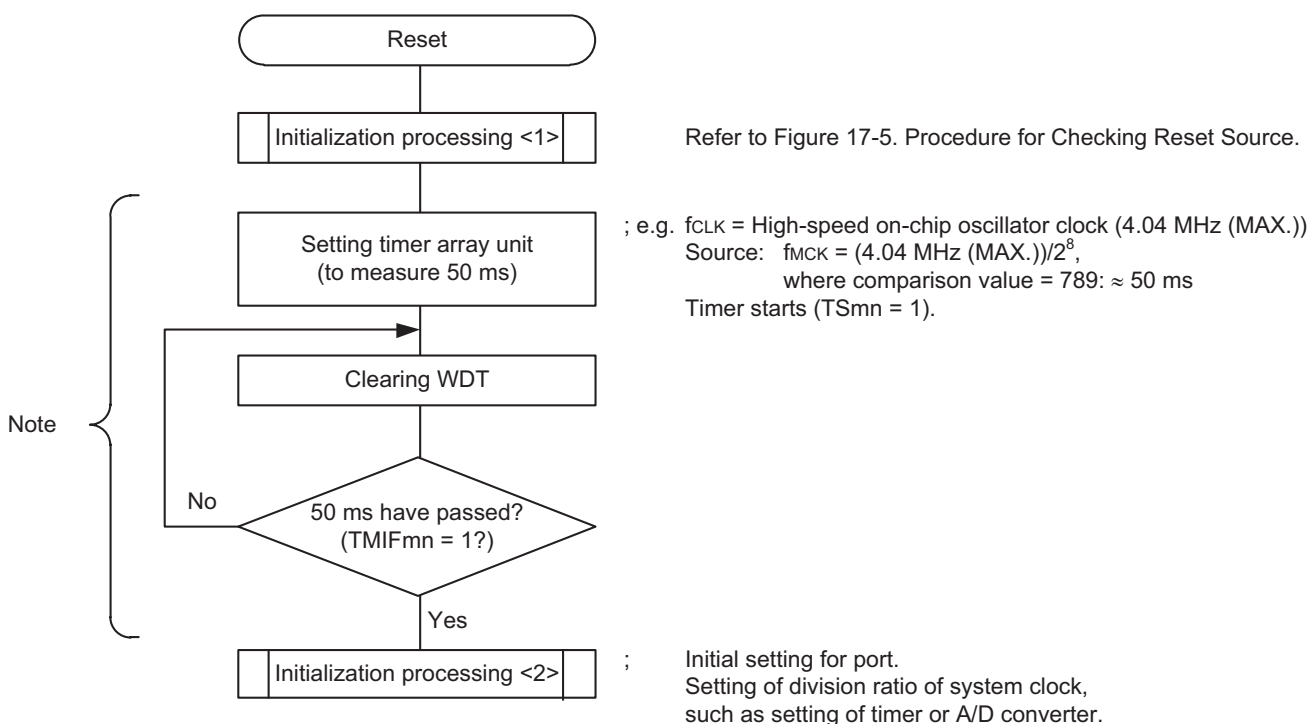
<R> (1) Voltage fluctuation when power is supplied

In a system where the supply voltage ( $V_{DD}$ ) fluctuates for a certain period in the vicinity of the LVD detection voltage, the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller can be arbitrarily set by taking the following action.

<Action>

After releasing the reset signal, wait for the supply voltage fluctuation period of each system by means of a software counter that uses a timer, and then initialize the ports.

Figure 19-9. Example of Software Processing If Supply Voltage Fluctuation is 50  $\mu$ s or Less in Vicinity of LVD Detection Voltage



**Note** If reset is generated again during this period, initialization processing <2> is not started.

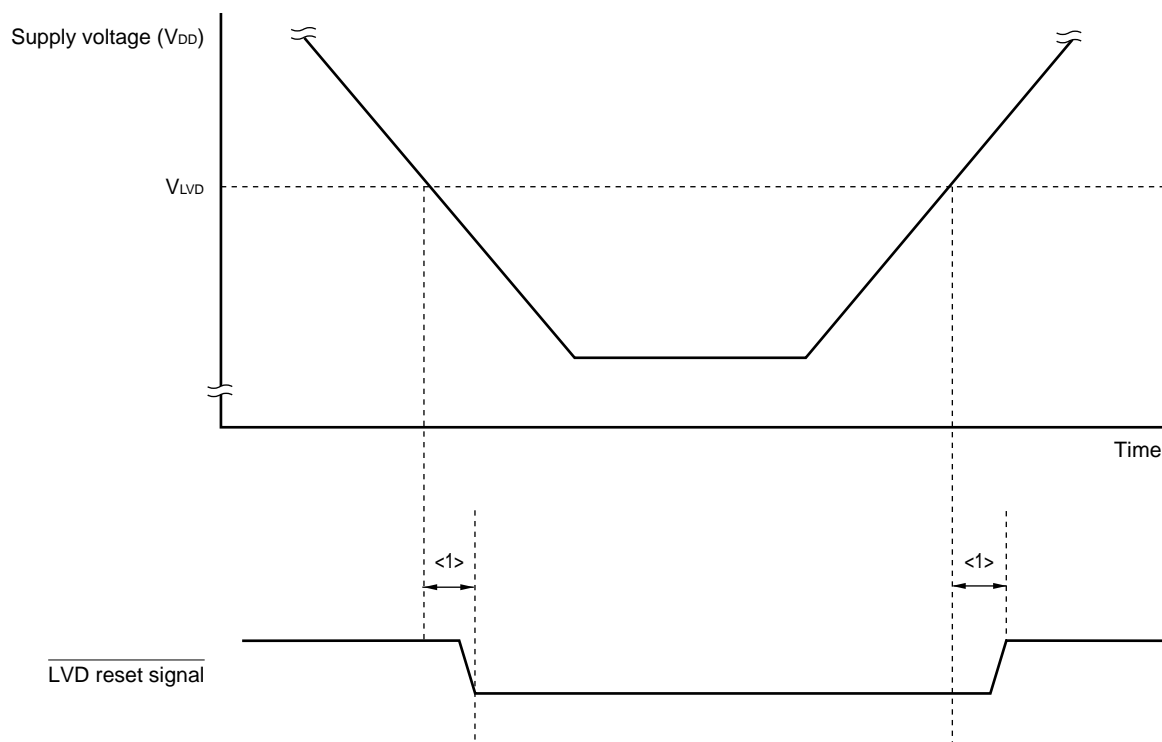
**Remark** Remark m = 0  
 n = 0 to 3

**(2) Delay from the time LVD reset source is generated until the time LVD reset has been generated or released**

There is some delay from the time supply voltage ( $V_{DD}$ ) < LVD detection voltage ( $V_{LVD}$ ) until the time LVD reset has been generated.

In the same way, there is also some delay from the time LVD detection voltage ( $V_{LVD}$ )  $\leq$  supply voltage ( $V_{DD}$ ) until the time LVD reset has been released (see **Figure 19-10**).

**Figure 19-10. Delay from the time LVD reset source is generated until the time LVD reset has been generated or released**



<1>: Detection delay (300  $\mu$ s (MAX.))

- <R> **(3)** However, when the operating voltage falls while interrupt mode is set, enter STOP mode, or enable the reset status using the external reset pin before the voltage falls below the operating voltage range shown in **27.4 AC Characteristics**. When restarting operation, confirm that the supply voltage has returned to the operating voltage range.
- (4)** When the LVD is off, it is necessary to perform an external reset. For an external reset, input a low level for 10  $\mu$ s or more to the  $\overline{\text{RESET}}$  pin. To perform an external reset when power is applied, input a low level to the  $\overline{\text{RESET}}$  pin before power-on, keep the low level for at least 10  $\mu$ s during the period in which the supply voltage is within the operating voltage range shown in 27.4 AC Characteristics, and then input a high level. After power is applied, do not input a high level to the  $\overline{\text{RESET}}$  pin during a period in which the supply voltage is not within the operating range shown in **27.4 AC Characteristics**.

## CHAPTER 20 SAFETY FUNCTIONS

### 20.1 Overview of Safety Functions

The following safety functions are provided in the R7F0C010 to comply with the IEC60730 and IEC61508 safety standards.

These functions enable the microcontroller to self-diagnose abnormalities and stop operating if an abnormality is detected.

#### (1) Flash memory CRC operation function (high-speed CRC, general-purpose CRC)

This detects data errors in the flash memory by performing CRC operations.

Two CRC functions are provided in the R7F0C01072DNP and R7F0C010B2DFP-C that can be used according to the application or purpose of use.

- High-speed CRC: The CPU can be stopped and a high-speed check executed on its entire code flash memory area during the initialization routine.
- General CRC: This can be used for checking various data in addition to the code flash memory area while the CPU is running.

#### (2) RAM parity error detection function

This detects parity errors when reading RAM data.

#### (3) RAM guard function

This prevents RAM data from being rewritten when the CPU freezes.

#### (4) SFR guard function

This prevents SFRs from being rewritten when the CPU freezes.

#### (5) Invalid memory access detection function

This detects illegal accesses to invalid memory areas (such as areas where no memory is allocated and areas to which access is restricted).

#### (6) Frequency detection function

<R> This uses the timer array unit to perform a self-check of the CPU/peripheral hardware clock frequency.

#### (7) A/D test function

<R> This is used to perform a self-check of A/D converter by performing A/D conversion on the positive internal reference voltage, negative reference voltage, analog input channel (ANI), temperature sensor output, and internal reference voltage output.

**Remark** Refer to the application note (R01AN0749) for the features required to comply with the IEC60730 standards.

## 20.2 Registers Used by Safety Functions

The safety functions use the following registers for each function.

Register	Each Function of Safety Function
<ul style="list-style-type: none"> <li>Flash memory CRC control register (CRC0CTL)</li> <li>Flash memory CRC operation result register (PGCRCL)</li> </ul>	Flash memory CRC operation function (high-speed CRC)
<ul style="list-style-type: none"> <li>CRC input register (CRCIN)</li> <li>CRC data register (CRCD)</li> </ul>	CRC operation function (general-purpose CRC)
<ul style="list-style-type: none"> <li>RAM parity error control register (RPECTL)</li> </ul>	RAM parity error detection function
<ul style="list-style-type: none"> <li>Invalid memory access detection control register (IAWCTL)</li> </ul>	RAM guard function
	SFR guard function
	Invalid memory access detection function
<ul style="list-style-type: none"> <li>Timer input select register 0 (TIS0)</li> </ul>	Frequency detection function
<ul style="list-style-type: none"> <li>A/D test register (ADTES)</li> </ul>	A/D test function

The content of each register is described in **20.3 Operation of Flash memory CRC operation function (high-speed CRC)**.

### 20.3 Operation of Flash memory CRC operation function (high-speed CRC)

The IEC60730 standard mandates the checking of data in the flash memory, and recommends using CRC to do it. The high-speed CRC provided in the R7F0C010 can be used to check the entire code flash memory area during the initialization routine. The high-speed CRC can be executed only when the program is allocated on the RAM and in the HALT mode of the main system clock.

<R> The high-speed CRC performs an operation by reading 32-bit data per clock from the flash memory while stopping the CPU. This function therefore can finish a check in a shorter time (for example, 171  $\mu$ s@24 MHz with 16 KB flash memory). The CRC generator polynomial used complies with " $X^{16} + X^{12} + X^5 + 1$ " of CRC-16-CCITT. The high-speed CRC operates in MSB first order from bit 31 to bit 0.

**Caution** The CRC operation result might differ during on-chip debugging because the monitor program is allocated.

**Remark** The operation result is different between the high-speed CRC and the general CRC, because the general CRC operates in LSB first order.

### 20.3.1 Flash memory CRC control register (CRC0CTL)

This register is used to control the operation of the high-speed CRC ALU, as well as to specify the operation range. The CRC0CTL register can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

**Figure 20-1. Format of Flash Memory CRC Control Register (CRC0CTL)**

Address: F02F0H After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	0
CRC0CTL	CRC0EN	0	FEA5	FEA4	FEA3	FEA2	FEA1	FEA0
CRC0EN	Control of CRC ALU operation							
0	Stop the operation.							
1	Start the operation according to HALT instruction execution.							
FEA5	FEA4	FEA3	FEA2	FEA1	FEA0	High-speed CRC operation range		
0	0	0	0	0	0	0000H to 3FFBH (16 K to 4 bytes)		
Other than above						Setting prohibited		

**Remark** Input the expected CRC operation result value to be used for comparison in the lowest 4 bytes of the flash memory. Note that the operation range will thereby be reduced by 4 bytes.

### 20.3.2 Flash memory CRC operation result register (PGCRCL)

This register is used to store the high-speed CRC operation results. The PGCRCL register can be set by a 16-bit memory manipulation instruction. Reset signal generation clears this register to 0000H.

**Figure 20-2. Format of Flash Memory CRC Operation Result Register (PGCRCL)**

Address: F02F2H After reset: 0000H R/W

Symbol	15	14	13	12	11	10	9	8
PGCRCL	PGCRC15	PGCRC14	PGCRC13	PGCRC12	PGCRC11	PGCRC10	PGCRC9	PGCRC8
	7	6	5	4	3	2	1	0
	PGCRC7	PGCRC6	PGCRC5	PGCRC4	PGCRC3	PGCRC2	PGCRC1	PGCRC0
PGCRC15 to PGCRC0	High-speed CRC operation results							
0000H to FFFFH	Store the high-speed CRC operation results.							

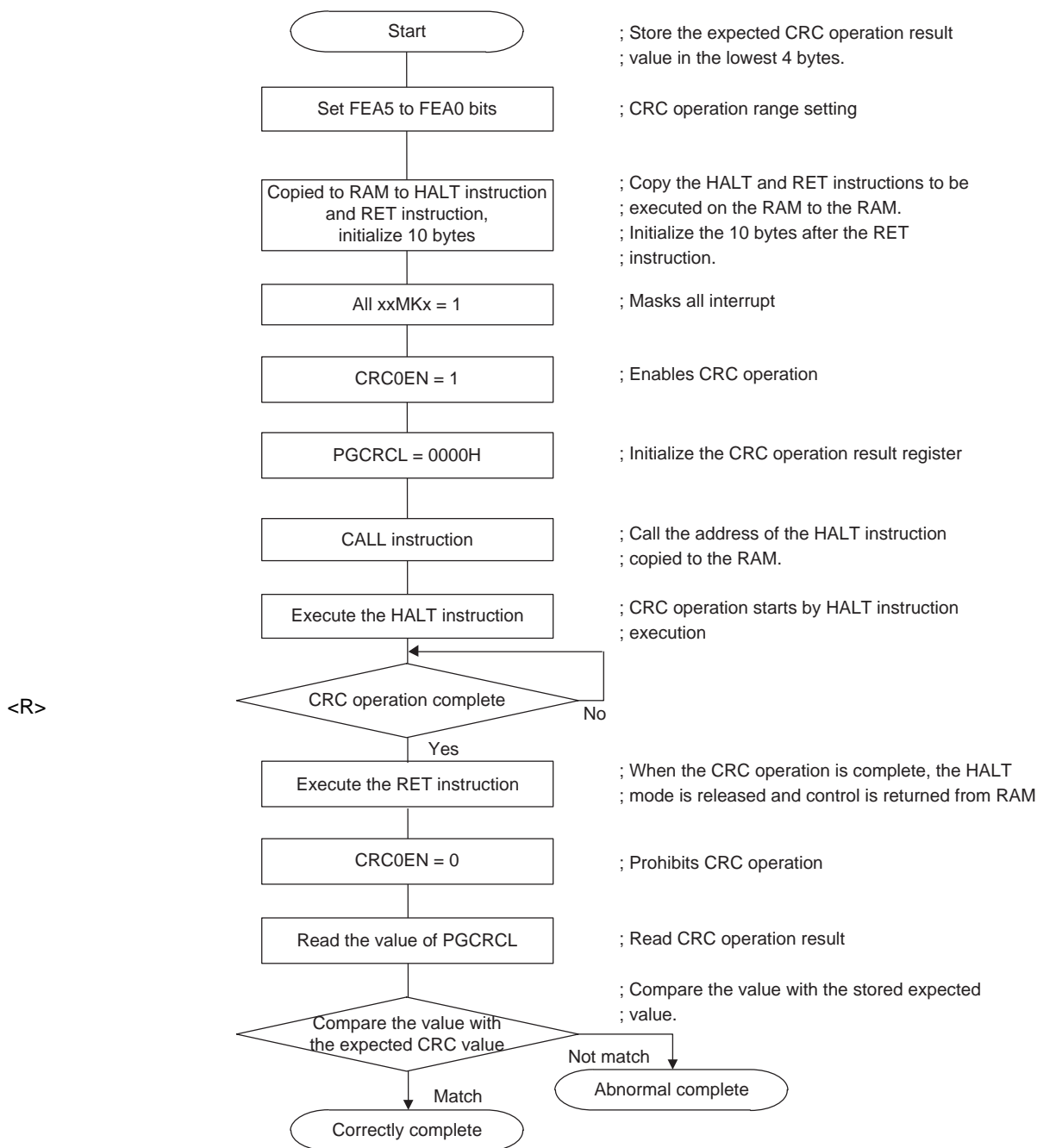
**Caution** The PGCRCL register can only be written if CRC0EN (bit 7 of the CRC0CTL register) = 1.

Figure 20-3 shows the flowchart of flash memory CRC operation function (high-speed CRC).

20.3.3 Operation flow

Figure 20-3 shows the Flowchart of Flash Memory CRC Operation Function (High-speed CRC).

Figure 20-3. Flowchart of Flash Memory CRC Operation Function (High-speed CRC)



- Cautions**
1. The CRC operation is executed only on the code flash.
  2. Store the expected CRC operation value in the area below the operation range in the code flash.
  3. The CRC operation is enabled by executing the HALT instruction in the RAM area.  
Be sure to execute the HALT instruction in RAM area.

The expected CRC operation value can be calculated by using the integrated development environment.

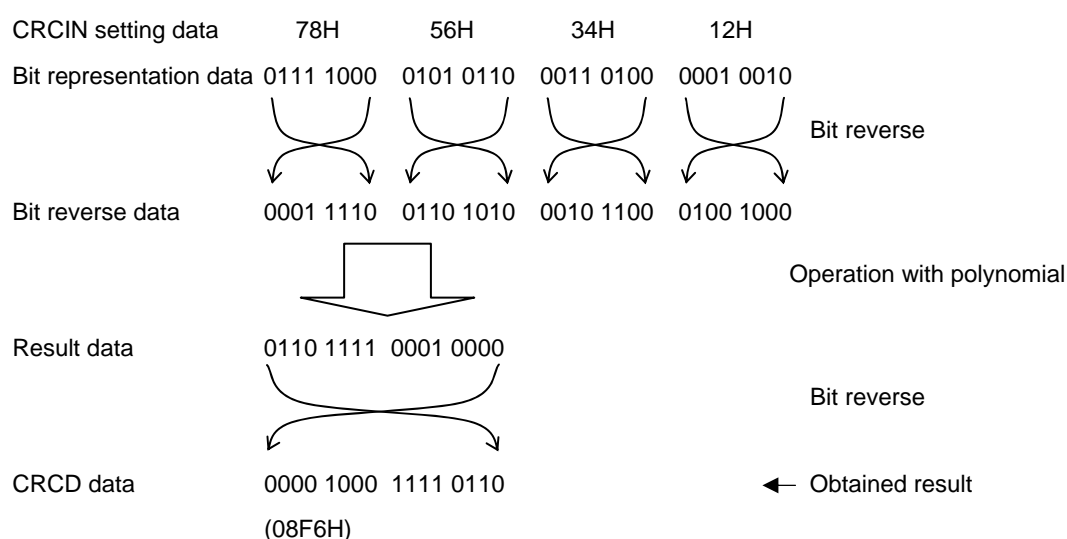
CubeSuite+ development environment. Refer to the CubeSuite+ integrated development environment user's manual for details.

### 20.4 CRC operation function (general-purpose CRC)

In order to guarantee safety during operation, the IEC61508 standard mandates the checking of data even while the CPU is operating.

In the R7F0C01072DNP and R7F0C010B2DFP-C, a general CRC operation can be executed as a peripheral function while the CPU is operating. The general CRC can be used for checking various data in addition to the code flash memory area. The data to be checked can be specified by using software (a user-created program). CRC calculation function in the HALT mode can be used only during the DMA transmission.

The CRC generator polynomial used is “ $X^{16} + X^{12} + X^5 + 1$ ” of CRC-16-CCITT. The data to be input is inverted in bit order and then calculated to allow for LSB-first communication. For example, if the data 12345678H is sent from the LSB, values are written to the CRCIN register in the order of 78H, 56H, 34H, and 12H, enabling a value of 08F6H to be obtained from the CRCD register. This is the result obtained by executing a CRC operation on the bit rows shown below, which consist of the data 12345678H inverted in bit order.



**Caution** Because the debugger rewrites the software break setting line to a break instruction during program execution, the CRC operation result differs if a software break is set in the CRC operation target area.

#### 20.4.1 CRC input register (CRCIN)

CRCIN register is an 8-bit register that is used to set the CRC operation data of general-purpose CRC.

The possible setting range is 00H to FFH.

The CRCIN register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 20-4. Format of CRC Input Register (CRCIN)

Address: FFFACH After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
CRCIN								
	Bits 7 to 0			Function				
	00H to FFH			Data input.				

**20.4.2 CRC data register (CRCD)**

This register is used to store the CRC operation result of the general-purpose CRC.

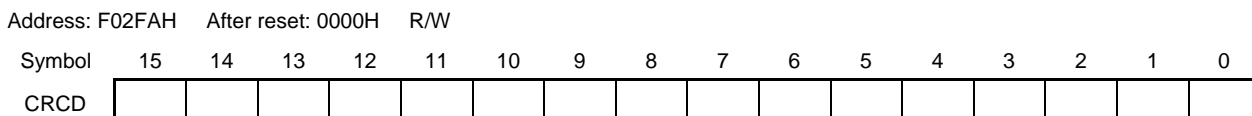
The setting range is 0000H to FFFFH.

After 1 clock of CPU/peripheral hardware clock ( $f_{CLK}$ ) has elapsed from the time CRCIN register is written, the CRC operation result is stored to the CRCD register.

The CRCD register can be set by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

**Figure 20-5. Format of CRC Data Register (CRCD)**

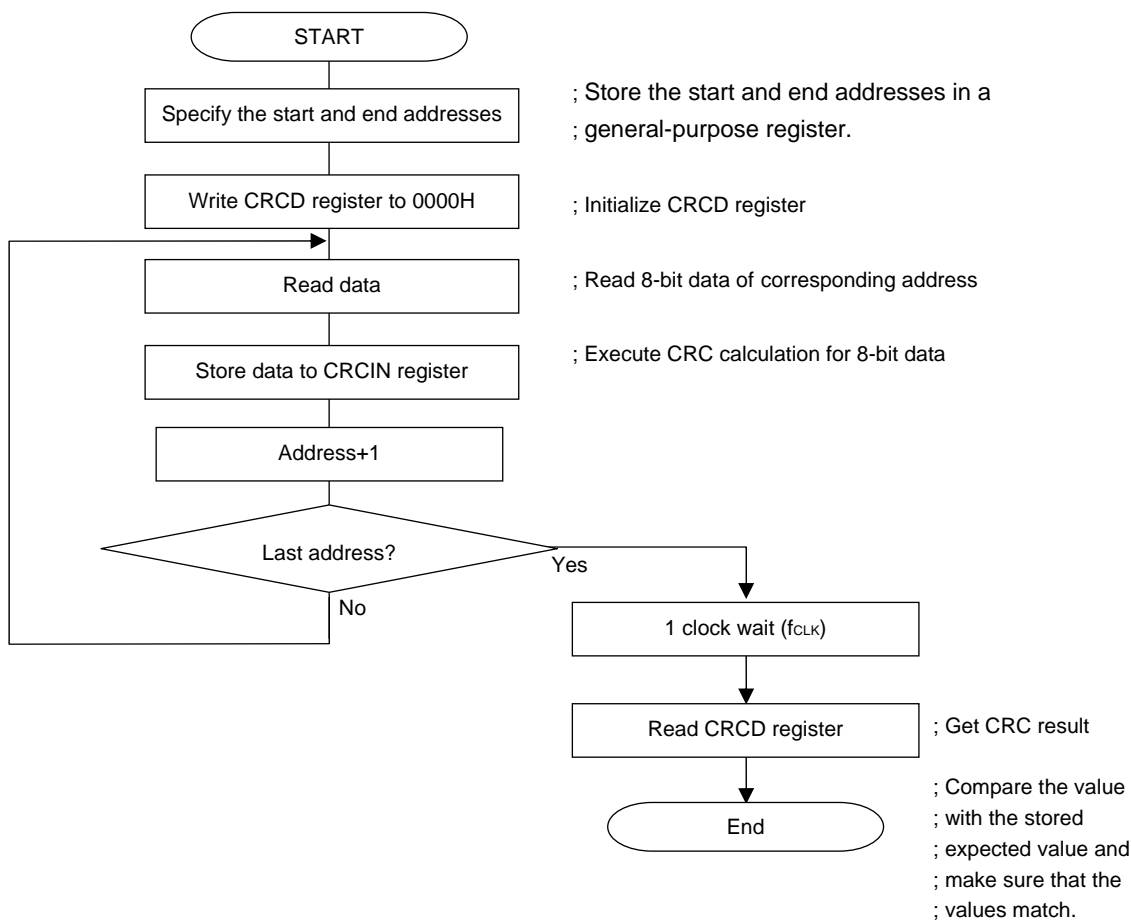


- Cautions**
1. Read the value written to CRCD register before writing to CRCIN register.
  2. If conflict between writing and storing operation result to CRCD register occurs, the writing is ignored.

**20.4.3 Operation flow**

Figure 20-6 shows the CRC Operation Function (General-Purpose CRC).

**Figure 20-6. CRC Operation Function (General-Purpose CRC)**





## 20.5 RAM parity error detection function

The IEC60730 standard mandates the checking of RAM data. A single-bit parity bit is therefore added to all 8-bit data in the RAM of the R7F0C010. By using this RAM parity error detection function, the parity bit is appended when data is written, and the parity is checked when the data is read. This function can also be used to trigger a reset when a parity error occurs.

### 20.5.1 RAM parity error control register (RPECTL)

This register is used to control parity error generation check bit and reset generation due to parity errors.

The RPECTL register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-7. Format of RAM Parity Error Control Register (RPECTL)**

Address: F00F5H After reset: 00H R/W

Symbol	<7>	6	5	4	3	2	1	<0>
RPECTL	RPERDIS	0	0	0	0	0	0	RPEF

RPERDIS	Parity error reset mask flag
0	Enable parity error resets.
1	Disable parity error resets.

RPEF	Parity error status flag
0	No parity error has occurred.
1	A parity error has occurred.

**Caution** The parity bit is appended when data is written, and the parity is checked when the data is read. Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize RAM areas where data access is to proceed before reading data.

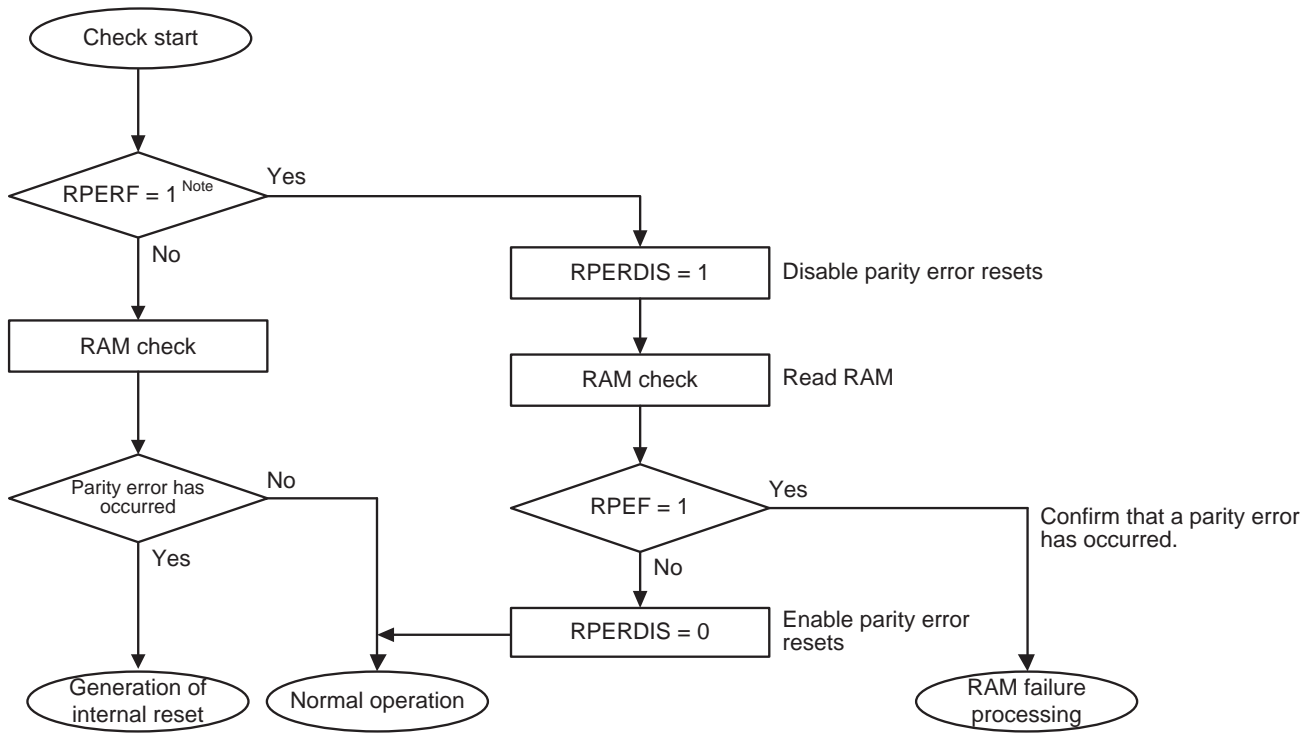
The RL78's CPU executes look-ahead due to the pipeline operation, the CPU might read an uninitialized RAM area that is allocated beyond the RAM used, which causes a RAM parity error.

Therefore, while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize the RAM area + 10 bytes when instructions are fetched from RAM areas. When using the self-programming function while RAM parity error resets are enabled (RPERDIS = 0), be sure to initialize the RAM area to overwrite + 10 bytes before overwriting.

- Remarks**
1. The RAM parity check is always on, and the result can be confirmed by checking the RPEF flag.
  2. The parity error reset is enabled by default (RPERDIS = 0).  
Even if the parity error reset is disabled (RPERDIS = 1), the RPEF flag will be set (1) if a parity error occurs.
  3. The RPEF flag is set (1) by RAM parity errors and cleared (0) by writing 0 to it or by any reset source.  
When RPEF = 1, the value is retained even if RAM for which no parity error has occurred is read.

<R>

Figure 20-8. RAM Parity Error Check Flow



**Note** See CHAPTER 17 RESET FUNCTION for details on how to confirm internal resets due to RAM parity errors.

## 20.6 RAM guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important data stored in the RAM be protected, even if the CPU freezes.

This RAM guard function is used to protect data in the specified memory space.

If the RAM guard function is specified, writing to the specified RAM space is disabled, but reading from the space can be carried out as usual.

### 20.6.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GRAM1 and GRAM0 bits are used in RAM guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-9. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

GRAM1	GRAM0	RAM guard space
0	0	Disabled. RAM can be written to.
0	1	The 128 bytes starting at the lower RAM address
1	0	The 256 bytes starting at the lower RAM address
1	1	The 512 bytes starting at the lower RAM address

## 20.7 SFR guard function

In order to guarantee safety during operation, the IEC61508 standard mandates that important SFRs be protected from being overwritten, even if the CPU freezes.

This SFR guard function is used to protect data in the control registers used by the port function, interrupt function, clock control function, voltage detection function, and RAM parity error detection function.

If the SFR guard function is specified, writing to the specified SFRs is disabled, but reading from the SFRs can be carried out as usual.

### 20.7.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

GPORT, GINT and GCSC bits are used in SFR guard function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-10. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

GPORT	Control registers of port function guard
0	Disabled. Control registers of port function can be read or written to.
1	Enabled. Writing to control registers of port function is disabled. Reading is enabled. [Guarded SFR] PMxx, PUxx, PMC1 <sup>Note 1</sup> , ADPC <sup>Note 2</sup>

GINT	Registers of interrupt function guard
0	Disabled. Registers of interrupt function can be read or written to.
1	Enabled. Writing to registers of interrupt function is disabled. Reading is enabled. [Guarded SFR] IFxx, MKxx, PRxx, EGP0, EGN0

GCSC <sup>Note 3</sup>	Control registers of clock control function, voltage detector and RAM parity error detection function guard
0	Disabled. Control registers of clock control function, voltage detector and RAM parity error detection function can be read or written to.
1	Enabled. Writing to control registers of clock control function, voltage detector and RAM parity error detection function is disabled. Reading is enabled. [Guarded SFR] CMC, CSC, OSTs, CKC, PERx, LVIM, LVIS, RPECTL

- Notes**
1. 24-pin products only
  2. Pxx (Port register) is not guarded.
  3. Clear GCSC bit to 0, during self programming /serial programming.

### 20.8 Invalid memory access detection function

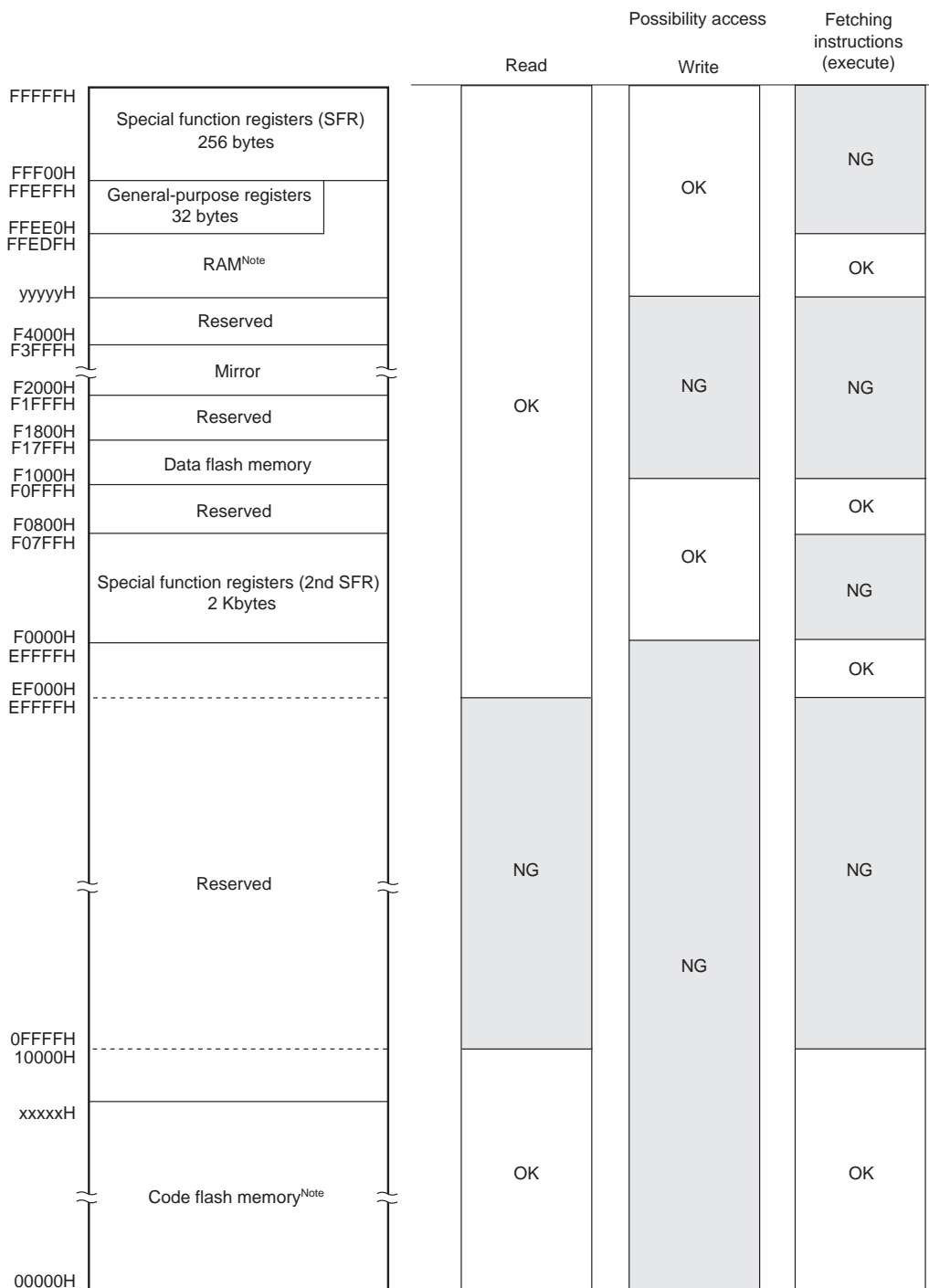
The IEC60730 standard mandates checking that the CPU and interrupts are operating correctly.

The illegal memory access detection function triggers a reset if a memory space specified as access-prohibited is accessed.

The illegal memory access detection function applies to the areas indicated by NG in Figure 20-10.

<R>

Figure 20-11. Invalid access detection area



**Note** Code flash memory and RAM address of each product are as follows.

Products	Code flash memory	RAM
R7F0C010	16384 × 8 bit (00000H to 03FFFH)	1536 × 8 bit (FF900H to FFEFFH)

### 20.8.1 Invalid memory access detection control register (IAWCTL)

This register is used to control the detection of invalid memory access and RAM/SFR guard function.

IAWEN bit is used in invalid memory access detection function.

The IAWCTL register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-12. Format of Invalid Memory Access Detection Control Register (IAWCTL)**

Address: F0078H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
IAWCTL	IAWEN	0	GRAM1	GRAM0	0	GPORT	GINT	GCSC

IAWEN <sup>Note</sup>	Control of invalid memory access detection
0	Disable the detection of invalid memory access.
1	Enable the detection of invalid memory access.

**Note** Only writing 1 to the IAWEN bit is enabled, not writing 0 to it after setting it to 1.

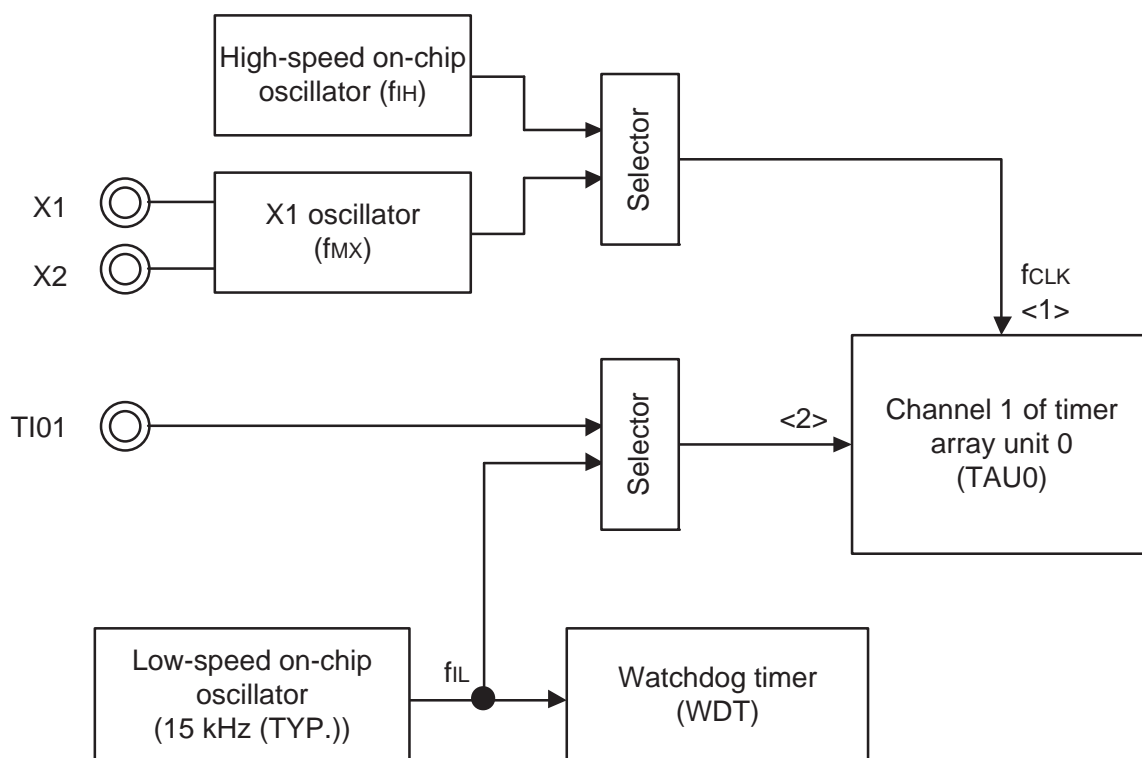
**Remark** By specifying WDTON = 1 for the option byte (watchdog timer operation enable), the invalid memory access function is enabled even IAWEN = 0.

**<R> 20.9 Frequency detection function**

The IEC60730 standard mandates checking that the oscillation frequency is correct.

By using the CPU/peripheral hardware clock frequency ( $f_{CLK}$ ) and measuring the pulse width of the input signal to channel 1 of the timer array unit 0 (TAU0), whether the proportional relationship between the two clock frequencies is correct can be determined. Note that, however, if one or both clock operations are completely stopped, the proportional relationship between the clocks cannot be determined.

**Figure 20-13. Configuration of Frequency Detection Function**

**<Operational overview>**

Whether the clock frequency is correct or not can be judged by measuring the pulse interval under the following conditions:

- The internal high-speed oscillation clock ( $f_{IH}$ ) or the external X1 oscillation clock ( $f_{MX}$ ) is selected as the CPU/peripheral hardware clock ( $f_{CLK}$ ).
- The internal low-speed oscillation clock ( $f_{IL}$ : 15 kHz) is selected as the timer input for channel 1 of timer array unit 0 (TAU0).

If pulse interval measurement results in an abnormal value, it can be concluded that the clock frequency is abnormal. For how to execute pulse interval measurement, see **6.8.3 Operation as input pulse interval measurement**.

### 20.9.1 Timer input select register 0 (TIS0)

&lt;R&gt;

The TIS0 register is used to select the timer input of channels 0 and 1 of the timer array unit 0 (TAU0).

By selecting the internal low-speed oscillation clock for the timer input, its pulse width can be measured to determine whether the proportional relationship between the internal low-speed oscillation clock and the timer operation clock is correct.

The TIS0 register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-14. Format of Timer Input Select Register 0 (TIS0)**

Address: F0074H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
TIS0	0	0	0	0	0	TIS02	TIS01	TIS00

TIS02	TIS01	TIS00	Selection of timer input used with channel 1
0	0	0	Input signal of timer input pin (TI01)
0	0	1	Input signal of timer input pin (TI01)
0	1	0	Input signal of timer input pin (TI01)
0	1	1	Input signal of timer input pin (TI01)
1	0	0	Low-speed on-chip oscillator clock ( $f_{IL}$ )
Other than above			Setting prohibited

&lt;R&gt;



**<R> 20.10 A/D test function**

The IEC60730 standard mandates testing the A/D converter. The A/D test function is used to check whether the A/D converter is operating normally by executing A/D conversions of the positive reference voltage and negative reference voltage of the A/D converter, analog input channel (ANI), temperature sensor output voltage, and internal reference voltage. For details on the checking method, refer to the safety function (A/D test) application note (R01AN0955)

The analog multiplexer can be checked using the following procedure.

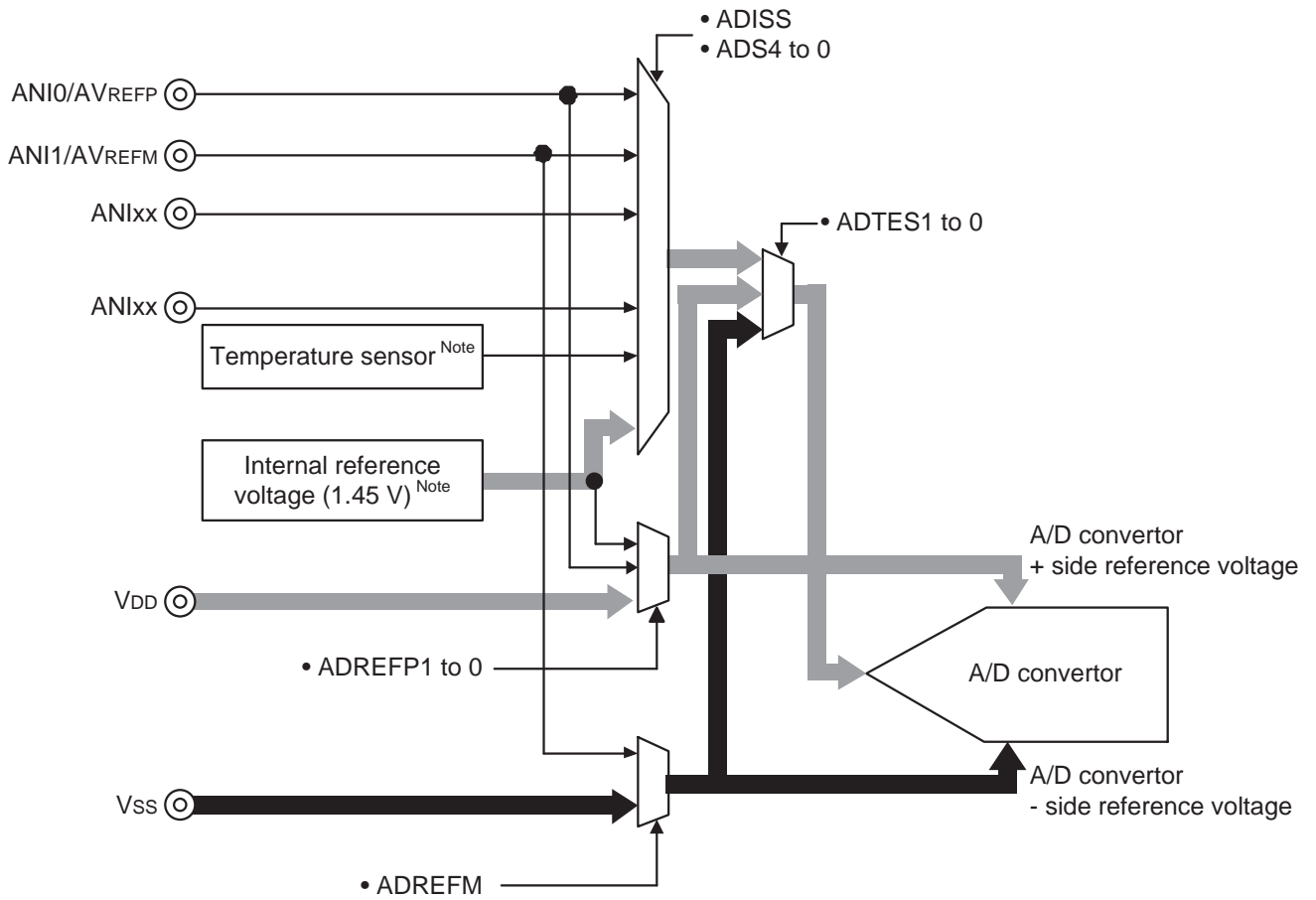
- <1> Select the ANIx pin as the target for A/D conversion by setting the ADTES register (ADTES1, ADTES0 = 0, 0).
- <2> Perform A/D conversion for the ANIx pin (conversion result 1-1).
- <3> Select the negative reference voltage of the A/D converter as the target for A/D conversion by setting the ADTES register (ADTES1, ADTES0 = 1, 0).
- <4> Perform A/D conversion of the negative reference voltage of the A/D converter (conversion result 2-1).
- <5> Select the ANIx pin as the target for A/D conversion by setting the ADTES register (ADTES1, ADTES0 = 0, 0).
- <6> Perform A/D conversion for the ANIx pin (conversion result 1-2)
- <7> Select the positive reference voltage of the A/D converter as the target for A/D conversion by setting the ADTES register (ADTES1, ADTES0 = 1, 1)
- <8> Perform A/D conversion of the positive reference voltage of the A/D converter (conversion result 2-2)
- <9> Select the ANIx pin as the target for A/D conversion by setting the ADTES register (ADTES1, ADTES0 = 0, 0)
- <10> Perform A/D conversion for the ANIx pin (conversion result 1-3)
- <11> Make sure that “conversion result 1-1” = “conversion result 1-2” = “conversion result 1-3”
- <12> Make sure that the A/D conversion results of “conversion result 2-1” are all 0 and those of “conversion result 2-2” are all 1

Using the procedure above can confirm that the analog multiplexer is selected and all wiring is connected.

- Remarks 1.** If the analog input voltage is variable during conversion in steps <1> to <10> above, use another method to check the analog multiplexer..
- 2.** The conversion results might contain an error. Consider an appropriate level of error when comparing the conversion results.

<R>

Figure 20-15. Configuration of A/D Test Function



**Note** Selectable only in HS (high-speed main) mode.

### 20.10.1 A/D test register (ADTES)

This register is used to select the A/D converter's positive reference voltage, the A/D converter's negative reference voltage, or the analog input channel (ANLxx), temperature sensor output voltage, or internal reference voltage (1.45 V) as the target of A/D conversion.

When using the A/D test function, specify the following settings:

- Select AV<sub>REFM</sub> as the target of A/D conversion when converting the internal 0 V.
- Select AV<sub>REFP</sub> as the target of A/D conversion when converting AV<sub>REF</sub>.

The ADTES register can be set by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-16. Format of A/D Test Register (ADTES)**

Address: F0013H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	0
ADTES	0	0	0	0	0	0	ADTES1	ADTES0

ADTES1	ADTES0	A/D conversion target
0	0	ANLxx/temperature sensor output <sup>Note</sup> /internal reference voltage (1.45 V) <sup>Note</sup> (This is specified using the analog input channel specification register (ADS).)
1	0	AV <sub>REFM</sub>
1	1	AV <sub>REFP</sub>
Other than above		Setting prohibited

**Note** Temperature sensor output/internal reference voltage (1.45 V) can be used only in HS (high-speed main) mode.

### 20.10.2 Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

Set A/D test register (ADTES) to 00H when measuring the ANIxx/temperature sensor output/internal reference voltage (1.45 V).

The ADS register can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 20-17. Format of Analog Input Channel Specification Register (ADS)**

Address: FFF31H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
ADS	ADISS	0	0	ADS4 <sup>Note 1</sup>	0	ADS2	ADS1	ADS0

○ Select mode (ADMMD = 0)

ADISS	ADS4 <sup>Note 1</sup>	ADS2	ADS1	ADS0	Analog input channel	Input source
0	0	0	0	0	ANI0	P20/ANI0/AV <sub>REFP</sub> pin
0	0	0	0	1	ANI1	P21/ANI1/AV <sub>REFM</sub> pin
0	0	0	1	0	ANI2	P22/ANI2 pin
0	0	0	1	1	ANI3	P23/ANI3 pin
0	0	1	0	0	ANI4 <sup>Note 2</sup>	P24/ANI4 pin
0	0	1	0	1	ANI5 <sup>Note 2</sup>	P25/ANI5 pin
0	0	1	1	0	ANI6 <sup>Note 2</sup>	P26/ANI6 pin
0	0	1	1	1	ANI7	P27/ANI7 pin
0	1	0	0	0	ANI16 <sup>Note 1</sup>	P03/ANI16 pin
1	0	0	0	0	–	Temperature sensor output <sup>Note 3</sup>
1	0	0	0	1	–	Internal reference voltage output (1.45 V) <sup>Note 3</sup>
Other than above					Setting prohibited	

- Notes**
1. 24-pin products only
  2. 32-pin products only
  3. This setting can be used only in HS (high-speed main) mode.

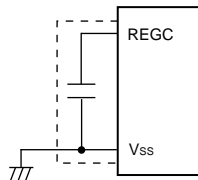
- Cautions**
1. Be sure to clear bits 3, 5, and 6 to “0”.
  2. Only rewrite the value of the ADISS bit while A/D voltage comparator operation is stopped (ADCS = 0 and ADCE = 0 in A/D converter mode register 0 (ADM0)).
  3. If using AV<sub>REFP</sub> as the + side reference voltage of the A/D converter, do not select ANI0 as an A/D conversion channel.
  4. If using AV<sub>REFM</sub> as the – side reference voltage of the A/D converter, do not select ANI1 as an A/D conversion channel.
  5. If ADISS is set to 1, the internal reference voltage (1.45 V) cannot be used for the + side reference voltage.

<R>

## CHAPTER 21 REGULATOR

## 21.1 Regulator Overview

The R7F0C010 contain a circuit for operating the device with a constant voltage. At this time, in order to stabilize the regulator output voltage, connect the REGC pin to V<sub>SS</sub> via a capacitor (0.47 to 1  $\mu$ F). Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.



**Caution** Keep the wiring length as short as possible for the broken-line part in the above figure.

The regulator output voltage, see **Table 21-1**.

**Table 21-1. Regulator Output Voltage Conditions**

Mode	Output Voltage	Condition
LS (low-speed main) mode	1.8 V	—
HS (high-speed main) mode	1.8 V	In STOP mode
	2.1 V	Other than above (include during OCD mode) <sup>Note</sup>

**Note** When it shifts to the STOP mode during the on-chip debugging, the regulator output voltage is kept at 2.1 V (not decline to 1.8 V).

## CHAPTER 22 OPTION BYTE

### 22.1 Functions of Option Bytes

Addresses 000C0H to 000C3H of the flash memory of the R7F0C010 form an option byte area.

Option bytes consist of user option byte (000C0H to 000C2H) and on-chip debug option byte (000C3H).

Upon power application or resetting and starting, an option byte is automatically referenced and a specified function is set. When using the product, be sure to set the following functions by using the option bytes.

For the bits to which no function is allocated, do not change their initial values.

<R>

#### 22.1.1 User option byte (000C0H to 000C2H)

##### (1) 000C0H

- Operation of watchdog timer
  - Enabling or disabling of counter operation.
  - Operation is stopped or enabled in the HALT or STOP mode.
- Setting of interval time of watchdog timer
- Setting of window open period of watchdog timer
- Setting of interval interrupt of watchdog timer
  - Used or not used

<R>

##### (2) 000C1H

- Setting of LVD operation mode
  - Interrupt & reset mode.
  - Reset mode.
  - Interrupt mode.
  - LVD off (external reset input from the  $\overline{\text{RESET}}$  pin is used).
- Setting of LVD detection level ( $V_{\text{LVDH}}$ ,  $V_{\text{LVDL}}$ ,  $V_{\text{LVD}}$ )

##### (3) 000C2H

- Setting of flash operation mode
  - LS (low-speed main) mode
  - HS (high-speed main) mode
- Setting of the frequency of the high-speed on-chip oscillator
  - Select from 1 MHz, 4 MHz, 8 MHz, 12 MHz, 16 MHz, 24 MHz, and 32 MHz.

#### 22.1.2 On-chip debug option byte (000C3H)

- Control of on-chip debug operation
  - On-chip debug operation is disabled or enabled.
- Handling of data of flash memory in case of failure in on-chip debug security ID authentication
  - Data of flash memory is erased or not erased in case of failure in on-chip debug security ID authentication.

## 22.2 Format of User Option Byte

Figure 22-1. Format of User Option Byte (000C0H)

Address: 000C0H

7	6	5	4	3	2	1	0
WDTINIT	WINDOW1	WINDOW0	WDTON	WDCS2	WDCS1	WDCS0	WDSTBYON
WDTINIT	Use of interval interrupt of watchdog timer						
0	Interval interrupt is not used.						
1	Interval interrupt is generated when $75\% + 1/2f_{IL}$ of the overflow time is reached.						
WINDOW1	WINDOW0	Watchdog timer window open period <sup>Note</sup>					
0	0	Setting prohibited					
0	1	50%					
1	0	75%					
1	1	100%					
WDTON	Operation control of watchdog timer counter						
0	Counter operation disabled (counting stopped after reset)						
1	Counter operation enabled (counting started after reset)						
WDCS2	WDCS1	WDCS0	Watchdog timer overflow time ( $f_{IL} = 17.25 \text{ kHz (MAX.)}$ )				
0	0	0	$2^6/f_{IL}$ (3.71 ms)				
0	0	1	$2^7/f_{IL}$ (7.42 ms)				
0	1	0	$2^8/f_{IL}$ (14.84 ms)				
0	1	1	$2^9/f_{IL}$ (29.68 ms)				
1	0	0	$2^{11}/f_{IL}$ (118.72 ms)				
Other than above			Setting prohibited				
WDSTBYON	Operation control of watchdog timer counter (HALT/STOP mode)						
0	Counter operation stopped in HALT/STOP mode <sup>Note</sup>						
1	Counter operation enabled in HALT/STOP mode						

&lt;R&gt;

**Note** The window open period is 100% when WDSTBYON = 0, regardless the value of the WINDOW1 and WINDOW0 bits.

**Caution** The watchdog timer continues its operation even during self-programming or data flash rewrite. During processing, the interrupt acknowledge time is delayed. Set the overflow time and window size taking this delay into consideration.

**Remark**  $f_{IL}$ : Low-speed on-chip oscillator clock frequency

Figure 22-2. Format of User Option Byte (000C1H)

Address: 000C1H

7	6	5	4	3	2	1	0
VPOC2	VPOC1	VPOC0	1	LVIS1	LVIS0	LVIMDS1	LVIMDS0

- LVD setting (interrupt & reset mode)

Detection voltage			Option byte setting value						
V <sub>LVDH</sub>		V <sub>LVDL</sub>	Mode setting		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0
Rising edge	Falling edge	Falling edge	LVIMDS1	LVIMDS0					
2.92 V	2.86 V	2.75 V	1	0	0	1	1	1	0
3.02 V	2.96 V							0	1
Other than above			Setting prohibited						

- LVD setting (reset mode)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	
Rising edge	Falling edge	LVIMDS1	LVIMDS0						
2.81 V	2.75 V	1	1	0	1	1	1	1	
2.92 V	2.86 V				1	1	1	0	
3.02 V	2.96 V				1	1	0	1	
3.13 V	3.06 V				0	1	0	0	
Other than above			Setting prohibited						

- LVD setting (interrupt mode)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	
Rising edge	Falling edge	LVIMDS1	LVIMDS0						
2.81 V	2.75 V	0	1	0	1	1	1	1	
2.92 V	2.86 V				1	1	1	0	
3.02 V	2.96 V				1	1	0	1	
3.13 V	3.06 V				0	1	0	0	
Other than above			Setting prohibited						

- LVD setting (LVDOFF)

Detection voltage			Option byte setting value						
V <sub>LVD</sub>		Mode setting		VPOC2	VPOC1	VPOC0	LVIS1	LVIS0	
Rising edge	Falling edge	LVIMDS1	LVIMDS0						
–	–	0/1	1	1	×	×	×	×	
Other than above			Setting prohibited						

**Caution** Be sure to set bit 4 to “1”.

<R> **Remarks 1.** For details on the LVD circuit, see **CHAPTER 19 VOLTAGE DETECTOR**

**2.** The detection voltage is a typical value. For details, see **27.6.5 LVD circuit characteristics.**



Figure 22-3. Format of Option Byte (000C2H)

Address: 000C2H

7	6	5	4	3	2	1	0
CMODE1	CMODE0	1	0	FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0

CMODE1	CMODE0	Setting of flash operation mode		
			Operating frequency range	Operating voltage range
1	0	LS (low-speed main) mode	1 to 8 MHz	2.7 to 3.6 V
1	1	HS (high-speed main) mode	1 to 32 MHz	2.7 to 3.6 V
Other than above		Setting prohibited		

FRQSEL3	FRQSEL2	FRQSEL1	FRQSEL0	Frequency of the high-speed on-chip oscillator
1	0	0	0	32 MHz
0	0	0	0	24 MHz
1	0	0	1	16 MHz
0	0	0	1	12 MHz
1	0	1	0	8 MHz
1	0	1	1	4 MHz
1	1	0	1	1 MHz
Other than above				Setting prohibited

<R> **Caution** Be sure to set bit 5 to 1.

### 22.3 Format of On-chip Debug Option Byte

The format of on-chip debug option byte is shown below.

**Figure 22-4. Format of On-chip Debug Option Byte (000C3H)**

Address: 000C3H

7	6	5	4	3	2	1	0
OCDENSET	0	0	0	0	1	0	OCDERSD

OCDENSET	OCDERSD	Control of on-chip debug operation
0	0	Disables on-chip debug operation.
0	1	Setting prohibited
1	0	Enables on-chip debugging. Erases data of flash memory in case of failures in authenticating on-chip debug security ID.
1	1	Enables on-chip debugging. Does not erases data of flash memory in case of failures in authenticating on-chip debug security ID.

**Caution** Bits 7 and 0 (OCDENSET and OCDERSD) can only be specified a value.

Be sure to set 000010B to bits 6 to 1.

**Remark** The value on bits 3 to 1 will be written over when the on-chip debug function is in use and thus it will become unstable after the setting.

However, be sure to set the default values (0, 1, and 0) to bits 3 to 1 at setting.

## 22.4 Setting of Option Byte

The user option byte and on-chip debug option byte can be set using the assembler linker option, in addition to describing to the source. When doing so, the contents set by using the linker option take precedence, even if descriptions exist in the source, as mentioned below.

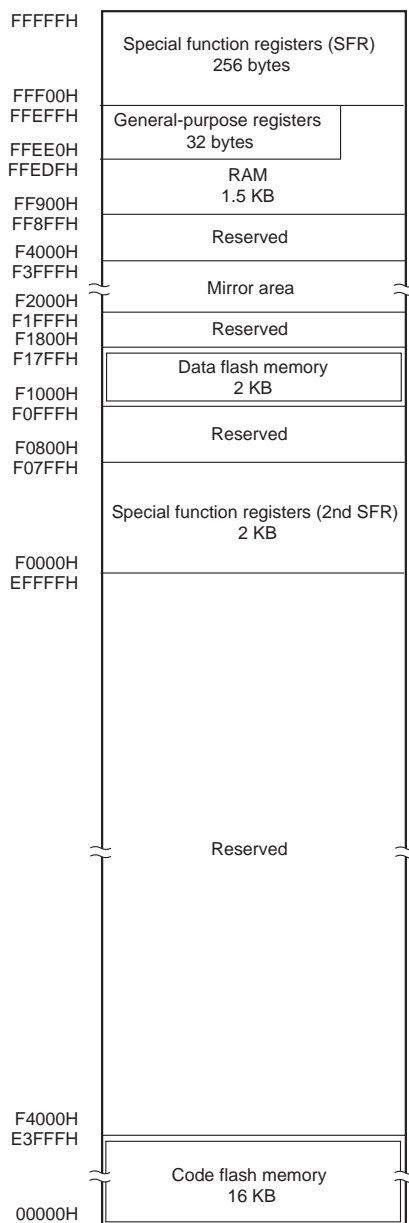
A software description example of the option byte setting is shown below.

OPT	CSEG	OPT_BYTE	
	DB	36H	; Does not use interval interrupt of watchdog timer, ; Enables watchdog timer operation, ; Window open period of watchdog timer is 50%, ; Overflow time of watchdog timer is $2^9/f_{IL}$ , ; Stops watchdog timer operation during HALT/STOP mode
	DB	7AH	; Select 2.75 V for $V_{LVDL}$ ; Select rising edge 1.77 V, falling edge 2.86 V for $V_{LVDH}$ ; Select the interrupt & reset mode as the LVD operation mode
	DB	ADH	; Select the LS (low-speed main) mode as the flash operation mode and 1 MHz as the frequency of the high-speed on-chip oscillator
	DB	85H	; Enables on-chip debug operation, does not erase flash memory data when security ID authorization fails

**Caution** To specify the option byte by using assembly language, use `OPT_BYTE` as the relocation attribute name of the `CSEG` pseudo instruction.

## CHAPTER 23 FLASH MEMORY

The R7F0C010 incorporate the flash memory to which a program can be written, erased, and overwritten while mounted on the board. The flash memory includes the “code flash memory”, in which programs can be executed, and the “data flash memory”, an area for storing data.



The following three methods for programming the flash memory are available:

- Writing to flash memory by using flash memory programmer (see **23.1**)  
Data can be written to the flash memory on-board or off-board by using a dedicated flash memory programmer.
- Writing to flash memory by using external device (that Incorporates UART) (see **23.2**)  
Data can be written to the flash memory on-board through UART communication with an external device (microcontroller or ASIC).
- Self-programming (see **23.7**)  
The user application can execute self-programming of the code flash memory by using the flash self-programming library.

**<R> 23.1 Serial Programming Using Flash Memory Programmer**

The following dedicated flash memory programmer can be used to write data to the internal flash memory of the R7F0C010.

- PG-FP5, FL-PR5
- E1 on-chip debugging emulator

Data can be written to the flash memory on-board or off-board, by using a dedicated flash memory programmer.

**(1) On-board programming**

The contents of the flash memory can be rewritten after the R7F0C010 have been mounted on the target system. The connectors that connect the dedicated flash memory programmer must be mounted on the target system.

**(2) Off-board programming**

Data can be written to the flash memory with a dedicated program adapter (FA series) before the R7F0C010 is mounted on the target system.

**Remark** FL-PR5 and FA series are products of Naito Densai Machida Mfg. Co., Ltd.

**Table 23-1. Wiring Between R7F0C010 and Dedicated Flash Memory Programmer**

Pin Configuration of Dedicated Flash Memory Programmer				Pin Name	Pin No.	
Signal Name		I/O	Pin Function		24-pin	32-pin
PG-FP5, FL-PR5	E1 On-chip Debugging Emulator				WQFN (4 x 4)	LQFP (7 x 7)
–	TOOL0	I/O	Transmit/receive signal	TOOL0/P40	23	1
SI/RxD	–	I/O	Transmit/receive signal			
SCK	–	Output	–	–	–	–
CLK	–	Output	–	–	–	–
–	RESET	Output	Reset signal	RESET	24	2
/RESET	–	Output				
FLMD0	–	Output	Mode signal	–	–	–
V <sub>DD</sub>		I/O	V <sub>DD</sub> voltage generation/ power monitoring	V <sub>DD</sub>	6	8
GND		–	Ground	V <sub>SS</sub>	5	7
				REGC <sup>Note</sup>	4	6
EMV <sub>DD</sub>		–	Driving power for TOOL pin	V <sub>DD</sub>	6	8

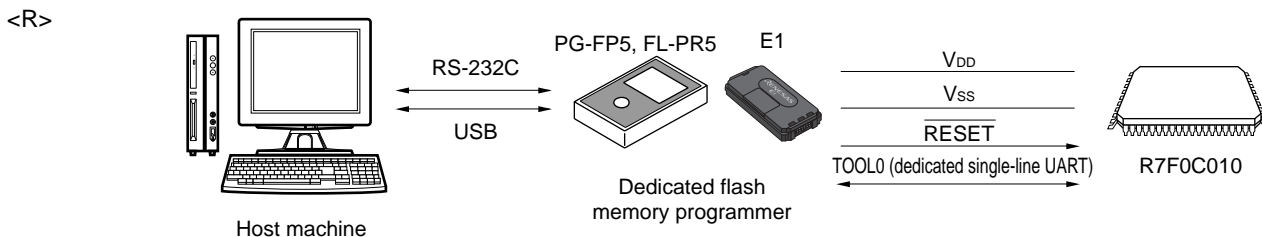
**Note** Connect REGC pin to ground via a capacitor (0.47 to 1  $\mu$ F).

**Remark** Pins that are not indicated in the above table can be left open when using the flash memory programmer for flash programming.

23.1.1 Programming environment

The environment required for writing a program to the flash memory of the R7F0C010 is illustrated below.

Figure 23-1. Environment for Writing Program to Flash Memory



A host machine that controls the dedicated flash memory programmer is necessary.

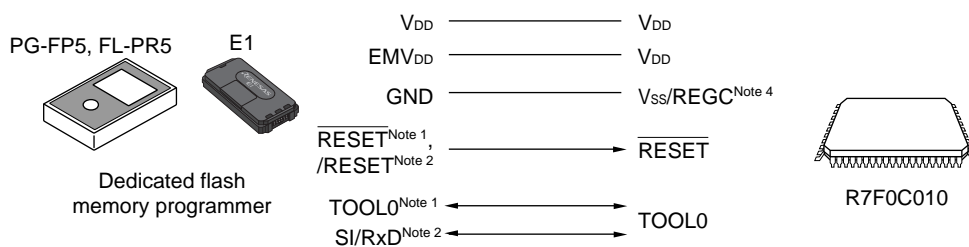
To interface between the dedicated flash memory programmer and the R7F0C010, the TOOL0 pin is used for manipulation such as writing and erasing via a dedicated single-line UART.

23.1.2 Communication mode

Communication between the dedicated flash memory programmer and the R7F0C010 is established by serial communication using the TOOL0 pin via a dedicated single-line UART of the R7F0C010.

Transfer rate: 1 M, 500 k, 250 k, 115.2 kbps

Figure 23-2. Communication with Dedicated Flash Memory Programmer



- Notes**
1. When using E1 on-chip debugging emulator.
  2. When using PG-FP5 or FL-PR5.
  3. Connect REGC pin to ground via a capacitor (0.47 to 1  $\mu$ F).

The dedicated flash memory programmer generates the following signals for the R7F0C010. See the manual of PG-FP5, FL-PR5, or E1 on-chip debugging emulator for details.

**Table 23-2. Pin Connection**

Dedicated Flash Memory Programmer			R7F0C010	Connection	
Signal Name		I/O	Pin Name		
PG-FP5, FL-PR5	E1 On-chip Debugging Emulator				
FLMD0	–	Output	Mode signal	–	×
$V_{DD}$		I/O	$V_{DD}$ voltage generation/power monitoring	$V_{DD}$	◎
GND		–	Ground	$V_{SS}$ , REGC <sup>Note</sup>	◎
$EMV_{DD}$		–	Driving power for TOOL pin	$V_{DD}$	◎
CLK	–	Output	Clock output	–	×
/RESET	–	Output	Reset signal	$\overline{\text{RESET}}$	◎
–	$\overline{\text{RESET}}$	Output			
–	TOOL0	I/O	Transmit/receive signal	TOOL0	◎
SI/RxD	–	I/O	Transmit/receive signal		
SCK	–	Output	Transfer clock	–	×

**Note** Connect REGC pin to ground via a capacitor (0.47 to 1  $\mu\text{F}$ ).

**Remark** ◎: Be sure to connect the pin.

×: The pin does not have to be connected.

## <R> 23.2 Serial Programming Using External Device (that Incorporates UART)

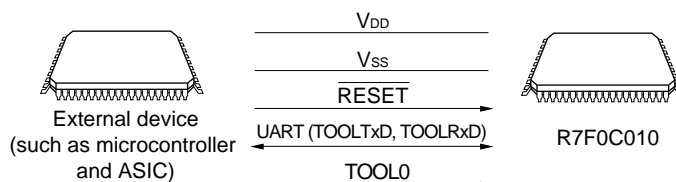
On-board data writing to the internal flash memory is possible by using the R7F0C010 and an external device (a microcontroller or ASIC) connected to a UART.

On the development of flash memory programmer by user, refer to the **RL78 Microcontrollers (RL78 Protocol A) Programmer Edition Application Note (R01AN0815)**.

### 23.2.1 Programming environment

The environment required for writing a program to the flash memory of the R7F0C010 is illustrated below.

**Figure 23-3. Environment for Writing Program to Flash Memory**



Processing to write data to or delete data from the R7F0C010 by using an external device is performed on-board. Off-board writing is not possible.

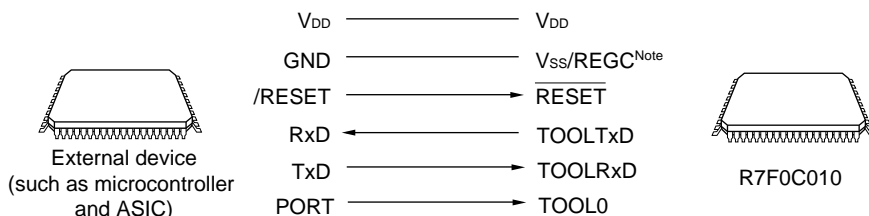


**23.2.2 Communication mode**

Communication between the external device and the R7F0C010 is established by serial communication using the TOOLTxD and TOOLRxD pins via the dedicated UART of the R7F0C010.

Transfer rate: 1 M, 500 k, 250 k, 115.2kbps

**Figure 23-4. Communication with External Device**



**Note** Connect REGC pin to ground via a capacitor (0.47 to 1  $\mu$ F).

The external device generates the following signals for the R7F0C010.

**Table 23-3. Pin Connection**

External Device			R7F0C010	Connection
Signal Name	I/O	Pin Function	Pin Name	
V <sub>DD</sub>	I/O	V <sub>DD</sub> voltage generation/power monitoring	V <sub>DD</sub>	◎
GND	–	Ground	V <sub>SS</sub> , REGC <sup>Note</sup>	◎
CLK	Output	Clock output	–	×
RESETOUT	Output	Reset signal output	RESET	◎
RxD	Input	Receive signal	TOOL0TxD	◎
TxD	Output	Transmit signal	TOOL0RxD	◎
PORT	Output	Mode signal	TOOL0	◎
SCK	Output	Transfer clock	–	×

**Note** Connect REGC pin to ground via a capacitor (0.47 to 1  $\mu$ F).

**Remark** ◎: Be sure to connect the pin.  
 ×: The pin does not have to be connected.

### 23.3 Connection of Pins on Board

To write the flash memory on-board by using the flash memory programmer, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

When the flash memory programming mode is set, all the pins not used for programming the flash memory are in the same status as immediately after reset. Therefore, if the external device does not recognize the state immediately after reset, the pins must be handled as described below.

<R> **Remark** Refer to flash programming mode, see **23.5.2 Flash memory programming mode**.

#### 23.3.1 P40/TOOL0 pin

In the flash memory programming mode, connect this pin to the dedicated flash memory programmer via an external 1 k $\Omega$  pull-up resistor.

When this pin is used as the port pin, use that by the following method.

When used as an input pin: Input of low-level is prohibited for  $t_{HD}$  period after external pin reset release. Furthermore, when this pin is used via pull-down resistors, use the 500 k $\Omega$  or more resistors.

When used as an output pin: When this pin is used via pull-down resistors, use the 500 k $\Omega$  or more resistors.

<R> **Remark 1.**  $t_{HD}$ : How long to keep the TOOL0 pin at the low level from when the external and internal resets end for setting of the flash memory programming mode (see **27.9 Timing for Switching Flash Memory Programming Modes**).

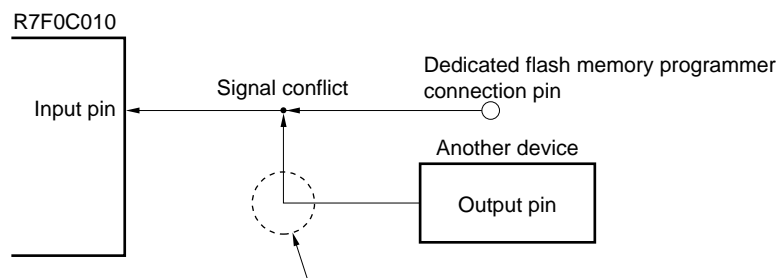
**Remark 2.** The SAU and IICA pins are not used for communication between the R7F0C010, and dedicated flash memory programmer, because single-line UART (TOOL0 pin) is used.

#### 23.3.2 $\overline{\text{RESET}}$ pin

Signal conflict will occur if the reset signal of the dedicated flash memory programmer and external device are connected to the  $\overline{\text{RESET}}$  pin that is connected to the reset signal generator on the board. To prevent this conflict, isolate the connection with the reset signal generator.

The flash memory will not be correctly programmed if the reset signal is input from the user system while the flash memory programming mode is set. Do not input any signal other than the reset signal of the dedicated flash memory programmer and external device.

**Figure 23-5. Signal Conflict ( $\overline{\text{RESET}}$  Pin)**



In the flash memory programming mode, a signal output by another device will conflict with the signal output by the dedicated flash memory programmer. Therefore, isolate the signal of another device.

### 23.3.3 Port pins

When the flash memory programming mode is set, all the pins not used for flash memory programming enter the same status as that immediately after reset. If external devices connected to the ports do not recognize the port status immediately after reset, the port pin must be connected to either to  $V_{DD}$  or  $V_{SS}$  via a resistor.

### 23.3.4 REGC pin

<R> Connect the REG C pin to GND via a capacitor having excellent characteristics (0.47 to 1  $\mu$ F) in the same manner as during normal operation. Also, use a capacitor with good characteristics, since it is used to stabilize internal voltage.

### 23.3.5 X1 and X2 pins

Connect X1 and X2 in the same status as in the normal operation mode.

**Remark** In the flash memory programming mode, the high-speed on-chip oscillator clock ( $f_{IH}$ ) is used.

### 23.3.6 Power supply

To use the supply voltage output of the flash memory programmer, connect the  $V_{DD}$  pin to  $V_{DD}$  of the flash memory programmer, and the  $V_{SS}$  pin to GND of the flash memory programmer.

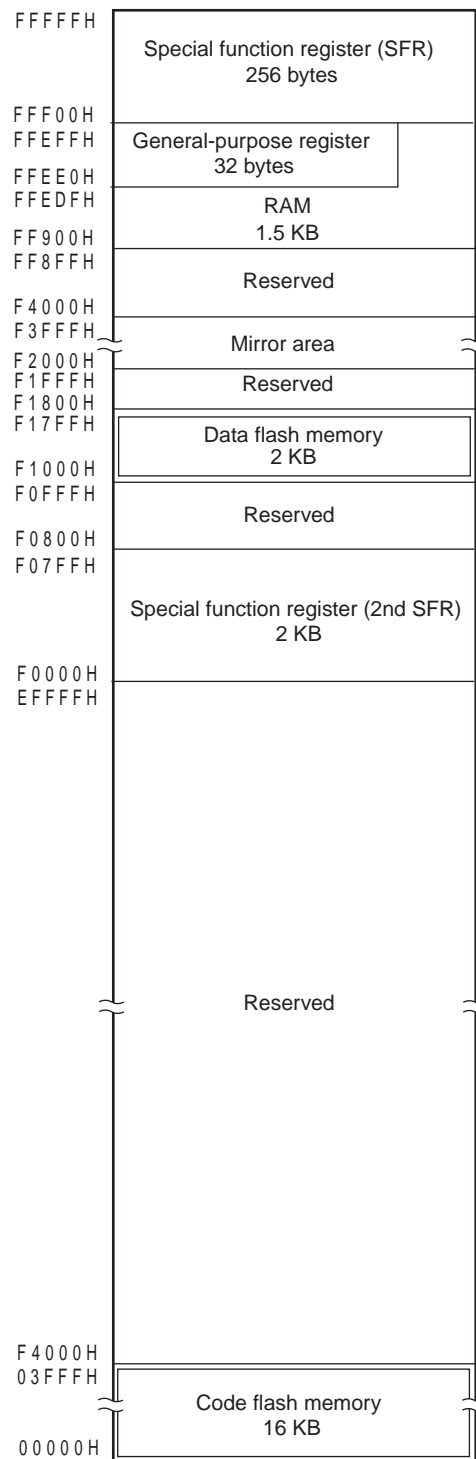
To use the on-board supply voltage, connect in compliance with the normal operation mode.

However, when writing to the flash memory by using the flash memory programmer and using the on-board supply voltage, be sure to connect the  $V_{DD}$  and  $V_{SS}$  pins to  $V_{DD}$  and GND of the flash memory programmer to use the power monitor function with the flash memory programmer.

## 23.4 Data Flash

### 23.4.1 Data flash overview

In addition to 16 KB of code flash memory, the R7F0C010 with data flash includes 2 KB of data flash memory for storing data.



An overview of the data flash memory is provided below. For details of a method for rewriting the data flash memory, refer to the **RL78 Family Data Flash Library User's Manual**.

- The data flash memory can be written to by using the flash memory programmer or an external device
- Programming is performed in 8-bit units
- Blocks can be deleted in 1 KB units
- The only access by CPU instructions is byte reading (1 clock cycle + wait 3 clock cycles)
- Because the data flash memory is an area exclusively used for data, it cannot be used to execute instructions (code fetching)
- Instructions can be executed from the code flash memory while rewriting the data flash memory (That is, back ground operation (BGO) is supported)
- Accessing the data flash memory is not possible while rewriting the code flash memory (during self-programming)
- Because the data flash memory is stopped after a reset ends, the data flash control register (DFLCTL) must be set up in order to use the data flash memory
- Manipulating the DFLCTL register is not possible while rewriting the data flash memory
- Transition the HALT/STOP status is not possible while rewriting the data flash memory

**Caution** The high-speed on-chip oscillator should be kept operating during data flash rewrite. If it is kept stopping, it should be operated (HIOSTOP = 0). The data flash library should be executed after 30  $\mu$ s have elapsed.

**Remark** Refer to flash programming mode, see 23.7 Flash Memory Programming by Self-programming.

## 23.4.2 Register controlling data flash memory

### 23.4.2.1 Data flash control register (DFLCTL)

This register is used to enable or disable accessing to the data flash.

The DFLCTL register is set by a 1-bit or 8-bit memory manipulation instruction.

Reset input sets this register to 00H.

**Figure 23-6. Format of Data Flash Control Register (DFLCTL)**

Address: F0090H    After reset: 00H    R/W

Symbol	7	6	5	4	3	2	1	$\square$
DFLCTL	0	0	0	0	0	0	0	DFLEN

DFLEN	Data flash access control
0	Disables data flash access
1	Enables data flash access

**Caution** Manipulating the DFLCTL register is not possible while rewriting the data flash memory.

### 23.4.3 Procedure for accessing data flash memory

<R> The data flash memory is stopped after a reset ends. To access the data flash, make initial settings according to the following procedure.

- <1> Set bit 0 (DFLEN) of the data flash control register (DFLCTL) to 1.
- <2> Wait for the setup to finish for software timer, etc.  
The time setup takes differs for each flash operation mode for the main clock.  
<Setup time for each flash operation mode>
  - HS (High speed main): 5  $\mu$ s
  - LS (Low speed main): 720 ns
- <3> After the wait, the data flash memory can be accessed.

- Cautions**
1. Accessing the data flash memory is not possible during the setup time.
  2. Transition to the STOP mode is not possible during the setup time. To enter the STOP mode during the setup time, clear DFLEN to 0 and then execute the STOP instruction.
  3. The high-speed on-chip oscillator should be kept operating during data flash rewrite. If it is kept stopping, the high-speed on-chip oscillator clock should be operated (HIOSTOP = 0). The data flash library should be executed after 30  $\mu$ s have elapsed.

After initial setting, the data flash can be read through CPU instructions and can be read or rewritten to by using the data flash library.

Follow one of the procedures below when the DMA controller operates during access to the data flash memory.

- (A) Hold DMA transfer pending or forcibly terminate it  
Before reading the data flash memory, hold the DMA transfer pending in all the channels which are in use. The data flash memory should be read 3 clocks ( $f_{CLK}$ ) or more after the DWAITn bit is set to 1. After reading the data flash memory, set the DWAITn bit to 0 and then cancel the pending status.  
Or, before reading the data flash memory, forcibly terminate the DMA transfer in accordance with the process described in **13.5.5 Forced termination by software**. Resume the DMA transfer after reading the data flash memory.

- (B) Access the data flash memory by using the library  
Access the data flash memory by using the latest data flash library.

- (C) Insert the NOP instruction  
Insert the NOP instruction immediately before the data flash read instruction.

<Example>

```
MOVW HL, !addr16 ; Read RAM
NOP                ; Insert the NOP instruction before reading the data flash memory
MOV A,[DE]         ; Read the data flash memory
```

If high-level language like C language is used, the compiler may generate two instructions per code. At that time, the NOP instruction is not inserted immediately before the data flash read instruction. Read the data flash memory by following procedure (A) or (B).

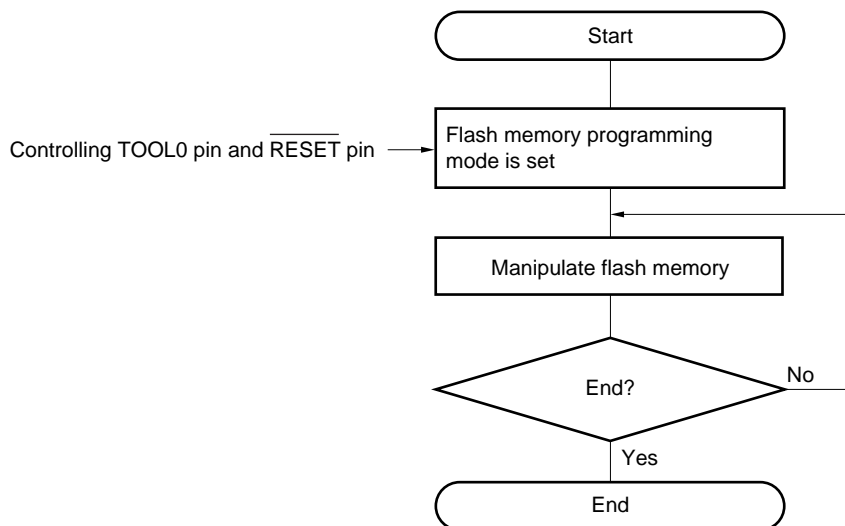
**Remark**  $f_{CLK}$ : CPU/peripheral hardware clock frequency

## 23.5 Programming Method

### 23.5.1 Controlling flash memory

<R> The following figure illustrates a flow for rewriting the code flash memory through serial programming.

Figure 23-7. Flash Memory Manipulation Procedure



### 23.5.2 Flash memory programming mode

To rewrite the contents of the flash memory, set the R7F0C010 in the flash memory programming mode. To enter the mode, set as follows.

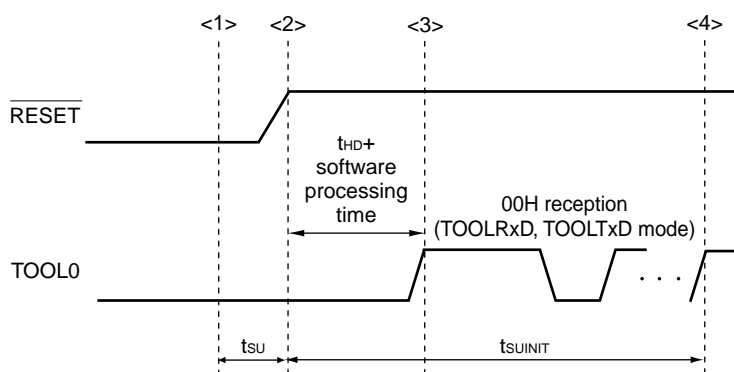
#### <When programming by using the dedicated flash memory programmer>

Communication from the dedicated flash memory programmer is performed to automatically switch to the flash memory programming mode.

#### <When programming by using an external device>

Set the TOOL0 pin to the low level, and then cancel the reset. Keep the TOOL0 pin at the low level from the reset ends to 1 ms + software processing end, and then use UART communication to send the data "00H" from the external device. Finish UART communication within 100 ms after the reset ends.

Figure 23-8. Setting of Flash Memory Programming Mode



<1> The low level is input to the TOOL0 pin.

<2> The external reset ends (POR and LVD reset must end before the external reset ends.).

<3> The TOOL0 pin is set to the high level.

<4> Setting of the flash memory programming mode by UART reception and complete the baud rate setting.

**Remark**  $t_{SUINIT}$ : The segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

$t_{SU}$ : How long from when the TOOL0 pin is placed at the low level until an external reset ends

$t_{HD}$ : How long to keep the TOOL0 pin at the low level from when the external and internal resets end (except soft processing time)

<R> For details, see 27.9 Timing for Switching Flash Memory Programming Modes.



**Table 23-4. Relationship Between TOOL0 Pin and Operation Mode After Reset Release**

TOOL0	Operation Mode
V <sub>DD</sub>	Normal operation mode
0	Flash memory programming mode

<R> There are two flash memory programming modes: wide voltage mode and full speed mode. The supply voltage value applied to the microcontroller during write operations and the setting information of the user option byte for setting of the flash memory programming mode determine which mode is selected.

When a dedicated flash memory programmer is used for serial programming, setting the voltage on GUI selects the mode automatically.

**Table 23-5. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified**

Mode	Voltages at Which Data can Be Written, Erased, or Verified	Writing Clock Frequency
Wide voltage mode	2.7 V to 3.6 V	8 MHz (MAX.)
	2.7 V to 3.6 V	16 MHz (MAX.)
	2.7 V to 3.6 V	32 MHz (MAX.)
Full speed mode <sup>Note</sup>	2.7 V to 3.6 V	16 MHz (MAX.)
	2.7 V to 3.6 V	32 MHz (MAX.)

**Note** This can only be specified if the CMODE1 and CMODE0 bits of the option byte 000C2H are 1.

Specify the mode that corresponds to the voltage range in which to write data. When programming by using the dedicated flash memory programmer, the mode is automatically selected by the voltage setting on GUI.

**Remarks 1.** Using both the wide voltage mode and full speed mode imposes no restrictions on writing, deletion, or verification.

**2.** For details about communication commands, see **23.5.4 Communication commands**.

### 23.5.3 Selecting communication mode

<R> Communication mode of the RL78 microcontroller as follows.

**Table 23-6. Communication Modes**

Communication Mode	Standard Setting <sup>Note 1</sup>				Pins Used
	Port	Speed <sup>Note 2</sup>	Frequency	Multiply Rate	
1-line mode (when flash memory programmer is used, or when external device is used)	UART	115200 bps, 250000 bps, 500000 bps, 1 Mbps	–	–	TOOL0
Dedicated UART (when external device is used)	UART	115200 bps, 250000 bps, 500000 bps, 1 Mbps	–	–	TOOLTxD , TOOLRxD

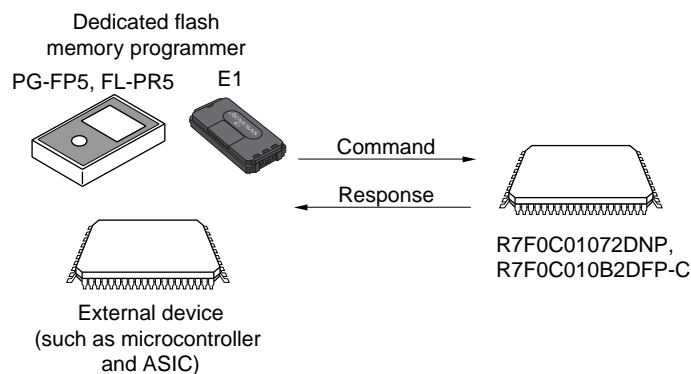
**Notes 1.** Selection items for Standard settings on GUI of the flash memory programmer.

**2.** Because factors other than the baud rate error, such as the signal waveform slew, also affect UART communication, thoroughly evaluate the slew as well as the baud rate error.

### 23.5.4 Communication commands

The R7F0C010 communicate with the dedicated flash memory programmer or external device by using commands. The signals sent from the flash memory programmer or external device to the R7F0C010 are called commands, and the signals sent from the R7F0C010 to the dedicated flash memory programmer or external device are called response.

**Figure 23-9. Communication Commands**



The flash memory control commands of the R7F0C01072DNP and R7F0C010B2DFP-C are listed in the table below. All these commands are issued from the programmer or external device, and the R7F0C01072DNP and R7F0C010B2DFP-C perform processing corresponding to the respective commands.

**Table 23-7. Flash Memory Control Commands**

Classification	Command Name	Function
Verify	Verify	Compares the contents of a specified area of the flash memory with data transmitted from the programmer.
Erase	Block Erase	Erases a specified area in the flash memory.
Blank check	Block Blank Check	Checks if a specified block in the flash memory has been correctly erased.
Write	Programming	Writes data to a specified area in the flash memory. <sup>Note</sup>
Getting information	Silicon Signature	Gets the R7F0C010 information (such as the part number, flash memory configuration, and programming firmware version).
	Checksum	Gets the checksum data for a specified area.
Security	Security Set	Sets security information.
	Security Get	Gets security information.
	Security Release	Release setting of prohibition of writing.
Others	Reset	Used to detect synchronization status of communication.
	Baud Rate Set	Sets baud rate when UART communication mode is selected.

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

The R7F0C010 return a response for the command issued by the dedicated flash memory programmer or external device. The response names sent from the R7F0C010 are listed below.

**Table 23-8. Response Names**

Response Name	Function
ACK	Acknowledges command/data.
NAK	Acknowledges illegal command/data.

### 23.5.5 Description of signature data

When the “silicon signature” command is performed, the R7F0C010 information (such as the part number, flash memory configuration, and programming firmware version) can be obtained.

Table 23-9 and 23-10 show signature data list and example of signature data list.

**Table 23-9. Signature Data List**

Field Name	Description	Number of Transmit Data
Device code	The serial number assigned to the device	3 bytes
Device name	Device name (ASCII code)	10 bytes
Code flash memory area last address	Last address of code flash memory area (Sent from lower address. Example: 00000H to 0FFFFH (64 KB) → FFH, 1FH, 00H)	3 bytes
Data flash memory area last address	Last address of data flash memory area (Sent from lower address. Example: F1000H to F1FFFH (4 KB) → FFH, 1FH, 0FH)	3 bytes
Firmware version	Version information of firmware for programming (Sent from upper address. Example: From Ver. 1.23 → 01H, 02H, 03H)	3 bytes

**Table 23-10. Example of Signature Data**

Field Name	Description	Number of Transmit Data	Data (Hexadecimal)
Device code	RL78 protocol A	3 bytes	10 00 06
Device name	R7F0C0107	10 bytes	52 = “R” 35 = “7” 46 = “F” 30 = “0” 43 = “C” 30 = “0” 31 = “1” 30 = “0” 37 = “7” 20 = “ ”
Code flash memory area last address	Code flash memory area 00000H to 03FFFH (16 KB)	3 bytes	FF 3F 00
Data flash memory area last address	Data flash memory area F1000H to F17FFFH (2 KB)	3 bytes	FF 17 0F
Firmware version	Ver.1.23	3 bytes	01 02 03

## 23.6 Security Settings

The R7F0C010 support a security function that prohibits rewriting the user program written to the internal flash memory, so that the program cannot be changed by an unauthorized person.

The operations shown below can be performed using the Security Set command.

- Disabling block erase

Execution of the block erase command for a specific block in the flash memory is prohibited during on-board/off-board programming. However, blocks can be erased by means of self-programming.

- Disabling write

Execution of the write command for entire blocks in the flash memory is prohibited during on-board/off-board programming. However, blocks can be written by means of self-programming.

<R> After the security settings are specified, releasing the security settings by the Security Release command is enabled by a reset.

After the security settings are specified, releasing the security settings by the Security Release command is enabled by a reset.

The block erase and write commands are enabled by the default setting when the flash memory is shipped. Security can be set by on-board/off-board programming and self-programming. Each security setting can be used in combination.

Table 23-11 shows the relationship between the erase and write commands when the R7F0C010 security function is enabled.

**Caution** The security function of the flash programmer does not support self-programming.

**Remark** To prohibit writing and erasing during self-programming, use the flash sealed window function (see 23.7.1 for detail).

**Table 23-11. Relationship Between Enabling Security Function and Command****(1) During on-board/off-board programming**

Valid Security	Executed Command	
	Block Erase	Write
Prohibition of block erase	Blocks cannot be erased.	Can be performed. <sup>Note</sup>
Prohibition of writing	Blocks can be erased.	Cannot be performed.

**Note** Confirm that no data has been written to the write area. Because data cannot be erased after block erase is prohibited, do not write data if the data has not been erased.

**(2) During self-programming**

Valid Security	Executed Command	
	Block Erase	Write
Prohibition of block erase	Blocks can be erased.	Can be performed.
Prohibition of writing		

**Remark** To prohibit writing and erasing during self-programming, use the flash sealed window function (see 23.7.1 for detail).

**Table 23-12. Setting Security in Each Programming Mode****(1) On-board/off-board programming**

Security	Security Setting	How to Disable Security Setting
Prohibition of block erase	Set via GUI of dedicated flash memory programmer, etc.	Cannot be disabled after set.
Prohibition of writing		Execute security release command

**Caution** The security release command can be applied only when the security is not set as the block erase prohibition with code flash memory area and data flash memory area being blanks.

### 23.7 Flash Memory Programming by Self-programming

The R7F0C010 support a self-programming function that can be used to rewrite the flash memory via a user program. Because this function allows a user application to rewrite the flash memory by using the R7F0C010 self-programming library, it can be used to upgrade the program in the field.

- Cautions**
1. To prohibit an interrupt during self-programming, in the same way as in the normal operation mode, execute the self-programming library in the state where the IE flag is cleared (0) by the DI instruction. To enable an interrupt, clear (0) the interrupt mask flag to accept in the state where the IE flag is set (1) by the EI instruction, and then execute the self-programming library.
  2. When enabling RAM parity error resets (RPERDIS = 0), be sure to initialize the RAM area to use +10 bytes before overwriting.
  3. The high-speed on-chip oscillator should be kept operating during self-programming. If it is kept stopped, it should be operated (HIOSTOP = 0). The flash self-programming library should be executed after 30  $\mu$ s have elapsed.

&lt;R&gt;

- Remarks**
1. For details of the self-programming function and the R7F0C010 self-programming library, refer to the **RL78 Microcontroller Self-programming Library Type01 User's Manual (R01AN0350E)**.
  2. For details of the time required to execute self-programming, see the notes on use that accompany the flash self-programming library tool.

Similar to when writing data by using the flash memory programmer, there are two flash memory programming modes for which the voltage range in which to write, erase, or verify data differs.

**Table 23-13. Programming Modes and Voltages at Which Data Can Be Written, Erased, or Verified**

Mode	Voltages at Which Data can Be Written, Erased, or Verified	Writing Clock Frequency
Wide voltage mode	2.7 V to 3.6 V	32 MHz (MAX.)
Full speed mode <sup>Note</sup>	2.7 V to 3.6 V	32 MHz (MAX.)

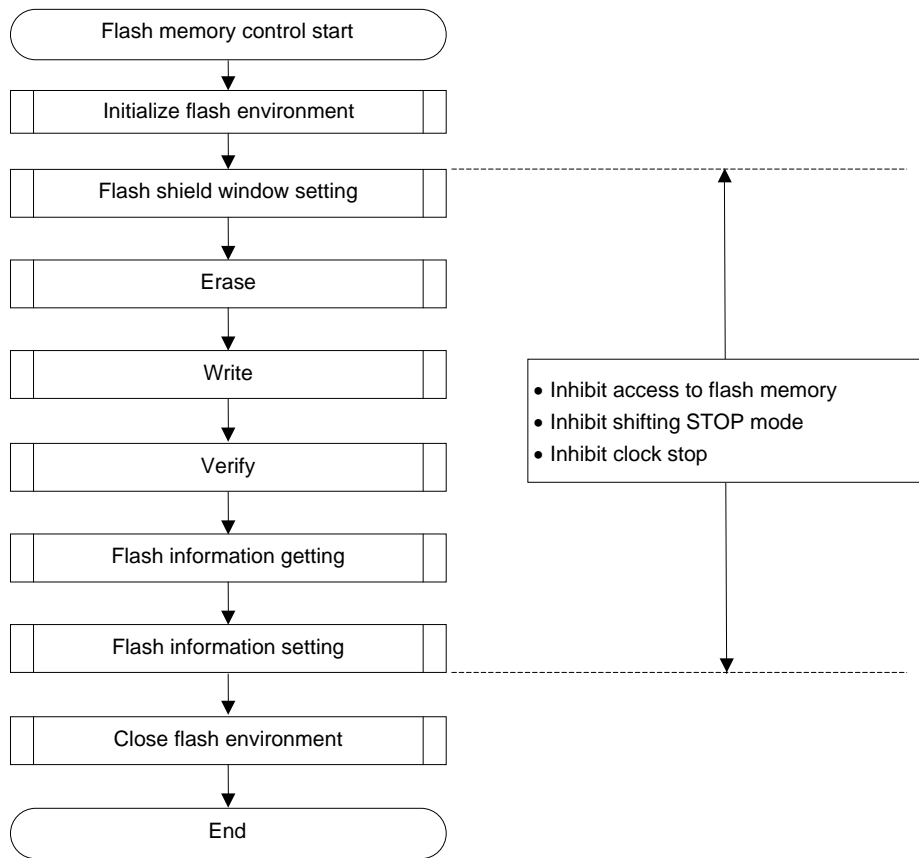
**Note** This can only be specified if the CMODE1 and CMODE0 bits of the option byte 000C2H are 1.

Specify the mode that corresponds to the voltage range in which to write data. If the argument `fsl_flash_voltage_u08` is other than 00H when the `FSL_Init` function of the self-programming library provided by Renesas Electronics is executed, wide-voltage mode is specified. If the argument is 00H, full-speed mode is specified.

- Remarks**
1. Using both the wide voltage mode and full speed mode imposes no restrictions on writing, deletion, or verification.
  2. For details of the self-programming function and the R7F0C010 self-programming library, refer to the **RL78 Microcontroller Self-programming Library Type01 User's Manual (R01AN0350E)**.

The following figure illustrates a flow of rewriting the flash memory by using a self-programming library.

**Figure 23-10. Flow of Self-programming (Rewriting Flash Memory)**





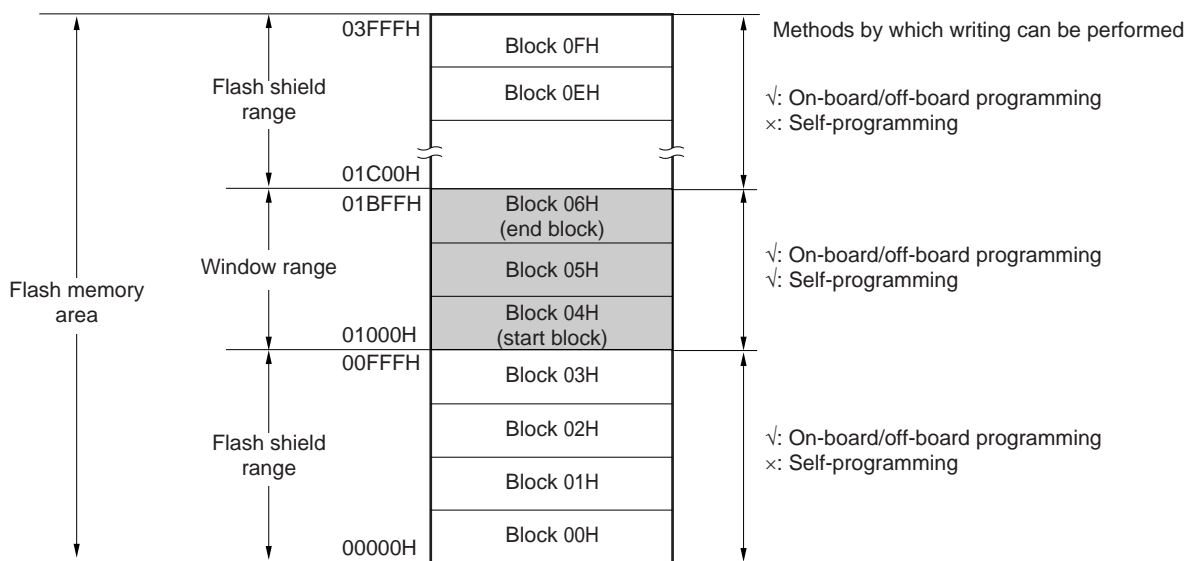
**23.7.1 Flash shield window function**

The flash shield window function is provided as one of the security functions for self-programming. It disables writing to and erasing areas outside the range specified as a window only during self-programming.

The window range can be set by specifying the start and end blocks. The window range can be set or changed only during on-board/off-board programming.

Writing to and erasing areas outside the window range are disabled during self-programming. During on-board/off-board programming, however, areas outside the range specified as a window can be written and erased.

**Figure 23-11. Flash Shield Window Setting Example**  
**(Target Devices: R7F0C010, Start Block: 04H, End Block: 06H)**



**Caution** The flash shield window can only be used for the code flash memory (and is not supported for the data flash memory).

**Table 23-14. Relationship Between Flash Shield Window Function Setting/Change Methods and Commands**

Programming Conditions	Window Range Setting/Change Methods	Execution Commands	
		Block Erase	Write
On-board/Off-board programming	Specify the starting and ending blocks on GUI of dedicated flash memory programmer, etc.	Block erasing is enabled also outside the window range.	Writing is enabled also outside the window range.

**Remark** See 23.6 Security Settings to prohibit writing/erasing during on-board/off-board programming.

&lt;R&gt;

**23.8 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)**

The following shows the processing time for each command (reference value) when PG-FP5 is used as a dedicated flash memory programmer.

**Table 25 - 15 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)**

PG-FP5 Command	Port: TOOL0 (UART)
	Speed: 1M bps
	16 Kbytes
Erasing	1 s
Writing 1	1.5 s
Verification	1.5 s
Writing after erasing	1.5 s

**Remark** The command processing times (reference values) shown in the table are typical values under the following conditions.

Port: TOOL0 (single-line UART)

Speed: 1,000,000 bps

Mode: Full speed mode (flash operation mode: HS (high speed main) mode).

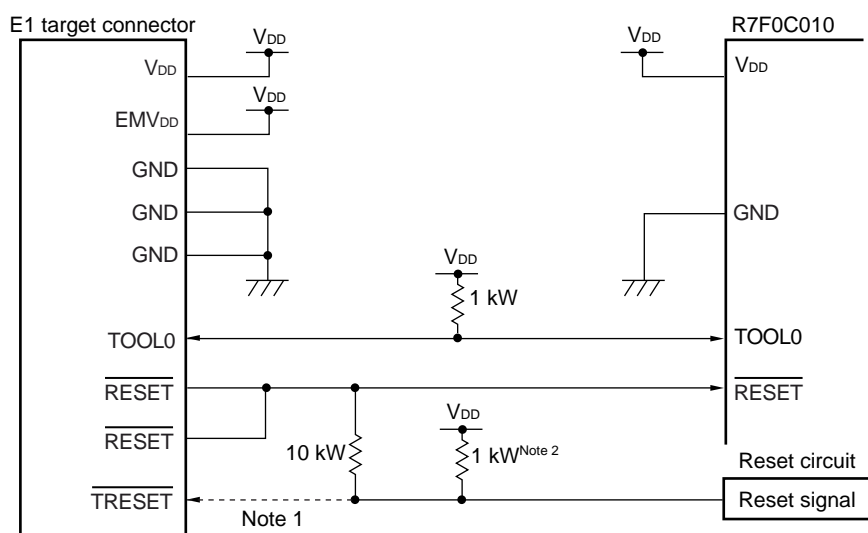
## CHAPTER 24 ON-CHIP DEBUG FUNCTION

## 24.1 Connecting E1 On-chip Debugging Emulator to R7F0C010

The R7F0C010 use the  $V_{DD}$ , RESET, TOOL0, and  $V_{SS}$  pins to communicate with the host machine via an E1 on-chip debugging emulator. Serial communication is performed by using a single-line UART that uses the TOOL0 pin.

**Caution** The R7F0C010 have an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.

Figure 24-1. Connection Example of E1 On-chip Debugging Emulator and R7F0C010



- Notes**
1. Connecting the dotted line is not necessary during flash programming.
  2. If the reset circuit on the target system does not have a buffer and generates a reset signal only with resistors and capacitors, this pull-up resistor is not necessary.

**Caution** This circuit diagram is assumed that the reset signal outputs from an N-ch O.D. buffer (output resistor: 100  $\Omega$  or less)

## 24.2 On-chip Debug Security ID

The R7F0C010 have an on-chip debug operation control bit in the flash memory at 000C3H (see **CHAPTER 22 OPTION BYTE**) and an on-chip debug security ID setting area at 000C4H to 000CDH, to prevent third parties from reading memory content.

**Table 24-1. On-chip Debug Security ID**

Address	On-chip Debug Security ID
000C4H to 000CDH	Any ID code of 10 bytes
010C4H to 010CDH	

## 24.3 Securing of User Resources

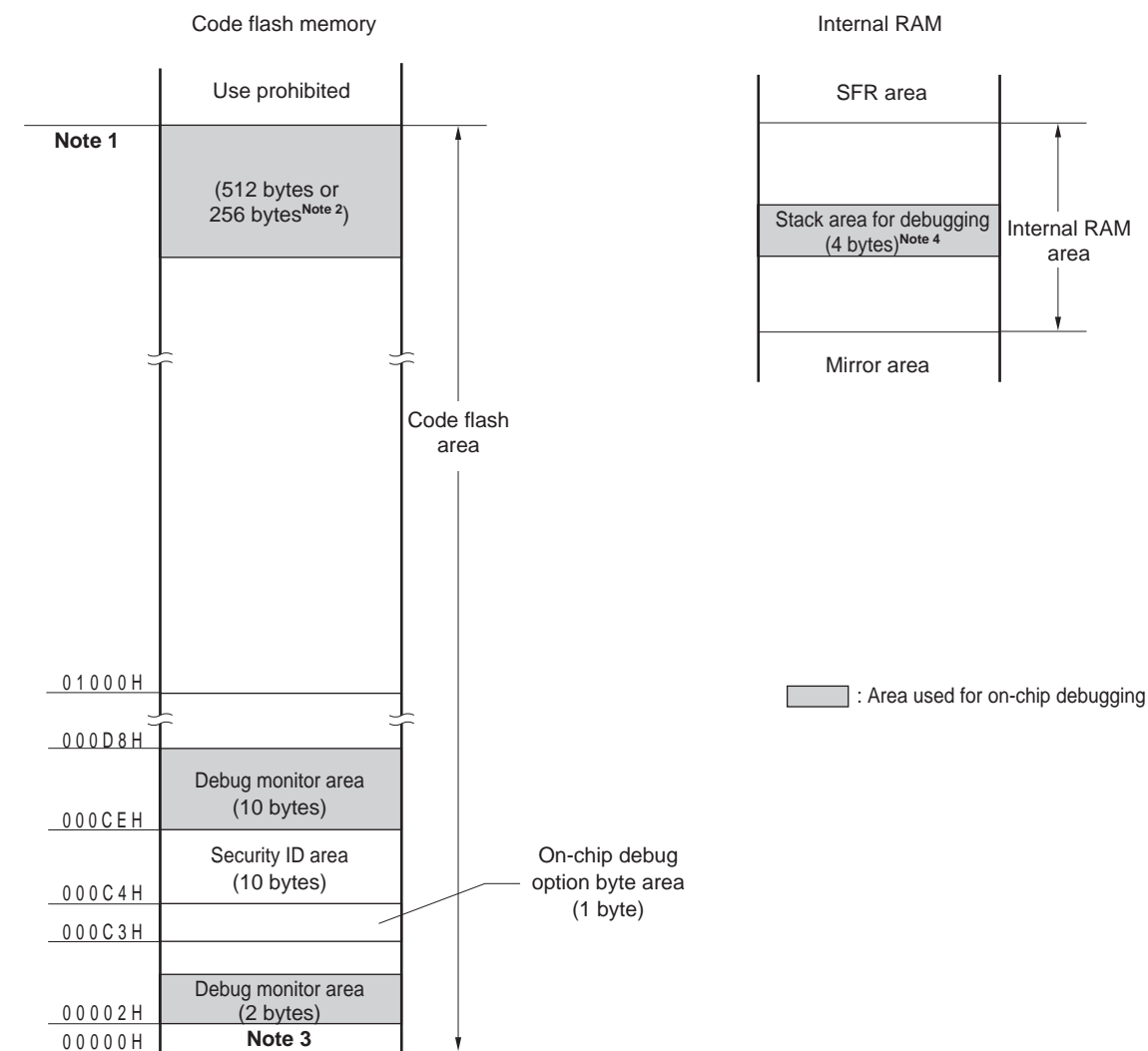
To perform communication between the R7F0C010 and E1 on-chip debugging emulator, as well as each debug function, the securing of memory space must be done beforehand.

If Renesas Electronics assembler or compiler is used, the items can be set by using linker options.

### (1) Securement of memory space

The shaded portions in Figure 24-2 are the areas reserved for placing the debug monitor program, so user programs or data cannot be allocated in these spaces. When using the on-chip debug function, these spaces must be secured so as not to be used by the user program. Moreover, this area must not be rewritten by the user program.

Figure 24-2. Memory Spaces Where Debug Monitor Programs Are Allocated



Notes 1. Address of code flash memory is as follows.

Products	Address of Note 1
R7F0C010	03FFFH

- When real-time RAM monitor (RRM) function and dynamic memory modification (DMM) function are not used, it is 256 bytes.
- In debugging, reset vector is rewritten to address allocated to a monitor program.
- Since this area is allocated immediately before the stack area, the address of this area varies depending on the stack increase and decrease. That is, 4 extra bytes are consumed for the stack area used. When using self-programming, 12 extra bytes are consumed for the stack area used.

## CHAPTER 25 BCD CORRECTION CIRCUIT

### 25.1 BCD Correction Circuit Function

The result of addition/subtraction of the BCD (binary-coded decimal) code and BCD code can be obtained as BCD code with this circuit.

The decimal correction operation result is obtained by performing addition/subtraction having the A register as the operand and then adding/subtracting the BCD correction result register (BCDADJ).

### 25.2 Registers Used by BCD Correction Circuit

The BCD correction circuit uses the following registers.

- BCD correction result register (BCDADJ)

#### 25.2.1 BCD correction result register (BCDADJ)

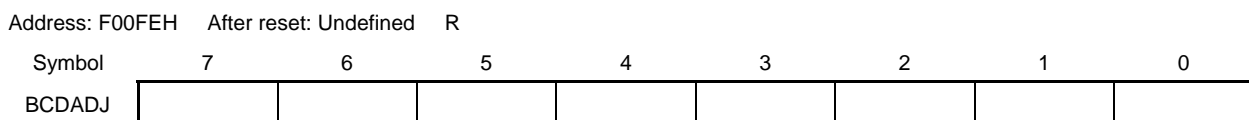
The BCDADJ register stores correction values for obtaining the add/subtract result as BCD code through add/subtract instructions using the A register as the operand.

The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags.

The BCDADJ register is read by an 8-bit memory manipulation instruction.

Reset input sets this register to undefined.

**Figure 25-1. Format of BCD Correction Result Register (BCDADJ)**



### 25.3 BCD Correction Circuit Operation

The basic operation of the BCD correction circuit is as follows.

**(1) Addition: Calculating the result of adding a BCD code value and another BCD code value by using a BCD code value**

- <1> The BCD code value to which addition is performed is stored in the A register.
- <2> By adding the value of the A register and the second operand (value of one more BCD code to be added) as are in binary, the binary operation result is stored in the A register and the correction value is stored in the BCD correction result register (BCDADJ).
- <3> Decimal correction is performed by adding in binary the value of the A register (addition result in binary) and the BCDADJ register (correction value), and the correction result is stored in the A register and CY flag.

**Caution** The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY flag and AC flag) is restored by the RETI instruction.

An example is shown below.

Examples 1:  $99 + 89 = 188$

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #99H ; <1>	99H	–	–	–
ADD A, #89H ; <2>	22H	1	1	66H
ADD A, !BCDADJ ; <3>	88H	1	0	–

Examples 2:  $85 + 15 = 100$

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #85H ; <1>	85H	–	–	–
ADD A, #15H ; <2>	9AH	0	0	66H
ADD A, !BCDADJ ; <3>	00H	1	1	–

Examples 3:  $80 + 80 = 160$

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #80H ; <1>	80H	–	–	–
ADD A, #80H ; <2>	00H	1	0	60H
ADD A, !BCDADJ ; <3>	60H	1	0	–

**(2) Subtraction: Calculating the result of subtracting a BCD code value from another BCD code value by using a BCD code value**

- <1> The BCD code value from which subtraction is performed is stored in the A register.
- <2> By subtracting the value of the second operand (value of BCD code to be subtracted) from the A register as is in binary, the calculation result in binary is stored in the A register, and the correction value is stored in the BCD correction result register (BCDADJ).
- <3> Decimal correction is performed by subtracting the value of the BCDADJ register (correction value) from the A register (subtraction result in binary) in binary, and the correction result is stored in the A register and CY flag.

**Caution** The value read from the BCDADJ register varies depending on the value of the A register when it is read and those of the CY and AC flags. Therefore, execute the instruction <3> after the instruction <2> instead of executing any other instructions. To perform BCD correction in the interrupt enabled state, saving and restoring the A register is required within the interrupt function. PSW (CY flag and AC flag) is restored by the RETI instruction.

An example is shown below.

Example:  $91 - 52 = 39$

Instruction	A Register	CY Flag	AC Flag	BCDADJ Register
MOV A, #91H ; <1>	91H	–	–	–
SUB A, #52H ; <2>	3FH	0	1	06H
SUB A, !BCDADJ ; <3>	39H	0	0	–



## CHAPTER 26 INSTRUCTION SET

This chapter lists the instructions in the RL78 microcontroller instruction set. For details of each operation and operation code, refer to the separate document **RL78 Family User's Manual: Software**.

## 26.1 Conventions Used in Operation List

### 26.1.1 Operand identifiers and specification methods

Operands are described in the “Operand” column of each instruction in accordance with the description method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more description methods, select one of them. Alphabetic letters in capitals and the symbols, #, !, !!, \$, \$!, [ ], and ES: are keywords and are described as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: 16-bit absolute address specification
- !!: 20-bit absolute address specification
- \$: 8-bit relative address specification
- \$!: 16-bit relative address specification
- [ ]: Indirect address specification
- ES:: Extension address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to describe the #, !, !!, \$, \$!, [ ], and ES: symbols.

For operand register identifiers, r and rp, either function names (X, A, C, etc.) or absolute names (names in parentheses in the table below, R0, R1, R2, etc.) can be used for description.

**Table 26-1. Operand Identifiers and Specification Methods**

Identifier	Description Method
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special-function register symbol (SFR symbol) FFF00H to FFFFFH
sfrp	Special-function register symbols (16-bit manipulatable SFR symbol. Even addresses only <sup>Note</sup> ) FFF00H to FFFFFH
saddr	FFE20H to FFF1FH Immediate data or labels
saddrp	FFE20H to FF1FH Immediate data or labels (even addresses only <sup>Note</sup> )
addr20	00000H to FFFFFH Immediate data or labels
addr16	0000H to FFFFH Immediate data or labels (only even addresses for 16-bit data transfer instructions <sup>Note</sup> )
addr5	0080H to 00BFH Immediate data or labels (even addresses only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

**Note** Bit 0 = 0 when an odd address is specified.

<R> **Remark** The special function registers can be described to operand sfr as symbols. See **Table 3-5 Special Function Register (SFR) List**. The extended special function registers can be described to operand !addr16 as symbols. See **Table 3-6 Extended Special Function Register (2nd SFR) List**.

### 26.1.2 Description of operation column

The operation when the instruction is executed is shown in the “Operation” column using the following symbols.

**Table 26-2. Symbols in “Operation” Column**

Symbol	Function
A	A register; 8-bit accumulator
X	X register
B	B register
C	C register
D	D register
E	E register
H	H register
L	L register
ES	ES register
CS	CS register
AX	AX register pair; 16-bit accumulator
BC	BC register pair
DE	DE register pair
HL	HL register pair
PC	Program counter
SP	Stack pointer
PSW	Program status word
CY	Carry flag
AC	Auxiliary carry flag
Z	Zero flag
RBS	Register bank select flag
IE	Interrupt request enable flag
()	Memory contents indicated by address or register contents in parentheses
X <sub>H</sub> , X <sub>L</sub>	16-bit registers: X <sub>H</sub> = higher 8 bits, X <sub>L</sub> = lower 8 bits
X <sub>S</sub> , X <sub>H</sub> , X <sub>L</sub>	20-bit registers: X <sub>S</sub> = (bits 19 to 16), X <sub>H</sub> = (bits 15 to 8), X <sub>L</sub> = (bits 7 to 0)
∧	Logical product (AND)
∨	Logical sum (OR)
⊕	Exclusive logical sum (exclusive OR)
–	Inverted data
addr5	16-bit immediate data (even addresses only in 0080H to 00BFH)
addr16	16-bit immediate data
addr20	20-bit immediate data
jdisp8	Signed 8-bit data (displacement value)
jdisp16	Signed 16-bit data (displacement value)

### 26.1.3 Description of flag operation column

The change of the flag value when the instruction is executed is shown in the “Flag” column using the following symbols.

**Table 26-3. Symbols in “Flag” Column**

Symbol	Change of Flag Value
(Blank)	Unchanged
0	Cleared to 0
1	Set to 1
×	Set/cleared according to the result
R	Previously saved value is restored

### 26.1.4 PREFIX instruction

Instructions with “ES:” have a PREFIX operation code as a prefix to extend the accessible data area to the 1 MB space (00000H to FFFFFH), by adding the ES register value to the 64 KB space from F0000H to FFFFFH. When a PREFIX operation code is attached as a prefix to the target instruction, only one instruction immediately after the PREFIX operation code is executed as the addresses with the ES register value added.

An interrupt and DTC transfer are not acknowledged between a PREFIX instruction code and the instruction immediately after.

**Table 26-4. Use Example of PREFIX Operation Code**

Instruction	Opcode				
	1	2	3	4	5
MOV !addr16, #byte	CFH	!addr16		#byte	–
MOV ES:!addr16, #byte	11H	CFH	!addr16		#byte
MOV A, [HL]	8BH	–	–	–	–
MOV A, ES:[HL]	11H	8BH	–	–	–

**Caution** Set the ES register value with MOV ES, A, etc., before executing the PREFIX instruction.

## 26.2 Operation List

Table 26-5. Operation List (1/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	r, #byte	2	1	–	r ← byte			
		PSW, #byte	3	3	–	PSW ← byte	x	x	x
		CS, #byte	3	1	–	CS ← byte			
		ES, #byte	2	1	–	ES ← byte			
		!addr16, #byte	4	1	–	(addr16) ← byte			
		ES:!addr16, #byte	5	2	–	(ES, addr16) ← byte			
		saddr, #byte	3	1	–	(saddr) ← byte			
		sfr, #byte	3	1	–	sfr ← byte			
		[DE+byte], #byte	3	1	–	(DE+byte) ← byte			
		ES:[DE+byte],#byte	4	2	–	((ES, DE)+byte) ← byte			
		[HL+byte], #byte	3	1	–	(HL+byte) ← byte			
		ES:[HL+byte],#byte	4	2	–	((ES, HL)+byte) ← byte			
		[SP+byte], #byte	3	1	–	(SP+byte) ← byte			
		word[B], #byte	4	1	–	(B+word) ← byte			
		ES:word[B], #byte	5	2	–	((ES, B)+word) ← byte			
		word[C], #byte	4	1	–	(C+word) ← byte			
		ES:word[C], #byte	5	2	–	((ES, C)+word) ← byte			
		word[BC], #byte	4	1	–	(BC+word) ← byte			
		ES:word[BC], #byte	5	2	–	((ES, BC)+word) ← byte			
		A, r <small>Note 3</small>	1	1	–	A ← r			
		r, A <small>Note 3</small>	1	1	–	r ← A			
		A, PSW	2	1	–	A ← PSW			
		PSW, A	2	3	–	PSW ← A	x	x	x
		A, CS	2	1	–	A ← CS			
		CS, A	2	1	–	CS ← A			
		A, ES	2	1	–	A ← ES			
		ES, A	2	1	–	ES ← A			
		A, !addr16	3	1	4	A ← (addr16)			
		A, ES:!addr16	4	2	5	A ← (ES, addr16)			
		!addr16, A	3	1	–	(addr16) ← A			
ES:!addr16, A	4	2	–	(ES, addr16) ← A					
A, saddr	2	1	–	A ← (saddr)					
saddr, A	2	1	–	(saddr) ← A					

- Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
- 2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.
- 3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (2/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	A, sfr	2	1	–	A ← sfr			
		sfr, A	2	1	–	sfr ← A			
		A, [DE]	1	1	4	A ← (DE)			
		[DE], A	1	1	–	(DE) ← A			
		A, ES:[DE]	2	2	5	A ← (ES, DE)			
		ES:[DE], A	2	2	–	(ES, DE) ← A			
		A, [HL]	1	1	4	A ← (HL)			
		[HL], A	1	1	–	(HL) ← A			
		A, ES:[HL]	2	2	5	A ← (ES, HL)			
		ES:[HL], A	2	2	–	(ES, HL) ← A			
		A, [DE+byte]	2	1	4	A ← (DE + byte)			
		[DE+byte], A	2	1	–	(DE + byte) ← A			
		A, ES:[DE+byte]	3	2	5	A ← ((ES, DE) + byte)			
		ES:[DE+byte], A	3	2	–	((ES, DE) + byte) ← A			
		A, [HL+byte]	2	1	4	A ← (HL + byte)			
		[HL+byte], A	2	1	–	(HL + byte) ← A			
		A, ES:[HL+byte]	3	2	5	A ← ((ES, HL) + byte)			
		ES:[HL+byte], A	3	2	–	((ES, HL) + byte) ← A			
		A, [SP+byte]	2	1	–	A ← (SP + byte)			
		[SP+byte], A	2	1	–	(SP + byte) ← A			
		A, word[B]	3	1	4	A ← (B + word)			
		word[B], A	3	1	–	(B + word) ← A			
		A, ES:word[B]	4	2	5	A ← ((ES, B) + word)			
		ES:word[B], A	4	2	–	((ES, B) + word) ← A			
		A, word[C]	3	1	4	A ← (C + word)			
		word[C], A	3	1	–	(C + word) ← A			
		A, ES:word[C]	4	2	5	A ← ((ES, C) + word)			
		ES:word[C], A	4	2	–	((ES, C) + word) ← A			
		A, word[BC]	3	1	4	A ← (BC + word)			
		word[BC], A	3	1	–	(BC + word) ← A			
A, ES:word[BC]	4	2	5	A ← ((ES, BC) + word)					
ES:word[BC], A	4	2	–	((ES, BC) + word) ← A					

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (3/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit data transfer	MOV	A, [HL+B]	2	1	4	$A \leftarrow (HL + B)$			
		[HL+B], A	2	1	–	$(HL + B) \leftarrow A$			
		A, ES:[HL+B]	3	2	5	$A \leftarrow ((ES, HL) + B)$			
		ES:[HL+B], A	3	2	–	$((ES, HL) + B) \leftarrow A$			
		A, [HL+C]	2	1	4	$A \leftarrow (HL + C)$			
		[HL+C], A	2	1	–	$(HL + C) \leftarrow A$			
		A, ES:[HL+C]	3	2	5	$A \leftarrow ((ES, HL) + C)$			
		ES:[HL+C], A	3	2	–	$((ES, HL) + C) \leftarrow A$			
		X, laddr16	3	1	4	$X \leftarrow (addr16)$			
		X, ES:laddr16	4	2	5	$X \leftarrow (ES, addr16)$			
		X, saddr	2	1	–	$X \leftarrow (saddr)$			
		B, laddr16	3	1	4	$B \leftarrow (addr16)$			
		B, ES:laddr16	4	2	5	$B \leftarrow (ES, addr16)$			
		B, saddr	2	1	–	$B \leftarrow (saddr)$			
		C, laddr16	3	1	4	$C \leftarrow (addr16)$			
		C, ES:laddr16	4	2	5	$C \leftarrow (ES, addr16)$			
		C, saddr	2	1	–	$C \leftarrow (saddr)$			
		ES, saddr	3	1	–	$ES \leftarrow (saddr)$			
	XCH	A, r <sup>Note 3</sup>	1 (r = X) 2 (other than r = X)	1	–	$A \leftrightarrow r$			
		A, laddr16	4	2	–	$A \leftrightarrow (addr16)$			
		A, ES:laddr16	5	3	–	$A \leftrightarrow (ES, addr16)$			
		A, saddr	3	2	–	$A \leftrightarrow (saddr)$			
		A, sfr	3	2	–	$A \leftrightarrow sfr$			
		A, [DE]	2	2	–	$A \leftrightarrow (DE)$			
		A, ES:[DE]	3	3	–	$A \leftrightarrow (ES, DE)$			
		A, [HL]	2	2	–	$A \leftrightarrow (HL)$			
		A, ES:[HL]	3	3	–	$A \leftrightarrow (ES, HL)$			
		A, [DE+byte]	3	2	–	$A \leftrightarrow (DE + \text{byte})$			
A, ES:[DE+byte]		4	3	–	$A \leftrightarrow ((ES, DE) + \text{byte})$				
A, [HL+byte]		3	2	–	$A \leftrightarrow (HL + \text{byte})$				
A, ES:[HL+byte]	4	3	–	$A \leftrightarrow ((ES, HL) + \text{byte})$					

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**3.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (4/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag			
				Note 1	Note 2		Z	AC	CY	
8-bit data transfer	XCH	A, [HL+B]	2	2	–	A $\leftrightarrow$ (HL+B)				
		A, ES:[HL+B]	3	3	–	A $\leftrightarrow$ ((ES, HL)+B)				
		A, [HL+C]	2	2	–	A $\leftrightarrow$ (HL+C)				
		A, ES:[HL+C]	3	3	–	A $\leftrightarrow$ ((ES, HL)+C)				
	ONEB	A	1	1	–	A $\leftarrow$ 01H				
		X	1	1	–	X $\leftarrow$ 01H				
		B	1	1	–	B $\leftarrow$ 01H				
		C	1	1	–	C $\leftarrow$ 01H				
		!addr16	3	1	–	(addr16) $\leftarrow$ 01H				
		ES:!addr16	4	2	–	(ES, addr16) $\leftarrow$ 01H				
		saddr	2	1	–	(saddr) $\leftarrow$ 01H				
	CLR B	A	1	1	–	A $\leftarrow$ 00H				
		X	1	1	–	X $\leftarrow$ 00H				
		B	1	1	–	B $\leftarrow$ 00H				
		C	1	1	–	C $\leftarrow$ 00H				
		!addr16	3	1	–	(addr16) $\leftarrow$ 00H				
		ES:!addr16	4	2	–	(ES, addr16) $\leftarrow$ 00H				
		saddr	2	1	–	(saddr) $\leftarrow$ 00H				
	MOVS	[HL+byte], X	3	1	–	(HL+byte) $\leftarrow$ X	x		x	
		ES:[HL+byte], X	4	2	–	(ES, HL+byte) $\leftarrow$ X	x		x	
	16-bit data transfer	MOVW	rp, #word	3	1	–	rp $\leftarrow$ word			
			saddrp, #word	4	1	–	(saddrp) $\leftarrow$ word			
sfrp, #word			4	1	–	sfrp $\leftarrow$ word				
AX, rp <sup>Note 3</sup>			1	1	–	AX $\leftarrow$ rp				
rp, AX <sup>Note 3</sup>			1	1	–	rp $\leftarrow$ AX				
AX, !addr16			3	1	4	AX $\leftarrow$ (addr16)				
!addr16, AX			3	1	–	(addr16) $\leftarrow$ AX				
AX, ES:!addr16			4	2	5	AX $\leftarrow$ (ES, addr16)				
ES:!addr16, AX			4	2	–	(ES, addr16) $\leftarrow$ AX				
AX, saddrp			2	1	–	AX $\leftarrow$ (saddrp)				
saddrp, AX			2	1	–	(saddrp) $\leftarrow$ AX				
AX, sfrp			2	1	–	AX $\leftarrow$ sfrp				
sfrp, AX			2	1	–	sfrp $\leftarrow$ AX				

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**3.** Except rp = AX

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.



Table 26-5. Operation List (5/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	AX, [DE]	1	1	4	AX ← (DE)			
		[DE], AX	1	1	–	(DE) ← AX			
		AX, ES:[DE]	2	2	5	AX ← (ES, DE)			
		ES:[DE], AX	2	2	–	(ES, DE) ← AX			
		AX, [HL]	1	1	4	AX ← (HL)			
		[HL], AX	1	1	–	(HL) ← AX			
		AX, ES:[HL]	2	2	5	AX ← (ES, HL)			
		ES:[HL], AX	2	2	–	(ES, HL) ← AX			
		AX, [DE+byte]	2	1	4	AX ← (DE+byte)			
		[DE+byte], AX	2	1	–	(DE+byte) ← AX			
		AX, ES:[DE+byte]	3	2	5	AX ← ((ES, DE) + byte)			
		ES:[DE+byte], AX	3	2	–	((ES, DE) + byte) ← AX			
		AX, [HL+byte]	2	1	4	AX ← (HL + byte)			
		[HL+byte], AX	2	1	–	(HL + byte) ← AX			
		AX, ES:[HL+byte]	3	2	5	AX ← ((ES, HL) + byte)			
		ES:[HL+byte], AX	3	2	–	((ES, HL) + byte) ← AX			
		AX, [SP+byte]	2	1	–	AX ← (SP + byte)			
		[SP+byte], AX	2	1	–	(SP + byte) ← AX			
		AX, word[B]	3	1	4	AX ← (B + word)			
		word[B], AX	3	1	–	(B+ word) ← AX			
		AX, ES:word[B]	4	2	5	AX ← ((ES, B) + word)			
		ES:word[B], AX	4	2	–	((ES, B) + word) ← AX			
		AX, word[C]	3	1	4	AX ← (C + word)			
		word[C], AX	3	1	–	(C + word) ← AX			
		AX, ES:word[C]	4	2	5	AX ← ((ES, C) + word)			
		ES:word[C], AX	4	2	–	((ES, C) + word) ← AX			
		AX, word[BC]	3	1	4	AX ← (BC + word)			
		word[BC], AX	3	1	–	(BC + word) ← AX			
		AX, ES:word[BC]	4	2	5	AX ← ((ES, BC) + word)			
		ES:word[BC], AX	4	2	–	((ES, BC) + word) ← AX			

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (6/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit data transfer	MOVW	BC, !addr16	3	1	4	BC ← (addr16)			
		BC, ES:!addr16	4	2	5	BC ← (ES, addr16)			
		DE, !addr16	3	1	4	DE ← (addr16)			
		DE, ES:!addr16	4	2	5	DE ← (ES, addr16)			
		HL, !addr16	3	1	4	HL ← (addr16)			
		HL, ES:!addr16	4	2	5	HL ← (ES, addr16)			
		BC, saddrp	2	1	–	BC ← (saddrp)			
		DE, saddrp	2	1	–	DE ← (saddrp)			
		HL, saddrp	2	1	–	HL ← (saddrp)			
	XCHW	AX, rp <sup>Note 3</sup>	1	1	–	AX ↔ rp			
	ONEW	AX	1	1	–	AX ← 0001H			
		BC	1	1	–	BC ← 0001H			
	CLRW	AX	1	1	–	AX ← 0000H			
		BC	1	1	–	BC ← 0000H			
8-bit operation	ADD	A, #byte	2	1	–	A, CY ← A + byte	x	x	x
		saddr, #byte	3	2	–	(saddr), CY ← (saddr)+byte	x	x	x
		A, r <sup>Note 4</sup>	2	1	–	A, CY ← A + r	x	x	x
		r, A	2	1	–	r, CY ← r + A	x	x	x
		A, !addr16	3	1	4	A, CY ← A + (addr16)	x	x	x
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16)	x	x	x
		A, saddr	2	1	–	A, CY ← A + (saddr)	x	x	x
		A, [HL]	1	1	4	A, CY ← A + (HL)	x	x	x
		A, ES:[HL]	2	2	5	A, CY ← A + (ES, HL)	x	x	x
		A, [HL+byte]	2	1	4	A, CY ← A + (HL+byte)	x	x	x
		A, ES:[HL+byte]	3	2	5	A, CY ← A + ((ES, HL)+byte)	x	x	x
		A, [HL+B]	2	1	4	A, CY ← A + (HL+B)	x	x	x
		A, ES:[HL+B]	3	2	5	A, CY ← A + ((ES, HL)+B)	x	x	x
		A, [HL+C]	2	1	4	A, CY ← A + (HL+C)	x	x	x
A, ES:[HL+C]	3	2	5	A, CY ← A + ((ES, HL) + C)	x	x	x		

- Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
- 2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.
- 3.** Except rp = AX
- 4.** Except r = A

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (7/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	ADDC	A, #byte	2	1	–	A, CY ← A+byte+CY	x	x	x
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) +byte+CY	x	x	x
		A, r <sup>Note 3</sup>	2	1	–	A, CY ← A + r + CY	x	x	x
		r, A	2	1	–	r, CY ← r + A + CY	x	x	x
		A, laddr16	3	1	4	A, CY ← A + (addr16)+CY	x	x	x
		A, ES:!addr16	4	2	5	A, CY ← A + (ES, addr16)+CY	x	x	x
		A, saddr	2	1	–	A, CY ← A + (saddr)+CY	x	x	x
		A, [HL]	1	1	4	A, CY ← A+ (HL) + CY	x	x	x
		A, ES:[HL]	2	2	5	A,CY ← A+ (ES, HL) + CY	x	x	x
		A, [HL+byte]	2	1	4	A, CY ← A+ (HL+byte) + CY	x	x	x
		A, ES:[HL+byte]	3	2	5	A,CY ← A+ ((ES, HL)+byte) + CY	x	x	x
		A, [HL+B]	2	1	4	A, CY ← A+ (HL+B) +CY	x	x	x
		A, ES:[HL+B]	3	2	5	A,CY ← A+((ES, HL)+B)+CY	x	x	x
		A, [HL+C]	2	1	4	A, CY ← A+ (HL+C)+CY	x	x	x
		A, ES:[HL+C]	3	2	5	A,CY ← A+ ((ES, HL)+C)+CY	x	x	x
	SUB	A, #byte	2	1	–	A, CY ← A – byte	x	x	x
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) – byte	x	x	x
		A, r <sup>Note 3</sup>	2	1	–	A, CY ← A – r	x	x	x
		r, A	2	1	–	r, CY ← r – A	x	x	x
		A, laddr16	3	1	4	A, CY ← A – (addr16)	x	x	x
		A, ES:!addr16	4	2	5	A, CY ← A – (ES, addr16)	x	x	x
		A, saddr	2	1	–	A, CY ← A – (saddr)	x	x	x
		A, [HL]	1	1	4	A, CY ← A – (HL)	x	x	x
		A, ES:[HL]	2	2	5	A,CY ← A – (ES, HL)	x	x	x
		A, [HL+byte]	2	1	4	A, CY ← A – (HL+byte)	x	x	x
		A, ES:[HL+byte]	3	2	5	A,CY ← A – ((ES, HL)+byte)	x	x	x
		A, [HL+B]	2	1	4	A, CY ← A – (HL+B)	x	x	x
		A, ES:[HL+B]	3	2	5	A,CY ← A – ((ES, HL)+B)	x	x	x
A, [HL+C]	2	1	4	A, CY ← A – (HL+C)	x	x	x		
A, ES:[HL+C]	3	2	5	A,CY ← A – ((ES, HL)+C)	x	x	x		

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**3.** Except  $r = A$

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (8/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	SUBC	A, #byte	2	1	–	A, CY ← A – byte – CY	x	x	x
		saddr, #byte	3	2	–	(saddr), CY ← (saddr) – byte – CY	x	x	x
		A, r <sup>Note 3</sup>	2	1	–	A, CY ← A – r – CY	x	x	x
		r, A	2	1	–	r, CY ← r – A – CY	x	x	x
		A, laddr16	3	1	4	A, CY ← A – (addr16) – CY	x	x	x
		A, ES:!addr16	4	2	5	A, CY ← A – (ES, addr16) – CY	x	x	x
		A, saddr	2	1	–	A, CY ← A – (saddr) – CY	x	x	x
		A, [HL]	1	1	4	A, CY ← A – (HL) – CY	x	x	x
		A, ES:[HL]	2	2	5	A, CY ← A – (ES, HL) – CY	x	x	x
		A, [HL+byte]	2	1	4	A, CY ← A – (HL+byte) – CY	x	x	x
		A, ES:[HL+byte]	3	2	5	A, CY ← A – ((ES, HL)+byte) – CY	x	x	x
		A, [HL+B]	2	1	4	A, CY ← A – (HL+B) – CY	x	x	x
		A, ES:[HL+B]	3	2	5	A, CY ← A – ((ES, HL)+B) – CY	x	x	x
		A, [HL+C]	2	1	4	A, CY ← A – (HL+C) – CY	x	x	x
		A, ES:[HL+C]	3	2	5	A, CY ← A – ((ES:HL)+C) – CY	x	x	x
	AND	A, #byte	2	1	–	A ← A ∧ byte	x		
		saddr, #byte	3	2	–	(saddr) ← (saddr) ∧ byte	x		
		A, r <sup>Note 3</sup>	2	1	–	A ← A ∧ r	x		
		r, A	2	1	–	R ← r ∧ A	x		
		A, laddr16	3	1	4	A ← A ∧ (addr16)	x		
		A, ES:!addr16	4	2	5	A ← A ∧ (ES:addr16)	x		
		A, saddr	2	1	–	A ← A ∧ (saddr)	x		
		A, [HL]	1	1	4	A ← A ∧ (HL)	x		
		A, ES:[HL]	2	2	5	A ← A ∧ (ES:HL)	x		
		A, [HL+byte]	2	1	4	A ← A ∧ (HL+byte)	x		
		A, ES:[HL+byte]	3	2	5	A ← A ∧ ((ES:HL)+byte)	x		
		A, [HL+B]	2	1	4	A ← A ∧ (HL+B)	x		
		A, ES:[HL+B]	3	2	5	A ← A ∧ ((ES:HL)+B)	x		
A, [HL+C]	2	1	4	A ← A ∧ (HL+C)	x				
A, ES:[HL+C]	3	2	5	A ← A ∧ ((ES:HL)+C)	x				

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**3.** Except  $r = A$

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (9/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	OR	A, #byte	2	1	–	$A \leftarrow A \vee \text{byte}$		x	
		saddr, #byte	3	2	–	$(\text{saddr}) \leftarrow (\text{saddr}) \vee \text{byte}$		x	
		A, r <sup>Note 3</sup>	2	1	–	$A \leftarrow A \vee r$		x	
		r, A	2	1	–	$r \leftarrow r \vee A$		x	
		A, laddr16	3	1	4	$A \leftarrow A \vee (\text{addr}16)$		x	
		A, ES:!addr16	4	2	5	$A \leftarrow A \vee (\text{ES}:\text{addr}16)$		x	
		A, saddr	2	1	–	$A \leftarrow A \vee (\text{saddr})$		x	
		A, [HL]	1	1	4	$A \leftarrow A \vee (\text{H})$		x	
		A, ES:[HL]	2	2	5	$A \leftarrow A \vee (\text{ES}:\text{HL})$		x	
		A, [HL+byte]	2	1	4	$A \leftarrow A \vee (\text{HL}+\text{byte})$		x	
		A, ES:[HL+byte]	3	2	5	$A \leftarrow A \vee ((\text{ES}:\text{HL})+\text{byte})$		x	
		A, [HL+B]	2	1	4	$A \leftarrow A \vee (\text{HL}+\text{B})$		x	
		A, ES:[HL+B]	3	2	5	$A \leftarrow A \vee ((\text{ES}:\text{HL})+\text{B})$		x	
		A, [HL+C]	2	1	4	$A \leftarrow A \vee (\text{HL}+\text{C})$		x	
	A, ES:[HL+C]	3	2	5	$A \leftarrow A \vee ((\text{ES}:\text{HL})+\text{C})$		x		
	XOR	A, #byte	2	1	–	$A \leftarrow A \oplus \text{byte}$		x	
		saddr, #byte	3	2	–	$(\text{saddr}) \leftarrow (\text{saddr}) \oplus \text{byte}$		x	
		A, r <sup>Note 3</sup>	2	1	–	$A \leftarrow A \oplus r$		x	
		r, A	2	1	–	$r \leftarrow r \oplus A$		x	
		A, laddr16	3	1	4	$A \leftarrow A \oplus (\text{addr}16)$		x	
		A, ES:!addr16	4	2	5	$A \leftarrow A \oplus (\text{ES}:\text{addr}16)$		x	
		A, saddr	2	1	–	$A \leftarrow A \oplus (\text{saddr})$		x	
		A, [HL]	1	1	4	$A \leftarrow A \oplus (\text{HL})$		x	
		A, ES:[HL]	2	2	5	$A \leftarrow A \oplus (\text{ES}:\text{HL})$		x	
		A, [HL+byte]	2	1	4	$A \leftarrow A \oplus (\text{HL}+\text{byte})$		x	
		A, ES:[HL+byte]	3	2	5	$A \leftarrow A \oplus ((\text{ES}:\text{HL})+\text{byte})$		x	
		A, [HL+B]	2	1	4	$A \leftarrow A \oplus (\text{HL}+\text{B})$		x	
		A, ES:[HL+B]	3	2	5	$A \leftarrow A \oplus ((\text{ES}:\text{HL})+\text{B})$		x	
A, [HL+C]		2	1	4	$A \leftarrow A \oplus (\text{HL}+\text{C})$		x		
A, ES:[HL+C]	3	2	5	$A \leftarrow A \oplus ((\text{ES}:\text{HL})+\text{C})$		x			

**Notes 1.** Number of CPU clocks ( $f_{\text{CLK}}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{\text{CLK}}$ ) when the program memory area is accessed.

**3.** Except  $r = A$

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (10/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	CMP	A, #byte	2	1	–	A – byte	x	x	x
		laddr16, #byte	4	1	4	(addr16) – byte	x	x	x
		ES:laddr16, #byte	5	2	5	(ES:addr16) – byte	x	x	x
		saddr, #byte	3	1	–	(saddr) – byte	x	x	x
		A, r <sup>Note3</sup>	2	1	–	A – r	x	x	x
		r, A	2	1	–	r – A	x	x	x
		A, laddr16	3	1	4	A – (addr16)	x	x	x
		A, ES:laddr16	4	2	5	A – (ES:addr16)	x	x	x
		A, saddr	2	1	–	A – (saddr)	x	x	x
		A, [HL]	1	1	4	A – (HL)	x	x	x
		A, ES:[HL]	2	2	5	A – (ES:HL)	x	x	x
		A, [HL+byte]	2	1	4	A – (HL+byte)	x	x	x
		A, ES:[HL+byte]	3	2	5	A – ((ES:HL)+byte)	x	x	x
		A, [HL+B]	2	1	4	A – (HL+B)	x	x	x
		A, ES:[HL+B]	3	2	5	A – ((ES:HL)+B)	x	x	x
		A, [HL+C]	2	1	4	A – (HL+C)	x	x	x
	A, ES:[HL+C]	3	2	5	A – ((ES:HL)+C)	x	x	x	
	CMP0	A	1	1	–	A – 00H	x	0	0
		X	1	1	–	X – 00H	x	0	0
		B	1	1	–	B – 00H	x	0	0
		C	1	1	–	C – 00H	x	0	0
		laddr16	3	1	4	(addr16) – 00H	x	0	0
		ES:laddr16	4	2	5	(ES:addr16) – 00H	x	0	0
		saddr	2	1	–	(saddr) – 00H	x	0	0
	CMPS	X, [HL+byte]	3	1	4	X – (HL+byte)	x	x	x
		X, ES:[HL+byte]	4	2	5	X – ((ES:HL)+byte)	x	x	x

- Notes**
1. Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
  2. Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.
  3. Except  $r = A$

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (11/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
16-bit operation	ADDW	AX, #word	3	1	–	AX, CY ← AX+word	x	x	x
		AX, AX	1	1	–	AX, CY ← AX+AX	x	x	x
		AX, BC	1	1	–	AX, CY ← AX+BC	x	x	x
		AX, DE	1	1	–	AX, CY ← AX+DE	x	x	x
		AX, HL	1	1	–	AX, CY ← AX+HL	x	x	x
		AX, !addr16	3	1	4	AX, CY ← AX+(addr16)	x	x	x
		AX, ES:!addr16	4	2	5	AX, CY ← AX+(ES:addr16)	x	x	x
		AX, saddrp	2	1	–	AX, CY ← AX+(saddrp)	x	x	x
		AX, [HL+byte]	3	1	4	AX, CY ← AX+(HL+byte)	x	x	x
		AX, ES: [HL+byte]	4	2	5	AX, CY ← AX+((ES:HL)+byte)	x	x	x
	SUBW	AX, #word	3	1	–	AX, CY ← AX – word	x	x	x
		AX, BC	1	1	–	AX, CY ← AX – BC	x	x	x
		AX, DE	1	1	–	AX, CY ← AX – DE	x	x	x
		AX, HL	1	1	–	AX, CY ← AX – HL	x	x	x
		AX, !addr16	3	1	4	AX, CY ← AX – (addr16)	x	x	x
		AX, ES:!addr16	4	2	5	AX, CY ← AX – (ES:addr16)	x	x	x
		AX, saddrp	2	1	–	AX, CY ← AX – (saddrp)	x	x	x
		AX, [HL+byte]	3	1	4	AX, CY ← AX – (HL+byte)	x	x	x
		AX, ES: [HL+byte]	4	2	5	AX, CY ← AX – ((ES:HL)+byte)	x	x	x
		CMPW	AX, #word	3	1	–	AX – word	x	x
	AX, BC		1	1	–	AX – BC	x	x	x
	AX, DE		1	1	–	AX – DE	x	x	x
	AX, HL		1	1	–	AX – HL	x	x	x
	AX, !addr16		3	1	4	AX – (addr16)	x	x	x
	AX, ES:!addr16		4	2	5	AX – (ES:addr16)	x	x	x
	AX, saddrp		2	1	–	AX – (saddrp)	x	x	x
	AX, [HL+byte]		3	1	4	AX – (HL+byte)	x	x	x
AX, ES: [HL+byte]	4		2	5	AX – ((ES:HL)+byte)	x	x	x	

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (12/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
Multiply, Divide, Multiply & accumulate	MULU	X	1	1	–	$AX \leftarrow A \times X$			

**Notes** 1. Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

2. Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.



Table 26-5. Operation List (13/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Increment/ decrement	INC	r	1	1	–	$r \leftarrow r+1$	x	x	
		laddr16	3	2	–	$(addr16) \leftarrow (addr16)+1$	x	x	
		ES:laddr16	4	3	–	$(ES, addr16) \leftarrow (ES, addr16)+1$	x	x	
		saddr	2	2	–	$(saddr) \leftarrow (saddr)+1$	x	x	
		[HL+byte]	3	2	–	$(HL+byte) \leftarrow (HL+byte)+1$	x	x	
		ES: [HL+byte]	4	3	–	$((ES:HL)+byte) \leftarrow ((ES:HL)+byte)+1$	x	x	
	DEC	r	1	1	–	$r \leftarrow r-1$	x	x	
		laddr16	3	2	–	$(addr16) \leftarrow (addr16)-1$	x	x	
		ES:laddr16	4	3	–	$(ES, addr16) \leftarrow (ES, addr16)-1$	x	x	
		saddr	2	2	–	$(saddr) \leftarrow (saddr)-1$	x	x	
		[HL+byte]	3	2	–	$(HL+byte) \leftarrow (HL+byte)-1$	x	x	
		ES: [HL+byte]	4	3	–	$((ES:HL)+byte) \leftarrow ((ES:HL)+byte)-1$	x	x	
	INCW	rp	1	1	–	$rp \leftarrow rp+1$			
		laddr16	3	2	–	$(addr16) \leftarrow (addr16)+1$			
		ES:laddr16	4	3	–	$(ES, addr16) \leftarrow (ES, addr16)+1$			
		saddrp	2	2	–	$(saddrp) \leftarrow (saddrp)+1$			
		[HL+byte]	3	2	–	$(HL+byte) \leftarrow (HL+byte)+1$			
		ES: [HL+byte]	4	3	–	$((ES:HL)+byte) \leftarrow ((ES:HL)+byte)+1$			
	DECW	rp	1	1	–	$rp \leftarrow rp-1$			
		laddr16	3	2	–	$(addr16) \leftarrow (addr16)-1$			
		ES:laddr16	4	3	–	$(ES, addr16) \leftarrow (ES, addr16)-1$			
saddrp		2	2	–	$(saddrp) \leftarrow (saddrp)-1$				
[HL+byte]		3	2	–	$(HL+byte) \leftarrow (HL+byte)-1$				
ES: [HL+byte]		4	3	–	$((ES:HL)+byte) \leftarrow ((ES:HL)+byte)-1$				
Shift	SHR	A, cnt	2	1	–	$(CY \leftarrow A_0, A_{m-1} \leftarrow A_m, A_7 \leftarrow 0) \times cnt$			x
	SHRW	AX, cnt	2	1	–	$(CY \leftarrow AX_0, AX_{m-1} \leftarrow AX_m, AX_{15} \leftarrow 0) \times cnt$			x
	SHL	A, cnt	2	1	–	$(CY \leftarrow A_7, A_m \leftarrow A_{m-1}, A_0 \leftarrow 0) \times cnt$			x
		B, cnt	2	1	–	$(CY \leftarrow B_7, B_m \leftarrow B_{m-1}, B_0 \leftarrow 0) \times cnt$			x
		C, cnt	2	1	–	$(CY \leftarrow C_7, C_m \leftarrow C_{m-1}, C_0 \leftarrow 0) \times cnt$			x
	SHLW	AX, cnt	2	1	–	$(CY \leftarrow AX_{15}, AX_m \leftarrow AX_{m-1}, AX_0 \leftarrow 0) \times cnt$			x
		BC, cnt	2	1	–	$(CY \leftarrow BC_{15}, BC_m \leftarrow BC_{m-1}, BC_0 \leftarrow 0) \times cnt$			x
	SAR	A, cnt	2	1	–	$(CY \leftarrow A_0, A_{m-1} \leftarrow A_m, A_7 \leftarrow A_7) \times cnt$			x
SARW	AX, cnt	2	1	–	$(CY \leftarrow AX_0, AX_{m-1} \leftarrow AX_m, AX_{15} \leftarrow AX_{15}) \times cnt$			x	

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remarks 1.** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**2.** cnt indicates the bit shift count.

Table 26-5. Operation List (14/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Rotate	ROR	A, 1	2	1	–	$(CY, A_7 \leftarrow A_0, A_{m-1} \leftarrow A_m) \times 1$			x
	ROL	A, 1	2	1	–	$(CY, A_0 \leftarrow A_7, A_{m+1} \leftarrow A_m) \times 1$			x
	RORC	A, 1	2	1	–	$(CY \leftarrow A_0, A_7 \leftarrow CY, A_{m-1} \leftarrow A_m) \times 1$			x
	ROLC	A, 1	2	1	–	$(CY \leftarrow A_7, A_0 \leftarrow CY, A_{m+1} \leftarrow A_m) \times 1$			x
	ROLWC	AX,1	2	1	–	$(CY \leftarrow AX_{15}, AX_0 \leftarrow CY, AX_{m+1} \leftarrow AX_m) \times 1$			x
		BC,1	2	1	–	$(CY \leftarrow BC_{15}, BC_0 \leftarrow CY, BC_{m+1} \leftarrow BC_m) \times 1$			x
Bit manipulate	MOV1	CY, A.bit	2	1	–	$CY \leftarrow A.bit$			x
		A.bit, CY	2	1	–	$A.bit \leftarrow CY$			
		CY, PSW.bit	3	1	–	$CY \leftarrow PSW.bit$			x
		PSW.bit, CY	3	4	–	$PSW.bit \leftarrow CY$	x	x	
		CY, saddr.bit	3	1	–	$CY \leftarrow (saddr).bit$			x
		saddr.bit, CY	3	2	–	$(saddr).bit \leftarrow CY$			
		CY, sfr.bit	3	1	–	$CY \leftarrow sfr.bit$			x
		sfr.bit, CY	3	2	–	$sfr.bit \leftarrow CY$			
		CY,[HL].bit	2	1	4	$CY \leftarrow (HL).bit$			x
		[HL].bit, CY	2	2	–	$(HL).bit \leftarrow CY$			
	CY, ES:[HL].bit	3	2	5	$CY \leftarrow (ES, HL).bit$			x	
	ES:[HL].bit, CY	3	3	–	$(ES, HL).bit \leftarrow CY$				
	AND1	CY, A.bit	2	1	–	$CY \leftarrow CY \wedge A.bit$			x
		CY, PSW.bit	3	1	–	$CY \leftarrow CY \wedge PSW.bit$			x
		CY, saddr.bit	3	1	–	$CY \leftarrow CY \wedge (saddr).bit$			x
		CY, sfr.bit	3	1	–	$CY \leftarrow CY \wedge sfr.bit$			x
		CY,[HL].bit	2	1	4	$CY \leftarrow CY \wedge (HL).bit$			x
		CY, ES:[HL].bit	3	2	5	$CY \leftarrow CY \wedge (ES, HL).bit$			x
	OR1	CY, A.bit	2	1	–	$CY \leftarrow CY \vee A.bit$			x
		CY, PSW.bit	3	1	–	$CY \leftarrow CY \vee PSW.bit$			x
CY, saddr.bit		3	1	–	$CY \leftarrow CY \vee (saddr).bit$			x	
CY, sfr.bit		3	1	–	$CY \leftarrow CY \vee sfr.bit$			x	
CY, [HL].bit		2	1	4	$CY \leftarrow CY \vee (HL).bit$			x	
CY, ES:[HL].bit		3	2	5	$CY \leftarrow CY \vee (ES, HL).bit$			x	

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (15/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Bit manipulate	XOR1	CY, A.bit	2	1	–	$CY \leftarrow CY \oplus A.bit$			x
		CY, PSW.bit	3	1	–	$CY \leftarrow CY \oplus PSW.bit$			x
		CY, saddr.bit	3	1	–	$CY \leftarrow CY \oplus (saddr).bit$			x
		CY, sfr.bit	3	1	–	$CY \leftarrow CY \oplus sfr.bit$			x
		CY, [HL].bit	2	1	4	$CY \leftarrow CY \oplus (HL).bit$			x
		CY, ES:[HL].bit	3	2	5	$CY \leftarrow CY \oplus (ES, HL).bit$			x
	SET1	A.bit	2	1	–	$A.bit \leftarrow 1$			
		PSW.bit	3	4	–	$PSW.bit \leftarrow 1$	x	x	x
		!addr16.bit	4	2	–	$(addr16).bit \leftarrow 1$			
		ES:!addr16.bit	5	3	–	$(ES, addr16).bit \leftarrow 1$			
		saddr.bit	3	2	–	$(saddr).bit \leftarrow 1$			
		sfr.bit	3	2	–	$sfr.bit \leftarrow 1$			
		[HL].bit	2	2	–	$(HL).bit \leftarrow 1$			
		ES:[HL].bit	3	3	–	$(ES, HL).bit \leftarrow 1$			
	CLR1	A.bit	2	1	–	$A.bit \leftarrow 0$			
		PSW.bit	3	4	–	$PSW.bit \leftarrow 0$	x	x	x
		!addr16.bit	4	2	–	$(addr16).bit \leftarrow 0$			
		ES:!addr16.bit	5	3	–	$(ES, addr16).bit \leftarrow 0$			
		saddr.bit	3	2	–	$(saddr).bit \leftarrow 0$			
		sfr.bit	3	2	–	$sfr.bit \leftarrow 0$			
		[HL].bit	2	2	–	$(HL).bit \leftarrow 0$			
		ES:[HL].bit	3	3	–	$(ES, HL).bit \leftarrow 0$			
	SET1	CY	2	1	–	$CY \leftarrow 1$			1
	CLR1	CY	2	1	–	$CY \leftarrow 0$			0
	NOT1	CY	2	1	–	$CY \leftarrow \overline{CY}$			x

**Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed

**2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (16/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Call/ return	CALL	rp	2	3	–	(SP – 2) ← (PC+2) <sub>s</sub> , (SP – 3) ← (PC+2) <sub>H</sub> , (SP – 4) ← (PC+2) <sub>L</sub> , PC ← CS, rp, SP ← SP – 4			
		\$!addr20	3	3	–	(SP – 2) ← (PC+3) <sub>s</sub> , (SP – 3) ← (PC+3) <sub>H</sub> , (SP – 4) ← (PC+3) <sub>L</sub> , PC ← PC+3+jdisp16, SP ← SP – 4			
		!addr16	3	3	–	(SP – 2) ← (PC+3) <sub>s</sub> , (SP – 3) ← (PC+3) <sub>H</sub> , (SP – 4) ← (PC+3) <sub>L</sub> , PC ← 0000, addr16, SP ← SP – 4			
		!!addr20	4	3	–	(SP – 2) ← (PC+4) <sub>s</sub> , (SP – 3) ← (PC+4) <sub>H</sub> , (SP – 4) ← (PC+4) <sub>L</sub> , PC ← addr20, SP ← SP – 4			
	CALLT	[addr5]	2	5	–	(SP – 2) ← (PC+2) <sub>s</sub> , (SP – 3) ← (PC+2) <sub>H</sub> , (SP – 4) ← (PC+2) <sub>L</sub> , PC <sub>s</sub> ← 0000, PC <sub>H</sub> ← (0000, addr5+1), PC <sub>L</sub> ← (0000, addr5), SP ← SP – 4			
	BRK	-	2	5	–	(SP – 1) ← PSW, (SP – 2) ← (PC+2) <sub>s</sub> , (SP – 3) ← (PC+2) <sub>H</sub> , (SP – 4) ← (PC+2) <sub>L</sub> , PC <sub>s</sub> ← 0000, PC <sub>H</sub> ← (0007FH), PC <sub>L</sub> ← (0007EH), SP ← SP – 4, IE ← 0			
	RET	-	1	6	–	PC <sub>L</sub> ← (SP), PC <sub>H</sub> ← (SP+1), PC <sub>s</sub> ← (SP+2), SP ← SP+4			
RETI	-	2	6	–	PC <sub>L</sub> ← (SP), PC <sub>H</sub> ← (SP+1), PC <sub>s</sub> ← (SP+2), PSW ← (SP+3), SP ← SP+4	R	R	R	
RETB	-	2	6	–	PC <sub>L</sub> ← (SP), PC <sub>H</sub> ← (SP+1), PC <sub>s</sub> ← (SP+2), PSW ← (SP+3), SP ← SP+4	R	R	R	

- Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
- 2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (17/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Stack manipulate	PUSH	PSW	2	1	–	(SP – 1) ← PSW, (SP – 2) ← 00H, SP ← SP – 2			
		rp	1	1	–	(SP – 1) ← rp <sub>H</sub> , (SP – 2) ← rp <sub>L</sub> , SP ← SP – 2			
	POP	PSW	2	3	–	PSW ← (SP+1), SP ← SP + 2	R	R	R
		rp	1	1	–	rp <sub>L</sub> ← (SP), rp <sub>H</sub> ← (SP+1), SP ← SP + 2			
	MOVW	SP, #word	4	1	–	SP ← word			
		SP, AX	2	1	–	SP ← AX			
		AX, SP	2	1	–	AX ← SP			
		HL, SP	3	1	–	HL ← SP			
		BC, SP	3	1	–	BC ← SP			
		DE, SP	3	1	–	DE ← SP			
ADDW	SP, #byte	2	1	–	SP ← SP + byte				
SUBW	SP, #byte	2	1	–	SP ← SP – byte				
Un-conditional branch	BR	AX	2	3	–	PC ← CS, AX			
		\$addr20	2	3	–	PC ← PC + 2 + jdisp8			
		!addr20	3	3	–	PC ← PC + 3 + jdisp16			
		!addr16	3	3	–	PC ← 0000, addr16			
		!!addr20	4	3	–	PC ← addr20			
Conditional branch	BC	\$addr20	2	2/4 <sup>Note3</sup>	–	PC ← PC + 2 + jdisp8 if CY = 1			
	BNC	\$addr20	2	2/4 <sup>Note3</sup>	–	PC ← PC + 2 + jdisp8 if CY = 0			
	BZ	\$addr20	2	2/4 <sup>Note3</sup>	–	PC ← PC + 2 + jdisp8 if Z = 1			
	BNZ	\$addr20	2	2/4 <sup>Note3</sup>	–	PC ← PC + 2 + jdisp8 if Z = 0			
	BH	\$addr20	3	2/4 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if (Z∨CY)=0			
	BNH	\$addr20	3	2/4 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if (Z∨CY)=1			
	BT	saddr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if (saddr).bit = 1			
		sfr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if sfr.bit = 1			
		A.bit, \$addr20	3	3/5 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if A.bit = 1			
		PSW.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if PSW.bit = 1			
		[HL].bit, \$addr20	3	3/5 <sup>Note3</sup>	6/7	PC ← PC + 3 + jdisp8 if (HL).bit = 1			
ES:[HL].bit, \$addr20	4	4/6 <sup>Note3</sup>	7/8	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1					

- Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
- 2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.
- 3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

Table 26-5. Operation List (18/18)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Clocks	Flag		
				Note 1	Note 2		Z	AC	CY
Conditional branch	BF	saddr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if (saddr).bit = 0			
		sfr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if sfr.bit = 0			
		A.bit, \$addr20	3	3/5 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if A.bit = 0			
		PSW.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if PSW.bit = 0			
		[HL].bit, \$addr20	3	3/5 <sup>Note3</sup>	6/7	PC ← PC + 3 + jdisp8 if (HL).bit = 0			
		ES:[HL].bit, \$addr20	4	4/6 <sup>Note3</sup>	7/8	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 0			
	BTCLR	saddr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if (saddr).bit = 1 then reset (saddr).bit			
		sfr.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit			
		A.bit, \$addr20	3	3/5 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit			
		PSW.bit, \$addr20	4	3/5 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	x	x	x
		[HL].bit, \$addr20	3	3/5 <sup>Note3</sup>	–	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit			
		ES:[HL].bit, \$addr20	4	4/6 <sup>Note3</sup>	–	PC ← PC + 4 + jdisp8 if (ES, HL).bit = 1 then reset (ES, HL).bit			
Conditional skip	SKC	–	2	1	–	Next instruction skip if CY = 1			
	SKNC	–	2	1	–	Next instruction skip if CY = 0			
	SKZ	–	2	1	–	Next instruction skip if Z = 1			
	SKNZ	–	2	1	–	Next instruction skip if Z = 0			
	SKH	–	2	1	–	Next instruction skip if (Z∨CY)=0			
	SKNH	–	2	1	–	Next instruction skip if (Z∨CY)=1			
CPU control	SEL <sup>Note4</sup>	Rbn	2	1	–	RBS[1:0] ← n			
	NOP	–	1	1	–	No Operation			
	EI	–	3	4	–	IE ← 1 (Enable Interrupt)			
	DI	–	3	4	–	IE ← 0 (Disable Interrupt)			
	HALT	–	2	3	–	Set HALT Mode			
	STOP	–	2	3	–	Set STOP Mode			

- Notes 1.** Number of CPU clocks ( $f_{CLK}$ ) when the internal RAM area, SFR area, or extended SFR area is accessed, or when no data is accessed.
- 2.** Number of CPU clocks ( $f_{CLK}$ ) when the program memory area is accessed.
- 3.** This indicates the number of clocks “when condition is not met/when condition is met”.

**Remark** Number of clock is when program exists in the internal ROM (flash memory) area. If fetching the instruction from the internal RAM area, the number becomes double number plus 3 clocks at a maximum.

**CHAPTER 27 ELECTRICAL SPECIFICATIONS**

- Cautions**
1. The R7F0C010 microcontrollers have an on-chip debug function, which is provided for development and evaluation. Do not use the on-chip debug function in products designated for mass production, because the guaranteed number of rewritable times of the flash memory may be exceeded when this function is used, and product reliability therefore cannot be guaranteed. Renesas Electronics is not liable for problems occurring when the on-chip debug function is used.
  2. The pins mounted depend on the product. See 2.1 Pin Function List to 2.1.3 Pins for each product (pins other than port pins).

## 27.1 Absolute Maximum Ratings

### Absolute Maximum Ratings ( $T_A = 25^\circ\text{C}$ ) (1/2)

Parameter	Symbols	Conditions	Ratings	Unit
Supply voltage	$V_{DD}$		-0.5 to +4.6	V
	$V_{SS}$		-0.5 to +0.3	V
REGC pin input voltage	$V_{IREGC}$	REGC	-0.3 to +2.8 and -0.3 to $V_{DD} + 0.3$ <sup>Note 1</sup>	V
Input voltage	$V_{I1}$	P10 to P17, P20 to P27, P30 to P35, P40, P121, P122, P137	-0.3 to $V_{DD} + 0.3$ <sup>Note 2</sup>	V
	$V_{I2}$	P60, P61 (N-ch open-drain)	-0.3 to +6.5	V
Output voltage	$V_{O1}$	P10 to P17, P20 to P27, P30 to P35, P40, P60, P61, P121, P122, P137	-0.3 to $V_{DD} + 0.3$ <sup>Note 2</sup>	V
Analog input voltage	$V_{AI1}$	ANI0 to ANI7, ANI16	-0.3 to $V_{DD} + 0.3$ and -0.3 to $AV_{REF(+)} + 0.3$ <sup>Notes 2, 3</sup>	V

**Notes 1.** Connect the REGC pin to  $V_{SS}$  via a capacitor (0.47 to 1  $\mu\text{F}$ ). This value regulates the absolute maximum rating of the REGC pin. Do not use this pin with voltage applied to it.

**2.** Must be 4.6 V or lower.

**3.** Do not exceed  $AV_{REF(+)} + 0.3$  V in case of A/D conversion target pin.

**Caution** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

**Remarks 1.** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

**2.**  $AV_{REF(+)}$ : + side reference voltage of the A/D converter.



**Absolute Maximum Ratings (T<sub>A</sub> = 25°C) (2/2)**

Parameter	Symbols	Conditions		Ratings	Unit
Output current, high	I <sub>OH1</sub>	Per pin	P10 to P17, P30 to P35, P40	-40	mA
		Total of all pins -170 mA	P40	-40	mA
			P10 to P17, P30, P35	-100	mA
	I <sub>OH2</sub>	Per pin	P20 to P27	-0.5	mA
Total of all pins		-2		mA	
Output current, low	I <sub>OL1</sub>	Per pin	P10 to P17, P30 to P35, P40, P60, P61	40	mA
		Total of all pins 170 mA	P40	40	mA
			P10 to P17, P30 to P35, P60, P61	100	mA
	I <sub>OL2</sub>	Per pin	P20 to P27	1	mA
		Total of all pins		5	mA
Operating ambient temperature	T <sub>A</sub>	In normal operation mode		-40 to +85	°C
		In flash memory programming mode		0 to +40	
Storage temperature	T <sub>stg</sub>			-65 to +150	°C

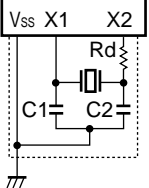
**Caution** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

## 27.2 Oscillator Characteristics

### 27.2.1 X1 oscillator characteristics

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Parameter	Resonator	Recommended Circuit	Conditions	MIN.	TYP.	MAX.	Unit
X1 clock oscillation frequency ( $f_x$ ) <sup>Note</sup>	Ceramic resonator/ crystal resonator		$2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	1.0		20.0	MHz

**Note** Indicates only oscillator characteristics. See 27.4 AC Characteristics for instruction execution time.

**Cautions 1.** When using the X1 oscillator, wire as follows in the area enclosed by the broken lines in the above figures to avoid an adverse effect from wiring capacitance.

- Keep the wiring length as short as possible.
  - Do not cross the wiring with the other signal lines.
  - Do not route the wiring near a signal line through which a high fluctuating current flows.
  - Always make the ground point of the oscillator capacitor the same potential as  $V_{SS}$ .
  - Do not ground the capacitor to a ground pattern through which a high current flows.
  - Do not fetch signals from the oscillator.
2. Since the CPU is started by the high-speed on-chip oscillator clock after a reset release, check the X1 clock oscillation stabilization time using the oscillation stabilization time counter status register (OSTC) by the user. Determine the oscillation stabilization time of the OSTC register and the oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.

### 27.2.2 On-chip oscillator characteristics

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Oscillators	Parameters	Conditions	MIN.	TYP.	MAX.	Unit
High-speed on-chip oscillator clock frequency <sup>Notes 1, 2</sup>	$f_{iH}$	32 MHz selected	31.36	32.00	32.64	MHz
		24 MHz selected	23.52	24.00	24.48	MHz
Low-speed on-chip oscillator clock frequency <sup>Note 2</sup>	$f_{iL}$		12.75	15.00	17.25	kHz

**Notes 1.** High-speed on-chip oscillator frequency is selected by bits 0 to 3 of option byte (000C2H) and bits 0 to 2 of HOCODIV register.

2. This indicates the oscillator characteristics only. See 27.4 AC Characteristics for instruction execution time.

## 27.3 DC Characteristics

### 27.3.1 Pin characteristics

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ ) (1/3)

Items	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Output current, high <sup>Note 1</sup>	I <sub>OH1</sub>	Per pin for P10 to P17, P30 to P35, P40			-2.0 <sup>Note 2</sup>	mA
		Total of P40 (When duty = 70% <sup>Note 3</sup> )			-2.0	mA
		Total of P10 to P17, P30 to P35 (When duty = 70% <sup>Note 3</sup> )			-19.0	mA
		Total of all pins (When duty = 70% <sup>Note 3</sup> )			-21.0	mA
	I <sub>OH2</sub>	Per pin for P20 to P27			-0.1	mA
		Total of all pins (When duty = 70% <sup>Note 3</sup> )			-0.8	mA

**Notes** 1. Value of current at which the device operation is guaranteed even if the current flows from the  $V_{DD}$  pin to an output pin.

2. Do not exceed the total current value.

3. Specification under conditions where the duty factor is 70%.

The output current value that has changed the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins =  $(I_{OH} \times 0.7)/(n \times 0.01)$

<Example> Where  $n = 50\%$  and  $I_{OH} = -10.0\text{ mA}$

$$\text{Total output current of pins} = (-10.0 \times 0.7)/(50 \times 0.01) = -14.0\text{ mA}$$

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

<R>

**(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V) (2/3)**

Items	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Output current, low <sup>Note 1</sup>	I <sub>OL1</sub>	Per pin for P10 to P17, P30 to P35, P40			3.0 <sup>Note 2</sup>	mA	
		Per pin for P60, P61			3.0 <sup>Note 2</sup>	mA	
		Total of P40 (When duty = 70% <sup>Note 3</sup> )			3.0	mA	
		Total of P10 to P17, P30 to P35, P60, P61 (When duty = 70% <sup>Note 3</sup> )			35.0	mA	
		Total of all pins (When duty = 70% <sup>Note 3</sup> )			38.0	mA	
	I <sub>OL2</sub>	Per pin for P20 to P27				0.4	mA
		Total of all pins (When duty = 70% <sup>Note 3</sup> )				3.2	mA

**Notes** 1. Value of current at which the device operation is guaranteed even if the current flows from an output pin to the V<sub>SS</sub> pin.

2. However, do not exceed the total current value.
3. Specification under conditions where the duty factor is 70%.

The output current value that has changed the duty ratio can be calculated with the following expression (when changing the duty factor from 70% to n%).

- Total output current of pins = (I<sub>OL</sub> × 0.7)/(n × 0.01)

<Example> Where n = 50% and I<sub>OL</sub> = 10.0 mA

$$\text{Total output current of pins} = (10.0 \times 0.7)/(50 \times 0.01) = 14.0 \text{ mA}$$

However, the current that is allowed to flow into one pin does not vary depending on the duty factor. A current higher than the absolute maximum rating must not flow into one pin.

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.

(TA = -40 to +85°C, 2.7 V ≤ VDD ≤ 3.6 V, VSS = 0 V) (3/3)

Items	Symbol	Conditions	MIN.	TYP.	MAX.	Unit		
Input voltage, high	V <sub>IH1</sub>	P10 to P17, P30 to P35, P40, P121, P122, P137, EXCLK, RESET	0.8V <sub>DD</sub>		V <sub>DD</sub>	V		
	V <sub>IH2</sub>	P20 to P27	0.7V <sub>DD</sub>		V <sub>DD</sub>	V		
	V <sub>IH3</sub>	P60, P61	0.7V <sub>DD</sub>		6.0	V		
Input voltage, low	V <sub>IL1</sub>	P10 to P17, P30 to P35, P40, P121, P122, P137, EXCLK, RESET	0		0.2V <sub>DD</sub>	V		
	V <sub>IL2</sub>	P20 to P27, P60, P61	0		0.3V <sub>DD</sub>	V		
Output voltage, high	V <sub>OH1</sub>	P10 to P17, P30 to P35, P40	I <sub>OH1</sub> = -2.0 mA	V <sub>DD</sub> - 0.6		V		
	V <sub>OH2</sub>	P20 to P27	I <sub>OH1</sub> = -100 μA	V <sub>DD</sub> - 0.5		V		
Output voltage, low	V <sub>OL1</sub>	P10 to P17, P30 to P35, P40	I <sub>OL1</sub> = 3.0 mA		0.6	V		
			I <sub>OL1</sub> = 1.5 mA		0.4	V		
	V <sub>OL2</sub>	P20 to P27	I <sub>OL2</sub> = 400 μA		0.4	V		
	V <sub>OL3</sub>	P60, P61	I <sub>OL3</sub> = 3.0 mA		0.4	V		
Input leakage current, high	I <sub>LIH1</sub>	P10 to P17, P20 to P27, P30 to P35, P40, P137, RESET	V <sub>I</sub> = V <sub>DD</sub>		1	μA		
			P121, P122 (X1, X2, EXCLK)	V <sub>I</sub> = V <sub>DD</sub>	In input port or external clock input	1	μA	
	In resonator connection	10			μA			
Input leakage current, low	I <sub>LIL1</sub>	P10 to P17, P20 to P27, P30 to P35, P40, P137, RESET	V <sub>I</sub> = V <sub>SS</sub>		-1	μA		
			P121, P122 (X1, X2, EXCLK)	V <sub>I</sub> = V <sub>SS</sub>	In input port or external clock input	-1	μA	
	In resonator connection	-10			μA			
	On-chip pull-up resistance	R <sub>U</sub>	P10 to P17, P30 to P35, P40	V <sub>I</sub> = V <sub>SS</sub> , In input port		10	20	100

**Remark** Unless specified otherwise, the characteristics of alternate-function pins are the same as those of the port pins.



(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V) (2/3)

Parameter	Symbol	Conditions				MIN.	TYP.	MAX.	Unit		
Supply current <sup>Note 1</sup>	I <sub>DD2</sub> <sup>Note 2</sup>	HALT mode	HS (high-speed main) mode <sup>Note 6</sup>	f <sub>IH</sub> = 32 MHz <sup>Note 4</sup>	V <sub>DD</sub> = 3.0 V		0.60	1.63	mA		
				f <sub>IH</sub> = 24 MHz <sup>Note 4</sup>	V <sub>DD</sub> = 3.0 V		0.49	1.28	mA		
				f <sub>IH</sub> = 16 MHz <sup>Note 4</sup>	V <sub>DD</sub> = 3.0 V		0.45	1.00	mA		
			LS (low-speed main) mode <sup>Note 6</sup>	f <sub>IH</sub> = 8 MHz <sup>Note 4</sup>	V <sub>DD</sub> = 3.0 V		320	530	μA		
				HS (high-speed main) mode <sup>Note 6</sup>	f <sub>MX</sub> = 20 MHz <sup>Note 3</sup>	V <sub>DD</sub> = 3.0 V	Square wave input		0.28	1.00	mA
					Resonator connection			0.49	1.17	mA	
		LS (low-speed main) mode <sup>Note 6</sup>	f <sub>MX</sub> = 10 MHz <sup>Note 3</sup>	V <sub>DD</sub> = 3.0 V	Square wave input		0.19	0.60	mA		
			Resonator connection			0.30	0.67	mA			
		LS (low-speed main) mode <sup>Note 6</sup>	f <sub>MX</sub> = 8 MHz <sup>Note 3</sup>	V <sub>DD</sub> = 3.0 V	Square wave input		95	330	μA		
					Resonator connection		145	380			
	I <sub>DD3</sub> <sup>Note 5</sup>	STOP mode	T <sub>A</sub> = -40°C					0.18		μA	
			T <sub>A</sub> = +25°C					0.23	0.50		
			T <sub>A</sub> = +50°C					0.26	1.10		
T <sub>A</sub> = +70°C					0.29	1.90					
T <sub>A</sub> = +85°C					0.90	3.30					

**Notes** 1. Total current flowing into V<sub>DD</sub>, including the input leakage current flowing when the level of the input pin is fixed to V<sub>DD</sub> or V<sub>SS</sub>. The values below the MAX. column include the peripheral operation current. However, not including the current flowing into the A/D converter, D/A converter, LVD circuit, I/O port, and on-chip pull-up/pull-down resistors.

2. During HALT instruction execution by flash memory.
3. When high-speed on-chip oscillator is stopped.
4. When high-speed system clock is stopped.
5. When high-speed on-chip oscillator and high-speed system clock are stopped. When watchdog timer is stopped. The values below the MAX. column include the leakage current.
6. Relationship between operation voltage width, operation frequency of CPU and operation mode is as below.
  - HS (high-speed main) mode: @1 MHz to 32 MHz
  - @1 MHz to 16 MHz
  - LS (low-speed main) mode: @1 MHz to 8 MHz

**Remarks** 1. f<sub>MX</sub>: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)

2. f<sub>IH</sub>: High-speed on-chip oscillator clock frequency
3. Except STOP mode, temperature condition of the TYP. value is T<sub>A</sub> = 25°C

**(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V) (3/3)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Watchdog timer operating current	I <sub>WDT</sub> <sup>Notes 1, 2</sup>	f <sub>IL</sub> = 15 kHz		0.22		μA
A/D converter operating current	I <sub>ADC</sub> <sup>Note 3</sup>	V <sub>DD</sub> = 3.0 V, When conversion at maximum speed		0.58	0.82	mA
A/D converter reference voltage current	I <sub>ADREF</sub>			75.0		μA
Temperature sensor operating current	I <sub>TMPS</sub>			75.0		μA
D/A converter operating current	I <sub>DAC</sub> <sup>Note 4</sup>				2	mA
LVD operating current	I <sub>LVD</sub> <sup>Note 5</sup>			0.08		μA
BGO operating current	I <sub>BGO</sub> <sup>Note 6</sup>			2.50	12.20	mA

- Notes**
- When high speed on-chip oscillator and high-speed system clock are stopped.
  - Current flowing only to the watchdog timer (including the operating current of the 15 kHz low-speed on-chip oscillator). The supply current of the R7F0C010 microcontrollers is the sum of I<sub>DD1</sub>, I<sub>DD2</sub> or I<sub>DD3</sub> and I<sub>WDT</sub> when the watchdog timer is in operation.
  - Current flowing only to the A/D converter. The supply current of the R7F0C010 microcontrollers is the sum of I<sub>DD1</sub> or I<sub>DD2</sub> and I<sub>ADC</sub> when the A/D converter operates in an operation mode or the HALT mode.
  - Current flowing only to the D/A converter. The supply current of the R7F0C010 microcontrollers is the sum of I<sub>DD1</sub> or I<sub>DD2</sub> and I<sub>DAC</sub> when the D/A converter operates in an operation mode or the HALT mode.
  - Current flowing only to the LVD circuit. The supply current value of the R7F0C010 microcontrollers is the sum of I<sub>DD1</sub>, I<sub>DD2</sub> or I<sub>DD3</sub> and I<sub>LVD</sub> when the LVD circuit is in operation.
  - Current flowing only to the BGO. The supply current of the R7F0C010 microcontrollers is the sum of I<sub>DD1</sub> or I<sub>DD2</sub> and I<sub>BGO</sub> when the BGO operates in an operation mode.

- Remarks**
- f<sub>IL</sub>: Low-speed on-chip oscillator clock frequency
  - f<sub>CLK</sub>: CPU/peripheral hardware clock frequency
  - Temperature condition of the TYP. value is T<sub>A</sub> = 25°C



## 27.4 AC Characteristics

(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

&lt;R&gt;

Items	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Instruction cycle (minimum instruction execution time)	T <sub>CY</sub>	Main system clock (f <sub>MAIN</sub> ) operation	HS (high-speed main) mode	0.03125		1	μs
			LS (low-speed main) mode	0.125		1	μs
		In the self-programming mode	HS (high-speed main) mode	0.03125		1	μs
			LS (low-speed main) mode	0.125		1	μs
External system clock frequency	f <sub>EX</sub>		1.0		20.0	MHz	
External system clock input high-level width, low-level width	t <sub>EXH</sub> , t <sub>EXL</sub>		24			ns	
TI00 to TI03 input high-level width, low-level width	t <sub>TIH</sub> , t <sub>TIL</sub>		1/f <sub>MCK</sub> +10			ns	
TO00 to TO03 output frequency	f <sub>TO</sub>	HS (high-speed main) mode			8	MHz	
		LS (low-speed main) mode			4	MHz	
PCLBUZ0, PCLBUZ1 output frequency	f <sub>PCL</sub>	HS (high-speed main) mode			8	MHz	
		LS (low-speed main) mode			4	MHz	
Interrupt input high-level width, low-level width	t <sub>INTH</sub> , t <sub>INTL</sub>	INTP0 to INTP5	1			μs	
RESET low-level width	t <sub>RSL</sub>		10			μs	

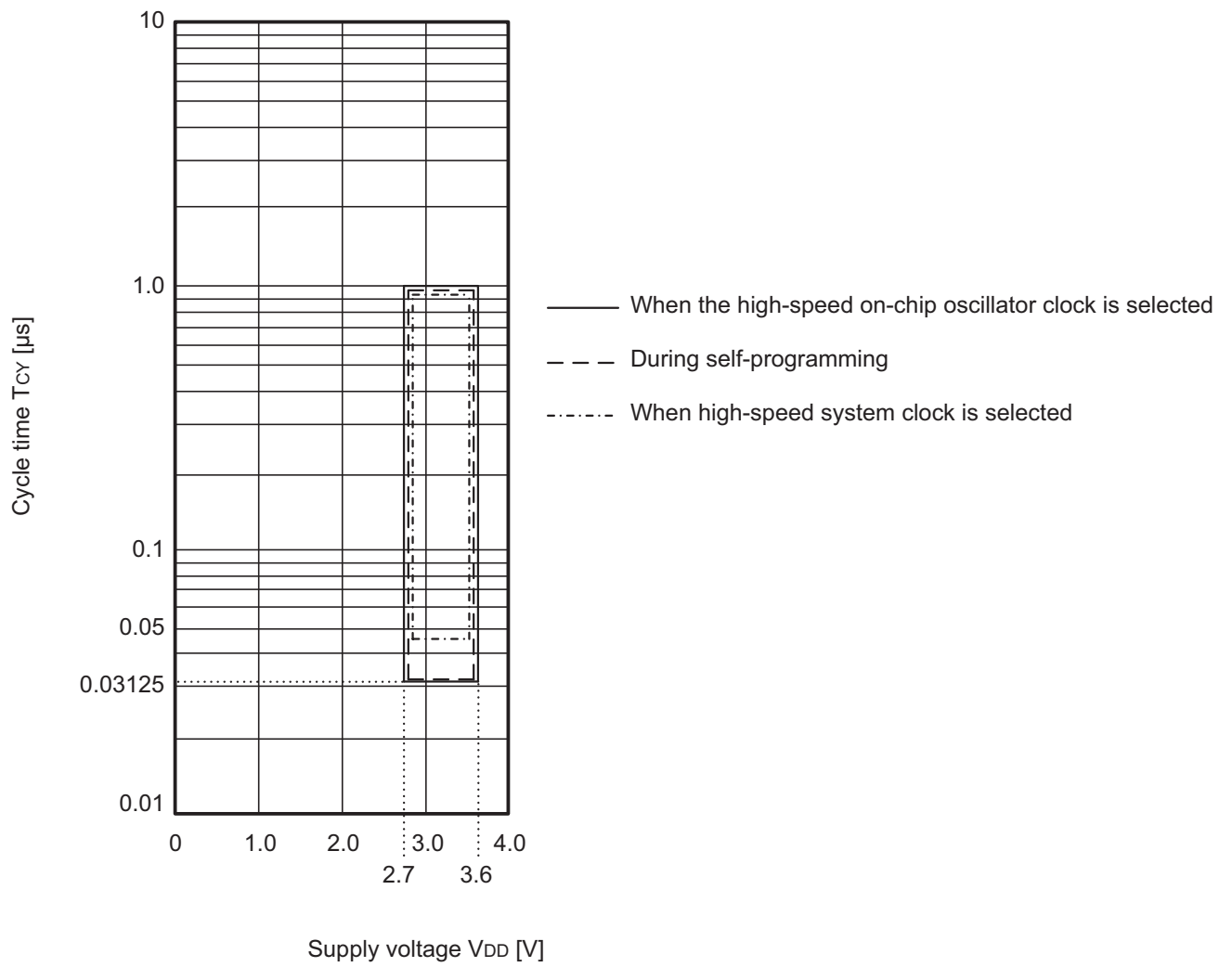
**Remark** f<sub>MCK</sub>: Timer array unit operation clock frequency

(Operation clock to be set by the CKSmn0 and CKSmn1 bits of timer mode register mn (TMRmn).

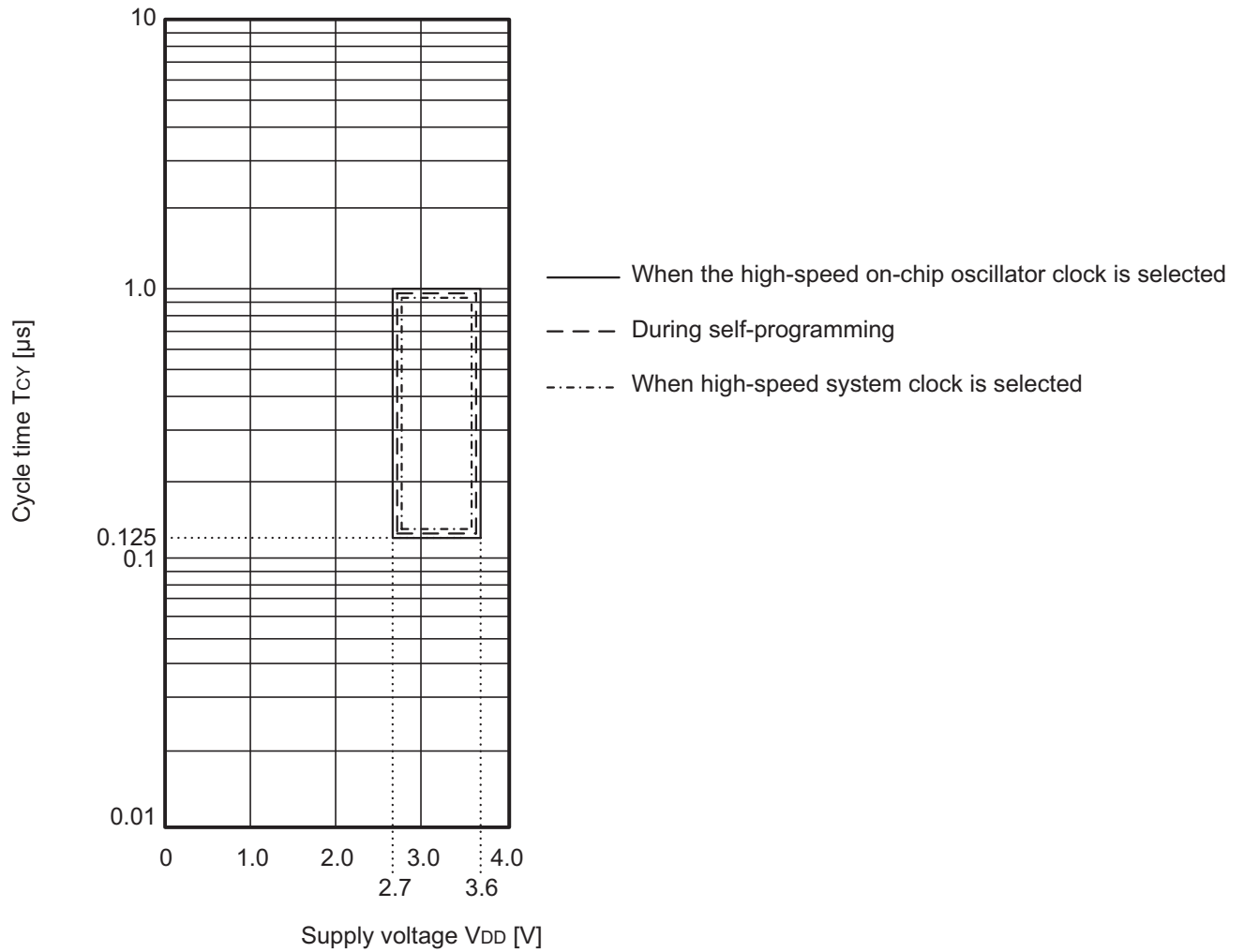
m: Unit number (m = 0), n: Channel number (n = 0 to 3))

<R> Minimum Instruction Execution Time during Main System Clock Operation

TCY vs V<sub>DD</sub> (HS (high-speed main) mode)



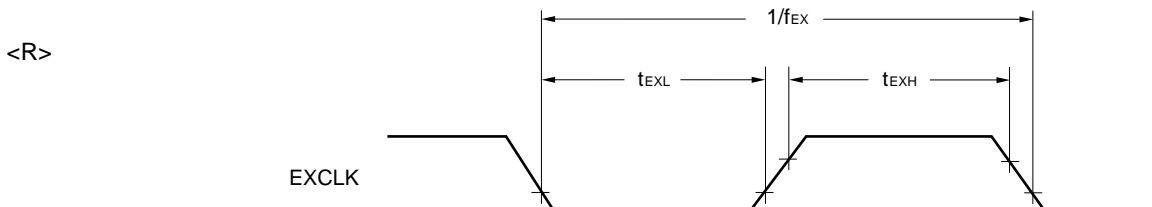
<R> TCY vs V<sub>DD</sub> (LS (low-speed main) mode)



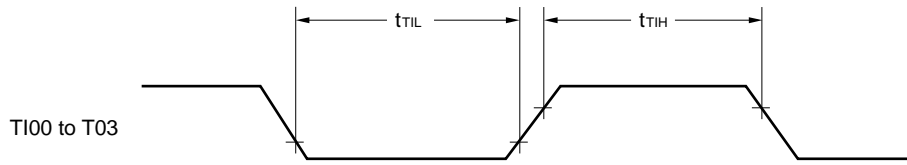
**AC Timing Test Points**



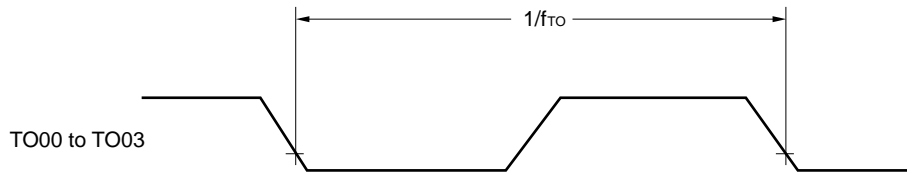
**External System Clock Timing**



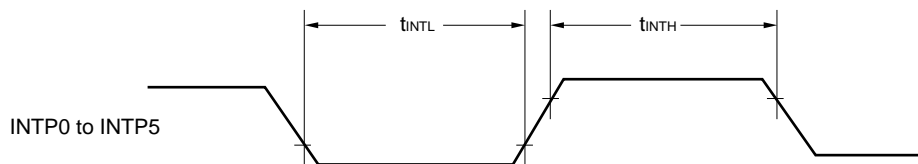
**TI/TO Timing**



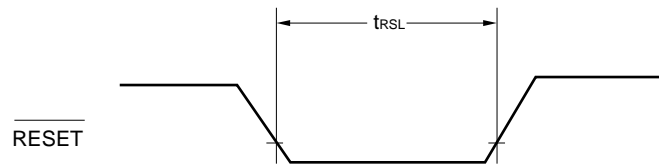
<R>



**Interrupt Request Input Timing**



**RESET Input Timing**



27.5 Peripheral Functions Characteristics

AC Timing Test Points

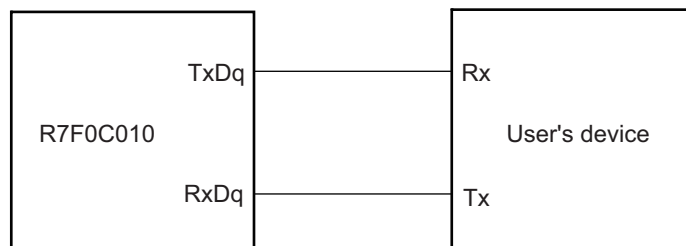


27.5.1 Serial array unit

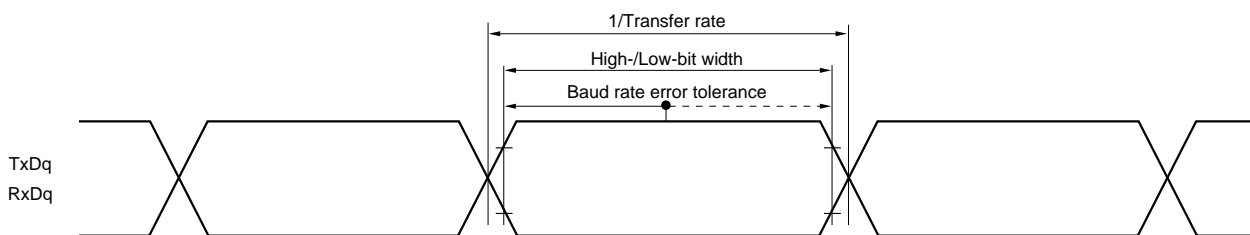
(1) During communication at same potential (UART mode) (dedicated baud rate generator output)  
 ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Parameter	Symbol	Conditions	HS (high-speed main) Mode		LS (low-speed main) Mode		Unit
			MIN.	TYP.	MIN.	MAX.	
Transfer rate <sup>Note</sup>		$2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$		$f_{MCK}/6$		$f_{MCK}/6$	bps
		Theoretical value of the maximum transfer rate $f_{MCK} = f_{CLK}$ <sup>Note 2</sup>		5.3		1.3	Mbps

UART mode connection diagram (during communication at same potential)



UART mode bit width (during communication at same potential) (reference)



- Notes**
- Transfer rate in the SNOOZE mode is 4800 bps only.
  - The maximum operating frequencies of the CPU/peripheral hardware clock ( $f_{CLK}$ ) are:  
 HS (high-speed main) mode: 32 MHz ( $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ )  
 LS (low-speed main) mode: 8 MHz ( $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ).

**Caution** Select the normal input buffer for the RxDq pin and the normal output mode for the TxDq pin by using port input mode register g (PIMg) and port output mode register g (POMg).

- Remarks**
- q: UART number (q = 0)
  - $f_{MCK}$ : Serial array unit operation clock frequency  
 (Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00, 01))

(2) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output , corresponding CSI00 only)

( $T_A = -40$  to  $+85$  °C,  $2.7\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

<R>

Parameter	Symbol	Conditions	HS (high-speed main) Mode		LS (low-speed main) Mode		Unit
			MIN.	TYP.	MIN.	MAX.	
SCKp cycle time	$t_{KCY1}$		83.3		250		ns
SCKp high-/low-level width	$t_{KH1}, t_{KL1}$		$t_{KCY1}/2 - 10$		$t_{KCY1}/2 - 50$		ns
SIp setup time (to SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{SIK1}$		33 <sup>Note 3</sup>		110		ns
SIp hold time (from SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{KSI1}$		10		10		ns
Delay time from SCKp $\downarrow$ to SOp output <sup>Note 2</sup>	$t_{KSO1}$	$C = 20\text{ pF}$ <sup>Note 4</sup>		10		10	ns

- Notes**
1. When  $DAPmn = 0$  and  $CKPmn = 0$ , or  $DAPmn = 1$  and  $CKPmn = 1$ . The SIp setup time becomes “to SCKp $\downarrow$ ” when  $DAPmn = 0$  and  $CKPmn = 1$ , or  $DAPmn = 1$  and  $CKPmn = 0$ .
  2. When  $DAPmn = 0$  and  $CKPmn = 0$ , or  $DAPmn = 1$  and  $CKPmn = 1$ . The SIp hold time becomes “from SCKp $\uparrow$ ” when  $DAPmn = 0$  and  $CKPmn = 1$ , or  $DAPmn = 1$  and  $CKPmn = 0$ .
  3. When  $DAPmn = 0$  and  $CKPmn = 0$ , or  $DAPmn = 1$  and  $CKPmn = 1$ . The delay time to SOp output becomes “from SCK p $\uparrow$ ” when  $DAPmn = 0$  and  $CKPmn = 1$ , or  $DAPmn = 1$  and  $CKPmn = 0$ .
  4. C is the load capacitance of the SCKp and SOp output lines.

**Caution** Select the normal input buffer for the SIp pin and the normal output mode for the SOp pin and SCKp pin by using port input mode register g (PIMg) and port output mode register g (POMg).

- Remarks**
1. p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0) , g: PIM number (g = 3, 5)
  2.  $f_{MCK}$ : Serial array unit operation clock frequency  
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00))

<R> **(3) During communication at same potential (CSI mode) (master mode, SCKp... internal clock output)**  
 ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Parameter	Symbol	Conditions	HS (high-speed main) Mode		LS (low-speed main) Mode		Unit
			MIN.	MAX.	MIN.	MAX.	
SCKp cycle time	$t_{KCY1}$		125		500		ns
SCKp high-/low-level width	$t_{KH1}, t_{KL1}$		$t_{KCY1}/2 - 18$		$t_{KCY1}/2 - 50$		ns
Slp setup time (to SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{SIK1}$		44		110		ns
Slp hold time (from SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{KSI1}$		19		19		ns
Delay time from SCKp $\downarrow$ to SOp output <sup>Note 2</sup>	$t_{KSO1}$	$C = 30\text{ pF}$ <sup>Note 3</sup>		25		25	ns

**Notes** 1. The value must also be  $2/f_{CLK}$  or more.

2. When  $DAPmn = 0$  and  $CKPmn = 0$ , or  $DAPmn = 1$  and  $CKPmn = 1$ . The Slp setup time becomes "to SCKp $\downarrow$ " when  $DAPmn = 0$  and  $CKPmn = 1$ , or  $DAPmn = 1$  and  $CKPmn = 0$ .

<R> 3. C is the load capacitance of the SCKp and SOp output lines.

**Remarks** 1. p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0)

2.  $f_{MCK}$ : Serial array unit operation clock frequency

(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00))

&lt;R&gt;

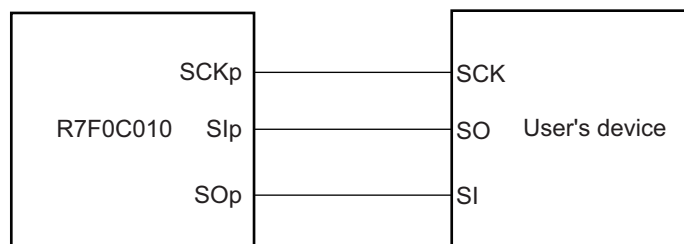
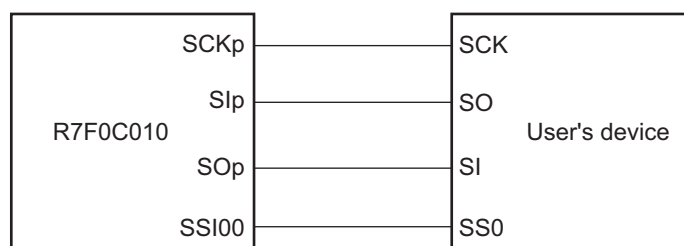
**(4) During communication at same potential (CSI mode) (slave mode, SCKp... external clock input)****( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )**

Parameter	Symbol	Conditions	HS (high-speed main) Mode		LS (low-speed main) Mode		Unit
			MIN.	MAX.	MIN.	MAX.	
SCKp cycle time <sup>Note 4</sup>	$t_{KCY2}$	$16\text{ MHz} < f_{MCK}$	$8/f_{MCK}$		-		ns
		$f_{MCK} \leq 16\text{ MHz}$	$6/f_{MCK}$		$6/f_{MCK}$		ns
SCKp high-/low-level width	$t_{KH2}$ , $t_{KL2}$		$t_{KCY2}/2-8$		$t_{KCY2}/2-8$		ns
Slp setup time (to SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{SIK2}$		$1/f_{MCK}+20$		$1/f_{MCK}+30$		ns
Slp hold time (from SCKp $\uparrow$ ) <sup>Note 1</sup>	$t_{SI2}$		$1/f_{MCK}+31$		$1/f_{MCK}+31$		ns
Delay time from SCKp $\downarrow$ to SOp output <sup>Note 2</sup>	$t_{KS02}$	$C = 30\text{ pF}$ <sup>Note 3</sup>		$2/f_{MCK}+44$		$2/f_{MCK}+110$	ns
SSI00 setup time	$t_{SSIK}$	DAPmn = 0	120		120		ns
		DAPmn = 1	$1/f_{MCK}+120$		$1/f_{MCK}+120$		ns
SSI00 hold time	$t_{KSSI}$	DAPmn = 0	$1/f_{MCK}+120$		$1/f_{MCK}+120$		ns
		DAPmn = 1	120		120		ns

- Notes**
- When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The Slp setup time becomes “to SCKp $\downarrow$ ” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.
  - When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1. The Slp hold time becomes “from SCKp $\downarrow$ ” when DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.
  - C is the load capacitance of the SOp output lines.
  - Transfer rate in the SNOOZE mode: MAX. 1 Mbps

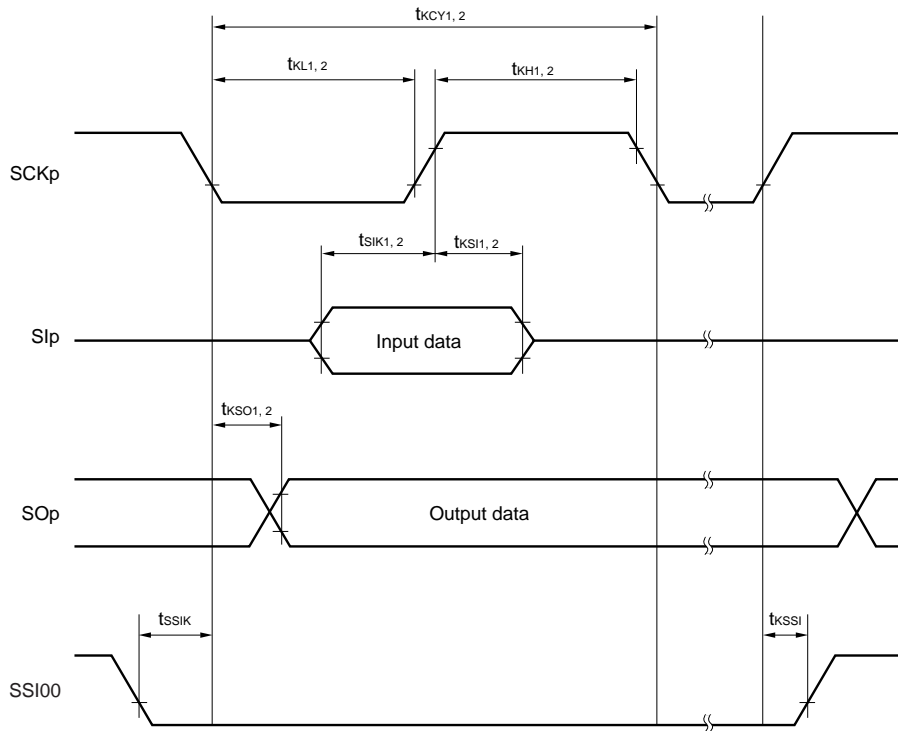
- Remarks**
- p: CSI number (p = 00), m: Unit number (m = 0), n: Channel number (n = 0)
  - $f_{MCK}$ : Serial array unit operation clock frequency  
(Operation clock to be set by the CKSmn bit of serial mode register mn (SMRmn). m: Unit number, n: Channel number (mn = 00))



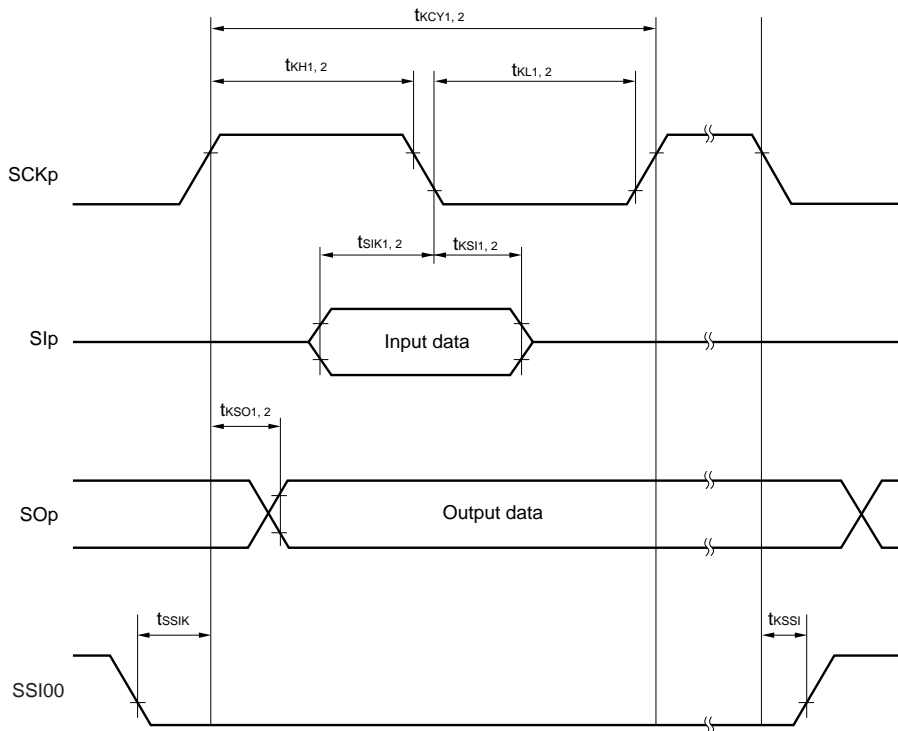
**CSI mode connection diagram (during communication at same potential)****CSI mode connection diagram (during communication at same potential)  
(When slave transmission of slave select input function)**

- Remarks**
1. p: CSI number (p = 00)
  2. m: Unit number, n: Channel number (mn = 00)

**CSI mode serial transfer timing (during communication at same potential)**  
**(When DAPmn = 0 and CKPmn = 0, or DAPmn = 1 and CKPmn = 1.)**



**CSI mode serial transfer timing (during communication at same potential)**  
**(When DAPmn = 0 and CKPmn = 1, or DAPmn = 1 and CKPmn = 0.)**



- Remarks**
1. p: CSI number (p = 00)
  2. m: Unit number, n: Channel number (mn = 00)

## 27.5.2 Serial interface IICA

(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions	Standard Mode		Fast Mode		Fast Mode Plus		Unit
			MIN.	MAX.	MIN.	MAX.	MIN.	MAX.	
SCLAn clock frequency	f <sub>SCL</sub>	Fast mode plus: f <sub>CLK</sub> ≥ 10 MHz					0	1000	kHz
		Fast mode: f <sub>CLK</sub> ≥ 3.5 MHz			0	400			kHz
		Normal mode: f <sub>CLK</sub> ≥ 1 MHz	0	100					kHz
Setup time of restart condition	t <sub>SU:STA</sub>		4.7		0.6		0.26		μs
Hold time <sup>Note 1</sup>	t <sub>HD:STA</sub>		4.0		0.6		0.26		μs
Hold time when SCLAn = "L"	t <sub>LOW</sub>		4.7		1.3		0.5		μs
Hold time when SCLAn = "H"	t <sub>HIGH</sub>		4.0		0.6		0.26		μs
Data setup time (reception)	t <sub>SU:DAT</sub>		250		100		50		ns
Data hold time (transmission) <sup>Note 2</sup>	t <sub>HD:DAT</sub>		0	3.45	0	0.9	0	0.45	μs
Setup time of stop condition	t <sub>SU:STO</sub>		4.0		0.6		0.26		μs
Bus-free time	t <sub>BUF</sub>		4.7		1.3		0.5		μs

- Notes**
- The first clock pulse is generated after this period when the start/restart condition is detected.
  - The maximum value (MAX.) of t<sub>HD:DAT</sub> is during normal transfer and a wait state is inserted in the  $\overline{\text{ACK}}$  (acknowledge) timing.

**Remarks 1.** The maximum value of C<sub>b</sub> (communication line capacitance) and the value of R<sub>b</sub> (communication line pull-up resistor) at that time in each mode are as follows.

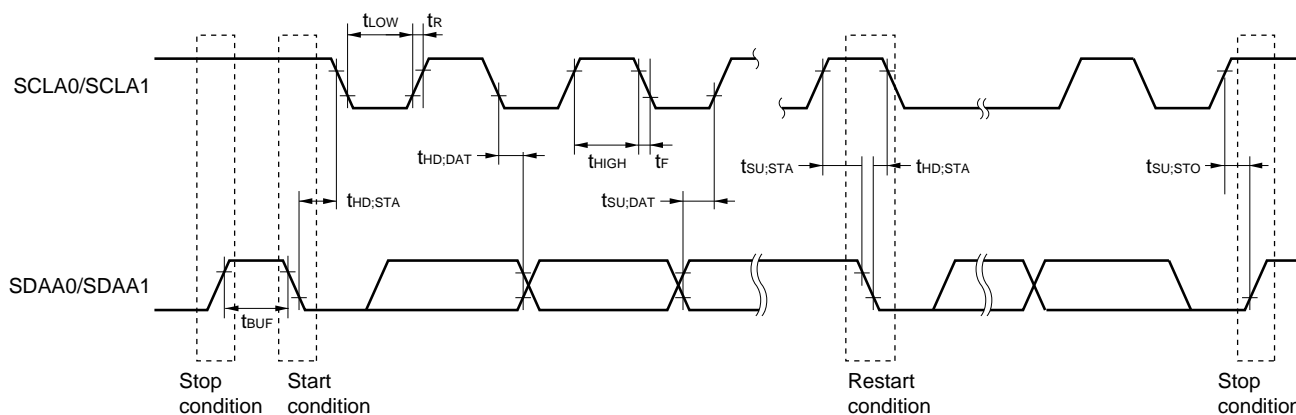
Standard mode: C<sub>b</sub> = 400 pF, R<sub>b</sub> = 2.7 kΩ

Fast mode: C<sub>b</sub> = 320 pF, R<sub>b</sub> = 1.1 kΩ

Fast mode plus: C<sub>b</sub> = 120 pF, R<sub>b</sub> = 1.1 kΩ

- n = 0, 1

## IICA serial transfer timing



### 27.5.3 On-chip debug (UART)

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Transfer rate			115.2 k		1 M	bps

## 27.6 Analog Characteristics

### Classification of A/D converter characteristics

Reference Voltage	Reference voltage (+) = $AV_{REFP}$	Reference voltage (+) = $V_{DD}$	Reference voltage (+) = $V_{BGR}$
Input channel	Reference voltage (-) = $AV_{REFM}$	Reference voltage (-) = $V_{SS}$	Reference voltage (-) = $AV_{REFM}$
ANI0 to ANI7	Refer to 27.6.1 (1)	Refer to 27.6.1 (2)	Refer to 27.6.1 (5)
ANI16	Refer to 27.6.1 (3)	Refer to 27.6.1 (4)	
Internal reference voltage	Refer to 27.6.1 (3)	Refer to 27.6.1 (4)	–
Temperature sensor output voltage			

### 27.6.1 A/D converter characteristics

(1) When  $AV_{REF(+)} = AV_{REFP}/ANI0$  ( $ADREFP1 = 0$ ,  $ADREFP0 = 1$ ),  $AV_{REF(-)} = AV_{REFM}/ANI1$  ( $ADREFM = 1$ ), target ANI pin: ANI0 to ANI7

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ , Reference voltage (+) =  $AV_{REFP}$ , Reference voltage (-) =  $AV_{REFM} = 0\text{ V}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	RES		8		12	bit
Overall error <sup>Notes 1</sup>	AINL	12-bit resolution $AV_{REFP} = V_{DD}$			$\pm 6.0$	LSB
Conversion time	$t_{CONV}$	12-bit resolution $AV_{REFP} = V_{DD}$	3.375		108	$\mu\text{s}$
Zero-scale error <sup>Notes 1, 2</sup>	$E_{ZS}$	12-bit resolution $AV_{REFP} = V_{DD}$			$\pm 0.10$	%FSR
Full-scale error <sup>Notes 1, 2</sup>	$E_{FS}$	12-bit resolution $AV_{REFP} = V_{DD}$			$\pm 0.10$	%FSR
Integral linearity error <sup>Note 1</sup>	ILE	12-bit resolution $AV_{REFP} = V_{DD}$			$\pm 2.5$	LSB
Differential linearity error <sup>Note 1</sup>	DLE	12-bit resolution $AV_{REFP} = V_{DD}$			$\pm 1.5$	LSB
Reference voltage (+)	$AV_{REFP}$		2.7		$V_{DD}$	V
Reference voltage (-)	$AV_{REFM}$		-0.5		0.3	V
Analog input voltage	$V_{AIN}$		0		$AV_{REFP}$	V
	$V_{BGR}$	Select internal reference voltage output $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ , HS (high-speed main) mode	1.38	1.45	1.5	V

**Notes 1.** Excludes quantization error ( $\pm 1/2$  LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

(2) When  $AV_{REF(+)} = V_{DD}$  ( $ADREFP1 = 0$ ,  $ADREFP0 = 0$ ),  $AV_{REF(-)} = V_{SS}$  ( $ADREFM = 0$ ), target ANI pin: ANI0 to ANI7

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ , Reference voltage (+) =  $V_{SS}$ , Reference voltage (-) =  $V_{SS}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	RES		8		12	bit
Overall error <sup>Notes 1</sup>	AINL	10-bit resolution			$\pm 7.0$	LSB
Conversion time	t <sub>CONV</sub>	10-bit resolution	3.375		108	$\mu\text{s}$
Zero-scale error <sup>Notes 1, 2</sup>	E <sub>ZS</sub>	10-bit resolution			$\pm 0.60$	%FSR
Full-scale error <sup>Notes 1, 2</sup>	E <sub>FS</sub>	10-bit resolution			$\pm 0.60$	%FSR
Integral linearity error <sup>Note 1</sup>	ILE	10-bit resolution			$\pm 4.0$	LSB
Differential linearity error <sup>Note 1</sup>	DLE	10-bit resolution			$\pm 2.0$	LSB
Analog input voltage	V <sub>AIN</sub>		0		AV <sub>REFP</sub>	V
	V <sub>BGR</sub>	Select internal reference voltage output 2.7 V $\leq$ V <sub>DD</sub> $\leq$ 3.6 V, HS (high-speed main) mode	1.38	1.45	1.5	V

**Notes 1.** Excludes quantization error ( $\pm 1/2$  LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

(3) When  $AV_{REF(+)} = AV_{REFP}/ANI0$  ( $ADREFP1 = 0$ ,  $ADREFP0 = 1$ ),  $AV_{REF(-)} = AV_{REFM}/ANI1$  ( $ADREFM = 1$ ), target ANI pin: ANI16, internal reference voltage, and temperature sensor output voltage

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ , Reference voltage (+) =  $AV_{REFP}$ , Reference voltage (-) =  $AV_{REFM} = 0\text{ V}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	RES		8		12	bit
Overall error <sup>Notes 1</sup>	AINL	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>			$\pm 7.0$	LSB
Conversion time	t <sub>CONV</sub>	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>	4.125		132	$\mu\text{s}$
Zero-scale error <sup>Notes 1, 2</sup>	E <sub>ZS</sub>	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>			$\pm 0.10$	%FSR
Full-scale error <sup>Notes 1, 2</sup>	E <sub>FS</sub>	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>			$\pm 0.10$	%FSR
Integral linearity error <sup>Note 1</sup>	ILE	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>			$\pm 3.0$	LSB
Differential linearity error <sup>Note 1</sup>	DLE	12-bit resolution AV <sub>REFP</sub> = V <sub>DD</sub>			$\pm 2.0$	LSB
Reference voltage (+)	AV <sub>REFP</sub>		2.7		V <sub>DD</sub>	V
Reference voltage (-)	AV <sub>REFM</sub>		-0.5		0.3	V
Analog input voltage	V <sub>AIN</sub>		0		AV <sub>REFP</sub>	V
	V <sub>BGR</sub>	Select internal reference voltage output 2.7 V $\leq$ V <sub>DD</sub> $\leq$ 3.6 V, HS (high-speed main) mode	1.38	1.45	1.5	V

**Notes 1.** Excludes quantization error ( $\pm 1/2$  LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

(4) When  $AV_{REF(+)} = V_{DD}$  ( $ADREFP1 = 0$ ,  $ADREFP0 = 0$ ),  $AV_{REF(-)} = V_{SS}$  ( $ADREFM = 0$ ), target ANI pin: ANI16, internal reference voltage, and temperature sensor output voltage

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ , Reference voltage (+) =  $V_{DD}$ , Reference voltage (-) =  $V_{SS}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	RES		8		12	bit
Overall error <sup>Notes 1</sup>	AINL	10-bit resolution			$\pm 7.0$	LSB
Conversion time	$t_{CONV}$	10-bit resolution	4.125		132	$\mu\text{s}$
Zero-scale error <sup>Notes 1, 2</sup>	$E_{ZS}$	10-bit resolution			$\pm 0.60$	%FSR
Full-scale error <sup>Notes 1, 2</sup>	$E_{FS}$	10-bit resolution			$\pm 0.60$	%FSR
Integral linearity error <sup>Note 1</sup>	ILE	10-bit resolution			$\pm 4.0$	LSB
Differential linearity error <sup>Note 1</sup>	DLE	10-bit resolution			$\pm 2.5$	LSB
Analog input voltage	$V_{AIN}$		0		$V_{DD}$	V
	$V_{BGR}$	Select internal reference voltage output $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ , HS (high-speed main) mode	1.38	1.45	1.5	V

**Notes 1.** Excludes quantization error ( $\pm 1/2$  LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

(5) When  $AV_{REF(+)} =$  Internal reference voltage ( $ADREFP1 = 1$ ,  $ADREFP0 = 0$ ),  $AV_{REF(-)} = AV_{REFM}/ANI1$  ( $ADREFM = 1$ ), target ANI pin: ANI0 to ANI7, ANI16

( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ , Reference voltage (+) =  $V_{BGR}$ , Reference voltage (-) =  $AV_{REFM} = 0\text{ V}$ , HS (high-speed main) mode)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	RES		8			bit
Conversion time	$t_{CONV}$	8-bit resolution	16		108	$\mu\text{s}$
Zero-scale error <sup>Notes 1, 2</sup>	$E_{ZS}$	8-bit resolution			$\pm 1.60$	%FSR
Integral linearity error <sup>Note 1</sup>	ILE	8-bit resolution			$\pm 2.5$	LSB
Differential linearity error <sup>Note 1</sup>	DLE	8-bit resolution			$\pm 2.5$	LSB
Reference voltage (+)	$V_{BGR}$		1.38	1.45	1.5	V
Analog input voltage	$V_{AIN}$		0		$V_{BGR}$	V

**Notes 1.** Excludes quantization error ( $\pm 1/2$  LSB).

**2.** This value is indicated as a ratio (%FSR) to the full-scale value.

## 27.6.2 Temperature sensor/internal reference voltage characteristics

(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V, HS (high-speed main) mode)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Temperature coefficient	SL	Temperature sensor that depends on the temperature		-3.6		mV/°C
Temperature coefficient error <sup>Note</sup>	E <sub>SL</sub>			43		μV/°C
Offset	OFST25	T <sub>A</sub> = +25°C		1050		mV
Offset error <sup>Note</sup>	E <sub>OFST25</sub>	T <sub>A</sub> = +25°C		9.0		mV
Operation stabilization wait time	t <sub>AMP</sub>				5	μs

**Note** Standard deviation**Caution** Measurement for mass production was not performed.

## 27.6.3 D/A converter

(T<sub>A</sub> = -40 to +85°C, 2.7 V ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Resolution	R <sub>ES</sub>				10	bit
Overall error <sup>Notes 1</sup>	A <sub>INL</sub>	Load current = 0 mA, 0.2 V ≤ output voltage ≤ V <sub>DD</sub> - 0.2 V		±2.0	±4.0	LSB
Settling time <sup>Note 1</sup>	t <sub>SET</sub>	R <sub>load</sub> = 47 kΩ, C <sub>load</sub> = 20 pF			10	μs
D/A output resistance <sup>Note 1</sup>	R <sub>o</sub>	Per channel Output volage = 0 V to 0.2 V or Output volage = V <sub>DD</sub> - 0.2 V to V <sub>DD</sub>		40	60	Ω
		Per channel Output volage = 0.2 V to V <sub>DD</sub> - 0.2 V		8	12	Ω
Standby current <sup>Note 1</sup>	I <sub>sb</sub>	Per D/A converter channel			0.1	μA
DAC operating current <sup>Notes 1, 2</sup>	I <sub>DAC</sub>	R <sub>load</sub> = 47 kΩ, C <sub>load</sub> = 20 pF, V <sub>I</sub> = 0.5V <sub>DD</sub> Per D/A converter channel			2	mA

**Notes 1.** Measurement for mass production was not performed.**2.** Current flowing only to the D/A converter

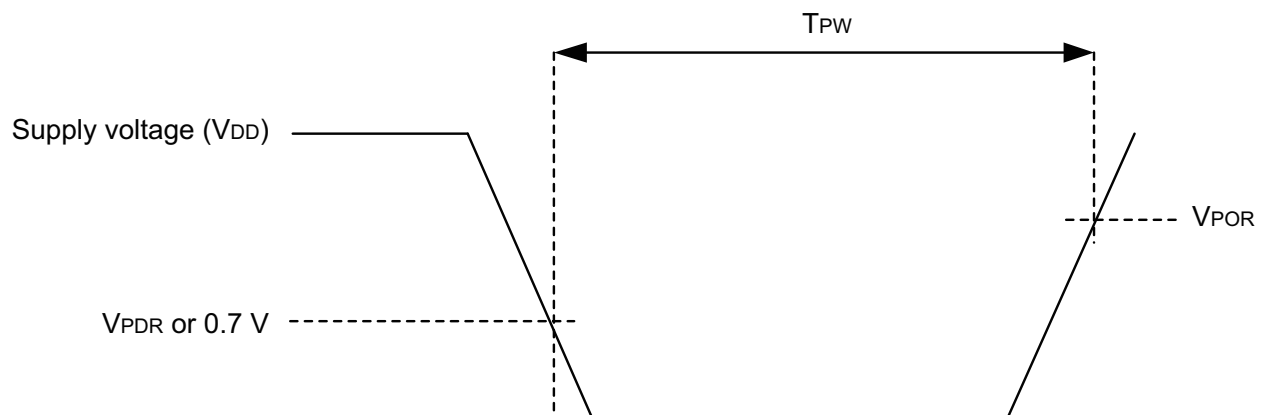
## 27.6.4 POR circuit characteristics

(T<sub>A</sub> = -40 to +85°C, V<sub>SS</sub> = 0 V)

&lt;R&gt;

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	V <sub>POR</sub>	Power supply rise time	1.47	1.51	1.55	V
	V <sub>PDR</sub>	Power supply fall time	1.46	1.50	1.54	V
Minimum pulse width <sup>Note</sup>	T <sub>PW</sub>		300			μs

**Note** Minimum time required for a POR reset when VDD exceeds below V<sub>PDR</sub>. This is also the minimum time required for a POR reset from when VDD exceeds below 0.7 V to when VDD exceeds V<sub>POR</sub> while STOP mode is entered or the main system clock is stopped through setting bit 0 (HIOSTOP) and bit 7 (MSTOP) in the clock operation status control register (CSC).





## 27.6.5 LVD circuit characteristics

## LVD Detection Voltage of Reset Mode and Interrupt Mode

(T<sub>A</sub> = -40 to +85°C, V<sub>PDR</sub> ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter		Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Detection voltage	Supply voltage level	V <sub>LVD2</sub>	Power supply rise time	3.07	3.13	3.19	V
			Power supply fall time	3.00	3.06	3.12	V
		V <sub>LVD3</sub>	Power supply rise time	2.96	3.02	3.08	V
			Power supply fall time	2.90	2.96	3.02	V
		V <sub>LVD4</sub>	Power supply rise time	2.86	2.92	2.97	V
			Power supply fall time	2.80	2.86	2.91	V
		V <sub>LVD5</sub>	Power supply rise time	2.76	2.81	2.87	V
			Power supply fall time	2.70	2.75	2.81	V
Minimum pulse width		t <sub>LW</sub>		300			μs
Detection delay time						300	μs

## LVD Detection Voltage of Interrupt &amp; Reset Mode

(T<sub>A</sub> = -40 to +85°C, V<sub>PDR</sub> ≤ V<sub>DD</sub> ≤ 3.6 V, V<sub>SS</sub> = 0 V)

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
Interrupt and reset mode	V <sub>LVD5</sub>	V <sub>POC2</sub> , V <sub>POC1</sub> , V <sub>POC0</sub> = 0, 1, 1, falling reset voltage	2.70	2.75	2.81	V	
	V <sub>LVD4</sub>	LVIS1, LVIS0 = 1, 0 (+0.1 V)	Rising release reset voltage	2.86	2.92	2.97	V
			Falling interrupt voltage	2.80	2.86	2.91	V
	V <sub>LVD3</sub>	LVIS1, LVIS0 = 0, 1 (+0.2 V)	Rising release reset voltage	2.96	3.02	3.08	V
			Falling interrupt voltage	2.90	2.96	3.02	V

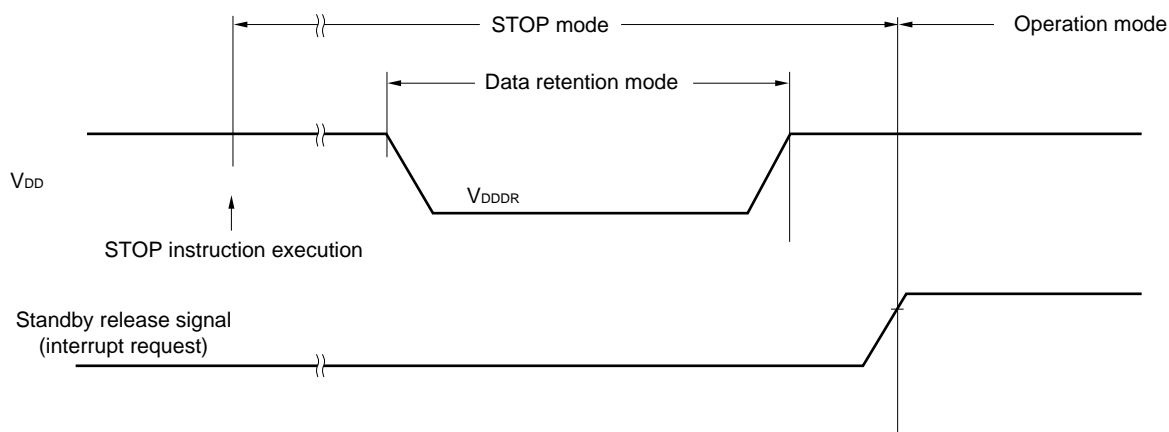
## 27.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics

( $T_A = -40$  to  $+85^\circ\text{C}$ )

<R>

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Data retention supply voltage	$V_{DDDR}$		1.46 <sup>Note</sup>		3.6	V

**Note** The value depends on the POR detection voltage. When the voltage drops, the data is retained before a POR reset is effected, but data is not retained when a POR reset is effected.



## 27.8 Flash Memory Programming Characteristics

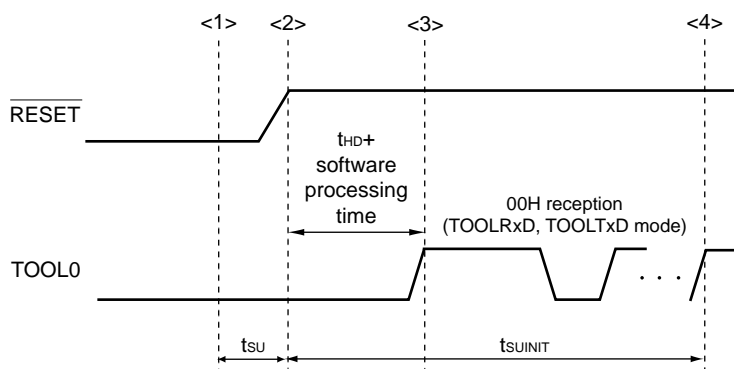
( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$ ,  $V_{SS} = 0\text{ V}$ )

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit	
System clock frequency	$f_{CLK}$	$2.7\text{ V} \leq V_{DD} \leq 3.6\text{ V}$	1		32	MHz	
Number of code flash rewrites <sup>Note 1, 2, 3</sup>	$C_{erwr}$	1 erase + 1 write after the erase is regarded as 1 rewrite. The retaining years are until next rewrite after the rewrite.	Retained for 20 years	1,000		Times	
Number of data flash rewrites <sup>Note 1, 2, 3</sup>			Retained for 1 years		1,000,000		
			Retained for 5 years	100,000			
Operating ambient temperature		In flash memory programming mode	0 to +40			$^\circ\text{C}$	
		In the self-programming mode	-40 to +85			$^\circ\text{C}$	

- Notes**
- 1 erase + 1 write after the erase is regarded as 1 rewrite.  
The retaining years are until next rewrite after the rewrite.
  2. When using flash memory programmer and Renesas Electronics self programming library
  3. These are the characteristics of the flash memory and the results obtained from reliability testing by Renesas Electronics Corporation.

## 27.9 Timing for Switching Flash Memory Programming Modes

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
How long from when an external reset ends until the initial communication settings are specified	$t_{\text{SUIINIT}}$	POR and LVD reset must end before the external reset ends.			100	ms
How long from when the TOOL0 pin is placed at the low level until an external reset ends	$t_{\text{SU}}$	POR and LVD reset must end before the external reset ends.	10			$\mu\text{s}$
How long the TOOL0 pin must be kept at the low level after a reset ends (except software processing time)	$t_{\text{HD}}$	POR and LVD reset must end before the external reset ends.	1			ms



- <1> The low level is input to the TOOL0 pin.
- <2> The external reset ends (POR and LVD reset must end before the external reset ends.).
- <3> The TOOL0 pin is set to the high level.
- <4> Setting of the flash memory programming mode by UART reception and complete the baud rate setting.

**Remark**  $t_{\text{SUIINIT}}$ : The segment shows that it is necessary to finish specifying the initial communication settings within 100 ms from when the resets end.

$t_{\text{SU}}$ : How long from when the TOOL0 pin is placed at the low level until an external reset ends (MIN. 10  $\mu\text{s}$ )

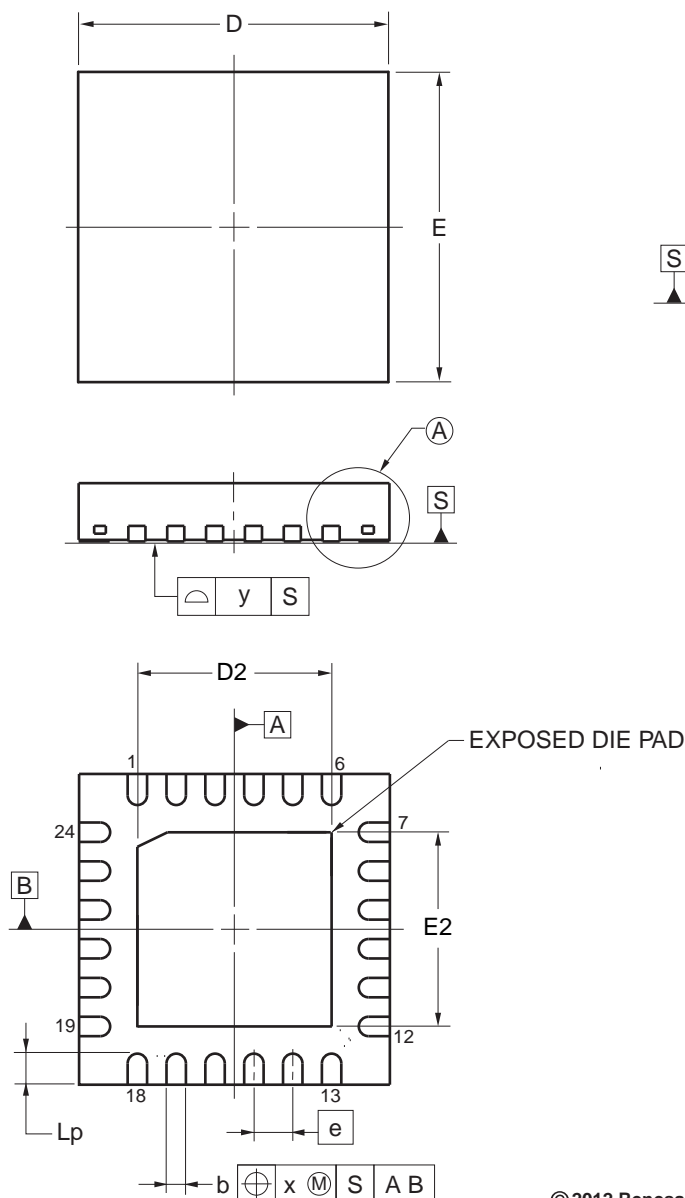
$t_{\text{HD}}$ : How long to keep the TOOL0 pin at the low level from when the external and internal resets end (except software processing time)

CHAPTER 28 PACKAGE DRAWINGS

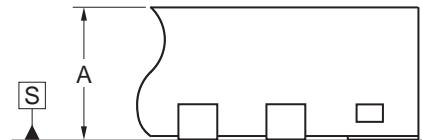
28.1 24-pin Products

R7F0C01072DNP

JEITA Package Code	RENESAS Code	Previous Code	MASS (TYP.) [g]
P-HWQFN24-4x4-0.50	PWQN0024KE-A	P24K8-50-CAB-2	0.04



DETAIL OF (A) PART



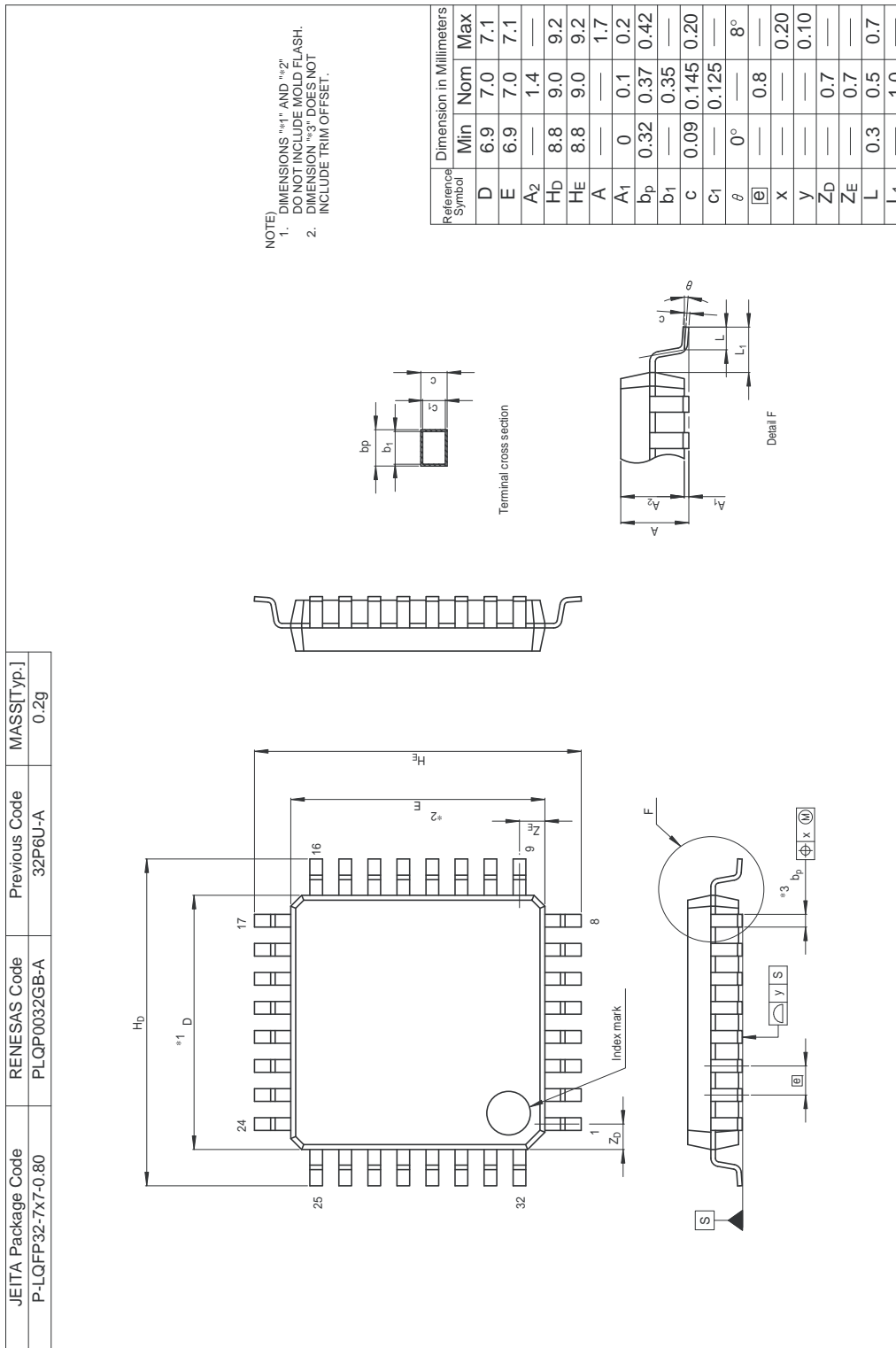
Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	3.95	4.00	4.05
E	3.95	4.00	4.05
A	0.70	0.75	0.80
b	0.18	0.25	0.30
e	—	0.50	—
Lp	0.30	0.40	0.50
x	—	—	0.05
y	—	—	0.05

ITEM	A	D2			E2		
		MIN	NOM	MAX	MIN	NOM	MAX
EXPOSED DIE PAD VARIATIONS	A	2.45	2.50	2.55	2.45	2.50	2.55

©2012 Renesas Electronics Corporation. All rights reserved.

28.2 32-pin Products

R7F0C010B2DFP-C



## APPENDIX A REVISION HISTORY

### A.1 Major Revisions in This Edition

Page	Description	Classification
<b>CHAPTER 1 OUTLINE</b>		
p.1	Modification of 1.1 Features	(c)
p.2	Modification of 1.2 List of Part Numbers	(c)
p.6 to p.7	Modification of 1.5 Block Diagram	(c)
p.8	Modification of 1.6 Outline of Functions	(c)
<b>CHAPTER 2 PIN FUNCTIONS</b>		
p.10	Modification of 2.1 Port Function	(c)
p.13 to p.14	Addition of 2.2 Functions Other Than Port Pins	(c)
p.18	Modification of 2.3.4 P40 (port 4)	(c)
p.21	Modification of 2.4 Connection of Unused Pins	(c)
<b>CHAPTER 3 CPU ARCHITECTURE</b>		
p.25	Modification of Figure 3 - 1 Memory Map	(c)
p.31	Modification of Figure 3 - 2 Format of Processor Mode Control Register (PMC)	(c)
p.32	Modification of 3.1.3 Internal data memory space	(c)
p.33	Modification of Figure 3 - 3 Correspondence between Data Memory and Addressing	(c)
p.35	Modification of Figure 3 - 6 Format of Stack Pointer	(c)
p.35	Modification of (3) in 3.2.1 Control registers	(c)
p.36	Modification of 3.2.2 General-purpose registers	(c)
p.37	Modification of 3.2.3 ES and CS registers	(c)
p.37	Addition of Figure 3 - 9 Extension of Data Area Which Can Be Accessed	(c)
p.38	Modification of 3.2.4 Special function registers (SFRs)	(c)
p.41	Modification of Table 3-5 Special Function Register (SFR) List (3/3)	(c)
p.42	Modification of 3.2.5 Extended special function registers (2nd SFRs: 2nd Special Function Registers)	(c)
p.53	Modification of Figure 3-20 Outline of SFR Addressing	(c)
<b>CHAPTER 4 PORT FUNCTIONS</b>		
p.83	Modification of 4.3 Registers Controlling Port Function	(c)
p.88	Modification of Figure 4 - 19 Format of Port Mode Control Register	(c)
p.89	Modification of Figure 4 - 20 Format of A/D Port Configuration Register (ADPC)	(c)
<b>CHAPTER 5 CLOCK GENERATOR</b>		
p.96	Modification of (1) Main system clock in 5.1 Functions of Clock Generator	(c)
p.99	Modification of 5.3 Registers Controlling Clock Generator	(c)
p.104	Modification of Figure 5-5 Format of Oscillation Stabilization Time Counter Status Register (OSTC)	(c)
p.105	Modification of 5.3.5 Oscillation stabilization time select register (OSTS)	(c)
p.106	Modification of Figure 5 - 6 Format of Oscillation stabilization time select register (OSTS)	(c)
p.109	Modification of 5.3.7 High-speed on-chip oscillator frequency select register (HOCODIV)	(c)
p.110	Modification of 5.3.8 High-speed on-chip oscillator trimming register (HIOTRM)	(c)
p.114	Modification of 5.4.2 High-speed on-chip oscillator	(c)

Page	Description	Classification
p.114	Modification of 5.4.3 Low-speed on-chip oscillator	(c)
p.116	Modification of Figure 5 - 13 Clock Generator Operation When Power Supply Voltage Is Turned On	(c)
p.118	Modification of 5.6.2 Example of setting X1 oscillation clock	(c)
p.119	Modification of Figure 5 - 14 CPU Clock Status Transition Diagram	(c)
p.120	Modifications of Table 5 - 4 CPU Clock Transition and SFR Register Setting Examples (1/3)	(c)
p.121	Modification of (4) of 5.6.3 CPU clock status transition diagram	(c)
p.123	Modification of Table 5 - 4 Changing CPU Clock	(c)
p.125, p.126	Modification of 5.7 Resonator and Oscillator Constants	(c)
<b>CHAPTER 6 TIMER ARRAY UNIT</b>		
p.127	Addition of the table	(c)
p.128	Deletion of Note of table	(c)
p.130	Modification of (6) of 6.1 Functions of Timer Array Unit	(c)
p.132	Modification of Table 6 - 1 Configuration of Timer Array Unit	(c)
p.134, p.135	Modifications of Figure 6 - 2 Internal Block Diagram of Channel 0 of Timer Array Unit 0 to Figure 6 - 5 Internal Block Diagram of Channel 3 of Timer Array Unit 0	(c)
p.138	Modification of 6.3 Registers Controlling Timer Array Unit	(c)
p.139	Modification of Figure 6 - 9 Format of Peripheral enable register 0 (PER0)	(c)
p.140	Modification of 6.3.2 Timer clock select register m (TPSm)	(c)
p.141, p.142	Modifications of Figure 6 - 10 Format of Timer clock select register m (TPSm) (1/2) and Figure 6 - 10 Format of Timer clock select register m (TPSm) (2/2)	(c)
p.145	Modification of Figure 6 - 11 Format of Timer mode register mn (TMRmn) (2/4)	(c)
p.151	Modification of Figure 6 - 15 Format of Timer channel stop register m (TTm)	(c)
p.153	Modification of Figure 6 -17 Format of Timer output enable register m (TOEm)	(c)
p.157	Modification of Figure 6 -21 Format of Noise filter enable register 1 (NFEN1)	(c)
p.166	Modification of Figure 6 - 26 Operation Timing (Event Counter Mode)	(c)
p.167	Modification of Figure 6 - 27 Operation Timing (In Capture Mode: Input Pulse Interval Measurement)	(c)
p.170	Modification of Figure 6-30 Output Circuit Configuration	(c)
p.179, p.180	Addition of 6.7 Timer Input (TImn) Control	(c)
p.183	Modification of Figure 6 - 43 Example of Set Contents of Registers during Operation as Interval Timer/Square Wave Output	(c)
p.185, p.186	Modification of Figure 6 - 44 Operation Procedure of Interval Timer/Square Wave Output Function	(c)
p.187	Modification of Figure 6 - 45 Block Diagram of Operation as External Event Counter	(c)
p.189	Modification of Figure 6 - 47 Example of Set Contents of Registers in External Event Counter Mode	(c)
p.191	Modification of Figure 6 - 48 Operation Procedure When External Event Counter Function Is Used	(c)
p.192	Modification of 6.8.3 Operation as input pulse interval measurement	(c)
p.192	Modification of Figure 6 - 49 Block Diagram of Operation as Input Pulse Interval Measurement	(c)
p.195	Modification of Figure 6 -52 Operation Procedure When Input Pulse Interval Measurement Function Is Used	(c)
p.197	Modification of Figure 6 - 53 Block Diagram of Operation as Input Signal High-/Low-level Width Measurement	(c)

Page	Description	Classification
p.199	Modification of Figure 6 - 56 Operation Procedure When Input Signal High-/Low-level Width Measurement Function Is Used	(c)
p.200	Modification of Figure 6 - 57 Block Diagram of Operation as Delay Counter	(c)
p.202	Modification of Figure 6 - 59 Example of Set Contents of Registers to Delay Counter	(c)
p.204	Modification of Figure 6 - 60 Operation Procedure When Delay Counter Function Is Used	(c)
p.205 to p.218	Modification from "p: Slave channel number ( $n < p \leq 3$ )" to "p: Slave channel number ( $n = 0: p = 1, 2, 3, n = 2: p = 3$ )"	(a)
p.206	Modification of Figure 6 - 61 Block Diagram of Operation as One-Shot Pulse Output Function	(c)
p.208	Modification of Figure 6 - 63 Example of Set Contents of Registers When One-Shot Pulse Output Function Is Used (Master Channel)	(c)
p.210, p.211	Modifications of Figure 6 - 65 Operation Procedure of One-Shot Pulse Output Function (1/2) and Figure 6 - 65 Operation Procedure of One-Shot Pulse Output Function (2/2)	(c)
p.215	Modification of Figure 6 - 68 Example of Set Contents of Registers When PWM Function (Master Channel) Is Used	(c)
p.216	Modification of Figure 6 - 69 Example of Set Contents of Registers When PWM Function (Slave Channel) Is Used	(c)
p.217, p.218	Modification of Figure 6-70 Operation Procedure When PWM Function Is Used (1/2) and Figure 6-70. Operation Procedure When PWM Function Is Used (2/2)	(c)
p.223	Modification of Figure 6 - 73 Example of Set Contents of Registers When Multiple PWM Output Function (Master Channel) Is Used	(c)
p.225, p.226	Modification of Figure 6-75 Operation Procedure When Multiple PWM Output Function Is Used (1/2) and Figure 6-75 Operation Procedure When Multiple PWM Output Function Is Used (2/2)	(c)
<b>CHAPTER 7 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER</b>		
p.230	Modification of 7.3 Registers Controlling Clock Output/Buzzer Output Controller	(c)
p.232	Modification of 7.3.2 Registers controlling port functions of pins to be used for clock or buzzer output	(c)
p.233	Modification of 7.4.1 Operation as output pin	(c)
p.233	Modification of 7.5 Cautions of Clock Output/Buzzer Output Controller	(c)
<b>CHAPTER 8 WATCHDOG TIMER</b>		
p.234	Modification of 8.1 Functions of Watchdog Timer	(c)
p.235	Modification of Table 8 - 1 Configuration of Watchdog Timer	(c)
p.235	Modification of Figure 8 - 1 Block Diagram of Watchdog Timer	(c)
p.237	Modification of 8.4.1 Controlling operation of watchdog timer	(c)
p.238	Modification of Table 8 - 3 Setting of Overflow Time of Watchdog Timer	(b)
p.240	Modification of the table in 8.4.3 Setting window open period of watchdog timer	(c)
<b>CHAPTER 9 A/D CONVERTER</b>		
p.241	Modification of 9.1 Function of A/D Converter	(c)
p.242	Modification of Figure 9 - 1 Block Diagram of A/D Converter	(c)
p.243, p.244	Modification of (1) and (8) of 9.2 Configuration of A/D Converter	(c)
p.246	Modification of Figure 9 - 2 Format of Peripheral enable register 0 (PER0)	(c)
p.247	Modification of 9.3.2 A/D converter mode register 0 (ADM0)	(c)
p.248	Modification of Table 9 - 1 Settings of ADCS and ADCE Bits	(c)
p.249	Modification of Figure 9 - 4 Timing Chart When A/D Voltage Comparator Is Used	(c)
p.250 to p.254	Modifications of Table 9 - 3 A/D Conversion Time Selection (1/4) to Table 9 - 3 A/D Conversion Time Selection (4/4)	(c)



Page	Description	Classification
p.256	Modification of Figure 9 - 5 A/D Converter Sampling and A/D Conversion Timing (Example for Software Trigger Mode)	(c)
p.257	Figure 9 - 6 Format of A/D converter mode register 1 (ADM1)	(c)
p.258 to p.260	Modification of 9.3.4 A/D converter mode register 2 (ADM2)	(c)
p.261	Modification of 9.3.5 10-bit A/D conversion result register (ADCR)	(c)
p.263, p.264	Modification of Figure 9 - 11 Format of Analog Input Channel Specification Register (ADS) (1/2) and Figure 9 - 11 Format of Analog Input Channel Specification Register (ADS) (2/2)	(c)
p.265	Modification of 9.3.8 Conversion result comparison upper limit setting register (ADUL)	(c)
p.265	Modification of Figure 9 - 13 Format of Conversion result comparison lower limit setting register (ADLL)	(c)
p.266	Modification of 9.3.10 A/D test register (ADTES)	(c)
p.267	Modification of 9.3.11 Registers controlling port function of analog input pins	(c)
—	Deletions of old 9.3.11 A/D port configuration register (ADPC), old 9.3.12 Port mode control register 1 (PMC1) (24-pin products only) and old 9.3.13 Port mode registers 1, 2 (PM1, PM2)	(c)
p.269	Modification of Figure 9 - 15 Conversion Operation of A/D Converter (Software Trigger Mode)	(c)
p.271 to p.282	Modifications of Figure 9 - 17 Example of Software Trigger Mode (Select Mode, Sequential Conversion Mode) Operation Timing to Figure 9 - 28 Example of Hardware Trigger Wait Mode (Scan Mode, One-Shot Conversion Mode) Operation Timing	(c)
p.284	Modification of Figure 9 - 29 Setting up Software Trigger Mode	(c)
p.285	Modification of Figure 9 - 30 Setting up Hardware Trigger No-Wait Mode	(c)
p.286	Modification of Figure 9 - 31 Setting up Hardware Trigger Wait Mode	(c)
p.287	Modification of 9.7.4 Setup when temperature sensor output voltage/internal reference voltage is selected (example for software trigger mode and one-shot conversion mode)	(c)
p.287	Modification of Figure 9 - 32 Setup when temperature sensor output voltage/internal reference voltage is selected (ADISS = 1)	(c)
p.288	Modification of Figure 9 - 33 Setting up Test Mode	(c)
p.289	Modification of 9.8 SNOOZE Mode Function	(c)
p.292	Addition of Figure 9 - 37 Flowchart for Setting up SNOOZE Mode	(c)
p.294	Modifications of Figure 9 - 40 Zero-Scale Error and Figure 9 - 43 Full-Scale Error	(c)
p.296	Modifications of (6) Input impedance of analog input (ANIn) pins	(c)
<b>CHAPTER 10 D/A CONVERTER</b>		
p.301	Modification of Figure 10-2 Format of A/D Port Configuration Register (ADPC)	(c)
<b>CHAPTER 11 SERIAL ARRAY UNIT</b>		
p.308	Modification of description	(c)
p.310	Modification of 11.1.2 UART (UART0)	(c)
p.312	Modification of Figure 11-1 shows the block diagram of the serial array unit 0	(c)
p.313	Modification of 11.2.2 Lower 8/9 bits of the serial data register mn (SDRmn)	(c)
p.316	Modification of Figure 11 - 3 Format of Peripheral enable register 0 (PER0)	(c)
p.317	Modification of Figure 11 - 4 Format of Serial clock select register m (SPSm)	(c)
p.320, p.321	Modifications of Figure 11 - 6 Format of Serial communication operation setting register mn (SCRmn) (1/2) and Figure 11 - 6 Format of Serial communication operation setting register mn (SCRmn) (2/2)	(c)
p.322	Modification of 11.3.5 Higher 7 bits of the serial data register mn (SDRmn)	(c)
p.325	Modification of Figure 11 - 9 Format of Serial status register mn (SSRmn) (2/2)	(c)
p.330	Modification of 11.3.12 Serial output register m (SOM)	(c)

Page	Description	Classification
p.330	Modification of Figure 11 - 14 Format of Serial output register m (SOm)	(c)
p.331	Modification of Figure 11 - 15 Format of Serial output level register m (SOLm)	(c)
p.331	Modification of 11.3.13 Serial output level register m (SOLm)	(c)
p.331	Addition of Figure 11 - 16 Examples of Reverse Transmit Data	(c)
p.332	Modification of 11.3.14 Serial standby control register m (SSCm)	(c)
p.332	Modification of Figure 11 - 17 Format of Serial standby control register m (SSCm)	(c)
p.332	Addition of Figure 11 - 18 Interrupt in UART Reception Operation in SNOOZE Mode	(c)
p.339	Modification of 11.5 Operation of 3-Wire Serial I/O (CSI00) Communication	(c)
p.341	Modification of 11.5.1 Master transmission	(c)
p.346 to p.349	Modification of Figure 11-28 Timing Chart of Master Transmission (in Single-Transmission Mode) (Type 1: DAPmn = 0, CKPmn = 0) to Figure 11-31 Flowchart of Master Transmission (in Continuous Transmission Mode)	(c)
p.350	Modification of 11.5.2 Master reception	(c)
p.352 to p.353	Modifications of Figure 11 - 33 Initial Setting Procedure for Master Reception to Figure 14 - 34 Procedure for Stopping Master Reception	(c)
p.356 to p.358	Modification of Figure 11-37 Flowchart of Master Reception (in Single-Reception Mode) to Figure 11-39 Flowchart of Master Reception (in Continuous Reception Mode)	(c)
p.359	Modification of 11.5.3 Master transmission/reception	(c)
p.361, p.362	Modifications of Figure 11 - 41 Initial Setting Procedure for Master Transmission/Reception and Figure 11-42 Procedure for Stopping Master Transmission/Reception	(c)
p.364 to p.367	Modification of Figure 11-44 Timing Chart of Master Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0) to Figure 11-47 Flowchart of Master Transmission/Reception (in Continuous Transmission/Reception Mode)	(c)
p.368	Modification of 11.5.4 Slave transmission	(c)
p.371 to p.376	Modifications of Figure 11 - 50 Procedure for Stopping Slave Transmission to Figure 11-55 Flowchart of Slave Transmission (in Continuous Transmission Mode)	(c)
p.377	Modification of 11.5.5 Slave reception	(c)
p.379 to p.382	Modifications of Figure 11 - 57 Initial Setting Procedure for Slave Reception to Figure 14 - 61 Flowchart of Slave Reception (in Single-Reception Mode)	(c)
p.383	Modification of 11.5.6 Slave transmission/reception	(c)
p.386	Modification of Figure 11-64 Procedure for Stopping Slave Transmission/Reception	(c)
p.388 to p.391	Modifications of Figure 11 - 66 Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0) to Figure 11 - 69 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)	(c)
p.392	Modification of 11.5.7 SNOOZE mode function	(c)
p.394, p.395	Modifications of Figure 11 - 72 Timing Chart of SNOOZE Mode Operation (Continuous Startup) (Type 1: DAPmn = 0, CKPmn = 0) and Figure 11 - 73 Flowchart of SNOOZE Mode Operation (Continuous Startup)	(c)
p.399	Modification of 11.6 Clock Synchronous Serial Communication with Slave Select Input Function	(c)
p.405 to p.411	Modifications of Figure 11 - 78 Initial Setting Procedure for Slave Transmission to Figure 14 - 84 Flowchart of Slave Transmission (in Continuous Transmission Mode)	(c)
p.412	Modification of 11.6.2 Slave reception	(c)
p.415 to p.418	Modifications of Figure 11 - 86 Initial Setting Procedure for Slave Reception to Figure 14 - 90 Flowchart of Slave Reception (in Single-Reception Mode)	(c)
p.419	Modification of 11.6.3 Slave transmission/reception	(c)

Page	Description	Classification
p.422 to p.428	Modifications of Figure 11 - 92 Initial Setting Procedure for Slave Transmission/Reception to Figure 11 - 98 Flowchart of Slave Transmission/Reception (in Continuous Transmission/Reception Mode)	(c)
p.429	Modification of 11.6.4 Calculating transfer clock frequency	(c)
p.432, p.433	Modification of 11.7 Operation of UART (UART0, UART1) Communication	(c)
p.434	Modification of 11.7.1 UART transmission	(c)
p.437 to p.441	Modifications of Figure 11 - 101 Initial Setting Procedure for UART Transmission to Figure 11 - 105 Flowchart of UART Transmission (in Single-Transmission Mode)	(c)
p.443	Modification of Figure 11-107 Flowchart of UART Transmission (in Continuous Transmission Mode)	(c)
p.446	Modification of Figure 11-108 Example of Contents of Registers for UART Reception of UART (UART0) (2/2)	(c)
p.447 to p.450	Modifications of Figure 11 - 110 Procedure for Stopping UART Reception to Figure 11 - 113 Flowchart of UART Reception	(c)
p.451	Modification of 11.7.3 SNOOZE mode function	(c)
p.451, p.452	Modifications of Figure 11 - 114 Timing Chart of SNOOZE Mode Operation (Normal Operation Mode) and Figure 11-115 Timing Chart of SNOOZE Mode Operation (Abnormal Operation <1>)	(c)
p.454	Modification of Figure 11 - 117 Timing Chart of SNOOZE Mode Operation (Abnormal Operation <2>)	(c)
<b>CHAPTER 13 DMA CONTROLLER</b>		
p.559	Modification of 13.5.2 Consecutive capturing of A/D conversion results	(c)
<b>CHAPTER 14 EVENT LINK CONTROLLER (ELC)</b>		
p.568	Modifications of 14.1 Functions of ELC and 15.2 Configuration of ELC	(c)
p.569	Modification of 14.3 Registers Controlling ELC	(c)
p.571	Modification of 14.4 Operation	(c)
p.571	Modification of Figure 14 - 3 Relationship Between Interrupt Handling and ELC	(c)
p.571	Modification of Table 14 - 4 Response of Peripheral Functions That Receive Events	(c)
<b>CHAPTER 15 INTERRUPT FUNCTIONS</b>		
p.572	Modification of description	(c)
p.576	Modification of Table 15 - 2 Flags Corresponding to Interrupt Request Sources	(c)
p.578	Modification of 15.3.1 Interrupt request flag registers (IF0L, IF0H, IF1L)	(a)
p.578	Modification of Figure 15 - 2 Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L)	(c)
p.579	Modification of Figure 15 - 3 Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L)	(c)
p.580	Modification of Figure 15 - 4 Format of Priority Specification Flag Registers (PR00L, PR00H, PR01L, PR01H, PR10L, PR10H, PR11L)	(c)
p.581	Modification of Table 15-3 Ports Corresponding to EGPn and EGNn bits	(c)
p.582	Modification of 15.3.5 Program status word (PSW)	(c)
p.585	Modifications of Figure 15 - 8 Interrupt Request Acknowledgment Timing (Minimum Time) and Figure 15 - 9 Interrupt Request Acknowledgment Timing (Maximum Time)	(c)
p.587	Modification of Table 15 - 5 Relationship between Interrupt Requests Enabled for Multiple Interrupt Servicing during Interrupt Servicing	(c)
p.590	Modification of 15.4.4 Interrupt request hold	(c)
<b>CHAPTER 16 STANDBY FUNCTION</b>		
p.592	Modification of 16.2 Registers Controlling Standby Function	(c)

Page	Description	Classification
p.593 to p.594	Modification of Figure 16 - 1 Format of Oscillation Stabilization Time Counter Status Register (OSTC) to Figure 16 - 2 Format of Oscillation Stabilization Time Select Register (OSTS)	(c)
p.595	Modification of 16.3.1 HALT mode	(c)
p.596	Modification of Table 16 - 1 Operating Statuses in HALT Mode	(c)
p.597	Modification of Figure 16 - 3 HALT Mode Release by Interrupt Request Generation	(c)
p.599	Modification of 16.3.2 STOP mode	(c)
p.600	Modification of Table 16 - 2 Operating Statuses in STOP Mode	(c)
p.601, p.602	Modifications of Figure 16 - 5 STOP Mode Release by Interrupt Request Generation (1/2) and Figure 16 - 5 STOP Mode Release by Interrupt Request Generation (2/2)	(c)
p.604	Modification of 16.3.3 SNOOZE mode	(c)
p.605	Modification of Table 16-3 Operating Statuses in SNOOZE Mode	(c)
<b>CHAPTER 17 RESET FUNCTION</b>		
p.606	Modification of description	(c)
p.608	Modification of 17.1 Timing of Reset Operation	(c)
p.610	Modification of Table 17 - 1 Operation Statuses during Reset Period	(c)
p.612	Modification of Table 17-2 Hardware Statuses After Reset Acknowledgment (2/3)	(c)
p.614	Modification of Figure 17 - 4 Format of Reset control flag register (RESF)	(c)
p.616	Addition of Figure 17 - 5 Procedure for Checking Reset Source	(c)
<b>CHAPTER 18 POWER-ON-RESET CIRCUIT</b>		
p.617	Modification of 18.1 Functions of Power-on-reset Circuit	(c)
p.618	Modification of 18.3 Operation of Power-on-reset Circuit	(c)
p.619	Modification of Figure 18 - 2 Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (1/3)	(c)
p.621	Modification of Figure 18 - 2 Timing of Generation of Internal Reset Signal by Power-on-reset Circuit and Voltage Detector (3/3)	(c)
<b>CHAPTER 19 VOLTAGE DETECTOR</b>		
p.624	Modification of 19.1 Functions of Voltage Detector	(c)
p.625	Modification of Figure 19 - 1 Block Diagram of Voltage Detector	(c)
p.626	Modification of Figure 19 - 2 Format of Voltage detection register (LVIM)	(c)
p.627	Modification of Figure 19 - 3 Format of Voltage detection level register (LVIS)	(c)
p.628	Modification of Table 19 - 1 LVD Operation Mode and Detection Voltage Settings for User Option Byte (000C1H)	(c)
p.629	Modification of 19.4.1 When used as reset mode	(c)
p.630	Modification of Figure 19 - 4 Timing of Voltage Detector Internal Reset Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 1)	(c)
p.631	Modification of 19.4.2 When used as interrupt mode	(c)
p.632	Modification of Figure 19 - 5 Timing of Voltage Detector Internal Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 0, 1)	(c)
p.633	Modification of 19.4.3 When used as interrupt and reset mode	(c)
p.634 to p.638	Modifications of Figure 19 - 6 Timing of Voltage Detector Reset Signal and Interrupt Signal Generation (Option Byte LVIMDS1, LVIMDS0 = 1, 0)(1/2) to Figure 19 - 8 Setting Procedure for Initial Setting of Interrupt and Reset Mode	(c)
p.639, p.640	Modification of 19.5 Cautions for Voltage Detector	(c)

Page	Description	Classification
<b>CHAPTER 20 SAFETY FUNCTIONS</b>		
p.641	Modification of 20.1 Overview of Safety Functions	(c)
p.642	Modification of 20.3 Operation of Flash Memory CRC Operation Function (High-speed CRC)	(c)
p.644	Modification of Figure 20 - 3 Flowchart of Flash Memory CRC Operation Function (High-speed CRC)	(a)
p.648	Addition of Figure 20 - 8 RAM Parity Error Check Flow	(c)
p.651	Modification of Figure 20 - 11 Invalid Access Detection Area	(a)
p.653	Modification of 20.9 Frequency Detection Function	(c)
p.654	Modification of 20.9.1 Timer input select register 0 (TIS0)	(c)
p.655	Modification of 20.10 A/D Test Function	(c)
p.656	Modification of Figure 20 - 15 Configuration of A/D Test Function	(c)
p.658	Modification of Figure 20 - 17 Format of Analog input channel specification register (ADS)	(c)
<b>CHAPTER 22 OPTION BYTE</b>		
p.660	Modification of 22.1 Functions of Option Bytes	(c)
p.660	Modification of 22.1.1 User option byte (000C0H to 000C2H)	(c)
p.661	Modification of Figure 22 - 1 Format of User Option Byte (000C0H)	(c)
p.662	Modification of Figure 22 - 2 Format of User Option Byte (000C1H) (1/2)	(c)
p.663	Modification of Figure 22 - 3 Format of Option Byte (000C2H)	(c)
<b>CHAPTER 23 FLASH MEMORY</b>		
p.667	Modification of 23.1 Serial Programming Using Flash Memory Programmer	(c)
p.669	Modification of Figure 23 - 1 Environment for Writing Program to Flash Memory	(c)
p.670	Modification of 23.2 Serial Programming Using External Device (that Incorporates UART)	(c)
p.672	Modification of 23.3 Connection of Pins on Board	(c)
p.672	Modification of 23.3.1 P40/TOOL0 pin	(c)
p.673	Modification of 23.3.4 REGC pin	(c)
p.676	Modification of 23.4.3 Procedure for accessing data flash memory	(c)
p.677	Modification of 23.5.1 Controlling flash memory	(c)
p.678, p.679	Modification of 23.5.2 Flash memory programming mode	(c)
p.679	Modification of 23.5.3 Selecting communication mode	(c)
p.679	Modification of Table 23 - 6. Communication Modes	(c)
p.680	Modification of Table 23 - 7 Flash Memory Control Commands	(c)
p.683	Modification of 23.6 Security Settings	(c)
p.685	Modification of 23.7 Flash Memory Programming by Self-Programming	(c)
p.688	Addition of 23.8 Processing Time for Each Command When PG-FP5 Is in Use (Reference Value)	(c)
<b>CHAPTER 26 INSTRUCTION SET</b>		
p.696	Modification of Table 26 - 1 Operand Identifiers and Specification Methods	(c)
<b>CHAPTER 27 ELECTRICAL SPECIFICATIONS</b>		
p.721	Modification of 27.3.1 Pin characteristics	(b)
p.727 to p.730	Modification of 27.4 AC Characteristics	(c)
p.731 to p.734	Modification of 29.5 Peripheral Functions Characteristics	(b)
p.738 to p.739	Modification of 27.6 Analog Characteristics	(c)
p.742	Modification of 27.6.4 POR circuit characteristics	(c)

Page	Description	Classification
p.744	Modification of 27.7 Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics	(c)

**Remark** "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documentsd

---

R7F0C01072DNP, R7F0C010B2DFP-C User's Manual: Hardware

Publication Date: Rev.2.00 Sep 12, 2014

Published by: Renesas Electronics Corporation

---

**SALES OFFICES****Renesas Electronics Corporation**<http://www.renesas.com>Refer to "<http://www.renesas.com/>" for the latest and detailed information.**Renesas Electronics America Inc.**2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.  
Tel: +1-408-588-6000, Fax: +1-408-588-6130**Renesas Electronics Canada Limited**1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada  
Tel: +1-905-898-5441, Fax: +1-905-898-3220**Renesas Electronics Europe Limited**Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K  
Tel: +44-1628-585-100, Fax: +44-1628-585-900**Renesas Electronics Europe GmbH**Arcadiastrasse 10, 40472 Düsseldorf, Germany  
Tel: +49-211-6503-0, Fax: +49-211-6503-1327**Renesas Electronics (China) Co., Ltd.**Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China  
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679**Renesas Electronics (Shanghai) Co., Ltd.**Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333  
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999**Renesas Electronics Hong Kong Limited**Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong  
Tel: +852-2265-6688, Fax: +852 2686-9022/9044**Renesas Electronics Taiwan Co., Ltd.**13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan  
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670**Renesas Electronics Singapore Pte. Ltd.**80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949  
Tel: +65-6213-0200, Fax: +65-6213-0300**Renesas Electronics Malaysia Sdn.Bhd.**Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia  
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510**Renesas Electronics Korea Co., Ltd.**12F., 234 Teheran-ro, Gangnam-Ku, Seoul, 135-920, Korea  
Tel: +82-2-558-3737, Fax: +82-2-558-5141



R7F0C01072DNP, R7F0C010B2DFP-C