

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

R7F0C011B, R7F0C012B, R7F0C013B

User's Manual: Hardware

8-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 R7F0C011B, R7F0C012B, R7F0C013B and design and develop application systems and programs for these devices.

The target products are as follows.

- R7F0C011B2DFP
- R7F0C012B2DFP
- R7F0C013B2DFP

Purpose

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

Organization

The R7F0C011B, R7F0C012B, R7F0C013B manual is separated into two parts: this manual and the instructions edition (common to the 78K0R Microcontroller).

| |
|--|
| R7F0C011B, R7F0C012B, R7F0C013B User's Manual (This Manual) |
|--|

| |
|---|
| 78K0 Series User's Manual Instructions |
|---|

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

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 R7F0C011B, R7F0C012B, R7F0C013B Microcontroller instructions:
→ Refer to the separate document **78K0 Series Instructions User's Manual (U12326E)**.

Conventions

| | |
|-----------------------------|---|
| Data significance: | Higher digits on the left and lower digits on the right |
| Active low representations: | \overline{xxx} (overscore over pin and signal name) |
| Note: | Footnote for item marked with Note in the text |
| Caution: | Information requiring particular attention |
| Remark: | 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. |
|---|--------------|
| R7F0C011B, R7F0C012B, R7F0C013B User's Manual | This manual |
| 78K/0 Series Instructions User's Manual | U12326E |
| 78K0/Kx2 Flash Memory Programming (Programmer) Application Note | U17739E |

Documents Related to Flash Memory Programming

| Document Name | Document No. |
|---|--------------|
| PG-FP5 Flash Memory Programmer User's Manual | R20UT0008E |
| QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual | R20UT0449E |
| QB-Programmer Programming GUI | U18257E |

Documents Related to Development Tools (Hardware)

| Document Name | Document No. |
|---|--------------|
| QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual | R20UT0449E |

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

Documents Related to Development Tools (Software)

| Document Name | | Document No. |
|--|------------------------------|------------------|
| RA78K0 Ver.3.80 Assembler Package User's Manual ^{Note 1} | Operation | U17199E |
| | Language | U17198E |
| | Structured Assembly Language | U17197E |
| 78K0 Assembler Package RA78K0 Ver.4.01 Operating Precautions (Notification Document) ^{Note 1} | | ZUD-CD-07-0181-E |
| CC78K0 Ver.3.70 C Compiler User's Manual ^{Note 2} | Operation | U17201E |
| | Language | U17200E |
| 78K0 C Compiler CC78K0 Ver. 4.00 Operating Precautions (Notification Document) ^{Note 2} | | ZUD-CD-07-0103-E |
| SM+ System Simulator User's Manual | Operation | U18601E |
| | User Open Interface | U18212E |
| ID78K0-QB Ver.2.94 Integrated Debugger User's Manual | Operation | U18330E |
| ID78K0-QB Ver.3.00 Integrated Debugger User's Manual | Operation | U18492E |
| PM plus Ver.5.20 ^{Note 3} User's Manual | | U16934E |
| PM+ Ver.6.30 ^{Note 4} User's Manual | | U18416E |

- Notes 1.** This document is installed into the PC together with the tool when installing RA78K0 Ver. 4.01. For descriptions not included in "78K0 Assembler Package RA78K0 Ver. 4.01 Operating Precautions", refer to the user's manual of RA78K0 Ver. 3.80.
- 2.** This document is installed into the PC together with the tool when installing CC78K0 Ver. 4.00. For descriptions not included in "78K0 C Compiler CC78K0 Ver. 4.00 Operating Precautions", refer to the user's manual of CC78K0 Ver. 3.70.
- 3.** PM plus Ver. 5.20 is the integrated development environment included with RA78K0 Ver. 3.80.
- 4.** PM+ Ver. 6.30 is the integrated development environment included with RA78K0 Ver. 4.01. Software tool (assembler, C compiler, debugger, and simulator) products of different versions can be managed.

Other Documents

| Document Name | Document No. |
|--|--------------|
| RENESAS MICROCOMPUTER GENERAL CATALOG | R01CS0001E |
| Semiconductor Package Mount Manual | Note |
| Quality Grades on NEC Semiconductor Devices | C11531E |
| NEC Semiconductor Device Reliability/Quality Control System | C10983E |
| Guide to Prevent Damage for Semiconductor Devices by Electrostatic Discharge (ESD) | C11892E |

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. Windows is a registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. 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

| | |
|---|-----------|
| CHAPTER 1 OUTLINE | 1 |
| 1.1 Features | 1 |
| 1.2 Applications | 2 |
| 1.3 Ordering Information | 2 |
| 1.4 Pin Configuration (Top View) | 3 |
| 1.5 Pin Identification | 5 |
| 1.6 Block Diagram | 6 |
| 1.7 Outline of Functions | 8 |
| | |
| CHAPTER 2 PIN FUNCTIONS | 10 |
| 2.1 Pin Function List | 10 |
| 2.2 Description of Pin Functions | 14 |
| 2.2.1 P00, P01 (port 0) | 14 |
| 2.2.2 P10 to P17 (port 1) | 14 |
| 2.2.3 P20 to P23 (port 2) | 16 |
| 2.2.4 P30 to P33 (port 3) | 16 |
| 2.2.5 P40 and P41 (port 4) | 17 |
| 2.2.6 P60 and P61 (port 6) | 17 |
| 2.2.7 P70 and P71 (port 7) | 17 |
| 2.2.8 P120 to P122 (port 12) | 17 |
| 2.2.9 V_{DD} , V_{SS} | 18 |
| 2.2.10 RESET | 18 |
| 2.2.11 REGC | 18 |
| 2.2.12 FLMD0 | 18 |
| 2.3 Pin I/O Circuits and Recommended Connection of Unused Pins | 19 |
| | |
| CHAPTER 3 CPU ARCHITECTURE | 22 |
| 3.1 Memory Space | 22 |
| 3.1.1 Internal program memory space | 27 |
| 3.1.2 Internal data memory space | 29 |
| 3.1.3 Special function register (SFR) area | 29 |
| 3.1.4 Data memory addressing | 29 |
| 3.2 Processor Registers | 33 |
| 3.2.1 Control registers | 33 |
| 3.2.2 General-purpose registers | 36 |
| 3.2.3 Special function registers (SFRs) | 38 |
| 3.3 Instruction Address Addressing | 43 |
| 3.3.1 Relative addressing | 43 |
| 3.3.2 Immediate addressing..... | 44 |
| 3.3.3 Table indirect addressing..... | 45 |
| 3.3.4 Register addressing | 45 |
| 3.4 Operand Address Addressing | 46 |
| 3.4.1 Implied addressing..... | 46 |
| 3.4.2 Register addressing..... | 47 |
| 3.4.3 Direct addressing | 48 |

| | | |
|---|---|------------|
| 3.4.4 | Short direct addressing | 49 |
| 3.4.5 | Special function register (SFR) addressing..... | 50 |
| 3.4.6 | Register indirect addressing | 51 |
| 3.4.7 | Based addressing | 52 |
| 3.4.8 | Based indexed addressing..... | 53 |
| 3.4.9 | Stack addressing | 54 |
| CHAPTER 4 PORT FUNCTIONS | | 55 |
| 4.1 | Port Functions | 55 |
| 4.2 | Port Configuration..... | 57 |
| 4.2.1 | Port 0 | 58 |
| 4.2.2 | Port 1 | 60 |
| 4.2.3 | Port 2 | 66 |
| 4.2.4 | Port 3 | 68 |
| 4.2.5 | Port 4 | 70 |
| 4.2.6 | Port 6 | 71 |
| 4.2.7 | Port 7 | 72 |
| 4.2.8 | Port 12 | 73 |
| 4.3 | Registers Controlling Port Function | 75 |
| 4.4 | Port Function Operations | 79 |
| 4.4.1 | Writing to I/O port..... | 79 |
| 4.4.2 | Reading from I/O port | 79 |
| 4.4.3 | Operations on I/O port | 79 |
| 4.5 | Settings of Port Mode Register and Output Latch When Using Alternate Function..... | 80 |
| 4.6 | Cautions on 1-Bit Manipulation Instruction for Port Register n (Pn)..... | 82 |
| CHAPTER 5 CLOCK GENERATOR | | 83 |
| 5.1 | Functions of Clock Generator..... | 83 |
| 5.2 | Configuration of Clock Generator | 84 |
| 5.3 | Registers Controlling Clock Generator..... | 86 |
| 5.4 | System Clock Oscillator | 93 |
| 5.4.1 | X1 oscillator | 93 |
| 5.4.2 | Internal high-speed oscillator | 95 |
| 5.4.3 | Internal low-speed oscillator | 95 |
| 5.4.4 | Prescaler..... | 95 |
| 5.5 | Clock Generator Operation | 96 |
| 5.6 | Controlling Clock | 99 |
| 5.6.1 | Example of controlling high-speed system clock | 99 |
| 5.6.2 | Example of controlling internal high-speed oscillation clock | 102 |
| 5.6.3 | Example of controlling internal low-speed oscillation clock..... | 105 |
| 5.6.4 | Clocks supplied to CPU and peripheral hardware | 105 |
| 5.6.5 | CPU clock status transition diagram | 106 |
| 5.6.6 | Condition before changing CPU clock and processing after changing CPU clock..... | 109 |
| 5.6.7 | Time required for switchover of CPU clock and main system clock..... | 110 |
| 5.6.8 | Conditions before clock oscillation is stopped..... | 111 |
| 5.6.9 | Peripheral hardware and source clocks..... | 111 |
| CHAPTER 6 16-BIT TIMER/EVENT COUNTER 00..... | | 112 |
| 6.1 | Functions of 16-Bit Timer/Event Counter 00 | 112 |

| | | |
|---|---|------------|
| 6.2 | Configuration of 16-Bit Timer/Event Counter 00 | 113 |
| 6.3 | Registers Controlling 16-Bit Timer/Event Counter 00 | 119 |
| 6.4 | Operation of 16-Bit Timer/Event Counter 00 | 127 |
| 6.4.1 | Interval timer operation | 127 |
| 6.4.2 | Square-wave output operation..... | 130 |
| 6.4.3 | External event counter operation | 134 |
| 6.4.4 | Operation in clear & start mode entered by TI000 pin valid edge input | 138 |
| 6.4.5 | Free-running timer operation | 154 |
| 6.4.6 | PPG output operation | 164 |
| 6.4.7 | One-shot pulse output operation..... | 168 |
| 6.4.8 | Pulse width measurement operation..... | 173 |
| 6.5 | Special Use of TM00 | 182 |
| 6.5.1 | Rewriting CR010 during TM00 operation..... | 182 |
| 6.5.2 | Setting LVS00 and LVR00 | 182 |
| 6.6 | Cautions for 16-Bit Timer/Event Counter 00 | 184 |
| CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 50 AND 51 | | 189 |
| 7.1 | Functions of 8-Bit Timer/Event Counters 50 and 51 | 189 |
| 7.2 | Configuration of 8-Bit Timer/Event Counters 50 and 51 | 189 |
| 7.3 | Registers Controlling 8-Bit Timer/Event Counters 50 and 51 | 192 |
| 7.4 | Operations of 8-Bit Timer/Event Counters 50 and 51 | 197 |
| 7.4.1 | Operation as interval timer..... | 197 |
| 7.4.2 | Operation as external event counter..... | 199 |
| 7.4.3 | Square-wave output operation..... | 200 |
| 7.4.4 | PWM output operation | 201 |
| 7.5 | Cautions for 8-Bit Timer/Event Counters 50 and 51 | 205 |
| CHAPTER 8 8-BIT TIMERS H0 AND H1 | | 206 |
| 8.1 | Functions of 8-Bit Timers H0 and H1 | 206 |
| 8.2 | Configuration of 8-Bit Timers H0 and H1 | 206 |
| 8.3 | Registers Controlling 8-Bit Timers H0 and H1 | 210 |
| 8.4 | Operation of 8-Bit Timers H0 and H1 | 215 |
| 8.4.1 | Operation as interval timer/square-wave output | 215 |
| 8.4.2 | Operation as PWM output..... | 218 |
| 8.4.3 | Carrier generator operation (8-bit timer H1 only) | 224 |
| CHAPTER 9 WATCH TIMER | | 231 |
| 9.1 | Functions of Watch Timer | 231 |
| 9.2 | Configuration of Watch Timer | 232 |
| 9.3 | Register Controlling Watch Timer | 233 |
| 9.4 | Watch Timer Operations | 234 |
| 9.4.1 | Watch timer operation..... | 234 |
| 9.4.2 | Interval timer operation | 235 |
| 9.5 | Cautions for Watch Timer | 235 |
| CHAPTER 10 WATCHDOG TIMER | | 236 |
| 10.1 | Functions of Watchdog Timer | 236 |
| 10.2 | Configuration of Watchdog Timer | 237 |

| | | |
|-------------------|---|------------|
| 10.3 | Register Controlling Watchdog Timer | 238 |
| 10.4 | Operation of Watchdog Timer..... | 239 |
| 10.4.1 | Controlling operation of watchdog timer..... | 239 |
| 10.4.2 | Setting overflow time of watchdog timer | 240 |
| 10.4.3 | Setting window open period of watchdog timer..... | 241 |
| CHAPTER 11 | A/D CONVERTER | 243 |
| 11.1 | Function of A/D Converter | 243 |
| 11.2 | Configuration of A/D Converter..... | 244 |
| 11.3 | Registers Used in A/D Converter..... | 246 |
| 11.4 | A/D Converter Operations | 252 |
| 11.4.1 | Basic operations of A/D converter | 252 |
| 11.4.2 | Input voltage and conversion results | 253 |
| 11.4.3 | A/D converter operation mode..... | 255 |
| 11.5 | How to Read A/D Converter Characteristics Table..... | 257 |
| 11.6 | Cautions for A/D Converter..... | 260 |
| CHAPTER 12 | SERIAL INTERFACE UART0 | 264 |
| 12.1 | Functions of Serial Interface UART0..... | 264 |
| 12.2 | Configuration of Serial Interface UART0 | 265 |
| 12.3 | Registers Controlling Serial Interface UART0..... | 268 |
| 12.4 | Operation of Serial Interface UART0..... | 273 |
| 12.4.1 | Operation stop mode | 273 |
| 12.4.2 | Asynchronous serial interface (UART) mode..... | 274 |
| 12.4.3 | Dedicated baud rate generator | 280 |
| 12.4.4 | Calculation of baud rate..... | 282 |
| CHAPTER 13 | SERIAL INTERFACE UART6 | 286 |
| 13.1 | Functions of Serial Interface UART6..... | 286 |
| 13.2 | Configuration of Serial Interface UART6 | 291 |
| 13.3 | Registers Controlling Serial Interface UART6..... | 294 |
| 13.4 | Operation of Serial Interface UART6..... | 304 |
| 13.4.1 | Operation stop mode | 304 |
| 13.4.2 | Asynchronous serial interface (UART) mode..... | 305 |
| 13.4.3 | Dedicated baud rate generator | 319 |
| 13.4.4 | Calculation of baud rate..... | 320 |
| CHAPTER 14 | SERIAL INTERFACE CSI10 | 326 |
| 14.1 | Functions of Serial Interface CSI10..... | 326 |
| 14.2 | Configuration of Serial Interface CSI10 | 326 |
| 14.3 | Registers Controlling Serial Interface CSI10..... | 328 |
| 14.4 | Operation of Serial Interface CSI10..... | 332 |
| 14.4.1 | Operation stop mode | 332 |
| 14.4.2 | 3-wire serial I/O mode..... | 333 |
| CHAPTER 15 | SERIAL INTERFACE IIC0..... | 343 |
| 15.1 | Functions of Serial Interface IIC0 | 343 |
| 15.2 | Configuration of Serial Interface IIC0..... | 346 |

| | | |
|---|---|------------|
| 15.3 | Registers to Control Serial Interface IIC0 | 349 |
| 15.4 | I²C Bus Mode Functions | 362 |
| 15.4.1 | Pin configuration | 362 |
| 15.5 | I²C Bus Definitions and Control Methods | 363 |
| 15.5.1 | Start conditions | 363 |
| 15.5.2 | Addresses | 364 |
| 15.5.3 | Transfer direction specification | 364 |
| 15.5.4 | Acknowledge (\overline{ACK}) | 365 |
| 15.5.5 | Stop condition | 366 |
| 15.5.6 | Wait | 367 |
| 15.5.7 | Canceling wait | 369 |
| 15.5.8 | Interrupt request (INTIIC0) generation timing and wait control | 370 |
| 15.5.9 | Address match detection method | 371 |
| 15.5.10 | Error detection | 371 |
| 15.5.11 | Extension code | 372 |
| 15.5.12 | Arbitration | 373 |
| 15.5.13 | Wakeup function | 375 |
| 15.5.14 | Communication reservation | 375 |
| 15.5.15 | Cautions | 379 |
| 15.5.16 | Communication operations | 380 |
| 15.5.17 | Timing of I ² C interrupt request (INTIIC0) occurrence | 388 |
| 15.6 | Timing Charts | 409 |
| CHAPTER 16 INTERRUPT FUNCTIONS..... | | 416 |
| 16.1 | Interrupt Function Types | 416 |
| 16.2 | Interrupt Sources and Configuration | 416 |
| 16.3 | Registers Controlling Interrupt Functions..... | 421 |
| 16.4 | Interrupt Servicing Operations | 429 |
| 16.4.1 | Maskable interrupt acknowledgment | 429 |
| 16.4.2 | Software interrupt request acknowledgment..... | 431 |
| 16.4.3 | Multiple interrupt servicing | 432 |
| 16.4.4 | Interrupt request hold..... | 435 |
| CHAPTER 17 STANDBY FUNCTION | | 436 |
| 17.1 | Standby Function and Configuration..... | 436 |
| 17.1.1 | Standby function | 436 |
| 17.1.2 | Registers controlling standby function | 437 |
| 17.2 | Standby Function Operation | 439 |
| 17.2.1 | HALT mode..... | 439 |
| 17.2.2 | STOP mode | 442 |
| CHAPTER 18 RESET FUNCTION..... | | 448 |
| 18.1 | Register for Confirming Reset Source..... | 457 |
| CHAPTER 19 POWER-ON-CLEAR CIRCUIT..... | | 458 |
| 19.1 | Functions of Power-on-Clear Circuit..... | 458 |
| 19.2 | Configuration of Power-on-Clear Circuit | 459 |
| 19.3 | Operation of Power-on-Clear Circuit..... | 459 |

| | | |
|--|---|------------|
| 19.4 | Cautions for Power-on-Clear Circuit | 462 |
| CHAPTER 20 LOW-VOLTAGE DETECTOR | | 464 |
| 20.1 | Functions of Low-Voltage Detector..... | 464 |
| 20.2 | Configuration of Low-Voltage Detector | 464 |
| 20.3 | Registers Controlling Low-Voltage Detector | 465 |
| 20.4 | Operation of Low-Voltage Detector..... | 468 |
| 20.4.1 | When used as reset..... | 469 |
| 20.4.2 | When used as interrupt..... | 474 |
| 20.5 | Cautions for Low-Voltage Detector | 479 |
| CHAPTER 21 OPTION BYTE..... | | 482 |
| 21.1 | Functions of Option Bytes | 482 |
| 21.2 | Format of Option Byte | 484 |
| CHAPTER 22 FLASH MEMORY | | 487 |
| 22.1 | Internal Memory Size Switching Register..... | 487 |
| 22.2 | Writing with Flash Memory Programmer..... | 488 |
| 22.3 | Programming Environment..... | 488 |
| 22.4 | Communication Mode..... | 489 |
| 22.5 | Connection of Pins on Board..... | 491 |
| 22.5.1 | FLMDO pin | 492 |
| 22.5.2 | Serial interface pins | 492 |
| 22.5.3 | $\overline{\text{RESET}}$ pin..... | 493 |
| 22.5.4 | Port pins..... | 494 |
| 22.5.5 | REGC pin..... | 494 |
| 22.5.6 | Other signal pins..... | 494 |
| 22.5.7 | Power supply | 494 |
| 22.6 | Programming Method | 495 |
| 22.6.1 | Controlling flash memory | 495 |
| 22.6.2 | Flash memory programming mode | 496 |
| 22.6.3 | Selecting communication mode | 497 |
| 22.6.4 | Communication commands | 497 |
| 22.7 | Security Settings..... | 499 |
| CHAPTER 23 ON-CHIP DEBUG FUNCTION (R7F0C999B ONLY)..... | | 501 |
| 23.1 | Connecting QB-MINI2 to R7F0C999B..... | 501 |
| 23.2 | On-Chip Debug Security ID | 506 |
| 23.3 | Securing of User Resources | 507 |
| CHAPTER 24 INSTRUCTION SET..... | | 508 |
| 24.1 | Conventions Used in Operation List | 509 |
| 24.1.1 | Operand identifiers and specification methods | 509 |
| 24.1.2 | Description of operation column | 510 |
| 24.1.3 | Description of flag operation column..... | 510 |
| 24.2 | Operation List..... | 511 |
| 24.3 | Instructions Listed by Addressing Type..... | 519 |

| | |
|---|------------|
| CHAPTER 25 ELECTRICAL SPECIFICATIONS (TARGET)..... | 523 |
| CHAPTER 26 PACKAGE DRAWINGS | 541 |
| CHAPTER 27 CAUTIONS FOR WAIT..... | 542 |
| 27.1 Cautions for Wait..... | 542 |
| 27.2 Peripheral Hardware That Generates Wait | 543 |

CHAPTER 1 OUTLINE

1.1 Features

- Minimum instruction execution time: 0.2 μ s (@ 10 MHz operation with high-speed system clock)
- General-purpose register: 8 bits \times 32 registers (8 bits \times 8 registers \times 4 banks)
- I/O ports, ROM, RAM capacities

| Item | I/O ports | Program Memory (Flash Memory) | Data Memory (Internal High-Speed RAM) |
|-------------------------------|---|----------------------------------|--|
| R7F0C011B2001DFP | 27 (CMOS I/O: 25, N-ch open drain I/O: 2) | 16 KB | 768 bytes |
| R7F0C012B2001DFP | | 24 KB | 1 KB |
| R7F0C013B2001DFP | | 32 KB | 1 KB |
| R7F0C999B2DFP ^{Note} | | | |

- On-chip single-power-supply flash memory
- On-chip power-on-clear (POC) circuit and low-voltage detector (LVI)
- On-chip watchdog timer (operable with the on-chip internal low-speed oscillation clock)
- Timer
 - 16-bit timer/event counter ... PPG output, capture input, external event counter input
 - 8-bit timers H0, H1 ... PWM output, operable with internal low-speed oscillation clock
 - 8-bit timer/event counters 50, 51 ... External event counter input
 - Watch timer
 - Watchdog timer ... Operable with internal low-speed oscillation clock

| Item | 16-Bit Timer/Event Counter | 8-Bit Timer | Watch Timer | Watchdog Timer |
|-------------------------------|----------------------------|--|-------------|----------------|
| R7F0C011B2001DFP | 1 channel | Timer H: 2 channels Timer 5: 2 channels | 1 channel | 1 channel |
| R7F0C012B2001DFP | | | | |
| R7F0C013B2001DFP | | | | |
| R7F0C999B2DFP ^{Note} | | | | |

- Serial interface
 - UART ... 2-wire serial interface supporting asynchronous communication
 - CSI ... 3-wire serial interface supporting clocked communication
 - IICA ... 2-wire serial interface supporting clocked communication. Supporting multi-master and, in slave mode, capable of releasing standby by address match

| Item | UART6 | CSI10/UART0 | IIC |
|-------------------------------|-----------|-------------|-----------|
| R7F0C011B2001DFP | 1 channel | 1 channel | 1 channel |
| R7F0C012B2001DFP | | | |
| R7F0C013B2001DFP | | | |
| R7F0C999B2DFP ^{Note} | | | |

Note R7F0C999B2DFP can be used only for an on-chip debug function.

- On-chip 10-bit resolution A/D converter ($AV_{REF} = 4.0$ to 5.5 V): 4 channels
- Power supply voltage: $V_{DD} = 4.0$ to 5.5 V
- Operating ambient temperature: $T_A = -40$ to $+85^\circ\text{C}$

1.2 Applications

- Household electrical appliances
 - Air conditioners

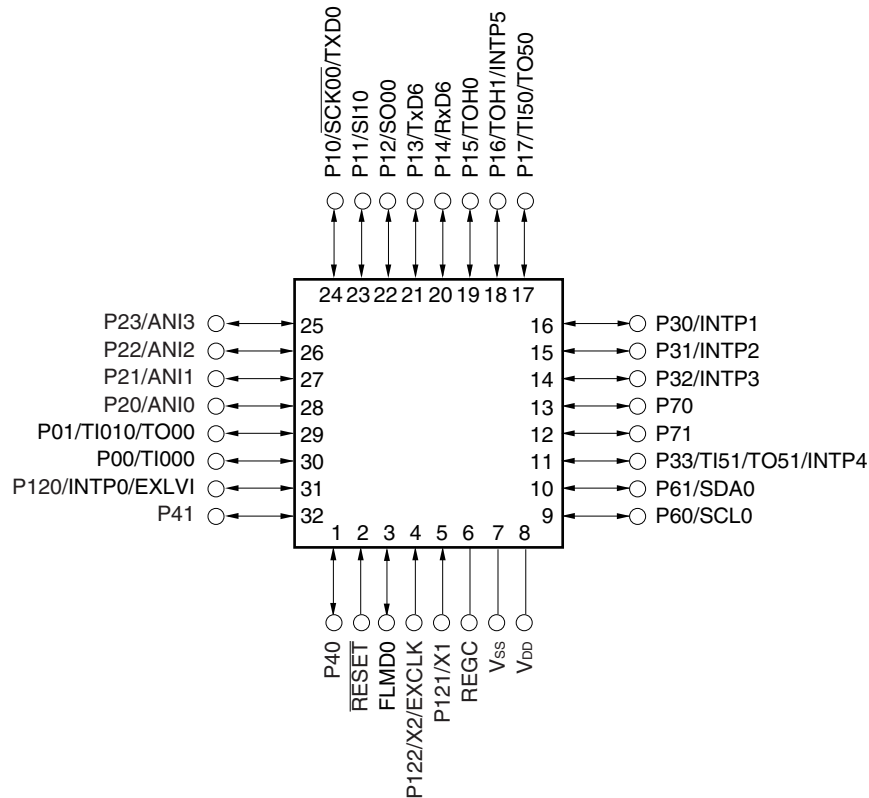
1.3 Ordering Information

| Pin Count | Package | ROM | RAM | Semiconductor Material | Part Number |
|-----------|-----------------------------|-------|-------|--|---|
| 32-pin | 32-pin plastic LQFP (7 × 7) | 16 KB | 768 B | Lead free product (External pin finish is Ni/Pd/Au plating) | R7F0C011B2001DFP |
| | | 24 KB | 1 KB | | R7F0C012B2001DFP |
| | | 32 KB | 1 KB | | R7F0C013B2001DFP |
| | | 32 KB | 1 KB | | R7F0C999B2DFP (for on-chip debug only) |

1.4 Pin Configuration (Top View)

(1) R7F0C011B, R7F0C012B, R7F0C013B

- 32-pin plastic LQFP (fine pitch) (7 × 7)

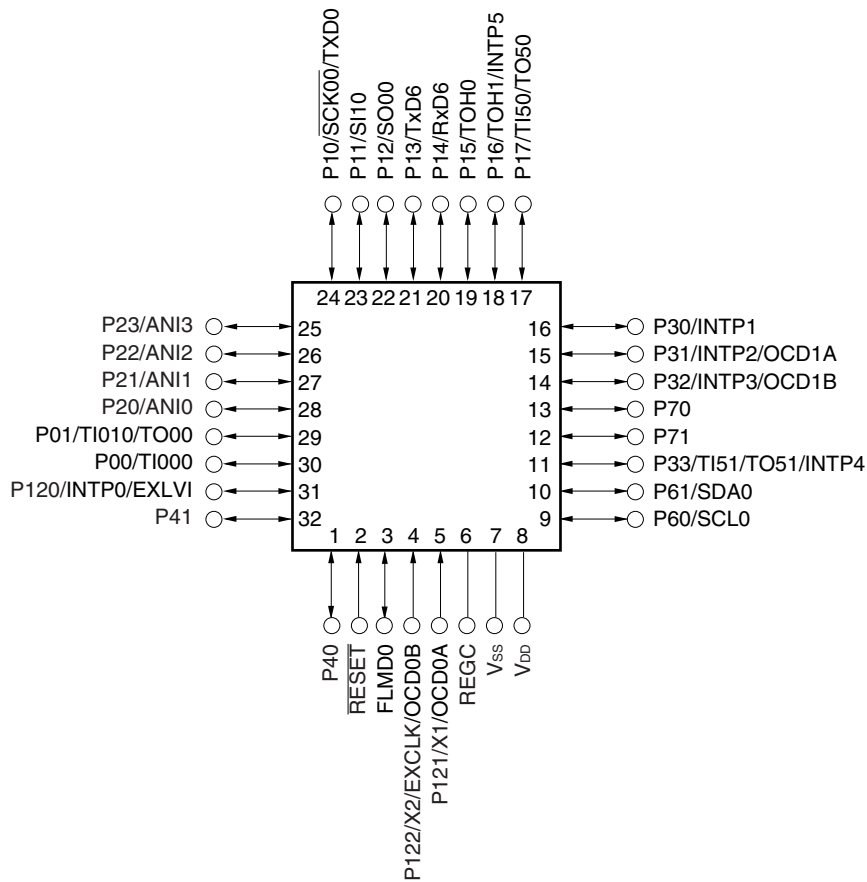


- Cautions**
1. Connect the REGC pin to V_{ss} via a capacitor (0.47 to 1 μ F).
 2. ANI0/P20 to ANI3/P23 are set in the analog input mode after release of reset.

Remark For pin identification, see 1.5 Pin Identification.

(2) R7F0C999B

- 32-pin plastic LQFP (fine pitch) (7 × 7)



- Cautions**
1. Connect the REGC pin to V_{SS} via a capacitor (0.47 to 1 μ F).
 2. ANI0/P20 to ANI3/P23 are set in the analog input mode after release of reset.

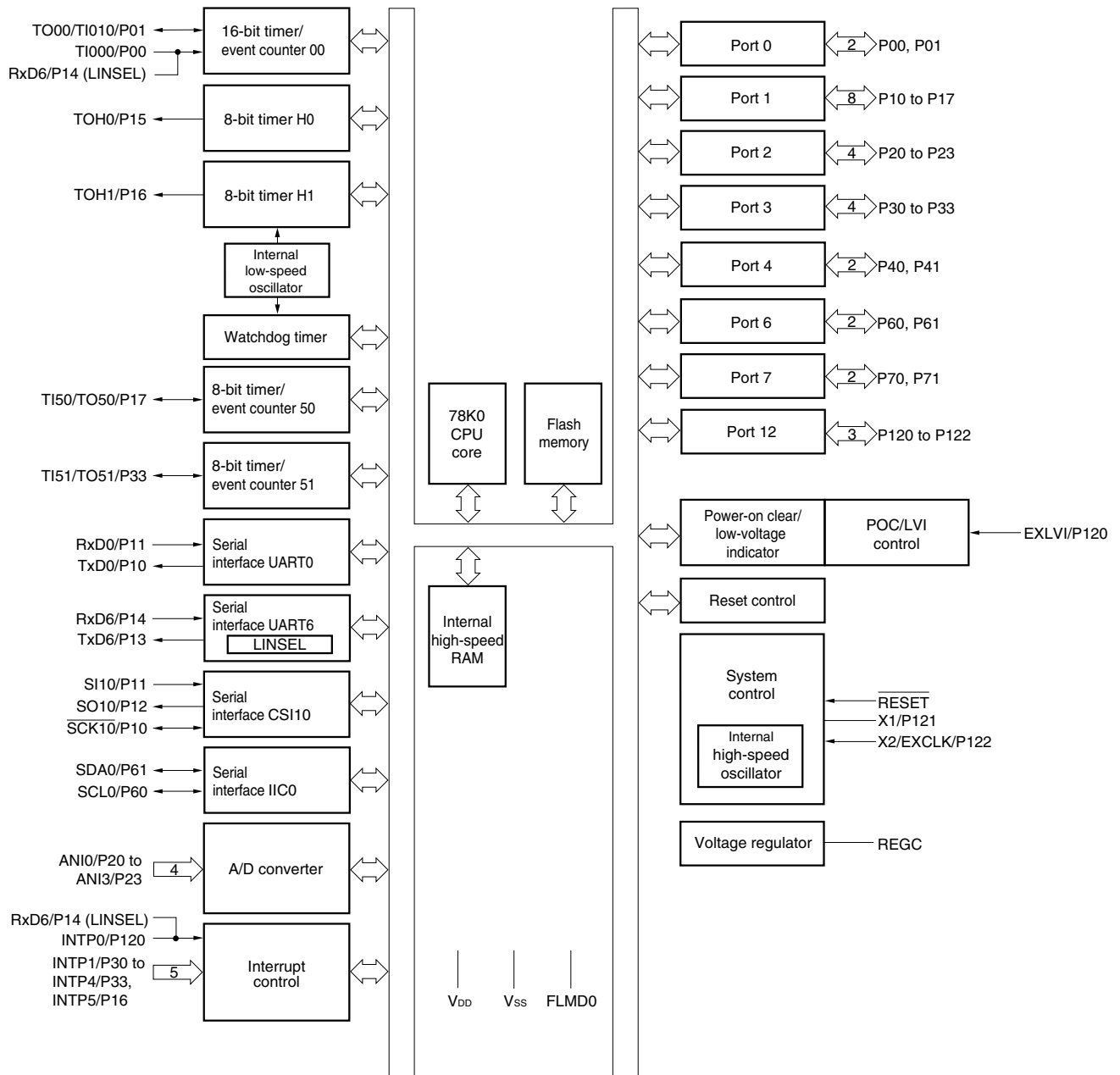
Remark For pin identification, see 1.5 Pin Identification.

1.5 Pin Identification

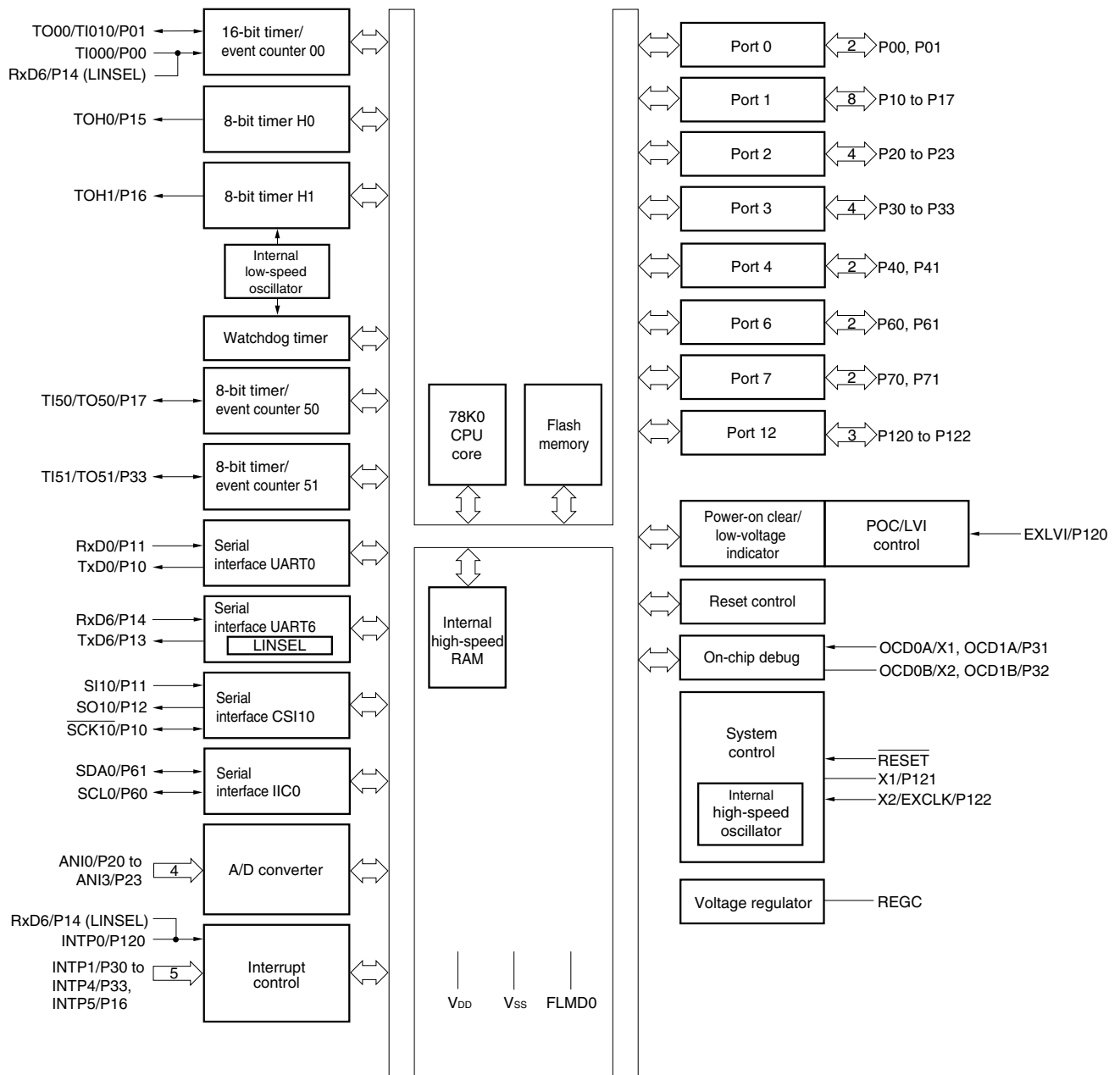
| | | | |
|-----------------|--|-----------------------------|-----------------------------------|
| ANI0 to ANI3: | Analog input | REGC: | Regulator capacitance |
| EXCLK: | External clock input | $\overline{\text{RESET}}$: | Reset |
| EXLVI: | External potential input for Low-voltage detector | RxD0, RxD6: | Receive data |
| FLMD0: | Flash programming mode | $\overline{\text{SCK10}}$: | Serial clock input/output |
| INTP0, INTP1: | External interrupt input | SCL0: | Serial clock input/output |
| INTP2 to INTP5: | External interrupt input | SDA0: | Serial data input/output |
| OCD0A, OCD0B | On-chip debug input/output | SI10: | Serial data input |
| OCD1A, OCD1B | On-chip debug input/output | SO10: | Serial data output |
| P00, P01: | Port 0 | TI000, TI010: | Timer input |
| P10 to P17: | Port 1 | TI50, TI51: | Timer input |
| P20 to P23: | Port 2 | TO00, TO01 : | Timer output |
| P30 to P33: | Port 3 | TO50, TO51: | Timer output |
| P40, P41: | Port 4 | TOH0, TOH1: | Timer output |
| P60, P61: | Port 6 | TxD0, TxD6: | Transmit data |
| P70, P71: | Port 7 | V _{DD} : | Power supply |
| P120 to P122: | Port 12 | V _{SS} : | Ground |
| | | X1, X2: | Crystal oscillator (system clock) |

1.6 Block Diagram

(1) R7F0C011B, R7F0C012B, R7F0C013B



(2) R7F0C999B



1.7 Outline of Functions

| Item | | Products | | | |
|------------------------------------|---------------------------------------|--|------------------|------------------|--------------------------------|
| | | R7F0C011B2001DFP | R7F0C012B2001DFP | R7F0C013B2001DFP | R7F0C9992DFP ^{Note 1} |
| Flash memory (KB) | | 16 | 24 | 32 | 32 |
| High-speed RAM (KB) | | 0.75 | 1 | 1 | 1 |
| Power supply voltage | | V _{DD} = 4.0 to 5.5 V | | | |
| Regulator | | Provided | | | |
| Minimum instruction execution time | | 0.2 μ s (10 MHz: V _{DD} = 4.0 to 5.5 V) | | | |
| Clock | High-speed system | 10 MHz: V _{DD} = 4.0 to 5.5 V | | | |
| | Internal high-speed oscillation | 8 MHz (TYP.): V _{DD} = 4.0 to 5.5 V | | | |
| | Internal low-speed oscillation | 240 kHz (TYP.): V _{DD} = 4.0 to 5.5 V | | | |
| Port | Total | 27 | | | |
| | N-ch O.D. (6 V tolerance) | 2 | | | |
| Timer | 16 bits (TM0) | 1 ch | | | |
| | 8 bits (TM5) | 2 ch | | | |
| | 8 bits (TMH) | 2 ch | | | |
| | Watch | 1 ch | | | |
| | Watchdog (WDT) | 1 ch | | | |
| Serial interface | 3-wire CSI | - | | | |
| | Automatic transmit/receive 3-wire CSI | - | | | |
| | UART/3-wire CSI ^{Note 2} | 1 ch | | | |
| | UART supporting LIN-bus | 1 ch | | | |
| | I ² C bus | 1 ch | | | |
| 10-bit A/D | | 4 ch | | | |
| Interrupt | External | 6 | | | |
| | Internal | 14 | | | |
| Reset | RESET pin | Provided | | | |
| | POC | 1.59 V \pm 0.15 V | | | |
| | LVI | The detection level of the supply voltage is selectable. | | | |
| | WDT | Provided | | | |
| On-chip debug function | | _Note 3 | | | Provided |
| Operating ambient temperature | | T _A = -40 to +85 °C | | | |

- Notes**
1. R7F0C999B2DFP can be used only for an on-chip debug function.
 2. Select either of the functions of these alternate-function pins.
 3. Supported by R7F0C999B2DFP

An outline of the timer is shown below.

| | | 16-Bit Timer/ Event Counter 00 | 8-Bit Timer/ Event Counters 50 and 51 | | 8-Bit Timers H0 and H1 | | Watch Timer | Watchdog Timer |
|------------------|-------------------------|-----------------------------------|---|-----------|------------------------|----------------------------|-----------------------------|-------------------|
| | | TM00 | TM50 | TM51 | TMH0 | TMH1 | | |
| Function | Interval timer | 1 channel | 1 channel | 1 channel | 1 channel | 1 channel | 1 channel ^{Note 1} | – |
| | External event counter | 1 channel | 1 channel | 1 channel | – | – | – | – |
| | PPG output | 1 output | – | – | – | – | – | – |
| | PWM output | – | 1 output | 1 output | 1 output | 1 output | – | – |
| | Pulse width measurement | 2 inputs | – | – | – | – | – | – |
| | Square-wave output | 1 output | 1 output | 1 output | 1 output | 1 output | – | – |
| | Carrier generator | – | – | – | – | 1 output ^{Note 2} | – | – |
| | Timer output | – | – | – | – | – | 1 channel ^{Note 1} | – |
| | Watchdog timer | – | – | – | – | – | – | 1 channel |
| Interrupt source | | 2 | 1 | 1 | 1 | 1 | 1 | – |

- Notes**
1. In the watch timer, the watch timer function and interval timer function can be used simultaneously.
 2. TM51 and TMH1 can be used in combination as a carrier generator mode.

CHAPTER 2 PIN FUNCTIONS**2.1 Pin Function List**

Pin I/O buffer power supplies include one V_{DD} system. The relationship between these power supplies and the pins is shown below.

Table 2-1. Pin I/O Buffer Power Supplies (AV_{REF} , V_{DD})

| Power Supply | Corresponding Pins |
|--------------|--------------------|
| V_{DD} | All pins |

(1) Port functions

| Function Name | I/O | Function | After Reset | Alternate Function |
|---------------|-----|---|--------------|----------------------------|
| P00 | I/O | Port 0. 2-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. | Input port | TI000 |
| P01 | | | | TI010/TO00 |
| 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. | Input port | SCK10/TxD0 |
| P11 | | | | SI10/RxD0 |
| P12 | | | | SO10 |
| P13 | | | | TxD6 |
| P14 | | | | RxD6 |
| P15 | | | | TOH0 |
| P16 | | | | TOH1/INTP5 |
| P17 | | | | TI50/TO50 |
| P20 to P23 | I/O | Port 2. 4-bit I/O port. Input/output can be specified in 1-bit units. | Analog input | ANI0 to ANI3 |
| P30 | I/O | Port 3. 4-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. | Input port | INTP1 |
| P31 | | | | INTP2 ^{Note 1} |
| P32 | | | | INTP3 ^{Note 2} |
| P33 | | | | INTP4/TI51/TO51 |
| P40, P41 | I/O | Port 4. 2-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. | Input port | — |
| P60 | I/O | Port 6. 2-bit I/O port. Output is N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units. | Input port | SCL0 |
| P61 | | | | SDA0 |
| P70, P71 | I/O | Port 7. 2-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. | Input port | — |
| P120 | I/O | Port 12. 3-bit I/O port. Input/output can be specified in 1-bit units. Only for P120, use of an on-chip pull-up resistor can be specified by a software setting. | Input port | INTP0/EXLVI |
| P121 | | | | X1 ^{Note 3} |
| P122 | | | | X2/EXCLK ^{Note 4} |

- Notes**
1. P31 of R7F0C999B2DFP is also used as INTP2/OCD1A.
 2. P32 of R7F0C999B2DFP is also used as INTP3/OCD1B.
 3. P121 of R7F0C999B2DFP is also used as X1/OCD0A.
 4. P122 of R7F0C999B2DFP is also used as X2/EXCLK/OCD0B.

(2) Non-port functions (1/2)

| Function Name | I/O | Function | After Reset | Alternate Function |
|---------------------------|--------|---|--------------|--------------------------------|
| ANI0 to ANI3 | Input | A/D converter analog input | Analog input | P20 to P23 |
| EXLVI | Input | Potential input for external low-voltage detection | Input port | P120/INTP0 |
| FLMD0 | – | Flash memory programming mode setting | – | – |
| INTP0 | Input | External interrupt request input for which the valid edge (rising edge, falling edge, or both rising and falling edges) can be specified | Input port | P120/EXLVI |
| INTP1 | | | | P30 |
| INTP2 | | | | P31 |
| INTP3 | | | | P32 |
| INTP4 | | | | P33/TI51/TO51 |
| INTP5 | | | | P16/TOH1 |
| REGC | – | Connecting regulator output (2.5 V) stabilization capacitance for internal operation. Connect to V _{SS} via a capacitor (0.47 to 1 μ F). | – | – |
| $\overline{\text{RESET}}$ | Input | System reset input | – | – |
| RxD0 | Input | Serial data input to UART0 | Input port | P11/SI10 |
| RxD6 | | Serial data input to UART6 | | P14 |
| TxD0 | Output | Serial data output from UART0 | Input port | P10/ $\overline{\text{SCK10}}$ |
| TxD6 | | Serial data output from UART6 | | P13 |
| $\overline{\text{SCK10}}$ | I/O | Clock input/output for CSI10 | Input port | P10/TxD0 |
| SI10 | Input | Serial data input to CSI10 | | P11/RxD0 |
| SO10 | Output | Serial data output from CSI10 | | P12 |
| SCL0 | I/O | Clock input/output for I ² C | Input port | P60 |
| SDA0 | | Serial data I/O for I ² C | | P61 |
| TI000 | Input | External count clock input to 16-bit timer/event counter 00 Capture trigger input to capture registers (CR000, CR010) of 16-bit timer/event counter 00 | Input port | P00 |
| TI010 | Input | Capture trigger input to capture register (CR000) of 16-bit timer/event counter 00 | Input port | P01/TO00 |
| TI50 | Input | External count clock input to 8-bit timer/event counter 50 | Input port | P17/TO50 |
| TI51 | | External count clock input to 8-bit timer/event counter 51 | | P33/TO51/INTP4 |
| TO00 | Output | 16-bit timer/event counter 00 output | Input port | P01/TI010 |
| TO50 | Output | 8-bit timer/event counter 50 output | Input port | P17/TI50 |
| TO51 | | 8-bit timer/event counter 51 output | | P33/TI51/INTP4 |
| TOH0 | Output | 8-bit timer H0 output | Input port | P15 |
| TOH1 | | 8-bit timer H1 output | | P16/INTP5 |
| X1 | – | Connecting resonator for main system clock | Input port | P121 |
| X2 | – | | Input port | P122/EXCLK |
| EXCLK | Input | External clock input for main system clock | Input port | P122/X2 |
| V _{DD} | – | Positive power supply and A/D converter reference voltage input | – | – |
| V _{SS} | – | Ground potential | – | – |

(2) Non-port functions (2/2)

| Function Name | I/O | Function | After Reset | Alternate Function |
|-----------------------|-------|-----------------------------|-------------|--------------------|
| OCD0A ^{Note} | Input | On-chip debugger connection | Input port | P121/X1 |
| OCD0B ^{Note} | – | On-chip debugger connection | Input port | P122/X2/EXCLK |
| OCD1A ^{Note} | Input | On-chip debugger connection | Input port | P31/INTP2 |
| OCD1B ^{Note} | – | On-chip debugger connection | Input port | P32/INTP3 |

Note R7F0C999B2DFP only

2.2 Description of Pin Functions

2.2.1 P00, P01 (port 0)

P00 and P01 function as an I/O port. These pins also function as timer I/O.

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

(1) Port mode

P00 and P01 function as an I/O port. P00 and P01 can be set to input or output port in 1-bit units using port mode register 0 (PM0). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 0 (PU0).

(2) Control mode

P00 and P01 function as timer I/O.

(a) TI000

This is a pin for inputting an external count clock to 16-bit timer/event counter 00 and is also for inputting a capture trigger signal to the capture registers (CR000, CR010) of 16-bit timer/event counter 00.

(b) TI010

This is a pin for inputting a capture trigger signal to the capture register (CR000 or CR001) of 16-bit timer/event counters 00 and 01.

(c) TO00

This is a timer output pin of 16-bit timer/event counter 00.

2.2.2 P10 to P17 (port 1)

P10 to P17 function as an I/O port. These pins also function as pins for serial interface data I/O, clock I/O, timer I/O, and external interrupt request input.

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

(2) Control mode

P10 to P17 function as serial interface data I/O, clock I/O, timer I/O, and external interrupt request input.

(a) SI10

This is a serial data input pin of serial interface CSI10.

(b) SO10

This is a serial data output pin of serial interface CSI10.

(c) $\overline{\text{SCK10}}$

This is a serial clock I/O pin of serial interface CSI10.

(d) RxD0

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

(e) RxD6

This is a serial data input pin of serial interface UART6.

(f) TxD0

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

(g) TxD6

This is a serial data output pin of serial interface UART6.

(h) TI50

This is the pin for inputting an external count clock to 8-bit timer/event counter 50.

(i) TO50

This is a timer output pin of 8-bit timer/event counter 50.

(j) TOH0, TOH1

These are the timer output pins of 8-bit timers H0 and H1.

(k) INTP5

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.

2.2.3 P20 to P23 (port 2)

P20 to P23 function as an I/O port. These pins also function as pins for A/D converter analog input.

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

(1) Port mode

P20 to P23 function as an I/O port. P20 to P23 can be set to input or output port in 1-bit units using port mode register 2 (PM2).

(2) Control mode

P20 to P23 function as A/D converter analog input pins (ANI0 to ANI3). When using these pins as analog input pins, see (5) ANI0/P20 to ANI3/P23 in 11.6 Cautions for A/D Converter.

Caution ANI0/P20 to ANI3/P23 are set in the analog input mode after release of reset.

2.2.4 P30 to P33 (port 3)

P30 to P33 function as an I/O port. These pins also function as pins for external interrupt request input, timer I/O, and on-chip debug I/O.

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

(1) Port mode

P30 to P33 function as an I/O port. P30 to P33 can be set to input or output port in 1-bit units using port mode register 3 (PM3). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 3 (PU3).

(2) Control mode

P30 to P33 function as external interrupt request input, timer I/O, and on-chip debug I/O.

(a) INTP2 to INTP4

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) TI51

This is an external count clock input pin to 8-bit timer/event counter 51.

(c) TO51

This is a timer output pin from 8-bit timer/event counter 51.

(d) INTP1

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.

(e) OCD1A, OCD1B (R7F0C999B2DFP only)

These are the on-chip debug connection pins.

2.2.5 P40 and P41 (port 4)

P40 and P41 function as an I/O port. P40 and P41 can be set to input or output port in 1-bit units using port mode register 4 (PM4). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 4 (PU4).

2.2.6 P60 and P61 (port 6)

P60 and P61 function as an I/O port. These pins also function as pins for serial interface data I/O and clock I/O. The following operation modes can be specified in 1-bit units.

(1) Port mode

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

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

(2) Control mode

P60 and P61 function as serial interface data I/O and clock I/O.

(a) SDA0

This is a serial data I/O pin for serial interface IIC0.

(b) SCL0

This is a serial clock I/O pin for serial interface IIC0.

2.2.7 P70 and P71 (port 7)

P70 and P71 function as an I/O port. P70 and P71 can be set to input or output port in 1-bit units using port mode register 7 (PM7). Use of an on-chip pull-up resistor can be specified by pull-up resistor option register 7 (PU7).

2.2.8 P120 to P122 (port 12)

P120 to P122 function as an I/O port. These pins also function as pins for external interrupt request input, potential input for external low-voltage detection, connecting resonator for main system clock, external clock input for main system clock, and connecting on-chip debugger.

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

(1) Port mode

P120 to P122 function as an I/O port. P120 to P122 can be set to input or output port using port mode register 12 (PM12). Only for P120, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

(2) Control mode

P120 to P122 function as pins for external interrupt request input, potential input for external low-voltage detection, connecting resonator for main system clock, external clock input for main system clock, and connecting on-chip debugger.

(a) INTPO

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

(b) EXLVI

This is a potential input pin for external low-voltage detection.

(c) X1, X2

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

(d) EXCLK

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

(e) OCD0A, OCD0B (R7F0C999B2DFP only)

These are the on-chip debug connection pins.

2.2.9 V_{DD}, V_{SS}

These are the power supply/ground pins.

(a) V_{DD}

V_{DD} is a positive power supply pin.

(b) V_{SS}

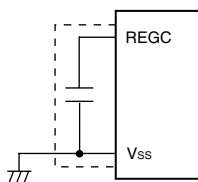
V_{SS} is a ground potential pin.

2.2.10 $\overline{\text{RESET}}$

This is the active-low system reset input pin.

2.2.11 REGC

This is the pin for connecting regulator output (2.5 V) stabilization capacitance for internal operation. Connect this pin to V_{SS} via a capacitor (0.47 to 1 μF).



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

2.2.12 FLMD0

This is a pin for setting flash memory programming mode.

Connect FLMD0 to V_{SS} in the normal operation mode.

In flash memory programming mode, connect this pin to the flash memory programmer.

2.3 Pin I/O Circuits and Recommended Connection of Unused Pins

Table 2-3 shows the types of pin I/O circuits and the recommended connections of unused pins.
See **Figure 2-1** for the configuration of the I/O circuit of each type.

Table 2-3. Pin I/O Circuit Types (1/2)

| Pin Name | I/O Circuit Type | I/O | Recommended Connection of Unused Pins |
|--------------------------------------|------------------|-----|---|
| P00/TI000 | 5-AQ | I/O | Input: Independently connect to EV_{DD} or EV_{SS} via a resistor. Output: Leave open. |
| P01/TI010/TO00 | | | |
| P10/ $\overline{SCK10}$ /TxD0 | 5-AQ | | |
| P11/SI10/RxD0 | | | |
| P12/SO10 | 5-AG | | |
| P13/TxD6 | | | |
| P14/RxD6 | 5-AQ | | |
| P15/TOH0 | 5-AG | | |
| P16/TOH1/INTP5 | 5-AQ | | |
| P17/TI50/TO50 | | | |
| ANI0/P20 to ANI3/P23 ^{Note} | 11-G | | |

Note ANI0/P20 to ANI3/P23 are set in the analog input mode after release of reset.

Table 2-3. Pin I/O Circuit Types (2/2)

| Pin Name | I/O Circuit Type | I/O | Recommended Connection of Unused Pins |
|---------------------------------|------------------|-------|---|
| P30/INTP1 | 5-AQ | I/O | Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open. |
| P31/INTP2 | | | |
| P32/INTP3 | | | |
| P33/TI51/TO51/INTP4 | | | |
| P40, P41 | 5-AG | I/O | Input: Independently connect to EV _{DD} or EV _{SS} via a resistor, or connect directly to EV _{SS} . Output: Leave this pin open at low-level output after clearing the output latch of the port to 0. |
| P60/SCL0 | 13-AI | | |
| P61/SDA0 | | | |
| P70, P71 | 5-AQ | I/O | Input: Independently connect to EV _{DD} or EV _{SS} via a resistor. Output: Leave open. |
| P120/INTP0/EXLVI | | | |
| P121/X1 ^{Note 1} | 37 | | |
| P122/X2/EXCLK ^{Note 1} | | | |
| FLMD0 | 38-A | – | Connect to EV _{SS} or V _{SS} ^{Note 2} . |
| RESET | 2 | Input | Connect directly to EV _{DD} or via a resistor. |
| REGC | – | – | Connect to V _{SS} via capacitor (0.47 to 1 μ F). |

- Notes**
1. Use recommended connection above in I/O port mode (see **Figure 5-2 Format of Clock Operation Mode Select Register (OSCCTL)**) when these pins are not used.
 2. FLMD0 is a pin that is used to write data to the flash memory. To rewrite the data of the flash memory on-board, connect this pin to V_{SS} via a resistor (10 k Ω : recommended).

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

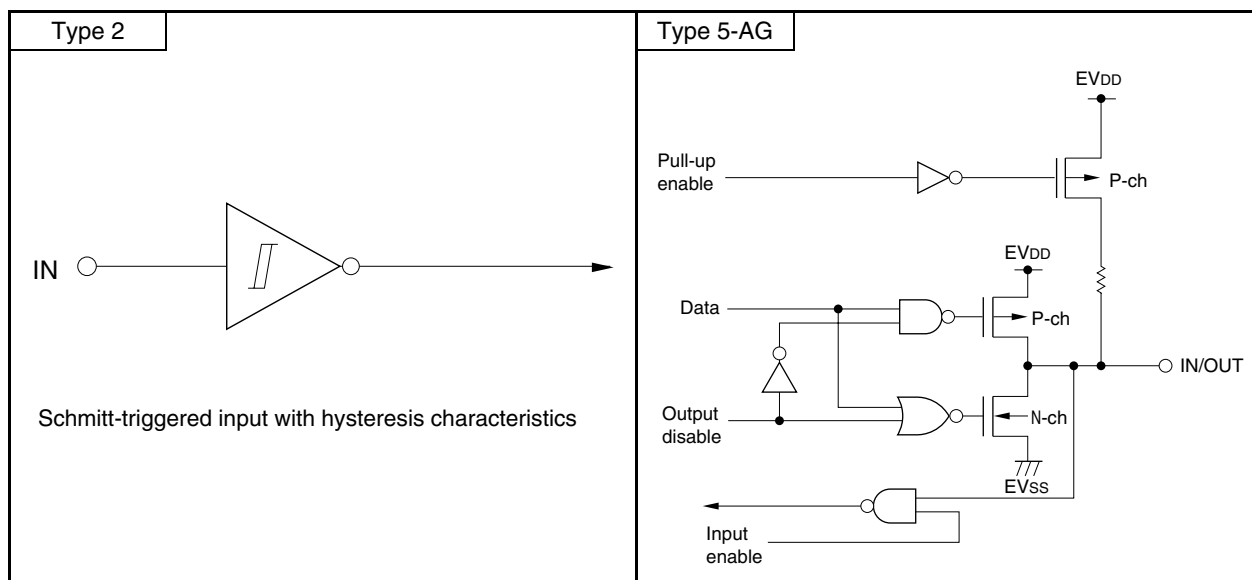
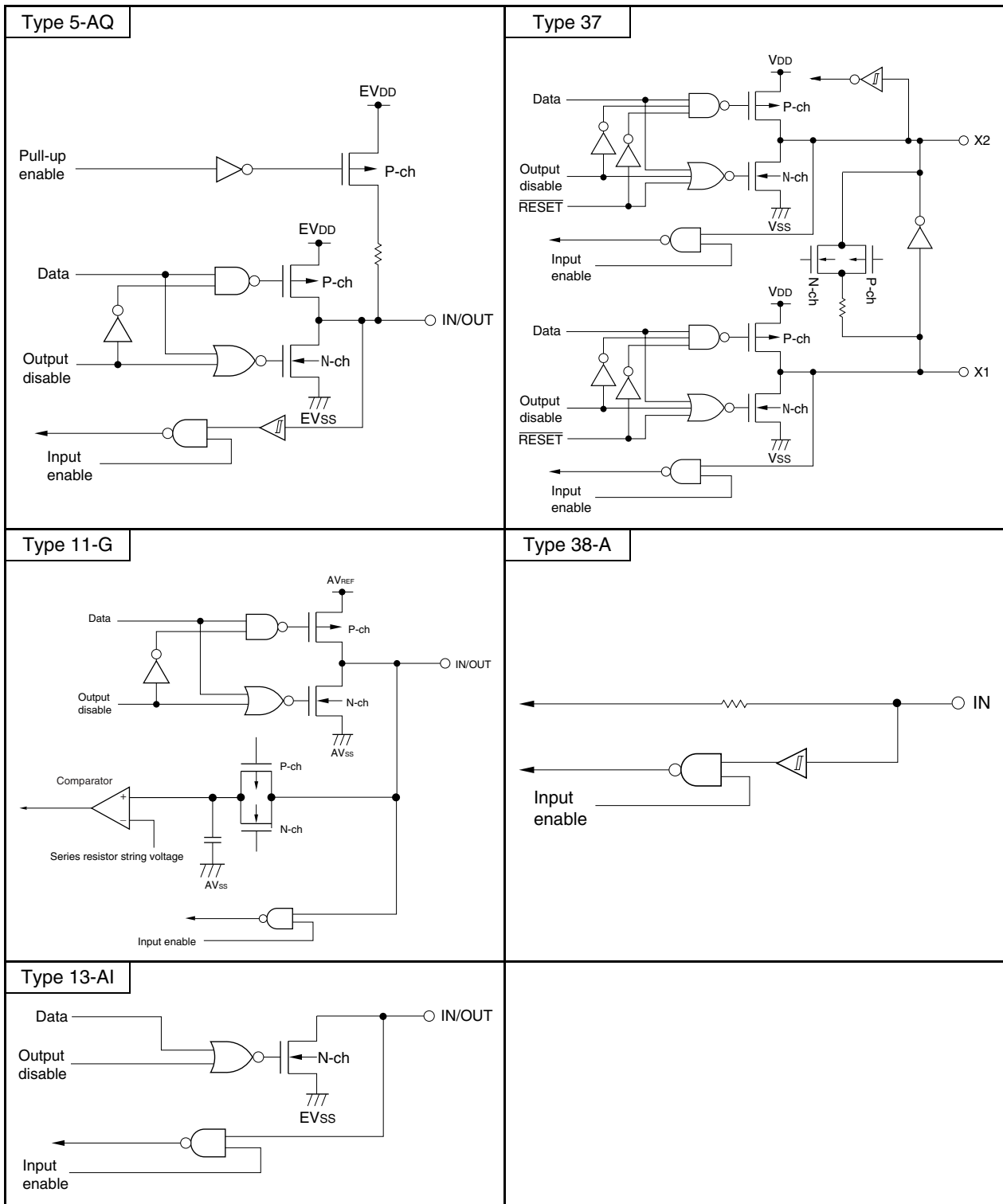


Figure 2-1. Pin I/O Circuit List (2/2)



CHAPTER 3 CPU ARCHITECTURE

3.1 Memory Space

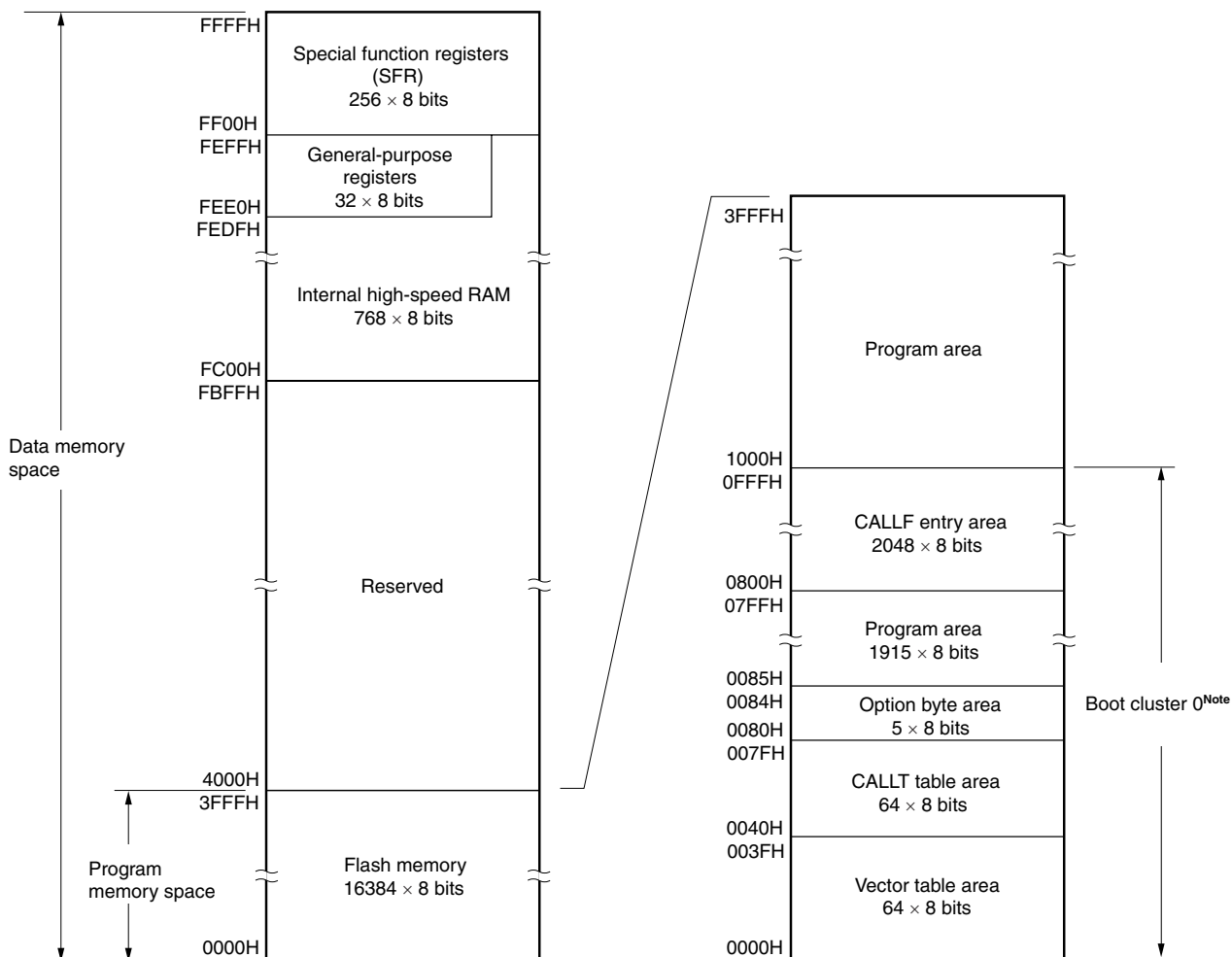
Products in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B can access a 64 KB memory space. Figures 3-1 to 3-3 show the memory maps.

Cautions 1. Regardless of the internal memory capacity, the initial values of the internal memory size switching register (IMS) of all products in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B are fixed (IMS = CFH). Therefore, set the value corresponding to each product as indicated below.

Table 3-1. Set Values of Internal Memory Size Switching Register (IMS)

| Product | IMS | ROM Capacity |
|-----------|-----|--------------|
| R7F0C011B | 04H | 16 KB |
| R7F0C012B | C6H | 24 KB |
| R7F0C013B | C8H | 32 KB |
| R7F0C999B | C8H | 32 KB |

Figure 3-1. Memory Map (R7F0C011B)



Note Writing boot cluster 0 can be prohibited depending on the setting of security (see 22.7 Security Settings).

Remark The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see Table 3-2 Correspondence Between Address Values and Block Numbers in Flash Memory.

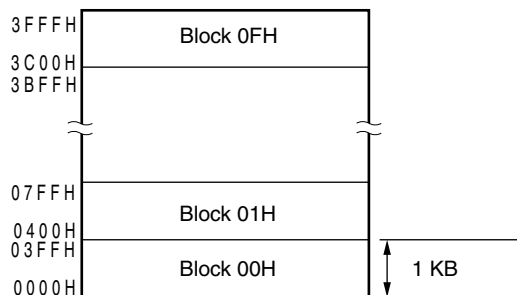
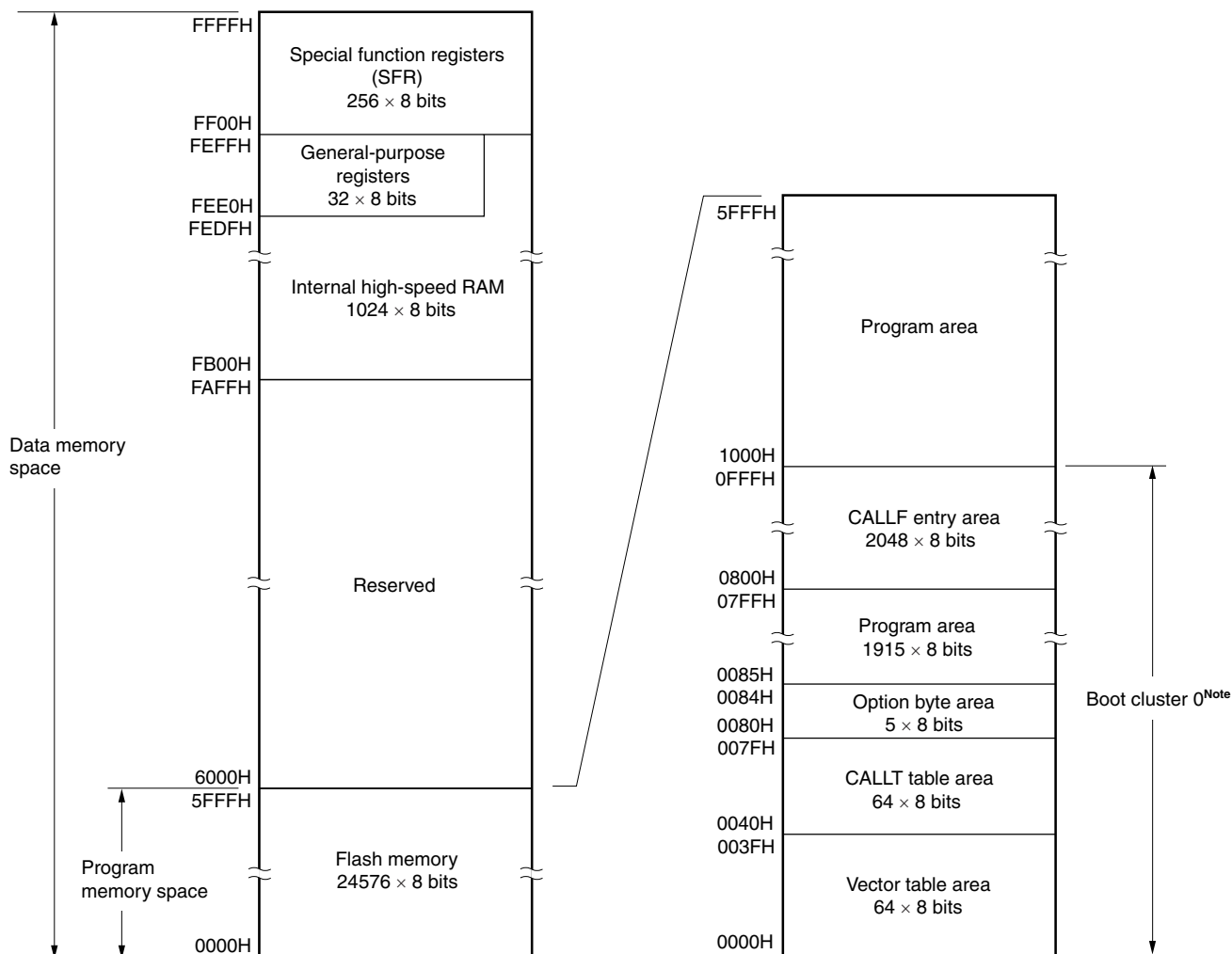


Figure 3-2. Memory Map (R7F0C012B)



Note Writing boot cluster 0 can be prohibited depending on the setting of security (see 22.7 Security Settings).

Remark The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see Table 3-2 Correspondence Between Address Values and Block Numbers in Flash Memory.

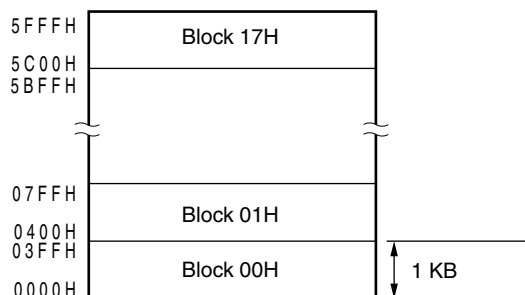
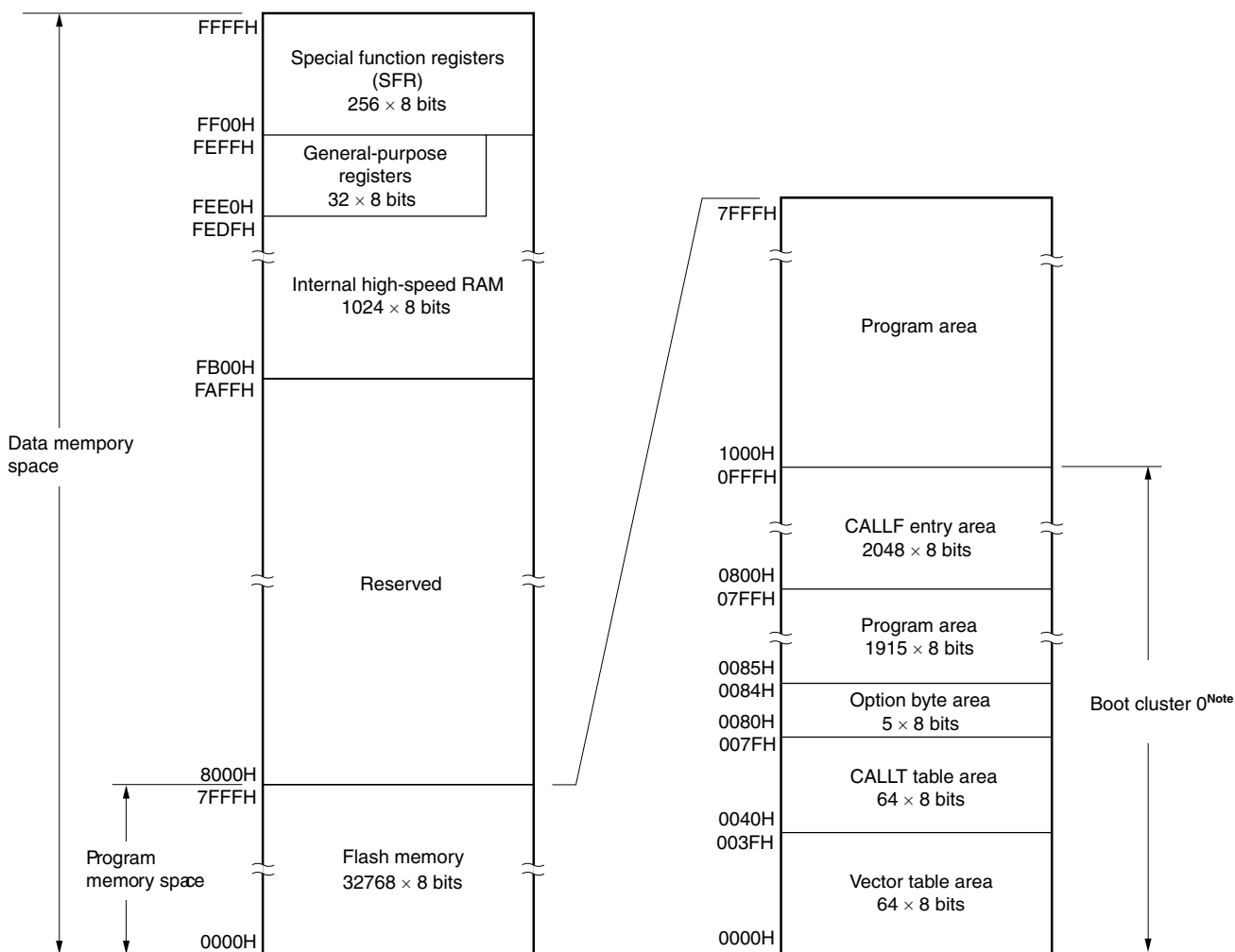
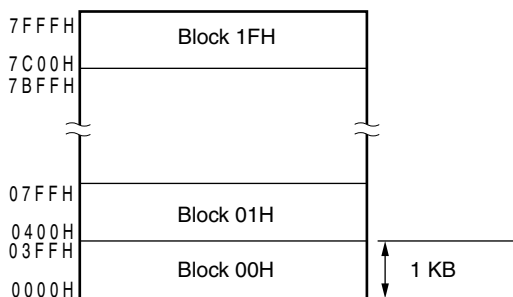


Figure 3-3. Memory Map (R7F0C013B, R7F0C999B)



Note Writing boot cluster 0 can be prohibited depending on the setting of security (see 22.7 Security Settings).

Remark The flash memory is divided into blocks (one block = 1 KB). For the address values and block numbers, see **Table 3-2 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 3-2. Correspondence Between Address Values and Block Numbers in Flash Memory

| Address Value | Block Number | Address Value | Block Number |
|----------------|--------------|----------------|--------------|
| 0000H to 03FFH | 00H | 4000H to 43FFH | 10H |
| 0400H to 07FFH | 01H | 4400H to 47FFH | 11H |
| 0800H to 0BFFH | 02H | 4800H to 4BFFH | 12H |
| 0C00H to 0FFFH | 03H | 4C00H to 4FFFH | 13H |
| 1000H to 13FFH | 04H | 5000H to 53FFH | 14H |
| 1400H to 17FFH | 05H | 5400H to 57FFH | 15H |
| 1800H to 1BFFH | 06H | 5800H to 5BFFH | 16H |
| 1C00H to 1FFFH | 07H | 5C00H to 5FFFH | 17H |
| 2000H to 23FFH | 08H | 6000H to 63FFH | 18H |
| 2400H to 27FFH | 09H | 6400H to 67FFH | 19H |
| 2800H to 2BFFH | 0AH | 6800H to 6BFFH | 1AH |
| 2C00H to 2FFFH | 0BH | 6C00H to 6FFFH | 1BH |
| 3000H to 33FFH | 0CH | 7000H to 73FFH | 1CH |
| 3400H to 37FFH | 0DH | 7400H to 77FFH | 1DH |
| 3800H to 3BFFH | 0EH | 7800H to 7BFFH | 1EH |
| 3C00H to 3FFFH | 0FH | 7C00H to 7FFFH | 1FH |

Remark R7F0C011B: Block numbers 00H to 07H
R7F0C012B: Block numbers 00H to 17H
R7F0C013B: Block numbers 00H to 1FH
R7F0C999B: Block numbers 00H to 1FH

3.1.1 Internal program memory space

The internal program memory space stores the program and table data. Normally, it is addressed with the program counter (PC).

R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B incorporate internal ROM (flash memory), as shown below.

Table 3-3. Internal ROM Capacity

| Product | Internal ROM (Flash Memory) |
|-----------|---------------------------------|
| R7F0C011B | 16384 × 8 bits (0000H to 3FFFH) |
| R7F0C012B | 24576 × 8 bits (0000H to 5FFFH) |
| R7F0C013B | 32768 × 8 bits (0000H to 7FFFH) |
| R7F0C999B | 32768 × 8 bits (0000H to 7FFFH) |

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

(1) Vector table area

The 64-byte area 0000H to 003FH 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.

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 3-4. Vector Table

| Vector Table Area | Interrupt Factors |
|-------------------|----------------------------|
| 0000H | RESET input, POC, LVI, WDT |
| 0004H | INTLVI |
| 0006H | INTP0 |
| 0008H | INTP1 |
| 000AH | INTP2 |
| 000CH | INTP3 |
| 000EH | INTP4 |
| 0010H | INTP5 |
| 0012H | INTSRE6 |
| 0014H | INTSR6 |
| 0016H | INTST6 |
| 0018H | INTCSI10/INTST0 |
| 001AH | INTTMH1 |
| 001CH | INTTMH0 |
| 001EH | INTTM50 |
| 0020H | INTTM000 |
| 0022H | INTTM010 |
| 0024H | INTAD |
| 0026H | INTSR0 |
| 0028H | INTWTI |
| 002AH | INTTM51 |
| 002EH | INTWT |
| 0034H | INTIIC0 |
| 003EH | BRK |

(2) CALLT instruction table area

The 64-byte area 0040H to 007FH can store the subroutine entry address of a 1-byte call instruction (CALLT).

(3) Option byte area

A 5-byte area of 0080H to 0084H can be used as an option byte area. Set the option byte at 0080H to 0084H. For details, see **CHAPTER 21 OPTION BYTE**.

(4) CALLF instruction entry area

The area 0800H to 0FFFH can perform a direct subroutine call with a 2-byte call instruction (CALLF).

3.1.2 Internal data memory space

R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B incorporate the following RAMs.

(1) Internal high-speed RAM

The 32-byte area FEE0H to FEFFH is assigned to four general-purpose register banks consisting of eight 8-bit registers per bank.

This area cannot be used as a program area in which instructions are written and executed.

The internal high-speed RAM can also be used as a stack memory.

Table 3-5. Internal High-Speed RAM Capacity

| Product | Internal High-Speed RAM |
|-----------|--------------------------------|
| R7F0C011B | 768 × 8 bits (FC00H to FEFFH) |
| R7F0C012B | 1024 × 8 bits (FB00H to FEFFH) |
| R7F0C013B | |
| R7F0C999B | |

3.1.3 Special function register (SFR) area

On-chip peripheral hardware special function registers (SFRs) are allocated in the area FF00H to FFFFH (see **Table 3-6 Special Function Register List** in **3.2.3 Special function registers (SFRs)**).

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

3.1.4 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 78K0/Kx2 microcontroller, based on operability and other considerations. For areas containing data memory in particular, special addressing methods designed for the functions of special function registers (SFR) and general-purpose registers are available for use. Figures 3-4 to 3-6 show correspondence between data memory and addressing. For details of each addressing mode, see **3.4 Operand Address Addressing**.

Figure 3-4. Correspondence Between Data Memory and Addressing (R7F0C011B)

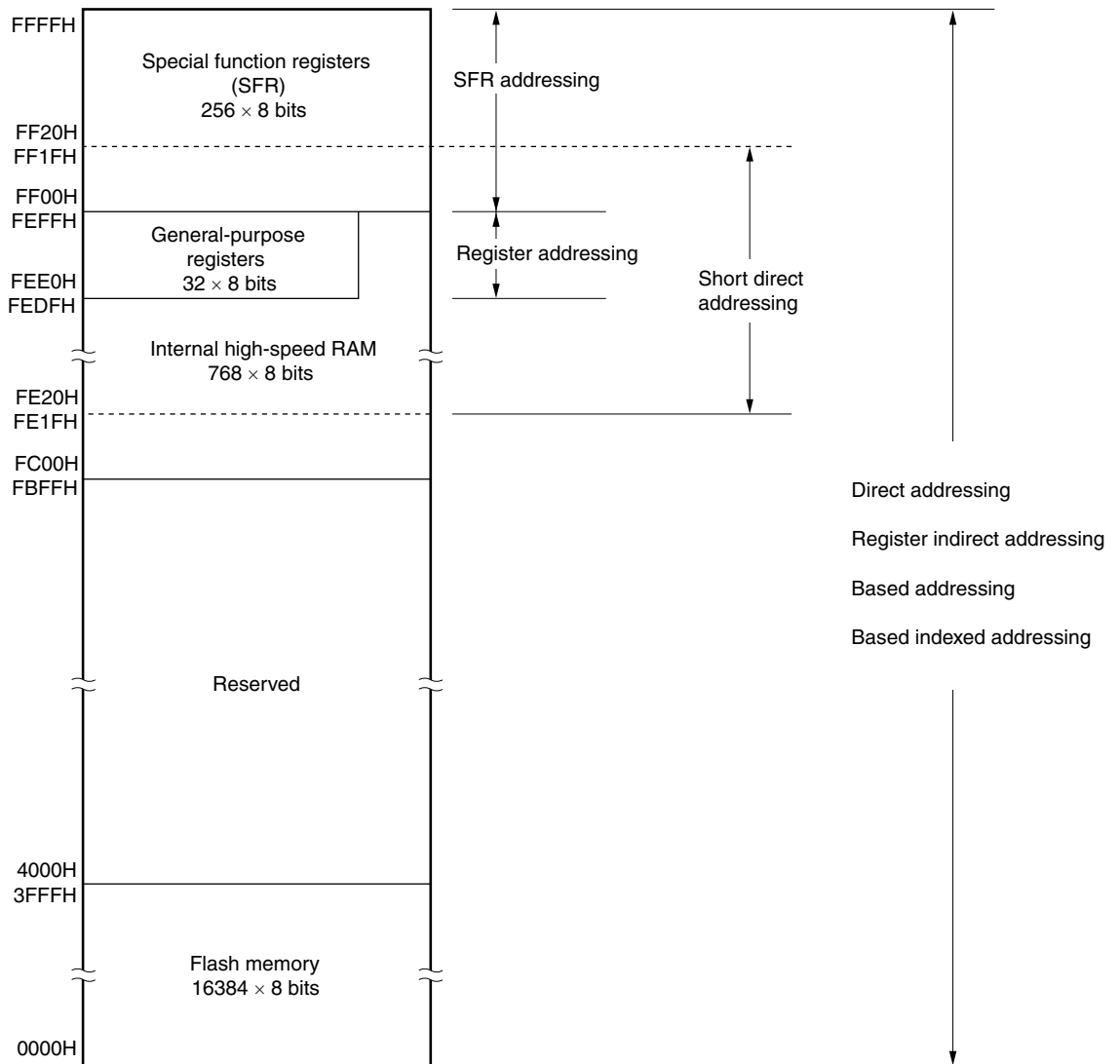


Figure 3-5. Correspondence Between Data Memory and Addressing (R7F0C012B)

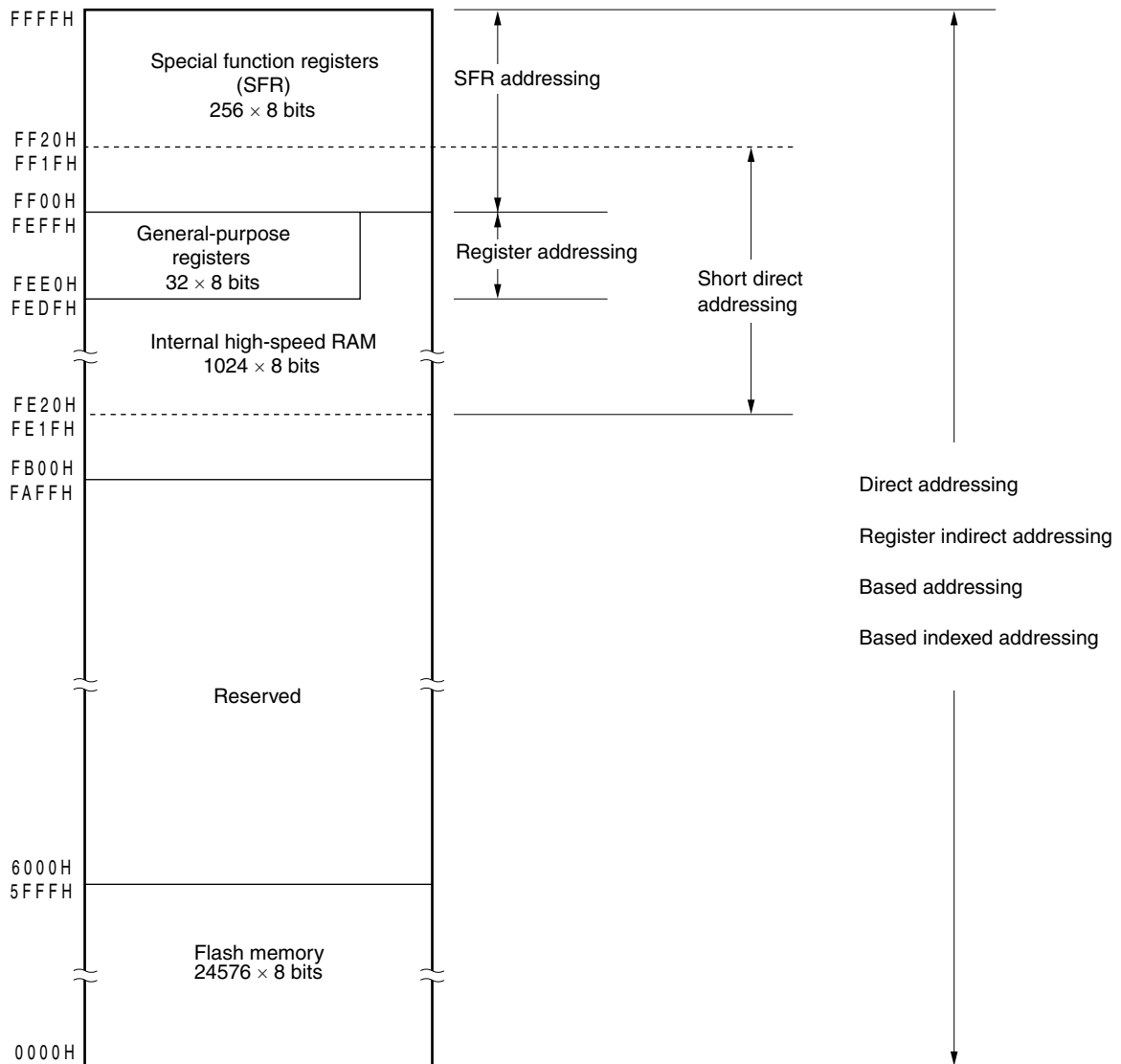
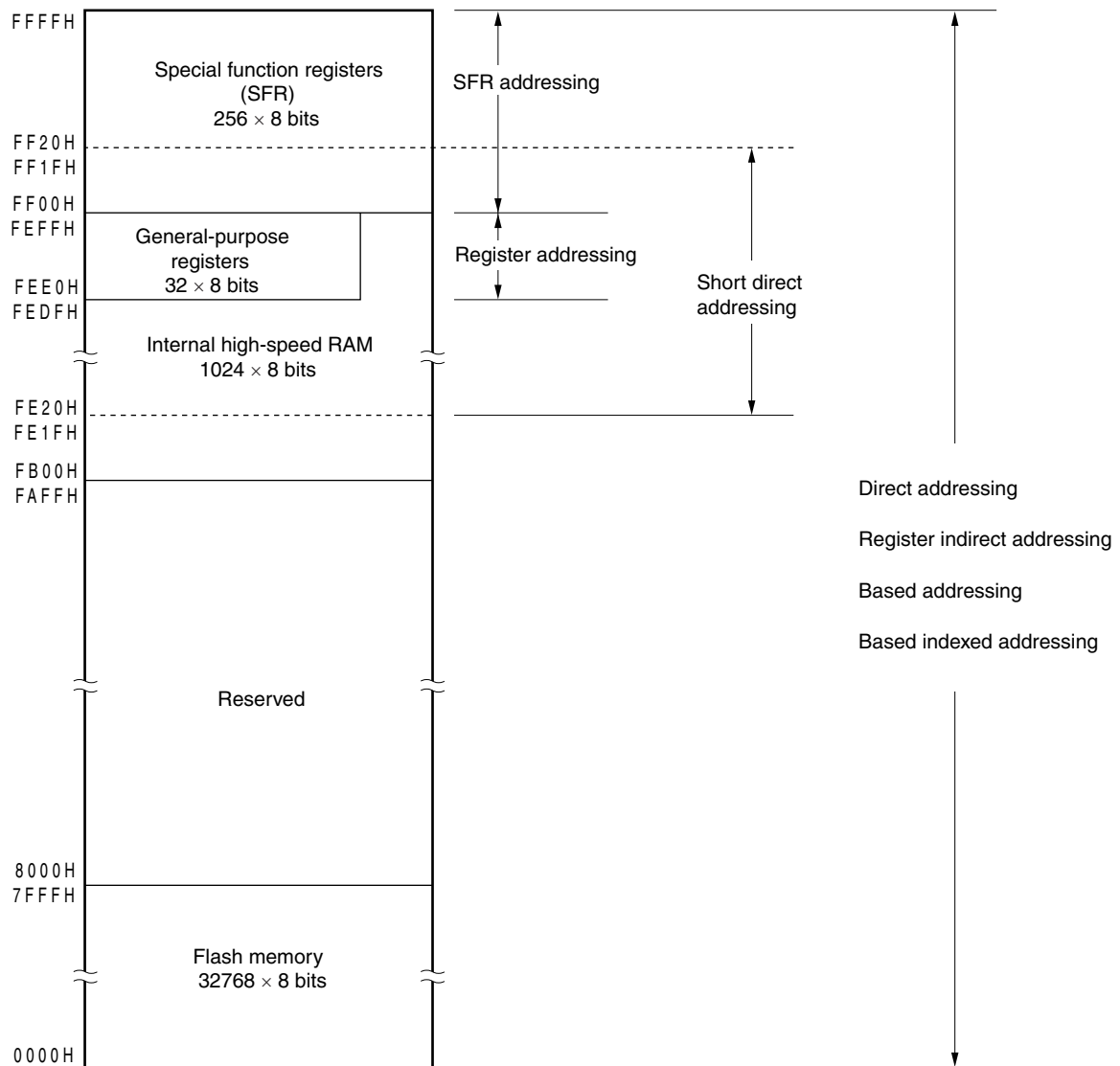


Figure 3-6. Correspondence Between Data Memory and Addressing (R7F0C013B)



3.2 Processor Registers

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B incorporate the following processor registers.

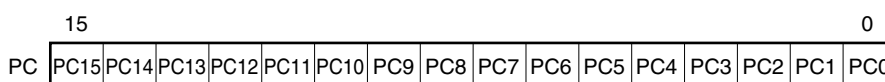
3.2.1 Control registers

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 16-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 program counter.

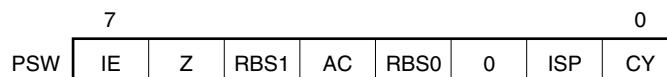
Figure 3-7. Format of Program Counter



(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 acknowledgement or PUSH PSW instruction execution and are restored upon execution of the RETB, RETI and POP PSW instructions. Reset signal generation sets PSW to 02H.

Figure 3-8. 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 interrupt request acknowledgment is controlled with an in-service priority flag (ISP), 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 result is zero, this flag is set (1). It is reset (0) in all other cases.

(c) Register bank select flags (RBS0 and 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 flag (ISP)

This flag manages the priority of acknowledgeable maskable vectored interrupts. When this flag is 0, low-level vectored interrupt requests specified by a priority specification flag register (PR0L, PR0H, PR1L, PR1H) (see **16.3 (3) Priority specification flag registers (PR0L, PR0H, PR1L, PR1H)**) can not be acknowledged. Actual request acknowledgment is controlled by the interrupt enable flag (IE).

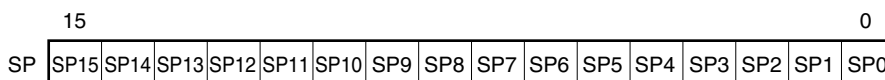
(f) Carry flag (CY)

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 high-speed RAM area can be set as the stack area.

Figure 3-9. Format of Stack Pointer



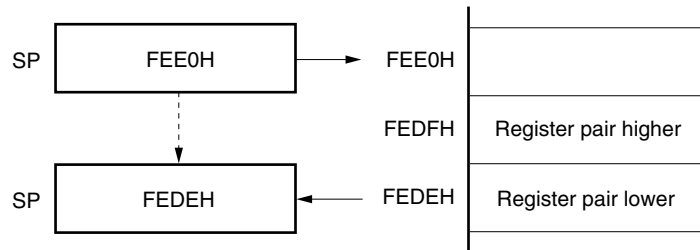
The SP is decremented ahead of write (save) to the stack memory and is incremented after read (restored) from the stack memory.

Each stack operation saves/restores data as shown in Figures 3-10 and 3-11.

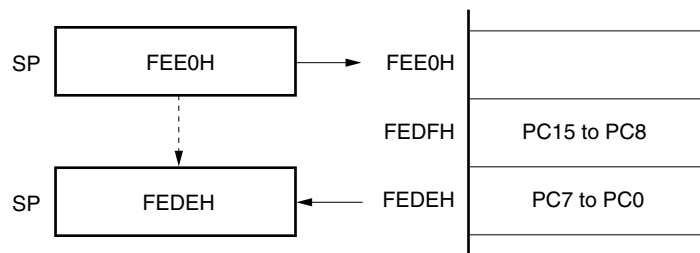
Caution Since reset signal generation makes the SP contents undefined, be sure to initialize the SP before using the stack.

Figure 3-10. Data to Be Saved to Stack Memory

(a) PUSH rp instruction (when SP = FEE0H)



(b) CALL, CALLF, CALLT instructions (when SP = FEE0H)



(c) Interrupt, BRK instructions (when SP = FEE0H)

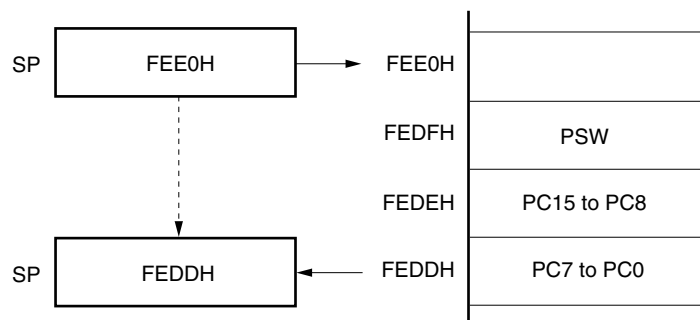
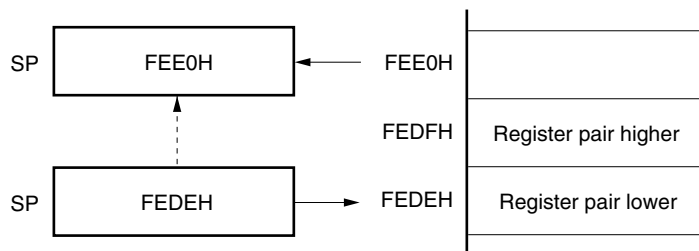
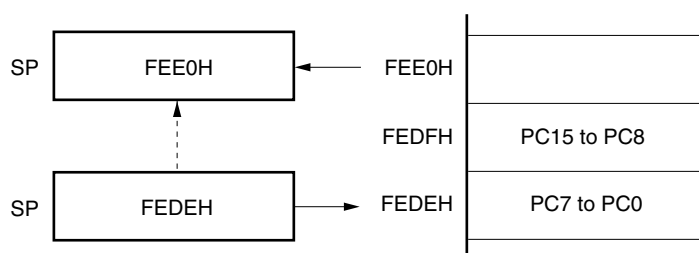


Figure 3-11. Data to Be Restored from Stack Memory

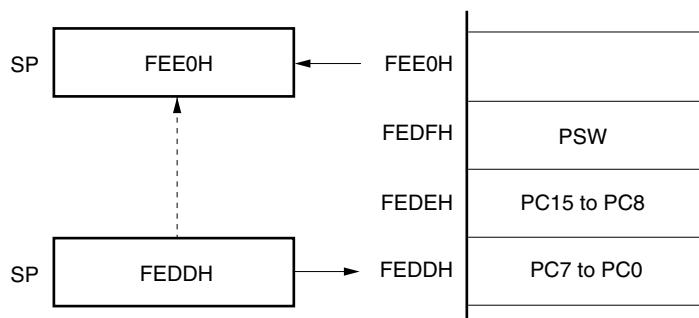
(a) POP rp instruction (when SP = FEDEH)



(b) RET instruction (when SP = FEDEH)



(c) RETI, RETB instructions (when SP = FEDDH)



3.2.2 General-purpose registers

General-purpose registers are mapped at particular addresses (FEE0H to FEFFH) 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).

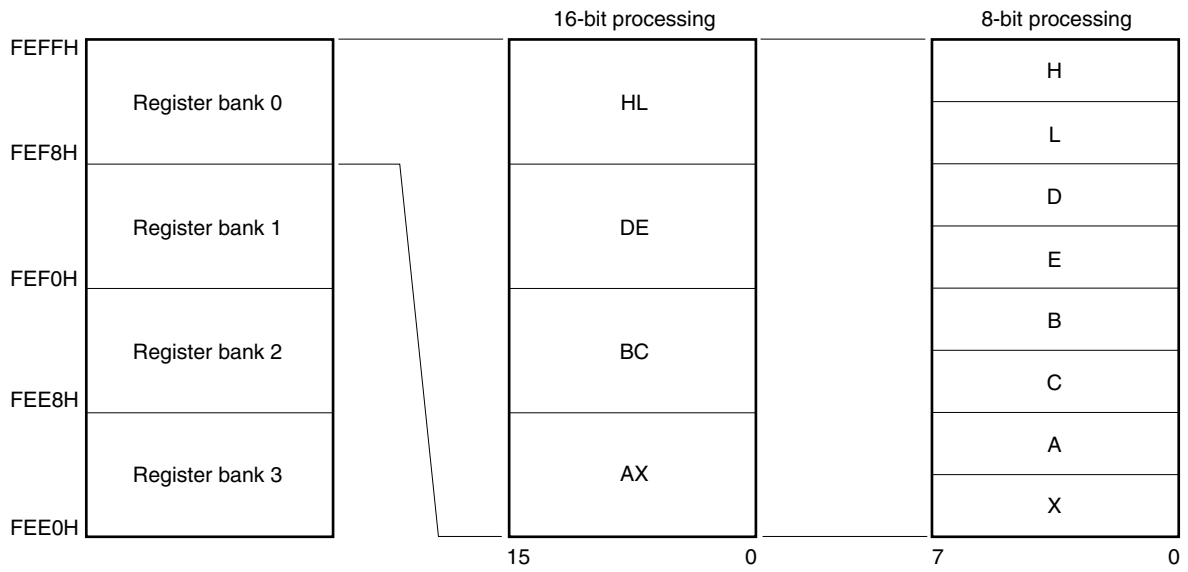
Each register can be used as an 8-bit register, and two 8-bit registers can also be used in a pair as a 16-bit register (AX, BC, DE, and HL).

These registers can be described in terms of function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL) and absolute names (R0 to R7 and RP0 to RP3).

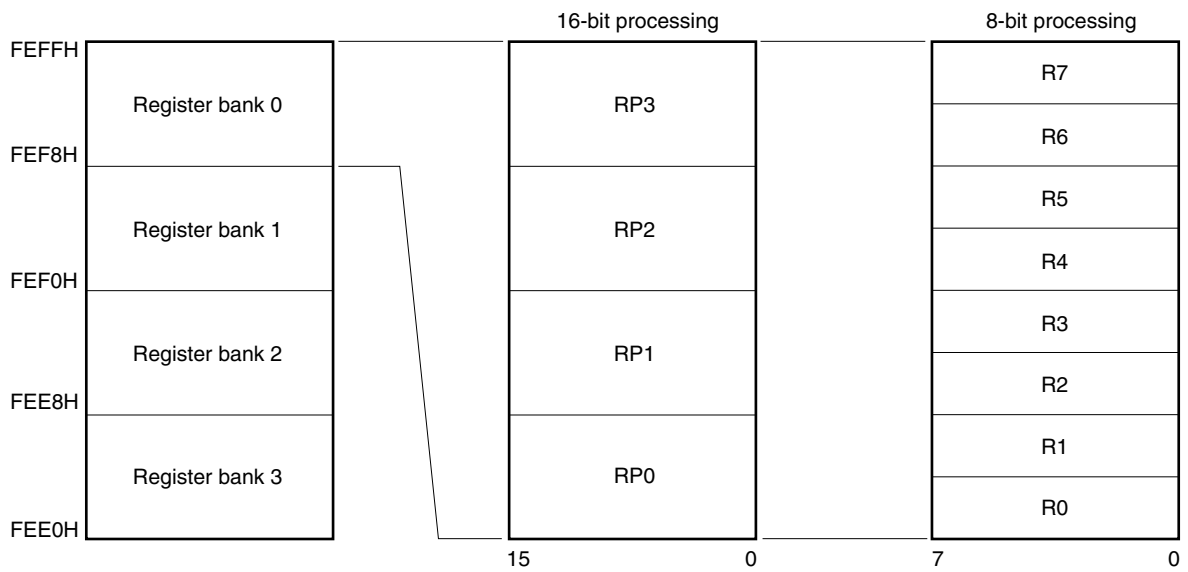
Register banks to be used for instruction execution are set by the CPU control instruction (SEL RBn). Because of the 4-register bank configuration, an efficient program can be created by switching between a register for normal processing and a register for interrupts for each bank.

Figure 3-12. Configuration of General-Purpose Registers

(a) Function name



(b) Absolute name



3.2.3 Special function registers (SFRs)

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

SFRs are allocated to the FF00H to FFFFH area.

Special function registers can be manipulated like general-purpose registers, using operation, transfer, and bit manipulation instructions. The manipulatable bit units, 1, 8, and 16, depend on the special function register type.

Each manipulation bit unit can be specified as follows.

- 1-bit manipulation
Describe the symbol reserved by the assembler for the 1-bit manipulation instruction operand (sfr.bit).
This manipulation can also be specified with an address.
- 8-bit manipulation
Describe the symbol reserved 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 reserved by the assembler for the 16-bit manipulation instruction operand (sfrp).
When specifying an address, describe an even address.

Table 3-6 gives a list of the special function registers. 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 RA78K0, and is defined as an sfr variable using the #pragma sfr directive in the CC78K0. When using the RA78K0 and ID78K0-QB, symbols can be written as an instruction operand.
- R/W
Indicates whether the corresponding special function register can be read or written.
R/W: Read/write enable
R: Read only
W: Write only
- Manipulatable bit units
Indicates the manipulatable 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.

Table 3-6. Special Function Register List (1/4)

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulatable Bit Unit | | | After Reset |
|---------|---|--------|-----|------------------------|--------|---------|-------------|
| | | | | 1 Bit | 8 Bits | 16 Bits | |
| FF00H | Port register 0 | P0 | R/W | √ | √ | – | 00H |
| FF01H | Port register 1 | P1 | R/W | √ | √ | – | 00H |
| FF02H | Port register 2 | P2 | R/W | √ | √ | – | 00H |
| FF03H | Port register 3 | P3 | R/W | √ | √ | – | 00H |
| FF04H | Port register 4 | P4 | R/W | √ | √ | – | 00H |
| FF06H | Port register 6 | P6 | R/W | √ | √ | – | 00H |
| FF07H | Port register 7 | P7 | R/W | √ | √ | – | 00H |
| FF08H | 10-bit A/D conversion result register | ADCR | R | – | – | √ | 0000H |
| FF09H | 8-bit A/D conversion result register | ADCRH | R | – | √ | – | 00H |
| FF0AH | Receive buffer register 6 | RXB6 | R | – | √ | – | FFH |
| FF0BH | Transmit buffer register 6 | TXB6 | R/W | – | √ | – | FFH |
| FF0CH | Port register 12 | P12 | R/W | √ | √ | – | 00H |
| FF0FH | Serial I/O shift register 10 | SIO10 | R | – | √ | – | 00H |
| FF10H | 16-bit timer counter 00 | TM00 | R | – | – | √ | 0000H |
| FF11H | | | | | | | |
| FF12H | 16-bit timer capture/compare register 000 | CR000 | R/W | – | – | √ | 0000H |
| FF13H | | | | | | | |
| FF14H | 16-bit timer capture/compare register 010 | CR010 | R/W | – | – | √ | 0000H |
| FF15H | | | | | | | |
| FF16H | 8-bit timer counter 50 | TM50 | R | – | √ | – | 00H |
| FF17H | 8-bit timer compare register 50 | CR50 | R/W | – | √ | – | 00H |
| FF18H | 8-bit timer H compare register 00 | CMP00 | R/W | – | √ | – | 00H |
| FF19H | 8-bit timer H compare register 10 | CMP10 | R/W | – | √ | – | 00H |
| FF1AH | 8-bit timer H compare register 01 | CMP01 | R/W | – | √ | – | 00H |
| FF1BH | 8-bit timer H compare register 11 | CMP11 | R/W | – | √ | – | 00H |
| FF1FH | 8-bit timer counter 51 | TM51 | R | – | √ | – | 00H |
| FF20H | Port mode register 0 | PM0 | R/W | √ | √ | – | FFH |
| FF21H | Port mode register 1 | PM1 | R/W | √ | √ | – | FFH |
| FF22H | Port mode register 2 | PM2 | R/W | √ | √ | – | FFH |
| FF23H | Port mode register 3 | PM3 | R/W | √ | √ | – | FFH |
| FF24H | Port mode register 4 | PM4 | R/W | √ | √ | – | FFH |
| FF26H | Port mode register 6 | PM6 | R/W | √ | √ | – | FFH |
| FF27H | Port mode register 7 | PM7 | R/W | √ | √ | – | FFH |
| FF28H | A/D converter mode register | ADM | R/W | √ | √ | – | 00H |
| FF29H | Analog input channel specification register | ADS | R/W | √ | √ | – | 00H |
| FF2CH | Port mode register 12 | PM12 | R/W | √ | √ | – | FFH |
| FF2EH | Port mode register 14 | PM14 | R/W | √ | √ | – | FFH |
| FF2FH | A/D port configuration register | ADPC | R/W | √ | √ | – | 00H |
| FF30H | Pull-up resistor option register 0 | PU0 | R/W | √ | √ | – | 00H |

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

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulatable Bit Unit | | | After Reset |
|---------|---|--------|-----|------------------------|--------|---------|-------------|
| | | | | 1 Bit | 8 Bits | 16 Bits | |
| FF31H | Pull-up resistor option register 1 | PU1 | R/W | √ | √ | – | 00H |
| FF33H | Pull-up resistor option register 3 | PU3 | R/W | √ | √ | – | 00H |
| FF34H | Pull-up resistor option register 4 | PU4 | R/W | √ | √ | – | 00H |
| FF37H | Pull-up resistor option register 7 | PU7 | R/W | √ | √ | – | 00H |
| FF3CH | Pull-up resistor option register 12 | PU12 | R/W | √ | √ | – | 00H |
| FF41H | 8-bit timer compare register 51 | CR51 | R/W | – | √ | – | 00H |
| FF43H | 8-bit timer mode control register 51 | TMC51 | R/W | √ | √ | – | 00H |
| FF48H | External interrupt rising edge enable register | EGP | R/W | √ | √ | – | 00H |
| FF49H | External interrupt falling edge enable register | EGN | R/W | √ | √ | – | 00H |
| FF4FH | Input switch control register | ISC | R/W | √ | √ | – | 00H |
| FF50H | Asynchronous serial interface operation mode register 6 | ASIM6 | R/W | √ | √ | – | 01H |
| FF53H | Asynchronous serial interface reception error status register 6 | ASIS6 | R | – | √ | – | 00H |
| FF55H | Asynchronous serial interface transmission status register 6 | ASIF6 | R | – | √ | – | 00H |
| FF56H | Clock selection register 6 | CKSR6 | R/W | – | √ | – | 00H |
| FF57H | Baud rate generator control register 6 | BRGC6 | R/W | – | √ | – | FFH |
| FF58H | Asynchronous serial interface control register 6 | ASICL6 | R/W | √ | √ | – | 16H |
| FF69H | 8-bit timer H mode register 0 | TMHMD0 | R/W | √ | √ | – | 00H |
| FF6AH | Timer clock selection register 50 | TCL50 | R/W | √ | √ | – | 00H |
| FF6BH | 8-bit timer mode control register 50 | TMC50 | R/W | √ | √ | – | 00H |
| FF6CH | 8-bit timer H mode register 1 | TMHMD1 | R/W | √ | √ | – | 00H |
| FF6DH | 8-bit timer H carrier control register 1 | TMCYC1 | R/W | √ | √ | – | 00H |
| FF6FH | Watch timer operation mode register | WTM | R/W | √ | √ | – | 00H |

Table 3-6. Special Function Register List (3/4)

| Address | Special Function Register (SFR) Name | Symbol | R/W | Manipulatable Bit Unit | | | After Reset |
|---------|---|--------|-----|------------------------|--------|---------|-------------------------------|
| | | | | 1 Bit | 8 Bits | 16 Bits | |
| FF70H | Asynchronous serial interface operation mode register 0 | ASIM0 | R/W | √ | √ | – | 01H |
| FF71H | Baud rate generator control register 0 | BRGC0 | R/W | – | √ | – | 1FH |
| FF72H | Receive buffer register 0 | RXB0 | R | – | √ | – | FFH |
| FF73H | Asynchronous serial interface reception error status register 0 | ASIS0 | R | – | √ | – | 00H |
| FF74H | Transmit shift register 0 | TXS0 | W | – | √ | – | FFH |
| FF80H | Serial operation mode register 10 | CSIM10 | R/W | √ | √ | – | 00H |
| FF81H | Serial clock selection register 10 | CSIC10 | R/W | √ | √ | – | 00H |
| FF84H | Transmit buffer register 10 | SOTB10 | R/W | – | √ | – | 00H |
| FF8CH | Timer clock selection register 51 | TCL51 | R/W | √ | √ | – | 00H |
| FF99H | Watchdog timer enable register | WDTE | R/W | – | √ | – | 1AH/ 9AH ^{Note 1} |
| FF9FH | Clock operation mode select register | OSCCTL | R/W | √ | √ | – | 00H |
| FFA0H | Internal oscillation mode register | RCM | R/W | √ | √ | – | 80H ^{Note 2} |
| FFA1H | Main clock mode register | MCM | R/W | √ | √ | – | 00H |
| FFA2H | Main OSC control register | MOC | R/W | √ | √ | – | 80H |
| FFA3H | Oscillation stabilization time counter status register | OSTC | R | √ | √ | – | 00H |
| FFA4H | Oscillation stabilization time select register | OSTS | R/W | – | √ | – | 05H |
| FFA5H | IIC shift register 0 | IIC0 | R/W | – | √ | – | 00H |
| FFA6H | IIC control register 0 | IICC0 | R/W | √ | √ | – | 00H |
| FFA7H | Slave address register 0 | SVA0 | R/W | – | √ | – | 00H |
| FFA8H | IIC clock selection register 0 | IICCL0 | R/W | √ | √ | – | 00H |
| FFA9H | IIC function expansion register 0 | IICX0 | R/W | √ | √ | – | 00H |
| FFAAH | IIC status register 0 | IICS0 | R | √ | √ | – | 00H |
| FFABH | IIC flag register 0 | IICF0 | R/W | √ | √ | – | 00H |
| FFACH | Reset control flag register | RESF | R | – | √ | – | 00H ^{Note 3} |

- Notes**
1. The reset value of WDTE is determined by setting of option byte.
 2. The value of this register is 00H immediately after a reset release but automatically changes to 80H after oscillation accuracy stabilization of high-speed internal oscillator has been waited.
 3. The reset value of RESF varies depending on the reset source.

Table 3-6. Special Function Register List (4/4)

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulatable Bit Unit | | | After Reset |
|---------|--|--------|------|-----|------------------------|--------|---------|-----------------------|
| | | | | | 1 Bit | 8 Bits | 16 Bits | |
| FFBAH | 16-bit timer mode control register 00 | TMC00 | | R/W | √ | √ | – | 00H |
| FFBBH | Prescaler mode register 00 | PRM00 | | R/W | √ | √ | – | 00H |
| FFBCH | Capture/compare control register 00 | CRC00 | | R/W | √ | √ | – | 00H |
| FFBDH | 16-bit timer output control register 00 | TOC00 | | R/W | √ | √ | – | 00H |
| FFBEH | Low-voltage detection register | LVIM | | R/W | √ | √ | – | 00H ^{Note 1} |
| FFBFH | Low-voltage detection level selection register | LVIS | | R/W | √ | √ | – | 00H ^{Note 1} |
| FFE0H | Interrupt request flag register 0L | IF0 | IF0L | R/W | √ | √ | √ | 00H |
| FFE1H | Interrupt request flag register 0H | | IF0H | R/W | √ | √ | | 00H |
| FFE2H | Interrupt request flag register 1L | IF1 | IF1L | R/W | √ | √ | √ | 00H |
| FFE3H | Interrupt request flag register 1H | | IF1H | R/W | √ | √ | | 00H |
| FFE4H | Interrupt mask flag register 0L | MK0 | MK0L | R/W | √ | √ | √ | FFH |
| FFE5H | Interrupt mask flag register 0H | | MK0H | R/W | √ | √ | | FFH |
| FFE6H | Interrupt mask flag register 1L | MK1 | MK1L | R/W | √ | √ | √ | FFH |
| FFE7H | Interrupt mask flag register 1H | | MK1H | R/W | √ | √ | | FFH |
| FFE8H | Priority specification flag register 0L | PR0 | PR0L | R/W | √ | √ | √ | FFH |
| FFE9H | Priority specification flag register 0H | | PR0H | R/W | √ | √ | | FFH |
| FFEAH | Priority specification flag register 1L | PR1 | PR1L | R/W | √ | √ | √ | FFH |
| FFEBH | Priority specification flag register 1H | | PR1H | R/W | √ | √ | | FFH |
| FFF0H | Internal memory size switching register ^{Note 2} | IMS | | R/W | – | √ | – | CFH |
| FFF4H | Internal expansion RAM size switching register ^{Note 2} | IXS | | R/W | – | √ | – | 0CH |
| FFFBH | Processor clock control register | PCC | | R/W | √ | √ | – | 01H |

- Notes 1.** The reset values of LVIM and LVIS vary depending on the reset source.
- 2.** Regardless of the internal memory capacity, the initial values of the internal memory size switching register (IMS) and internal expansion RAM size switching register (IXS) of all products in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B are fixed (IMS = CFH, IXS = 0CH). Therefore, set the value corresponding to each product as indicated in Tables 3-1.

3.3 Instruction Address Addressing

An instruction address is determined by contents of the program counter (PC) and memory bank select register (BANK), and is normally incremented (+1 for each byte) automatically according to the number of bytes of an instruction to be fetched each time another instruction is executed. When a branch instruction is executed, the branch destination information is set to PC and branched by the following addressing (for details of instructions, refer to the **78K/0 Series Instructions User’s Manual (U12326E)**).

3.3.1 Relative addressing

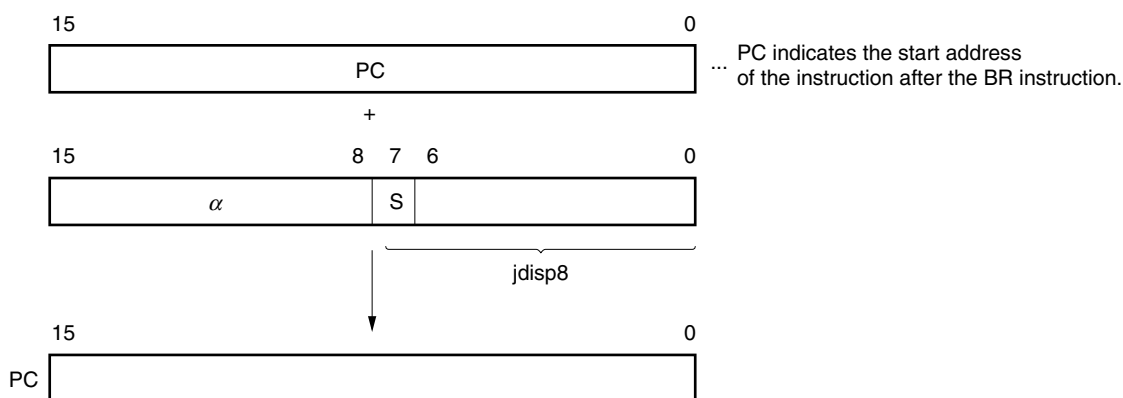
[Function]

The value obtained by adding 8-bit immediate data (displacement value: *jdisp8*) of an instruction code to the start address of the following instruction is transferred to the program counter (PC) and branched. The displacement value is treated as signed two’s complement data (–128 to +127) and bit 7 becomes a sign bit.

In other words, relative addressing consists of relative branching from the start address of the following instruction to the –128 to +127 range.

This function is carried out when the BR \$addr16 instruction or a conditional branch instruction is executed.

[Illustration]



When S = 0, all bits of *α* are 0.
 When S = 1, all bits of *α* are 1.

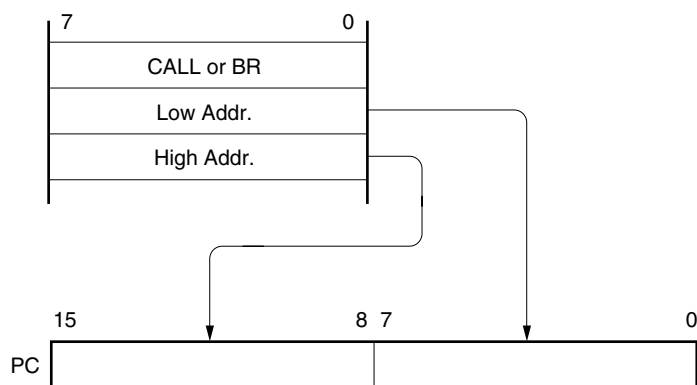
3.3.2 Immediate addressing

[Function]

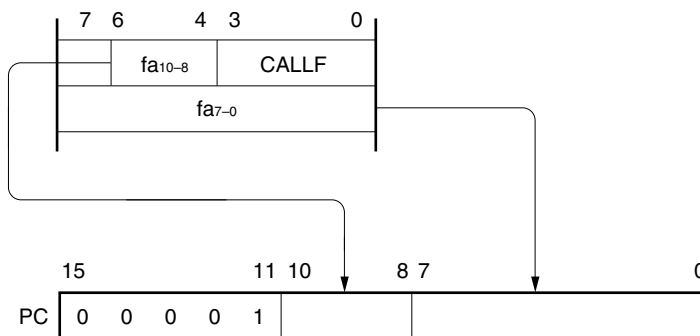
Immediate data in the instruction word is transferred to the program counter (PC) and branched.
 This function is carried out when the CALL !addr16 or BR !addr16 or CALLF !addr11 instruction is executed.
 CALL !addr16 and BR !addr16 instructions can be branched to the entire memory space. However, before branching to a memory bank that is not set by the memory bank select register (BANK), change the setting of the memory bank by using BANK.
 The CALLF !addr11 instruction is branched to the 0800H to 0FFFH area.

[Illustration]

In the case of CALL !addr16 and BR !addr16 instructions



In the case of CALLF !addr11 instruction



3.3.3 Table indirect addressing

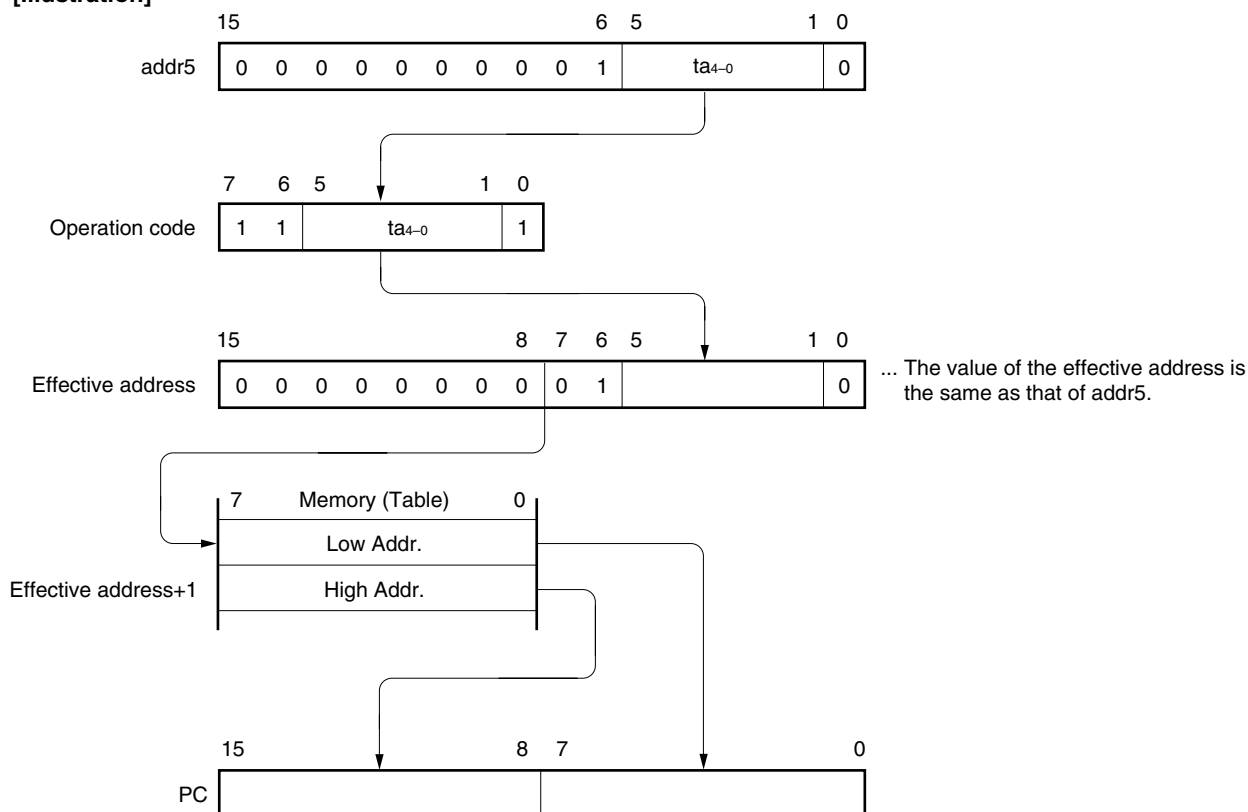
[Function]

Table contents (branch destination address) of the particular location to be addressed by bits 1 to 5 of the immediate data of an operation code are transferred to the program counter (PC) and branched.

This function is carried out when the CALLT [addr5] instruction is executed.

This instruction references the address that is indicated by addr5 and is stored in the memory table from 0040H to 007FH, and allows branching to the entire memory space.

[Illustration]



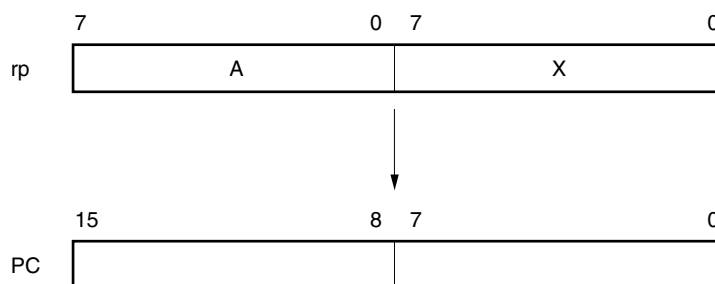
3.3.4 Register addressing

[Function]

Register pair (AX) contents to be specified with an instruction word are transferred to the program counter (PC) and branched.

This function is carried out when the BR AX instruction is executed.

[Illustration]



3.4 Operand Address Addressing

The following methods are available to specify the register and memory (addressing) to undergo manipulation during instruction execution.

3.4.1 Implied addressing

[Function]

The register that functions as an accumulator (A and AX) among the general-purpose registers is automatically (implicitly) addressed.

Of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B instruction words, the following instructions employ implied addressing.

| Instruction | Register to Be Specified by Implied Addressing |
|-------------|---|
| MULU | A register for multiplicand and AX register for product storage |
| DIVUW | AX register for dividend and quotient storage |
| ADJBA/ADJBS | A register for storage of numeric values that become decimal correction targets |
| ROR4/ROL4 | A register for storage of digit data that undergoes digit rotation |

[Operand format]

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

[Description example]

In the case of MULU X

With an 8-bit × 8-bit multiply instruction, the product of the A register and X register is stored in AX. In this example, the A and AX registers are specified by implied addressing.

3.4.2 Register addressing

[Function]

The general-purpose register to be specified is accessed as an operand with the register bank select flags (RBS0 to RBS1) and the register specify codes of an operation code.

Register addressing is carried out when an instruction with the following operand format is executed. When an 8-bit register is specified, one of the eight registers is specified with 3 bits in the operation code.

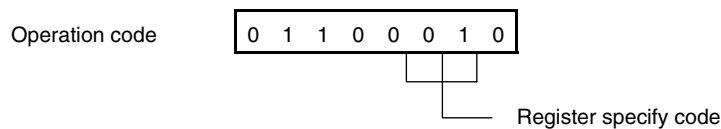
[Operand format]

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

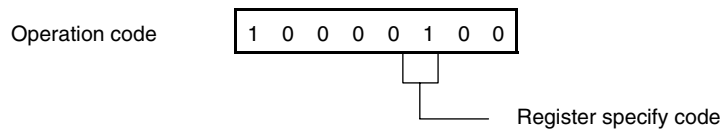
'r' and 'rp' can be described by absolute names (R0 to R7 and RP0 to RP3) as well as function names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

[Description example]

MOV A, C; when selecting C register as r



INCW DE; when selecting DE register pair as rp



3.4.3 Direct addressing

[Function]

The memory to be manipulated is directly addressed with immediate data in an instruction word becoming an operand address.

This addressing can be carried out for all of the memory spaces. However, before addressing a memory bank that is not set by the memory bank select register (BANK), change the setting of the memory bank by using BANK.

[Operand format]

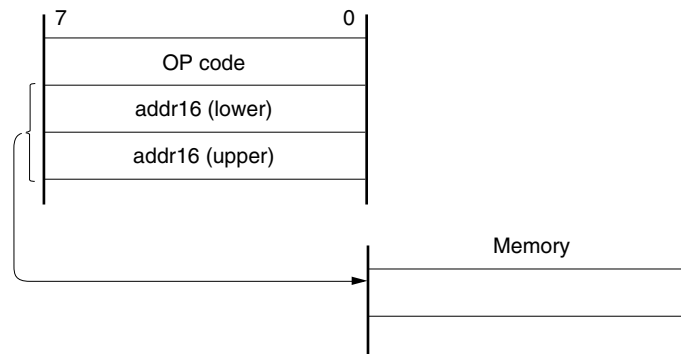
| Identifier | Description |
|------------|--------------------------------|
| addr16 | Label or 16-bit immediate data |

[Description example]

MOV A, !0FE00H; when setting !addr16 to FE00H

| | | | | | | | | | | |
|----------------|---|---|---|---|---|---|---|---|---|---------|
| Operation code | <table border="1"><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table> | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | OP code |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | | | |
| | <table border="1"><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table> | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 00H |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | |
| | <table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table> | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | FEH |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | | | |

[Illustration]



3.4.4 Short direct addressing

[Function]

The memory to be manipulated in the fixed space is directly addressed with 8-bit data in an instruction word. This addressing is applied to the 256-byte space FE20H to FF1FH. Internal high-speed RAM and special function registers (SFRs) are mapped at FE20H to FEFFH and FF00H to FF1FH, respectively. The SFR area (FF00H to FF1FH) where short direct addressing is applied is a part of the overall SFR area. Ports that are frequently accessed in a program and compare and capture registers of the timer/event counter are mapped in this area, allowing SFRs to be manipulated with a small number of bytes and clocks. When 8-bit immediate data is at 20H to FFH, bit 8 of an effective address is set to 0. When it is at 00H to 1FH, bit 8 is set to 1. See the [Illustration] shown below.

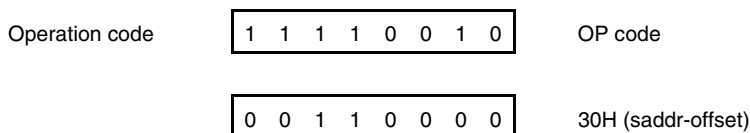
[Operand format]

| Identifier | Description |
|------------|--|
| saddr | Immediate data that indicate label or FE20H to FF1FH |
| saddrp | Immediate data that indicate label or FE20H to FF1FH (even address only) |

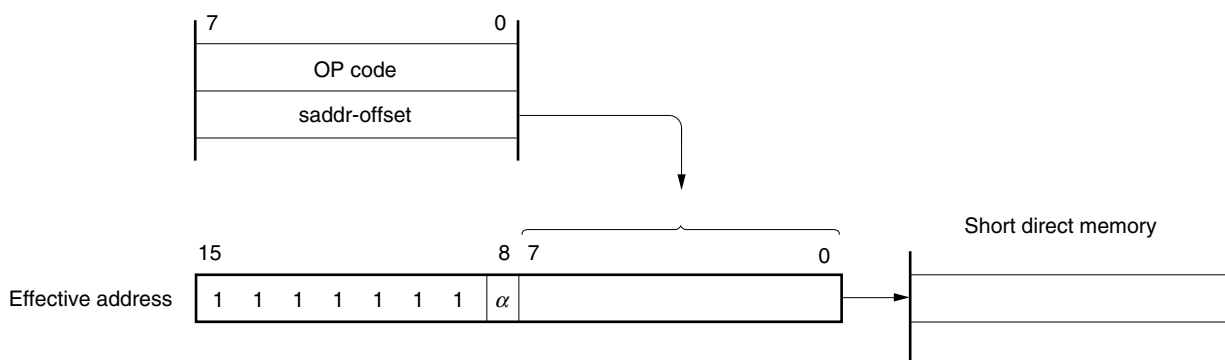
[Description example]

```

LB1 EQU 0FE30H ; Defines FE30H by LB1.
:
MOV LB1, A ; When LB1 indicates FE30H of the saddr area and the value of register A is transferred to that
            address
    
```



[Illustration]



When 8-bit immediate data is 20H to FFH, $\alpha = 0$
 When 8-bit immediate data is 00H to 1FH, $\alpha = 1$

3.4.5 Special function register (SFR) addressing

[Function]

A memory-mapped special function register (SFR) is addressed with 8-bit immediate data in an instruction word.

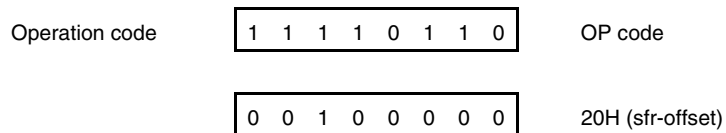
This addressing is applied to the 240-byte spaces FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped at FF00H to FF1FH can be accessed with short direct addressing.

[Operand format]

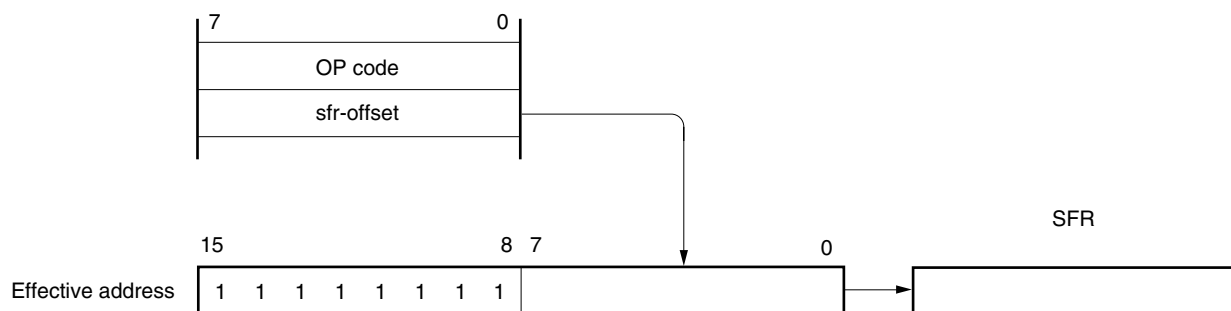
| Identifier | Description |
|------------|---|
| sfr | Special function register name |
| sfrp | 16-bit manipulatable special function register name (even address only) |

[Description example]

MOV PM0, A; when selecting PM0 (FF20H) as sfr



[Illustration]



3.4.6 Register indirect addressing

[Function]

Register pair contents specified by a register pair specify code in an instruction word and by a register bank select flag (RBS0 and RBS1) serve as an operand address for addressing the memory.

This addressing can be carried out for all of the memory spaces. However, before addressing a memory bank that is not set by the memory bank select register (BANK), change the setting of the memory bank by using BANK.

[Operand format]

| Identifier | Description |
|------------|-------------|
| – | [DE], [HL] |

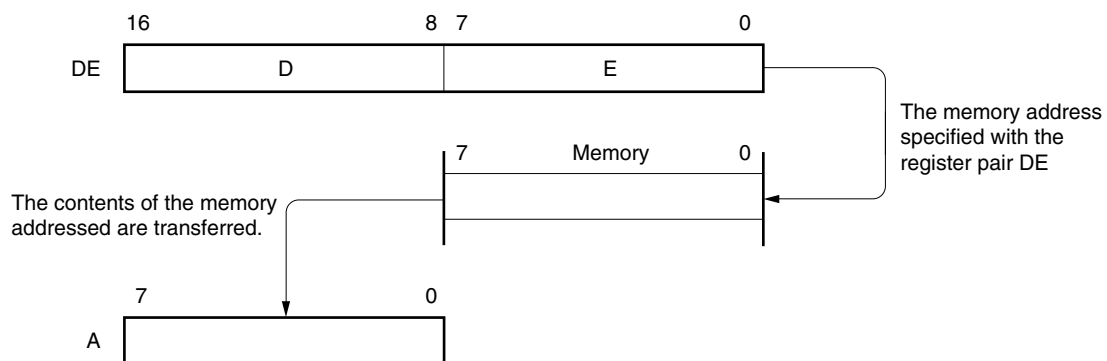
[Description example]

MOV A, [DE]; when selecting [DE] as register pair

Operation code

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|

[Illustration]



3.4.7 Based addressing

[Function]

8-bit immediate data is added as offset data to the contents of the base register, that is, the HL register pair in the register bank specified by the register bank select flag (RBS0 and RBS1), and the sum is used to address the memory. Addition is performed by expanding the offset data as a positive number to 16 bits. A carry from the 16th bit is ignored.

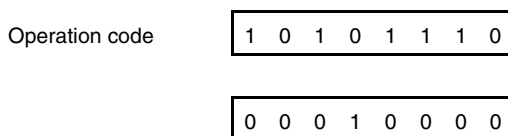
This addressing can be carried out for all of the memory spaces. However, before addressing a memory bank that is not set by the memory bank select register (BANK), change the setting of the memory bank by using BANK.

[Operand format]

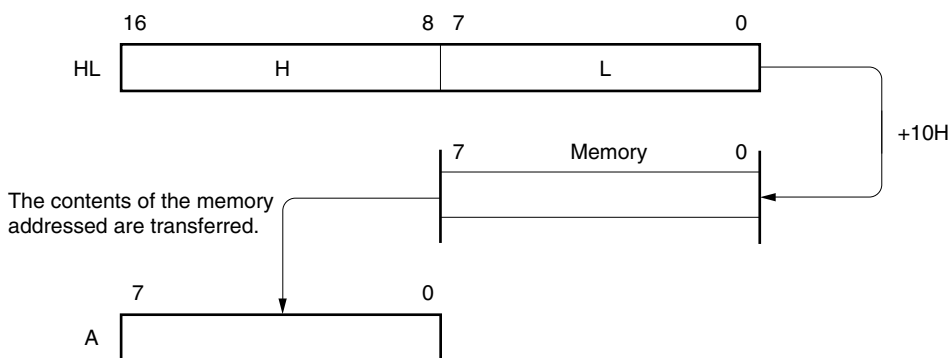
| Identifier | Description |
|------------|-------------|
| – | [HL + byte] |

[Description example]

MOV A, [HL + 10H]; when setting byte to 10H



[Illustration]



3.4.8 Based indexed addressing

[Function]

The B or C register contents specified in an instruction word are added to the contents of the base register, that is, the HL register pair in the register bank specified by the register bank select flag (RBS0 and RBS1), and the sum is used to address the memory. Addition is performed by expanding the B or C register contents as a positive number to 16 bits. A carry from the 16th bit is ignored.

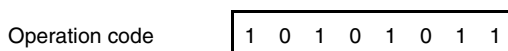
This addressing can be carried out for all of the memory spaces. However, before addressing a memory bank that is not set by the memory bank select register (BANK), change the setting of the memory bank by using BANK.

[Operand format]

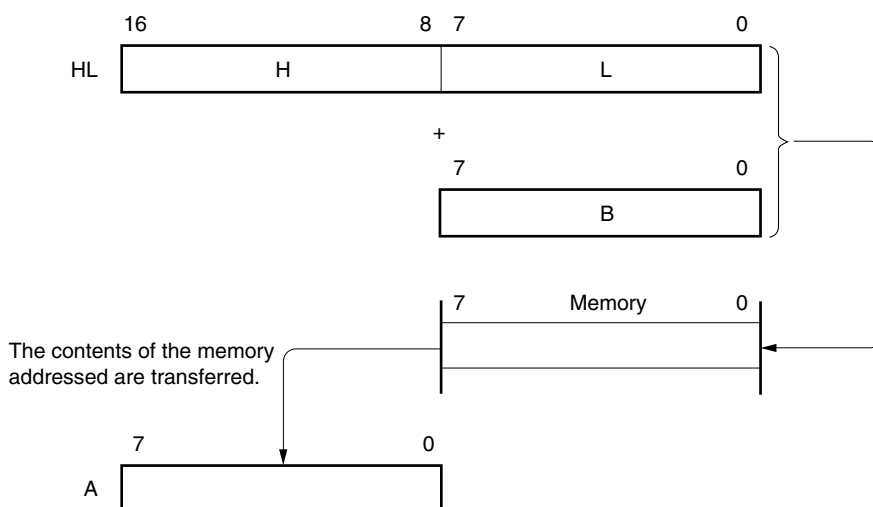
| Identifier | Description |
|------------|--------------------|
| - | [HL + B], [HL + C] |

[Description example]

MOV A, [HL +B]; when selecting B register



[Illustration]



3.4.9 Stack addressing

[Function]

The stack area is indirectly addressed with the stack pointer (SP) contents.

This addressing method is automatically employed when the PUSH, POP, subroutine call and return instructions are executed or the register is saved/reset upon generation of an interrupt request.

With stack addressing, only the internal high-speed RAM area can be accessed.

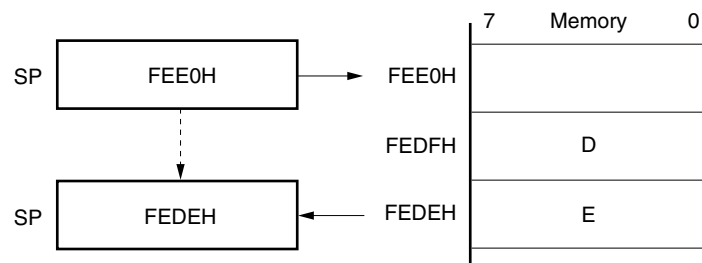
[Description example]

PUSH DE; when saving DE register

Operation code

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
|---|---|---|---|---|---|---|---|

[Illustration]



CHAPTER 4 PORT FUNCTIONS

4.1 Port Functions

Pin I/O buffer power supplies include one V_{DD} system. The relationship between these power supplies and the pins is shown below.

Table 4-1. Pin I/O Buffer Power Supplies

| Power Supply | Corresponding Pins |
|--------------|--------------------|
| V_{DD} | All pins |

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B, microcontrollers are provided with digital I/O ports, which enable variety of control operations. The functions of each port are shown in Table 4-2.

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**.

Table 4-2. Port Functions

| Function Name | I/O | Function | After Reset | Alternate Function |
|---------------|-----|---|--------------|---------------------------------------|
| P00 | I/O | Port 0. 2-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. | Input port | TI000 |
| P01 | | | | TI010/TO00 |
| 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. | Input port | $\overline{\text{SCK10}}/\text{TxD0}$ |
| P11 | | | | SI10/RxD0 |
| P12 | | | | SO10 |
| P13 | | | | TxD6 |
| P14 | | | | RxD6 |
| P15 | | | | TOH0 |
| P16 | | | | TOH1/INTP5 |
| P17 | | | | TI50/TO50 |
| P20 to P23 | I/O | Port 2. 4-bit I/O port. Input/output can be specified in 1-bit units. | Analog input | ANI0 to ANI3 |
| P30 | I/O | Port 3. 4-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. | Input port | INTP1 |
| P31 | | | | INTP2 |
| P32 | | | | INTP3 |
| P33 | | | | INTP4/TI51/TO51 |
| P40, P41 | I/O | Port 4. 2-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. | Input port | – |
| P60 | I/O | Port 6 2-bit I/O port. Output is N-ch open-drain output (6 V tolerance). Input/output can be specified in 1-bit units. | Input port | SCL0 |
| P61 | | | | SDA0 |
| P70, P71 | I/O | Port 7 2-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. | Input port | – |
| P120 | I/O | Port 12. 3-bit I/O port. Input/output can be specified in 1-bit units. Only for P120, use of an on-chip pull-up resistor can be specified by a software setting. | Input port | INTP0/EXLVI |
| P121 | | | | X1 |
| P122 | | | | X2/EXCLK |

4.2 Port Configuration

Ports include the following hardware.

Table 4-3. Port Configuration

| Item | Configuration |
|-------------------|---|
| Control registers | Port mode register (PMxx): PM0 to PM4, PM6, PM7, PM12 Port register (Pxx): P0 to P4, P6, P7, P12 Pull-up resistor option register (PUxx): PU0, PU1, PU3, PU4, PU7, PU12 A/D port configuration register (ADPC) |
| Port | Total: 27 (CMOS I/O: 25, N-ch open drain I/O: 2) |
| Pull-up resistor | Total: 19 |

4.2.1 Port 0

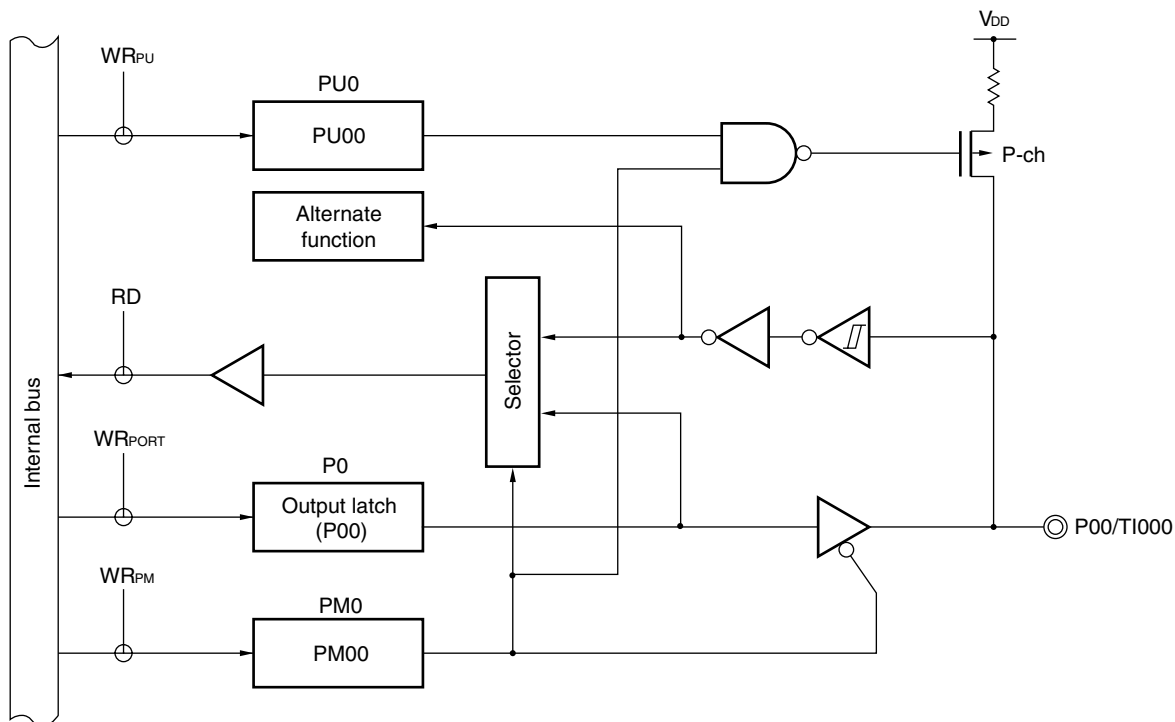
Port 0 is an I/O port with an output latch. Port 0 can be set to the input mode or output mode in 1-bit units using port mode register 0 (PM0). When the P00 and P01 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 0 (PU0).

This port can also be used for timer I/O.

Reset signal generation sets port 0 to input mode.

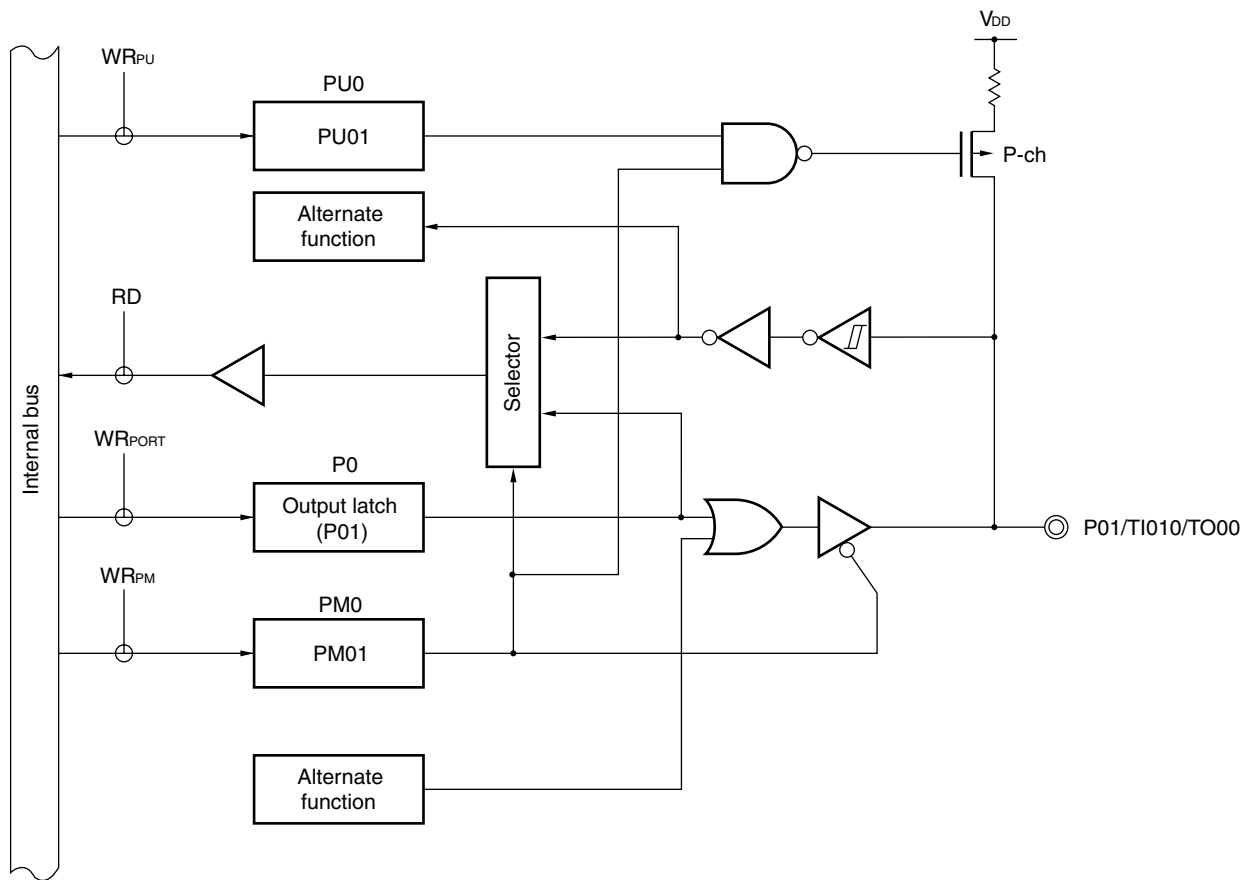
Figures 4-1 and 4-2 show block diagrams of port 0.

Figure 4-1. Block Diagram of P00



- P0: Port register 0
- PU0: Pull-up resistor option register 0
- PM0: Port mode register 0
- RD: Read signal
- WR_{xx}: Write signal

Figure 4-2. Block Diagram of P01



- P0: Port register 0
 PU0: Pull-up resistor option register 0
 PM0: Port mode register 0
 RD: Read signal
 WR_{xx} : Write signal

4.2.2 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).

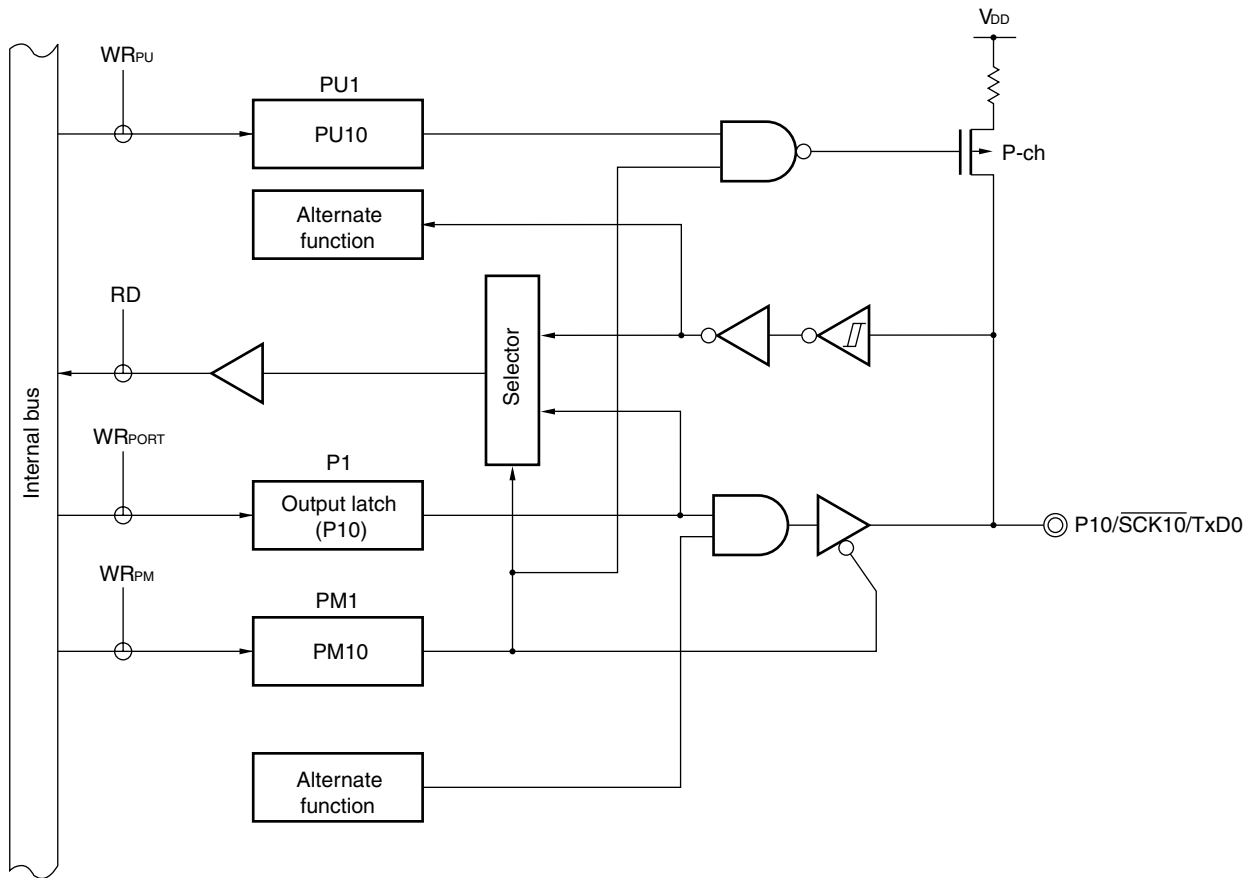
This port can also be used for, serial interface data I/O, clock I/O, timer I/O, and external interrupt request input.

Reset signal generation sets port 1 to input mode.

Figures 4-3 to 4-7 show block diagrams of port 1.

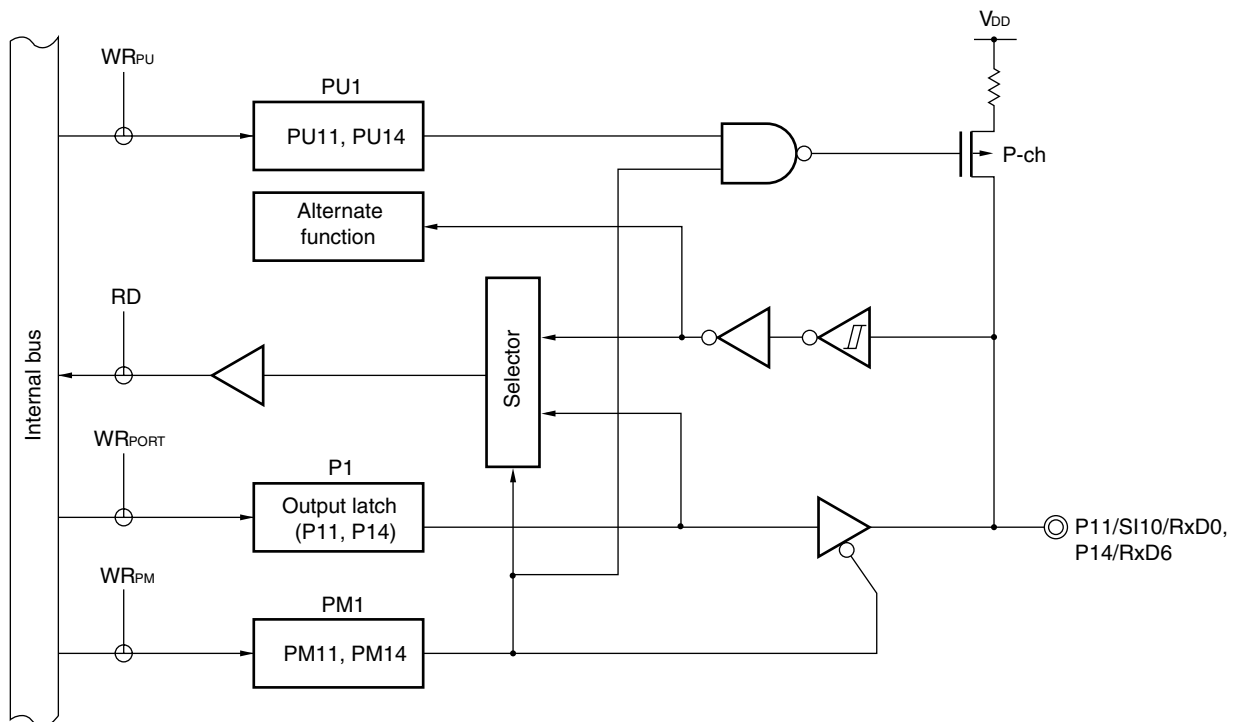
- Cautions**
1. To use P10/ $\overline{\text{SCK10}}$ /TxD0 and P12/SO10 as general-purpose ports, set serial operation mode register 10 (CSIM10) and serial clock selection register 10 (CSIC10) to the default status (00H).
 2. To use P13/TxD6 as general-purpose port, clear bit 0 (TXDLV6) of asynchronous serial interface control register 6 (ASICL6) to 0 (normal output of TxD6).

Figure 4-3. Block Diagram of P10



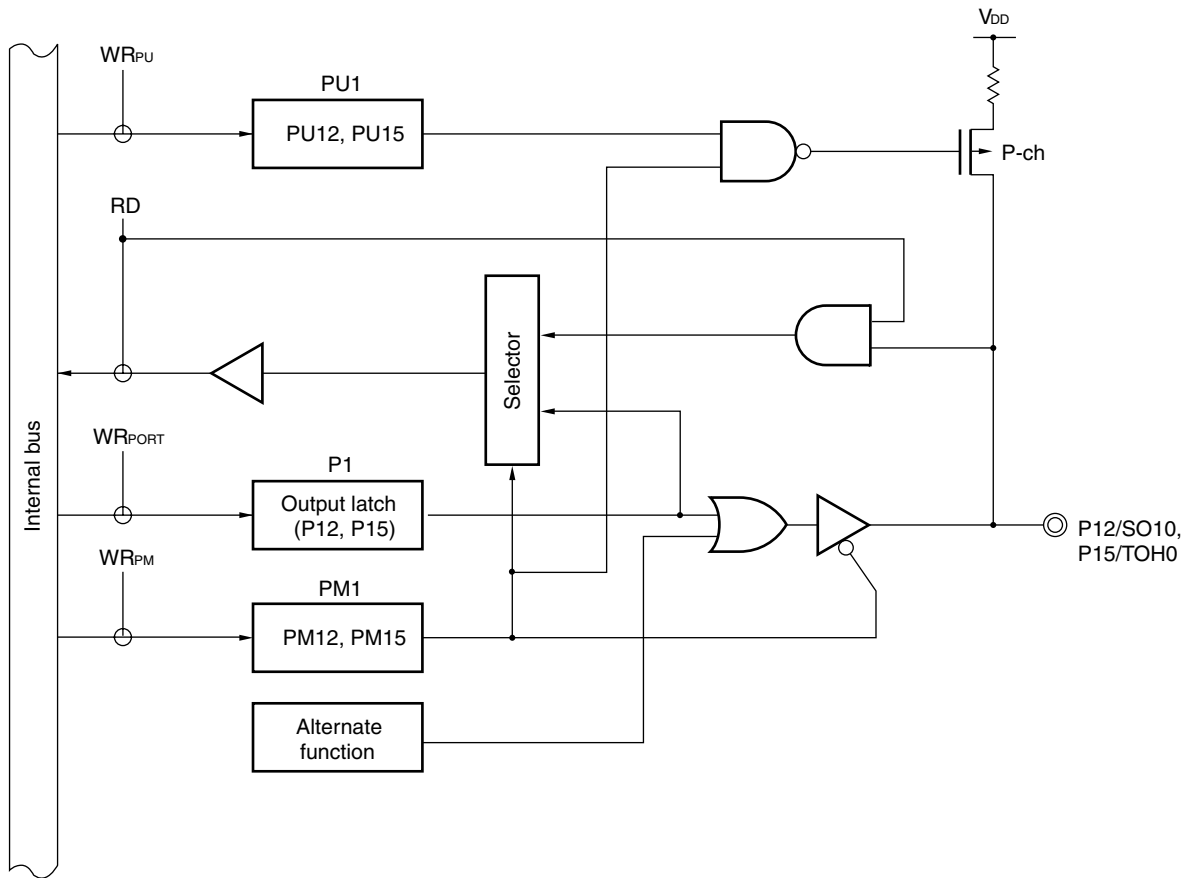
- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- WR_{xx}: Write signal

Figure 4-4. Block Diagram of P11 and P14



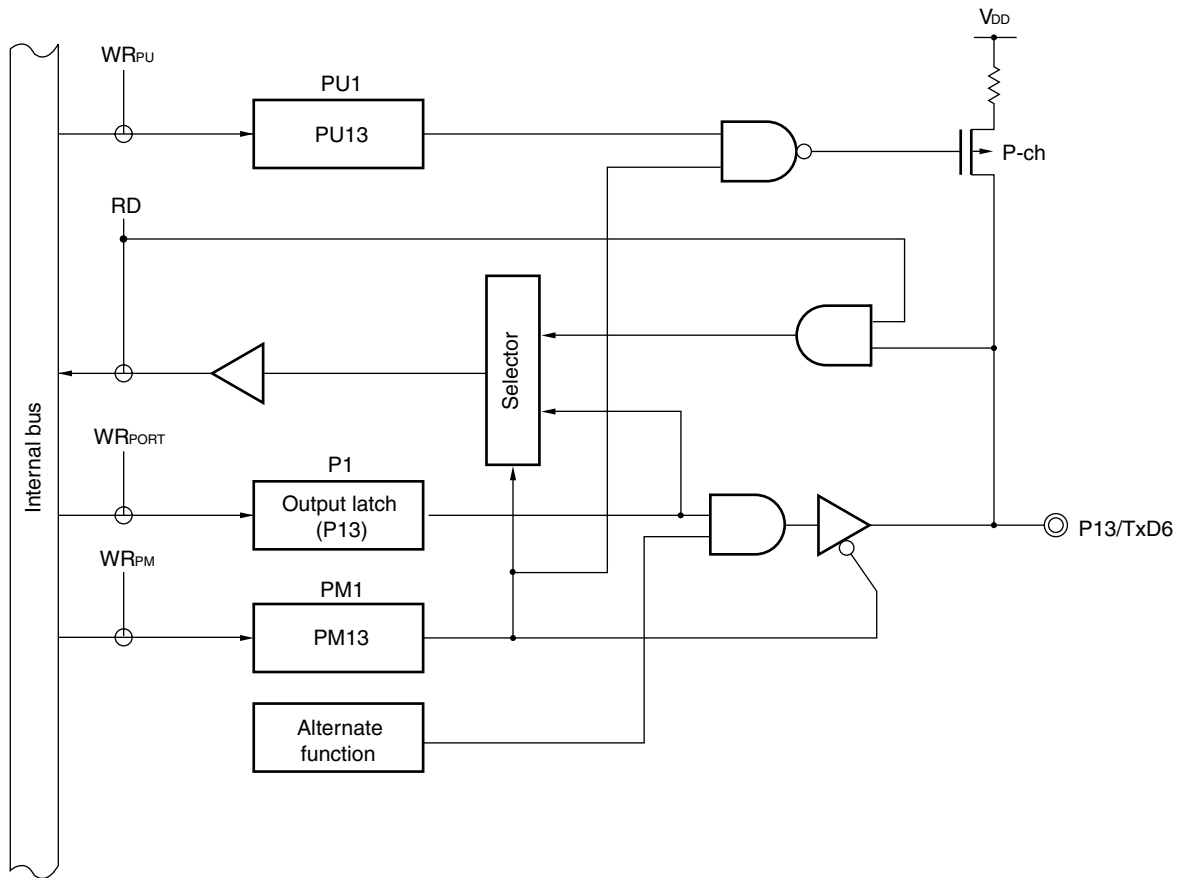
- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- WR_{xx} : Write signal

Figure 4-5. Block Diagram of P12 and P15



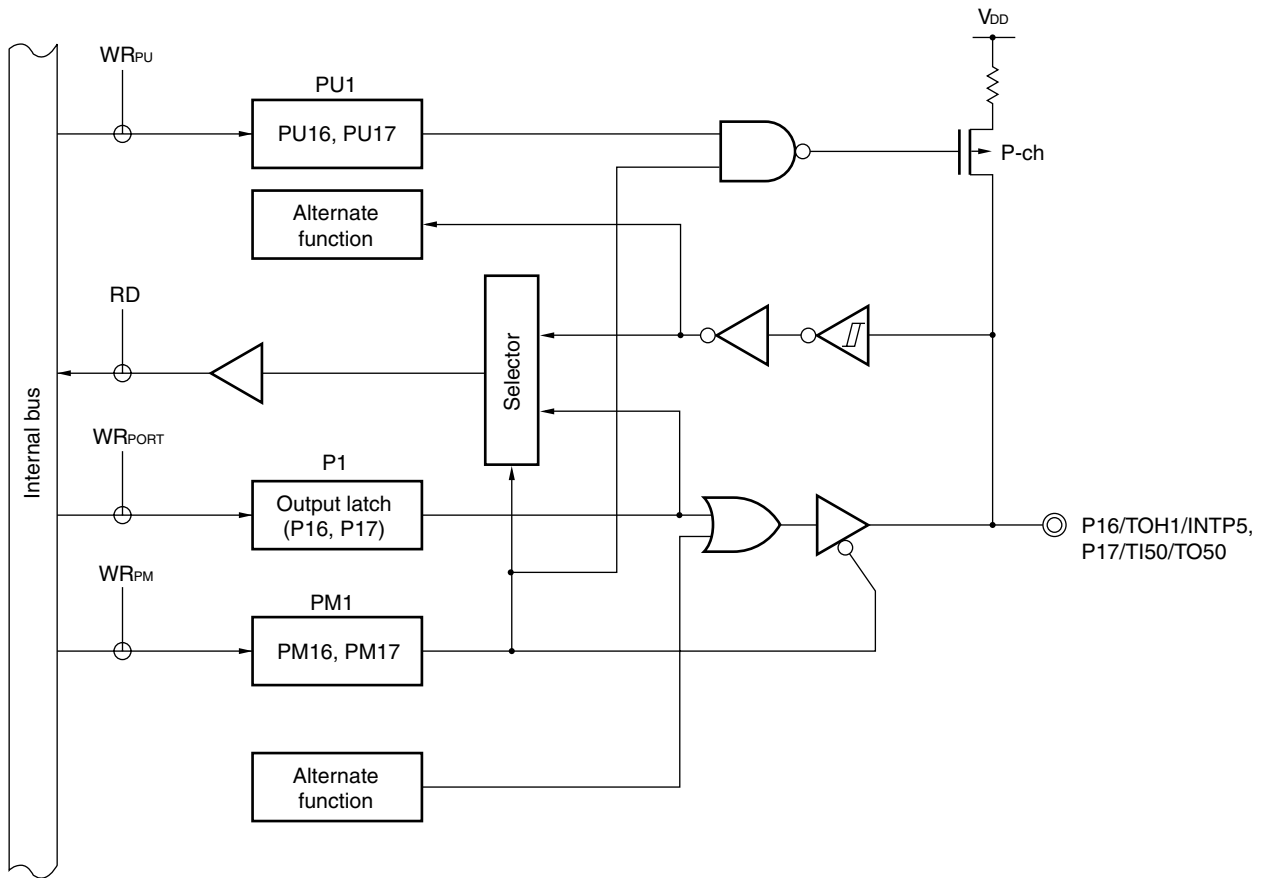
- P1: Port register 1
- PU1: Pull-up resistor option register 1
- PM1: Port mode register 1
- RD: Read signal
- WR_{xx} : Write signal

Figure 4-6. Block Diagram of P13



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

Figure 4-7. Block Diagram of P16 and P17



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

4.2.3 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.

To use P20/ANI0 to P23/ANI3 as digital input pins, set them in the digital I/O mode by using the A/D port configuration register (ADPC) and in the input mode by using PM2. Use these pins starting from the lower bit.

To use P20/ANI0 to P23/ANI3 as digital output pins, set them in the digital I/O mode by using ADPC and in the output mode by using PM2.

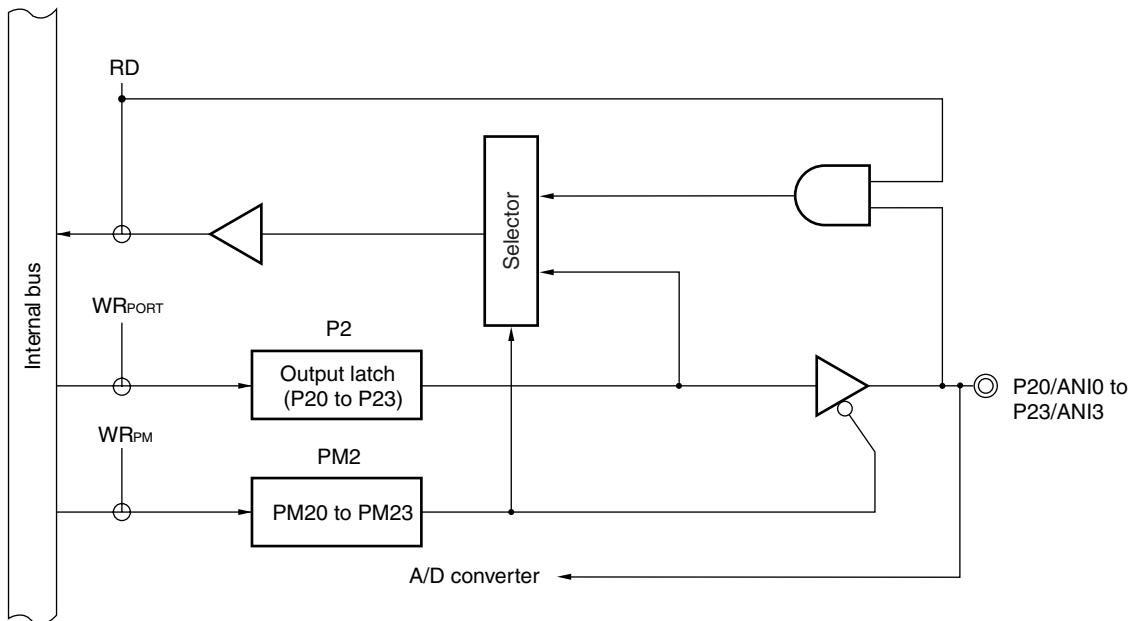
Table 4-4. Setting Functions of P20/ANI0 to P23/ANI3 Pins

| ADPC | PM2 | ADS | P20/ANI0 to P23/ANI3 Pin |
|------------------------|-------------|----------------------|------------------------------------|
| 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. | |

All P20/ANI0 to P23/ANI3 are set in the analog input mode when the reset signal is generated.

Figure 4-8 shows a block diagram of port 2.

Figure 4-8. Block Diagram of P20 to P23



- P2: Port register 2
 PM2: Port mode register 2
 RD: Read signal
 WR_{xx} : Write signal

4.2.4 Port 3

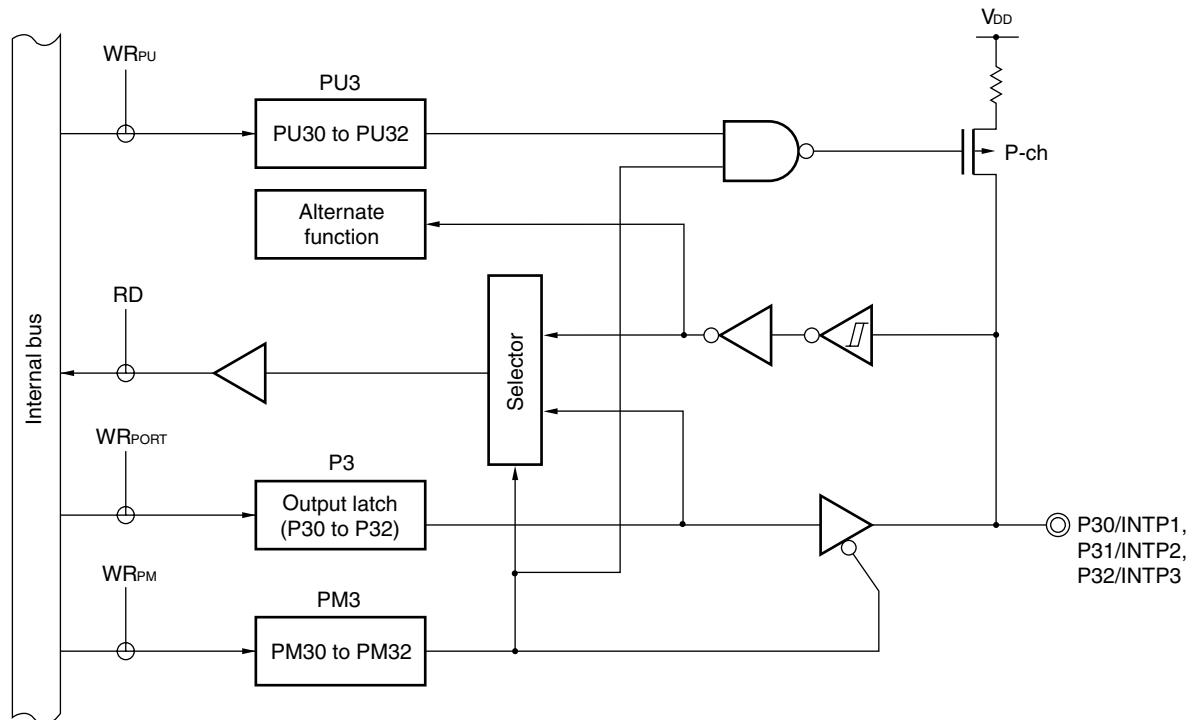
Port 3 is an I/O port with an output latch. Port 3 can be set to the input mode or output mode in 1-bit units using port mode register 3 (PM3). When the P30 to P33 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 3 (PU3).

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

Reset signal generation sets port 3 to input mode.

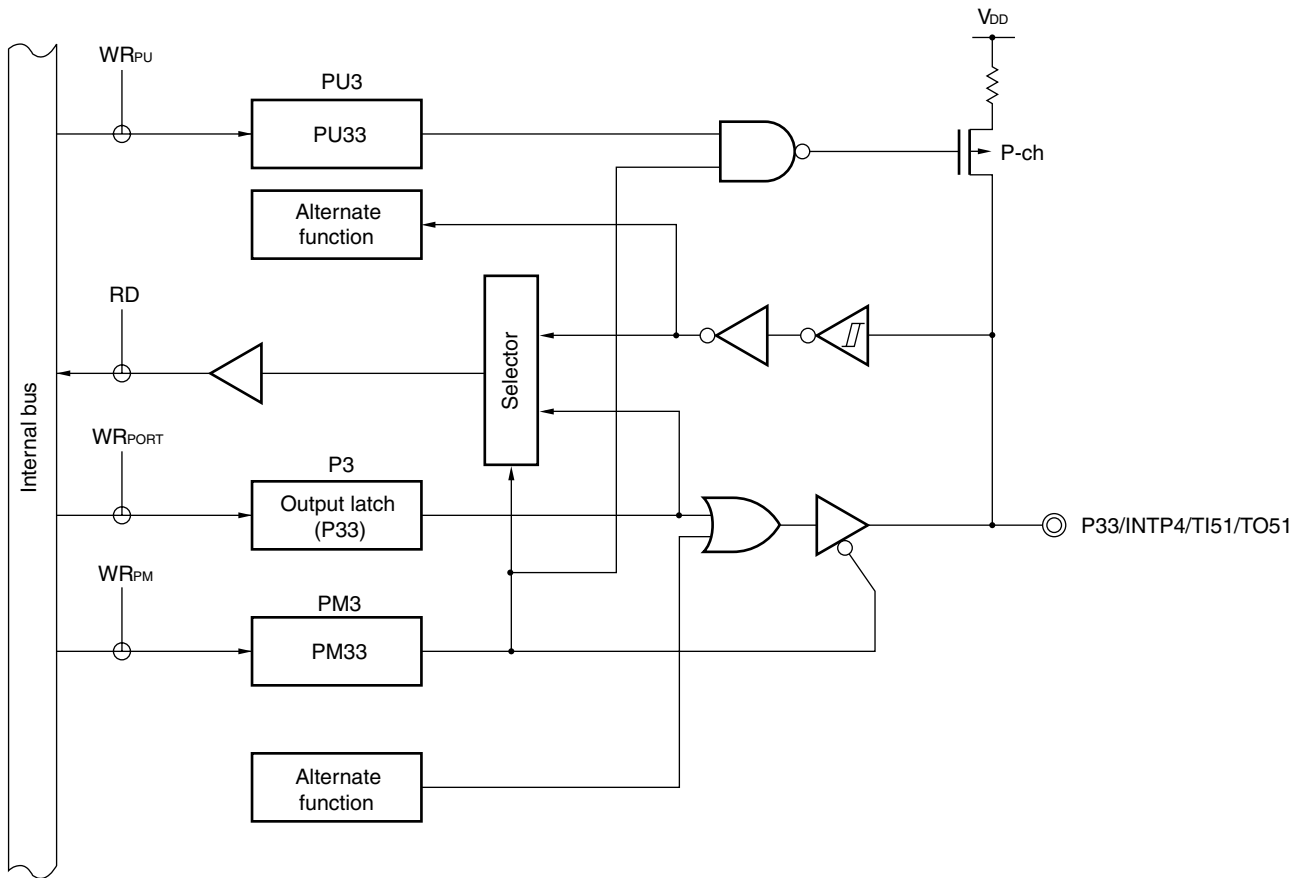
Figures 4-9 and 4-10 show block diagrams of port 3.

Figure 4-9. Block Diagram of P30 to P32



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

Figure 4-10. Block Diagram of P33



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

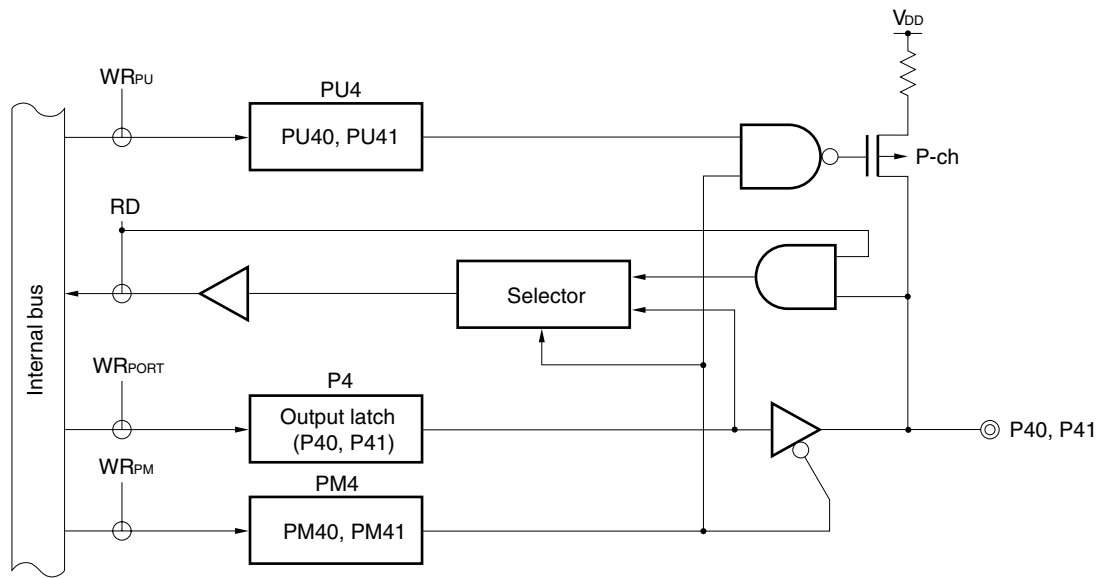
4.2.5 Port 4

Port 4 is an I/O port with an output latch. Port 4 can be set to the input mode or output mode in 1-bit units using port mode register 4 (PM4). When the P40 and P41 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 4 (PU4).

Reset signal generation sets port 4 to input mode.

Figure 4-11 shows a block diagram of port 4.

Figure 4-11. Block Diagram of P40 and P41



- P4: Port register 4
- PU4: Pull-up resistor option register 4
- PM4: Port mode register 4
- RD: Read signal
- WR_{xx} : Write signal

4.2.6 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).

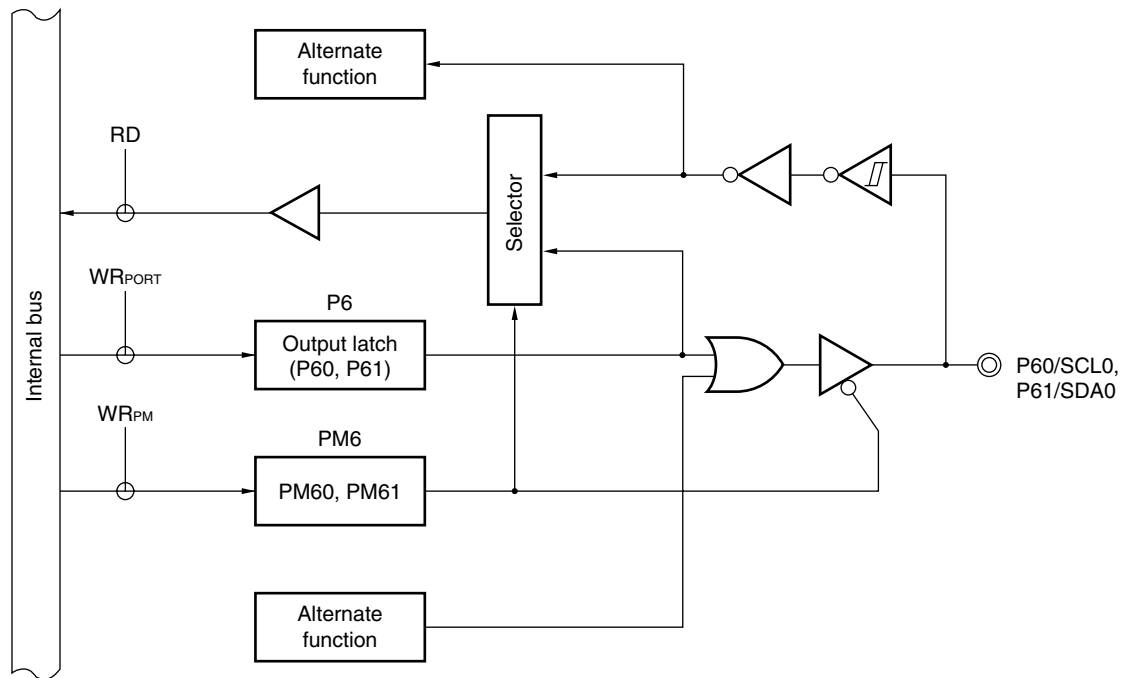
The output of the P60 and P61 pins is N-ch open-drain output (6 V tolerance).

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

Reset signal generation sets port 6 to input mode.

Figure 4-12 shows block diagram of port 6.

Figure 4-12. Block Diagram of P60 and P61



P6: Port register 6
 PM6: Port mode register 6
 RD: Read signal
 WR_{xx}: Write signal

Caution A through current flows through P60 and P61 if an intermediate potential is input to these pins, because the input buffer is also turned on when P60 and P61 are in output mode. Consequently, do not input an intermediate potential when P60 and P61 are in output mode.

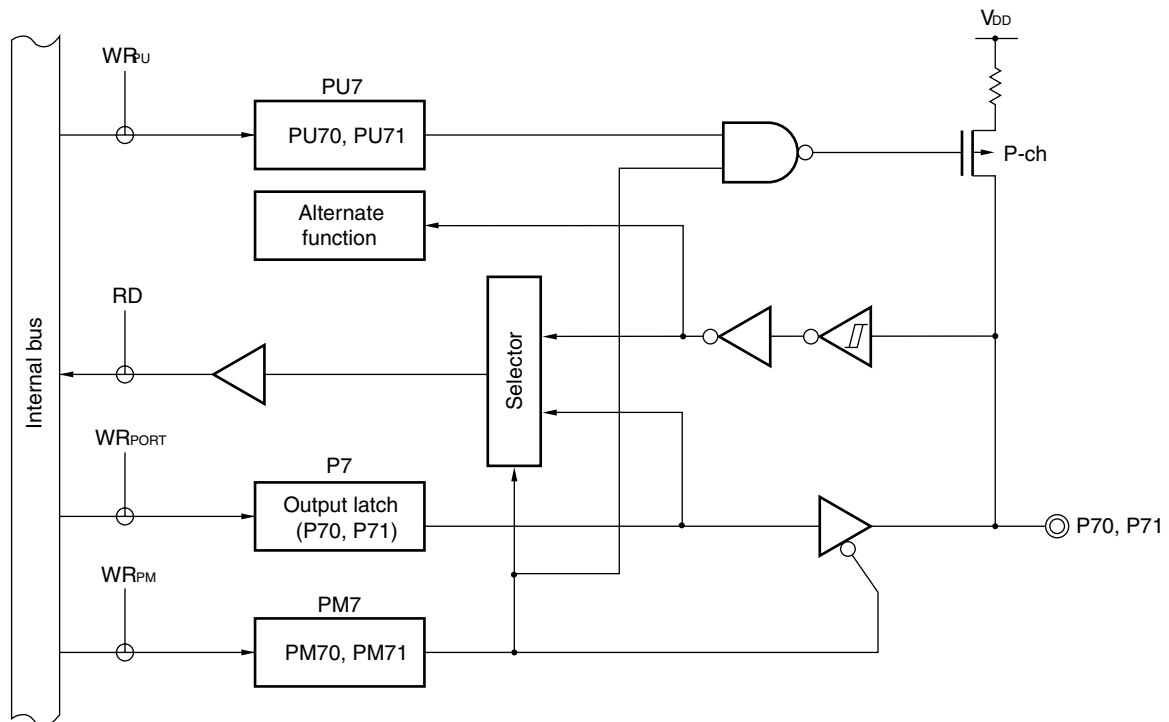
4.2.7 Port 7

Port 7 is an I/O port with an output latch. Port 7 can be set to the input mode or output mode in 1-bit units using port mode register 7 (PM7). When the P70 and P71 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 7 (PU7).

Reset signal generation sets port 7 to input mode.

Figure 4-13 shows a block diagram of port 7.

Figure 4-13. Block Diagram of P70 and P71



- P7: Port register 7
- PU7: Pull-up resistor option register 7
- PM7: Port mode register 7
- RD: Read signal
- WR_{xx}: Write signal

4.2.8 Port 12

Port 12 is an I/O port with an output latch. Port 12 can be set to the input mode or output mode in 1-bit units using port mode register 12 (PM12). When used as an input port only for P120, use of an on-chip pull-up resistor can be specified by pull-up resistor option register 12 (PU12).

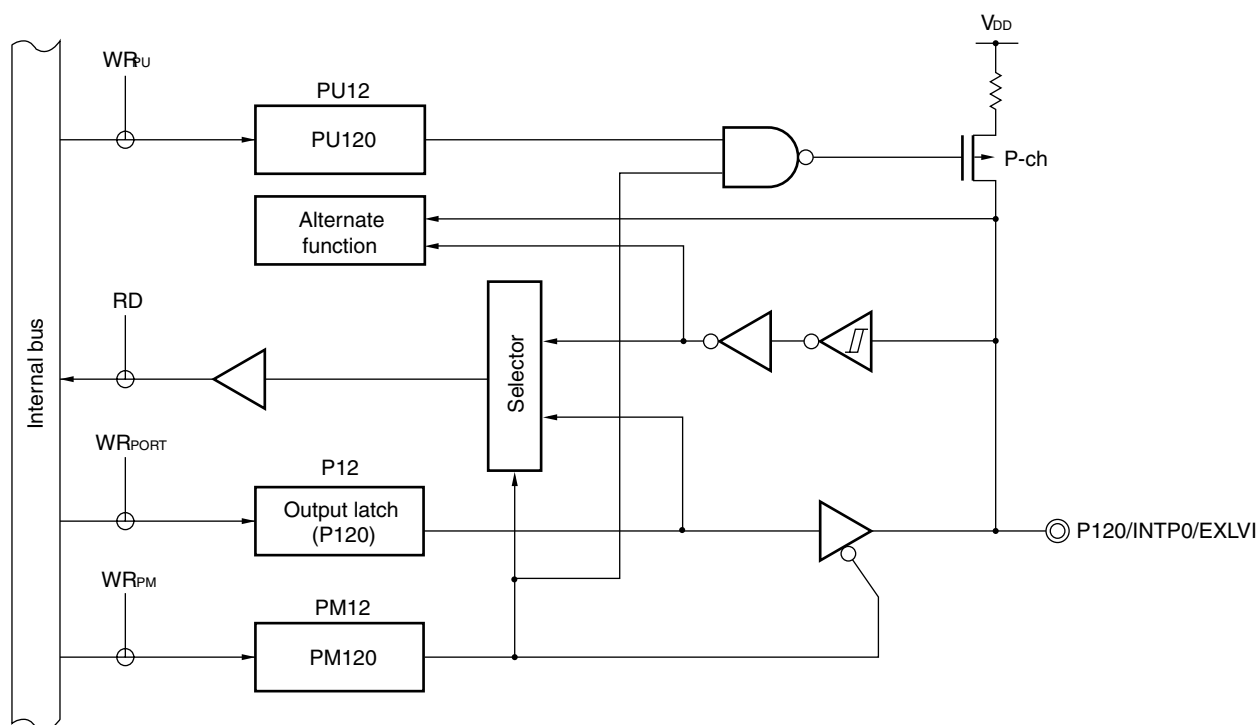
This port can also be used as pins for external interrupt request input, potential input for external low-voltage detection, connecting resonator for main system clock, and external clock input for main system clock.

Reset signal generation sets port 12 to input mode.

Figures 4-14 and 4-15 show block diagrams of port 12.

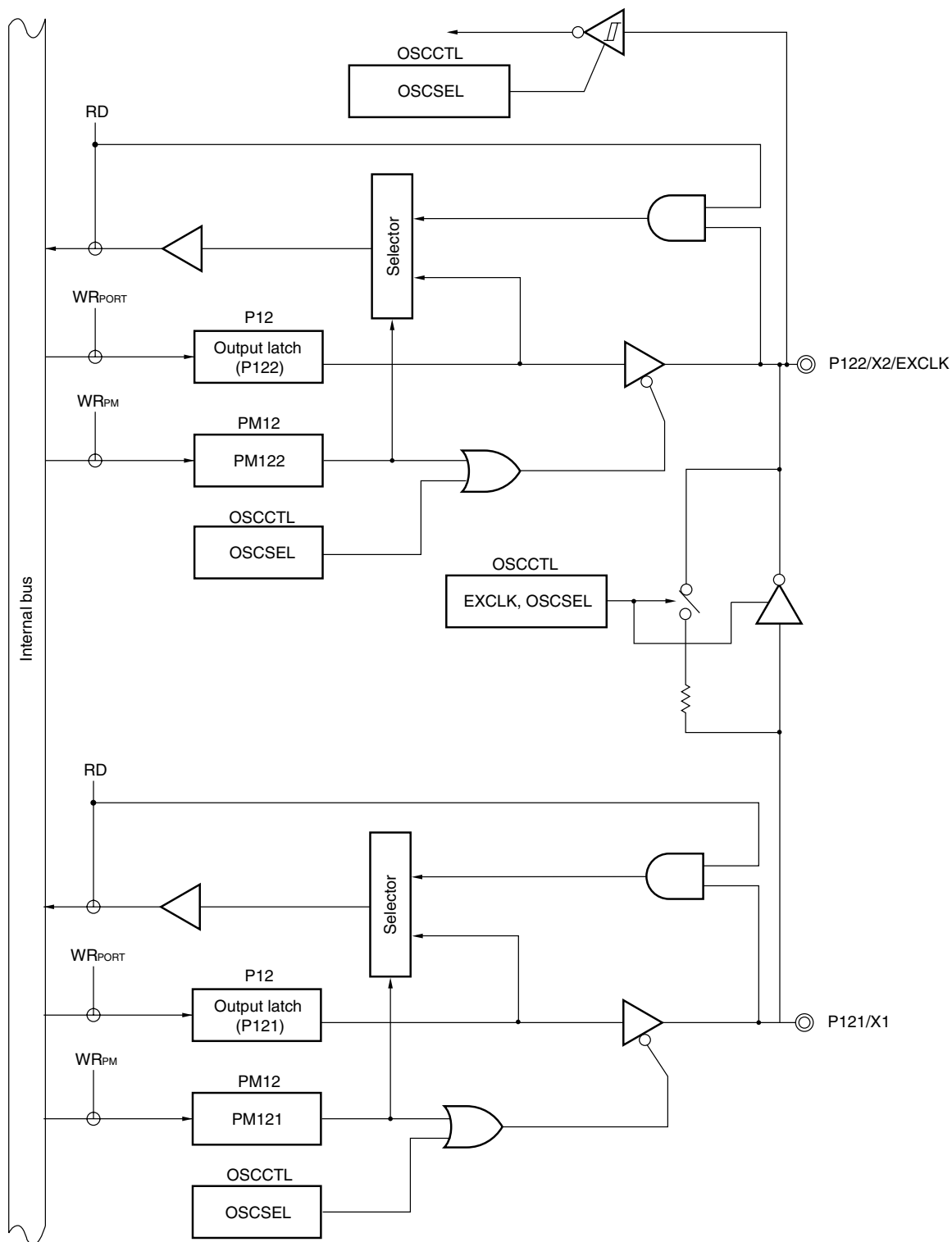
Caution When using the P121 and P122 pins to connect a resonator for the main system clock (X1, X2), or to input an external clock for the main system clock (EXCLK), the X1 oscillation mode, or external clock input mode must be set by using the clock operation mode select register (OSCCTL) (for details, see 5.3 (1) Clock operation mode select register (OSCCTL)). The reset value of OSCCTL is 00H (all of the P121 and P122 pins are I/O port pins). At this time, setting of the PM121, PM122, P121, and P122 pins is not necessary.

Figure 4-14. Block Diagram of P120



- P12: Port register 12
- PU12: Pull-up resistor option register 12
- PM12: Port mode register 12
- RD: Read signal
- WR_{xx}: Write signal

Figure 4-15. Block Diagram of P121 and P122



- P12: Port register 12
- PU12: Pull-up resistor option register 12
- PM12: Port mode register 12
- OSCCTL: Clock operation mode select register
- RD: Read signal
- WR_{xx}: Write signal

4.3 Registers Controlling Port Function

Port functions are controlled by the following four types of registers.

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

(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 Mode Register and Output Latch 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 |
|--------|--|------|------|------|------|-------|-------|-------|---------|-------------|-----|
| PM0 | 1 | 1 | 1 | 1 | 1 | 1 | PM01 | PM00 | FF20H | FFH | R/W |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 | FF21H | FFH | R/W |
| PM2 | 1 | 1 | 1 | 1 | PM23 | PM22 | PM21 | PM20 | FF22H | FFH | R/W |
| PM3 | 1 | 1 | 1 | 1 | PM33 | PM32 | PM31 | PM30 | FF23H | FFH | R/W |
| PM4 | 1 | 1 | 1 | 1 | 1 | 1 | PM41 | PM40 | FF24H | FFH | R/W |
| PM6 | 1 | 1 | 1 | 1 | 1 | 1 | PM61 | PM60 | FF26H | FFH | R/W |
| PM7 | 1 | 1 | 1 | 1 | 1 | 1 | PM71 | PM70 | FF27H | FFH | R/W |
| PM12 | 1 | 1 | 1 | 1 | 1 | PM122 | PM121 | PM120 | FF2CH | FFH | R/W |
| PMmn | Pmn pin I/O mode selection (m = 0 to 4, 6, 7, 12; n = 0 to 7) | | | | | | | | | | |
| 0 | Output mode (output buffer on) | | | | | | | | | | |
| 1 | Input mode (output buffer off) | | | | | | | | | | |

Caution Be sure to set bits 2 to 7 of PM0, bits 4 to 7 of PM2, bits 4 to 7 of PM3, bits 2 to 7 of PM4, bits 2 to 7 of PM6, bits 2 to 7 of PM7, and bits 3 to 7 of PM12 to “1”.

(2) Port registers (Pxx)

These registers write the data that is output from the chip when data is output from 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.

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

Reset signal generation clears these registers to 00H.

Figure 4-17. Format of Port Mode Register

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|--------|-----|-----|-----|-----|-----|----------------------|----------------------|------|---------|--------------------|-----|
| P0 | 0 | 0 | 0 | 0 | 0 | 0 | P01 | P00 | FF00H | 00H (output latch) | R/W |
| P1 | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | FF01H | 00H (output latch) | R/W |
| P2 | 0 | 0 | 0 | 0 | P23 | P22 | P21 | P20 | FF02H | 00H (output latch) | R/W |
| P3 | 0 | 0 | 0 | 0 | P33 | P32 | P31 | P30 | FF03H | 00H (output latch) | R/W |
| P4 | 0 | 0 | 0 | 0 | 0 | 0 | P41 | P40 | FF04H | 00H (output latch) | R/W |
| P6 | 0 | 0 | 0 | 0 | 0 | 0 | P61 | P60 | FF06H | 00H (output latch) | R/W |
| P7 | 0 | 0 | 0 | 0 | 0 | 0 | P71 | P70 | FF07H | 00H (output latch) | R/W |
| P12 | 0 | 0 | 0 | 0 | 0 | P122 ^{Note} | P121 ^{Note} | P120 | FF0CH | 00H (output latch) | R/W |

| PMmn | m = 0 to 4, 6, 7, 12; n = 0 to 7 | |
|------|--------------------------------------|------------------------------------|
| | Output data control (in output mode) | Input data reading (in input mode) |
| 0 | Output 0 | Input low level |
| 1 | Output 1 | Input high level |

Note “0” is always read from the output latch of P121 and P122 if the pin is in the external clock input mode.

Caution Be sure to set bits 2 to 7 of P0, bits 4 to 7 of P2, bits 4 to 7 of P3, bits 2 to 7 of P4, bits 2 to 7 of P6, bits 2 to 7 of P7, and bits 3 to 7 of P12 to “0”.

(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 of 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, 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.

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

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Address | After reset | R/W |
|--------|---|------|------|------|------|------|------|-------|---------|-------------|-----|
| PU0 | 0 | 0 | 0 | 0 | 0 | 0 | PU01 | PU00 | FF30H | 00H | R/W |
| PU1 | PU17 | PU16 | PU15 | PU14 | PU13 | PU12 | PU11 | PU10 | FF31H | 00H | R/W |
| PU3 | 0 | 0 | 0 | 0 | PU33 | PU32 | PU31 | PU30 | FF33H | 00H | R/W |
| PU4 | 0 | 0 | 0 | 0 | 0 | 0 | PU41 | PU40 | FF34H | 00H | R/W |
| PU7 | 0 | 0 | 0 | 0 | 0 | 0 | PU71 | PU70 | FF37H | 00H | R/W |
| PU12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PU120 | FF3CH | 00H | R/W |
| PUmn | Pmn pin on-chip pull-up resistor selection (m = 0, 1, 3, 4, 7, 12; n = 0 to 7) | | | | | | | | | | |
| 0 | On-chip pull-up resistor not connected | | | | | | | | | | |
| 1 | On-chip pull-up resistor connected | | | | | | | | | | |

(4) A/D port configuration register (ADPC)

This register switches the P20/ANI0 to P23/ANI3 pins to digital I/O of port or analog input of A/D converter. ADPC can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 4-19. Format of A/D Port Configuration Register (ADPC)

Address: FF2FH After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|-------|-------|-------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADPC | 0 | 0 | 0 | 0 | ADPC3 | ADPC2 | ADPC1 | ADPC0 |

| ADPC3 | ADPC2 | ADPC1 | ADPC0 | Digital I/O (D)/analog input (A) switching | | | |
|------------------|-------|-------|-------|--|----------|----------|----------|
| | | | | P23/ANI3 | P22/ANI2 | P21/ANI1 | P20/ANI0 |
| 0 | 0 | 0 | 0 | A | A | A | A |
| 0 | 0 | 0 | 1 | A | A | A | D |
| 0 | 0 | 1 | 0 | A | A | D | D |
| 0 | 0 | 1 | 1 | A | D | D | D |
| 0 | 1 | 0 | 0 | D | D | D | D |
| 1 | 0 | 0 | 0 | D | D | D | D |
| Other than above | | | | Setting prohibited | | | |

- Cautions**
1. Set the channel used for A/D conversion to the input mode by using port mode register 2 (PM2).
 2. If data is written to ADPC, a wait cycle is generated. Do not write data to ADPC when the peripheral hardware clock is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

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.

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.

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

4.5 Settings of Port Mode Register and Output Latch 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 4-5.

Table 4-5. Settings of Port Mode Register and Output Latch When Using Alternate Function (1/2)

| Pin Name | Alternate Function | | PM _{xx} | P _{xx} |
|----------------------------|------------------------------|--------|------------------|-----------------|
| | Function Name | I/O | | |
| P00 | TI000 | Input | 1 | × |
| P01 | TI010 | Input | 1 | × |
| | TO00 | Output | 0 | 0 |
| P10 | SCK10 | Input | 1 | × |
| | | Output | 0 | 1 |
| | TxD0 | Output | 0 | 1 |
| P11 | SI10 | Input | 1 | × |
| | RxD0 | Input | 1 | × |
| P12 | SO10 | Output | 0 | 0 |
| P13 | TxD6 | Output | 0 | 1 |
| P14 | RxD6 | Input | 1 | × |
| P15 | TOH0 | Output | 0 | 0 |
| P16 | TOH1 | Output | 0 | 0 |
| | INTP5 | Input | 1 | × |
| P17 | TI50 | Input | 1 | × |
| | TO50 | Output | 0 | 0 |
| P20 to P23 ^{Note} | ANI0 to ANI3 ^{Note} | Input | 1 | × |

Note The function of the ANI0/P20 to ANI3/P23 pins can be selected by using the A/D port configuration register (ADPC), the analog input channel specification register (ADS), and PM2.

| ADPC | PM2 | ADS | ANI0/P20 to ANI3/P23 Pins |
|------------------------|-------------|----------------------|------------------------------------|
| 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. | |
| Digital I/O selection | Input mode | – | Digital input |
| | Output mode | – | Digital output |

Remark ×: Don't care
 PM_{xx}: Port mode register
 P_{xx}: Port output latch

Table 4-5. Settings of Port Mode Register and Output Latch When Using Alternate Function (2/2)

| Pin Name | Alternate Function | | PM _{xx} | P _{xx} |
|------------|-----------------------|--------|------------------|-----------------|
| | Function Name | I/O | | |
| P30 to P32 | INTP1 to INTP3 | Input | 1 | × |
| P33 | INTP4 | Input | 1 | × |
| | TI51 | Input | 1 | × |
| | TO51 | Output | 0 | 0 |
| P60 | SCL0 | I/O | 0 | 0 |
| P61 | SDA0 | I/O | 0 | 0 |
| P120 | INTP0 | Input | 1 | × |
| | EXLVI | Input | 1 | × |
| P121 | X1 ^{Note} | – | × | × |
| P122 | X2 ^{Note} | – | × | × |
| | EXCLK ^{Note} | Input | × | × |

Note When using the P121 and P122 pins to connect a resonator for the main system clock (X1, X2), or to input an external clock for the main system clock (EXCLK), the X1 oscillation mode or external clock input mode must be set by using the clock operation mode select register (OSCCTL) (for details, see **5.3 (1) Clock operation mode select register (OSCCTL)**). The reset value of OSCCTL is 00H (all of the P121 and P122 are I/O port pins). At this time, setting of PM121, PM122, P121, and P122 is not necessary.

Remarks ×: Don't care

PM_{xx}: Port mode register

P_{xx}: Port output latch

4.6 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 R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B microcontrollers.

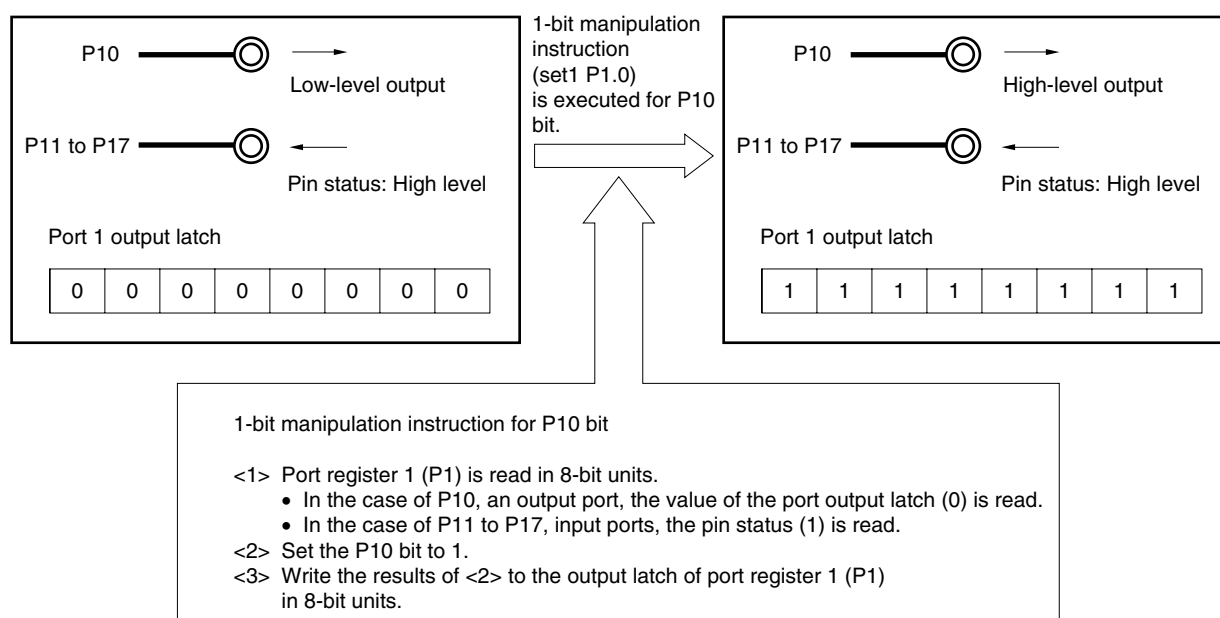
- <1> The Pn register is read in 8-bit units.
- <2> The targeted one bit is manipulated.
- <3> The Pn register is written in 8-bit units.

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 4-20. Bit Manipulation Instruction (P10)



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

<1> X1 oscillator

This circuit oscillates a clock of $f_x = 1$ to 10 MHz by connecting a resonator to X1 and X2.

Oscillation can be stopped by executing the STOP instruction or using the main OSC control register (MOC).

<2> Internal high-speed oscillator

This circuit oscillates a clock of $f_{RH} = 8$ MHz (TYP.). After a reset release, the CPU always starts operating with this internal high-speed oscillation clock. Oscillation can be stopped by executing the STOP instruction or using the internal oscillation mode register (RCM).

An external main system clock ($f_{EXCLK} = 1$ to 10 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 using RCM.

As the main system clock, a high-speed system clock (X1 clock or external main system clock) or internal high-speed oscillation clock can be selected by using the main clock mode register (MCM).

(2) Internal low-speed oscillation clock (clock for watchdog timer)

• Internal low-speed oscillator

This circuit oscillates a clock of $f_{RL} = 240$ kHz (TYP.). After a reset release, the internal low-speed oscillation clock always starts operating.

Oscillation can be stopped by using the internal oscillation mode register (RCM) when “internal low-speed oscillator can be stopped by software” is set by option byte.

The internal low-speed oscillation clock cannot be used as the CPU clock. The following hardware operates with the internal low-speed oscillation clock.

- Watchdog timer
- TMH1 (when f_{RL} , $f_{RL}/2^7$, or $f_{RL}/2^9$ is selected)

| | | |
|---------------|---------------|---|
| Remark | f_x : | X1 clock oscillation frequency |
| | f_{RH} : | Internal high-speed oscillation clock frequency |
| | f_{EXCLK} : | External main system clock frequency |
| | f_{RL} : | Internal low-speed oscillation clock frequency |

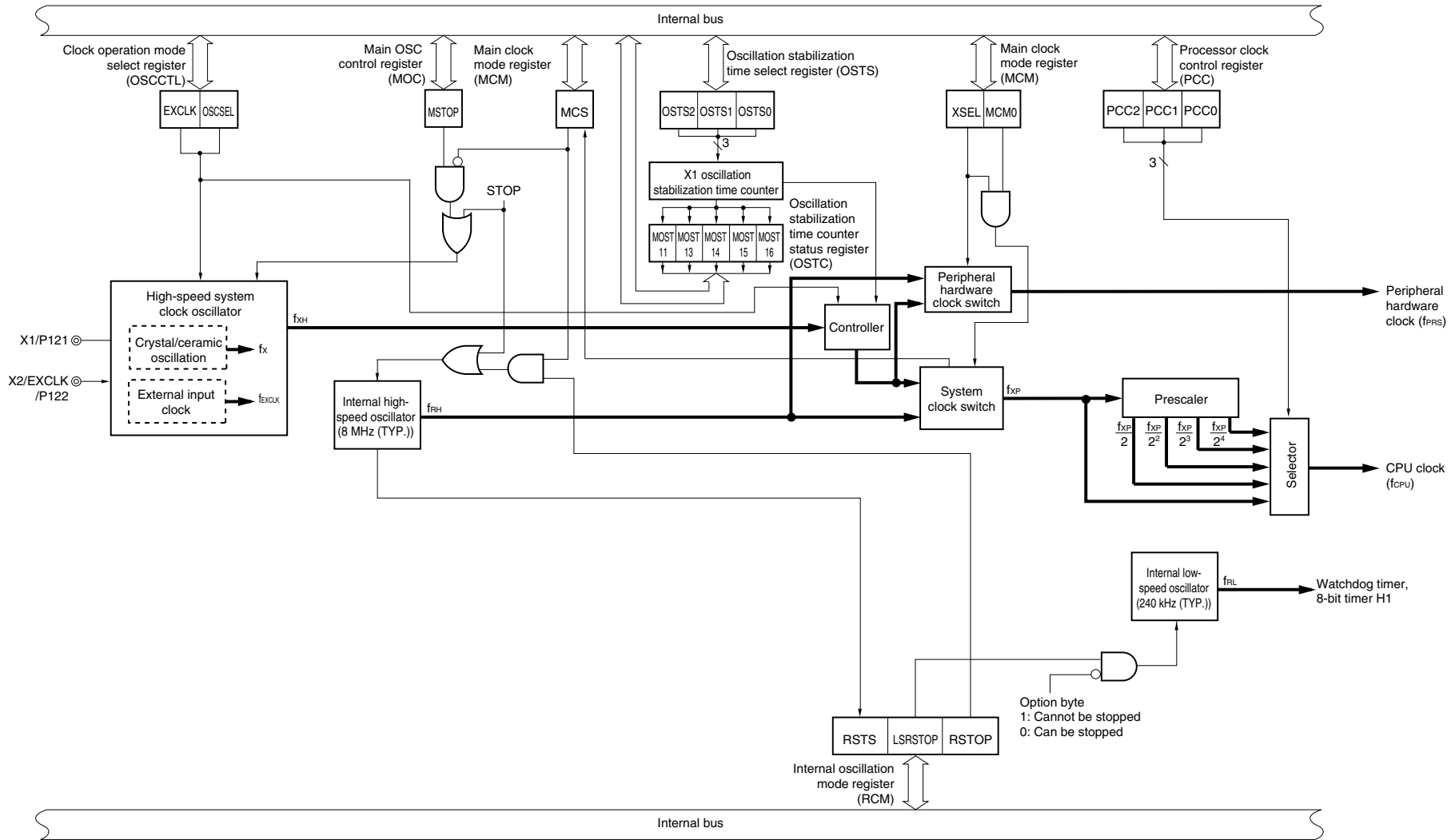
5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

Table 5-1. Configuration of Clock Generator

| Item | Configuration |
|-------------------|--|
| Control registers | Clock operation mode select register (OSCCTL) Processor clock control register (PCC) Internal oscillation mode register (RCM) Main OSC control register (MOC) Main clock mode register (MCM) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) |
| Oscillators | X1 oscillator Internal high-speed oscillator Internal low-speed oscillator |

Figure 5-1. Block Diagram of Clock Generator



| | | |
|---------------|----------------------|---|
| Remark | fx: | X1 clock oscillation frequency |
| | f _{RH} : | Internal high-speed oscillation clock frequency |
| | f _{EXCLK} : | External main system clock frequency |
| | f _{XH} : | High-speed system clock frequency |
| | f _{XP} : | Main system clock frequency |
| | f _{PRS} : | Peripheral hardware clock frequency |
| | f _{CPU} : | CPU clock frequency |
| | f _{RL} : | Internal low-speed oscillation clock frequency |

5.3 Registers Controlling Clock Generator

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

- Clock operation mode select register (OSCCTL)
- Processor clock control register (PCC)
- Internal oscillation mode register (RCM)
- Main OSC control register (MOC)
- Main clock mode register (MCM)
- Oscillation stabilization time counter status register (OSTC)
- Oscillation stabilization time select register (OSTS)

(1) Clock operation mode select register (OSCCTL)

This register selects the operation modes of the high-speed system clock.

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

Reset signal generation clears this register to 00H.

Figure 5-2. Format of Clock Operation Mode Select Register (OSCCTL)

Address: FF9FH After reset: 00H R/W

| | | | | | | | | |
|--------|-------|--------|--|--------------------------------------|---|----------------------|---|---|
| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
| OSCCTL | EXCLK | OSCSEL | 0 | 0 | 0 | 0 | 0 | 0 |
| | EXCLK | OSCSEL | High-speed system clock pin operation mode | P121/X1 pin | | P122/X2/EXCLK pin | | |
| | 0 | 0 | I/O port mode | I/O port | | | | |
| | 0 | 1 | X1 oscillation mode | Crystal/ceramic resonator connection | | | | |
| | 1 | 0 | I/O port mode | I/O port | | | | |
| | 1 | 1 | External clock input mode | I/O port | | External clock input | | |

- Cautions**
1. To change the value of EXCLK and OSCSEL, be sure to confirm that bit 7 (MSTOP) of the main OSC control register (MOC) is 1 (the X1 oscillator stops or the external clock from the EXCLK pin is disabled).
 2. Be sure to clear bits 0 to 5 to 0.

Remark f_{XH}: High-speed system clock oscillation frequency

(2) Processor clock control register (PCC)

This register is used to select the CPU clock and the division ratio.

PCC is set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets PCC to 01H.

Figure 5-3. Format of Processor Clock Control Register (PCC)

Address: FFFBH After reset: 01H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|---|---|------|------|------|
| PCC | 0 | 0 | 0 | 0 | 0 | PCC2 | PCC1 | PCC0 |

| PCC2 | PCC1 | PCC0 | CPU clock (f_{CPU}) selection |
|------------------|------|------|-----------------------------------|
| 0 | 0 | 0 | f_{XP} |
| 0 | 0 | 1 | $f_{XP}/2$ (default) |
| 0 | 1 | 0 | $f_{XP}/2^2$ |
| 0 | 1 | 1 | $f_{XP}/2^3$ |
| 1 | 0 | 0 | $f_{XP}/2^4$ |
| Other than above | | | Setting prohibited |

Cautions 1. Be sure to clear bits 3 to 7 to 0.

2. The peripheral hardware clock (f_{PRS}) is not divided when the division ratio of the PCC is set.

Remark f_{XP} : Main system clock oscillation frequency

The fastest instruction can be executed in 2 clocks of the CPU clock in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B. Therefore, the relationship between the CPU clock (f_{CPU}) and the minimum instruction execution time is as shown in Table 5-2.

Table 5-2. Relationship Between CPU Clock and Minimum Instruction Execution Time

| CPU Clock (f_{CPU}) | Minimum Instruction Execution Time: $2/f_{CPU}$ | |
|-------------------------|---|---|
| | Main System Clock | |
| | High-Speed System Clock ^{Note} | Internal High-Speed Oscillation Clock ^{Note} |
| | At 10 MHz Operation | At 8 MHz (TYP.) Operation |
| f_{XP} | 0.2 μ s | 0.25 μ s (TYP.) |
| $f_{XP}/2$ | 0.4 μ s | 0.5 μ s (TYP.) |
| $f_{XP}/2^2$ | 0.8 μ s | 1.0 μ s (TYP.) |
| $f_{XP}/2^3$ | 1.6 μ s | 2.0 μ s (TYP.) |
| $f_{XP}/2^4$ | 3.2 μ s | 4.0 μ s (TYP.) |

Note The main clock mode register (MCM) is used to set the main system clock supplied to CPU clock (high-speed system clock/internal high-speed oscillation clock) (see **Figure 5-6**).

(3) Internal oscillation mode register (RCM)

This register sets the operation mode of internal oscillator.

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

Reset signal generation sets this register to 80H^{Note 1}.

Figure 5-4. Format of Internal Oscillation Mode Register (RCM)

Address: FFA0H After reset: 80H^{Note 1} R/W^{Note 2}

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
|--------|------|---|---|---|---|---|---------|-------|
| RCM | RSTS | 0 | 0 | 0 | 0 | 0 | LSRSTOP | RSTOP |

| RSTS | Status of internal high-speed oscillator |
|------|--|
| 0 | Waiting for accuracy stabilization of internal high-speed oscillator |
| 1 | Stability operating of internal high-speed oscillator |

| LSRSTOP | Internal low-speed oscillator oscillating/stopped |
|---------|---|
| 0 | Internal low-speed oscillator oscillating |
| 1 | Internal low-speed oscillator stopped |

| RSTOP | Internal high-speed oscillator oscillating/stopped |
|-------|--|
| 0 | Internal high-speed oscillator oscillating |
| 1 | Internal high-speed oscillator stopped |

- Notes**
1. The value of this register is 00H immediately after a reset release but automatically changes to 80H after internal high-speed oscillator has been stabilized.
 2. Bit 7 is read-only.

Caution When setting RSTOP to 1, be sure to confirm that the CPU operates with a clock other than the internal high-speed oscillation clock. Specifically, set under either of the following conditions.

- When MCS = 1 (when CPU operates with the high-speed system clock)

In addition, stop peripheral hardware that is operating on the internal high-speed oscillation clock before setting RSTOP to 1.

(4) Main OSC control register (MOC)

This register selects the operation mode of the high-speed system clock.

This register is used to stop the X1 oscillator or to disable an external clock input from the EXCLK pin when the CPU operates with a clock other than the high-speed system clock.

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

Reset signal generation sets this register to 80H.

Figure 5-5. Format of Main OSC Control Register (MOC)

Address: FFA2H After reset: 80H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|-------|---|---|---|---|---|---|---|
| MOC | MSTOP | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| MSTOP | Control of high-speed system clock operation | |
|-------|--|---|
| | X1 oscillation mode | External clock input mode |
| 0 | X1 oscillator operating | External clock from EXCLK pin is enabled |
| 1 | X1 oscillator stopped | External clock from EXCLK pin is disabled |

Cautions 1. When setting MSTOP to 1, be sure to confirm that the CPU operates with a clock other than the high-speed system clock. Specifically, set under either of the following conditions.

- When MCS = 0 (when CPU operates with the internal high-speed oscillation clock)

In addition, stop peripheral hardware that is operating on the high-speed system clock before setting MSTOP to 1.

2. Do not clear MSTOP to 0 while bit 6 (OSCSEL) of the clock operation mode select register (OSCCTL) is 0 (I/O port mode).
3. The peripheral hardware cannot operate when the peripheral hardware clock is stopped. To resume the operation of the peripheral hardware after the peripheral hardware clock has been stopped, initialize the peripheral hardware.

(5) Main clock mode register (MCM)

This register selects the main system clock supplied to CPU clock and clock supplied to peripheral hardware clock. MCM can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 5-6. Format of Main Clock Mode Register (MCM)

Address: FFA1H After reset: 00H R/W^{Note}

| | | | | | | | | |
|--------|---|---|---|---|---|------|-----|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | <2> | <1> | <0> |
| MCM | 0 | 0 | 0 | 0 | 0 | XSEL | MCS | MCM0 |

| XSEL | MCM0 | Selection of clock supplied to main system clock and peripheral hardware | |
|------|------|--|--|
| | | Main system clock (f _{XP}) | Peripheral hardware clock (f _{PRS}) |
| 0 | 0 | Internal high-speed oscillation clock (f _{RH}) | Internal high-speed oscillation clock (f _{RH}) |
| 0 | 1 | | High-speed system clock (f _{XH}) |
| 1 | 0 | High-speed system clock (f _{XH}) | Internal high-speed oscillation clock (f _{RH}) |
| 1 | 1 | | High-speed system clock (f _{XH}) |

| MCS | Main system clock status |
|-----|---|
| 0 | Operates with internal high-speed oscillation clock |
| 1 | Operates with high-speed system clock |

Note Bit 1 is read-only.

Cautions 1. XSEL can be changed only once after a reset release.

2. A clock other than f_{PRS} is supplied to the following peripheral functions regardless of the setting of XSEL and MCM0.

- Watchdog timer (operates with internal low-speed oscillation clock)
- When “f_{RL}”, “f_{RL}/2⁷”, or “f_{RL}/2⁹” is selected as the count clock for 8-bit timer H1 (operates with internal low-speed oscillation clock)
- Peripheral hardware selects the external clock as the clock source (Except when the external count clock of TM00 is selected (TI000 pin valid edge))

(6) Oscillation stabilization time counter status register (OSTC)

This is the register that indicates the count status of the X1 clock oscillation stabilization time counter. When X1 clock oscillation starts with the internal high-speed oscillation clock used as the CPU clock, the X1 clock oscillation stabilization time can be checked.

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

When reset is released (reset by $\overline{\text{RESET}}$ input, POC, LVI, and WDT), the STOP instruction and MSTOP (bit 7 of MOC register) = 1 clear OSTC to 00H.

Figure 5-7. Format of Oscillation Stabilization Time Counter Status Register (OSTC)

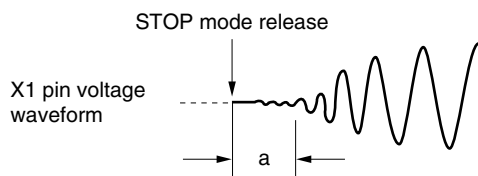
Address: FFA3H After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|--------|--------|--------|--------|--------|
| OSTC | 0 | 0 | 0 | MOST11 | MOST13 | MOST14 | MOST15 | MOST16 |

| MOST11 | MOST13 | MOST14 | MOST15 | MOST16 | Oscillation stabilization time status | |
|--------|--------|--------|--------|--------|---------------------------------------|---------------|
| | | | | | f _x = 10 MHz | |
| 1 | 0 | 0 | 0 | 0 | 2 ¹¹ /f _x min. | 204.8 μs min. |
| 1 | 1 | 0 | 0 | 0 | 2 ¹³ /f _x min. | 819.2 μs min. |
| 1 | 1 | 1 | 0 | 0 | 2 ¹⁴ /f _x min. | 1.64 ms min. |
| 1 | 1 | 1 | 1 | 0 | 2 ¹⁵ /f _x min. | 3.27 ms min. |
| 1 | 1 | 1 | 1 | 1 | 2 ¹⁶ /f _x min. | 6.55 ms min. |

- Cautions**
- After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.
 - The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTC. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.
 - Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTC

Note, therefore, that only the status up to the oscillation stabilization time set by OSTC is set to OSTC 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

(7) 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 OSTS after the STOP mode is released.

When the internal high-speed oscillation clock is selected as the CPU clock, confirm with 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 OSTC.

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

Reset signal generation sets OSTS to 05H.

Figure 5-8. Format of Oscillation Stabilization Time Select Register (OSTS)

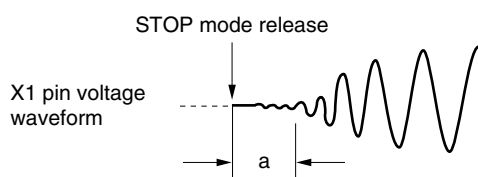
Address: FFA4H After reset: 05H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|-------|-------|-------|
| 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 | |
|------------------|-------|-------|--|------------------------|
| | | | | $f_x = 10 \text{ MHz}$ |
| 0 | 0 | 1 | $2^{11}/f_x$ | 204.8 μs |
| 0 | 1 | 0 | $2^{13}/f_x$ | 819.2 μs |
| 0 | 1 | 1 | $2^{14}/f_x$ | 1.64 ms |
| 1 | 0 | 0 | $2^{15}/f_x$ | 3.27 ms |
| 1 | 0 | 1 | $2^{16}/f_x$ | 6.55 ms |
| Other than above | | | Setting prohibited | |

- Cautions**
- To set the STOP mode when the X1 clock is used as the CPU clock, set OSTS before executing the STOP instruction.
 - 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 OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.
 - Desired OSTC oscillation stabilization time \leq Oscillation stabilization time set by OSTS

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC 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

5.4 System Clock Oscillator

5.4.1 X1 oscillator

The X1 oscillator oscillates with a crystal resonator or ceramic resonator (1 to 10 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.

Figure 5-9 shows an example of the external circuit of the X1 oscillator.

Figure 5-9. Example of External Circuit of X1 Oscillator



Caution When using the X1 oscillator, wire as follows in the area enclosed by the broken lines in the Figure 5-19 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.

Figure 5-10 shows examples of incorrect resonator connection.

Figure 5-10. Examples of Incorrect Resonator Connection (1/2)

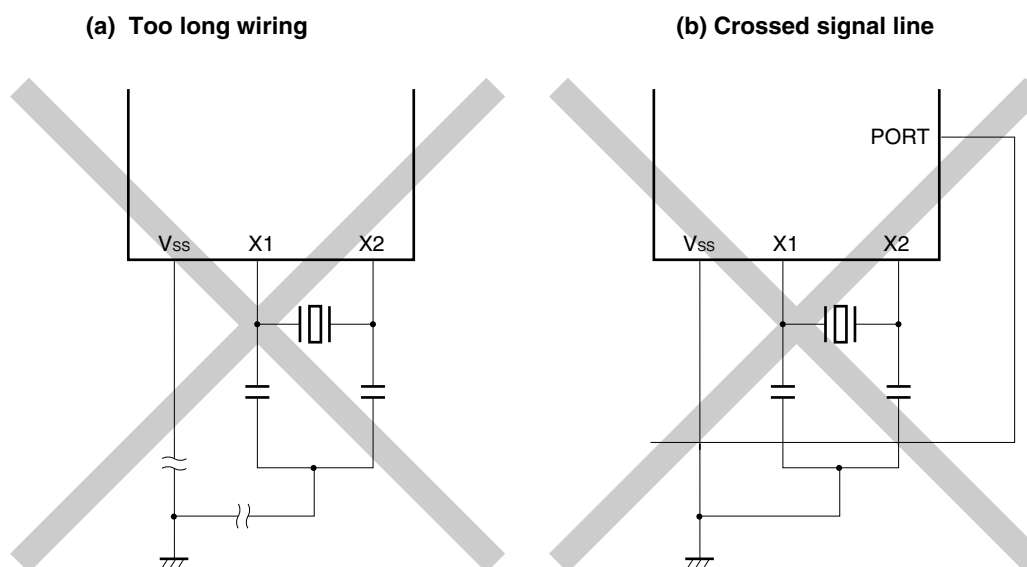
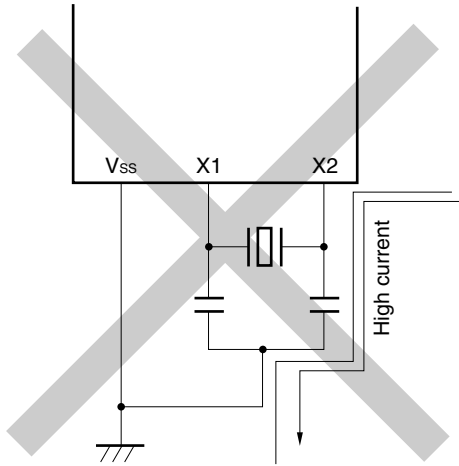
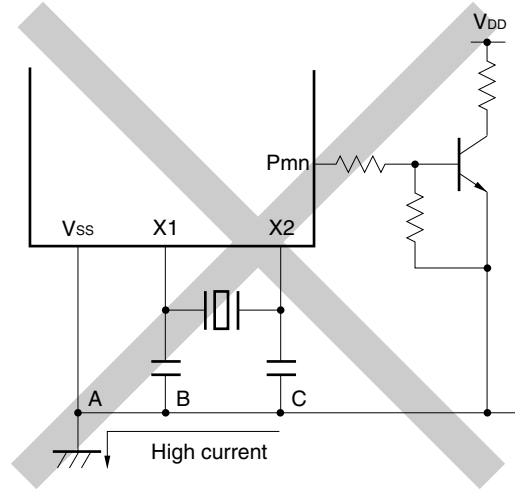


Figure 5-10. Examples of Incorrect Resonator Connection (2/2)

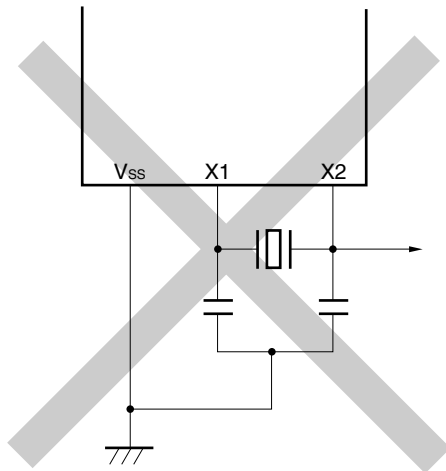
(c) Wiring near high alternating current



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



(e) Signals are fetched



5.4.2 Internal high-speed oscillator

The internal high-speed oscillator is incorporated in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B. Oscillation can be controlled by the internal oscillation mode register (RCM).

After a reset release, the internal high-speed oscillator automatically starts oscillation (8 MHz (TYP.)).

5.4.3 Internal low-speed oscillator

The internal low-speed oscillator is incorporated in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B.

The internal low-speed oscillation clock is only used as the watchdog timer and the clock of 8-bit timer H1. The internal low-speed oscillation clock cannot be used as the CPU clock.

“Can be stopped by software” or “Cannot be stopped” can be selected by the option byte. When “Can be stopped by software” is set, oscillation can be controlled by the internal oscillation mode register (RCM).

After a reset release, the internal low-speed oscillator automatically starts oscillation, and the watchdog timer is driven (240 kHz (TYP.)) if the watchdog timer operation is enabled using the option byte.

5.4.4 Prescaler

The prescaler generates the CPU clock by dividing the main system clock when the main system clock is selected as the clock to be supplied to the CPU.

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_{XP}
 - High-speed system clock f_{XH}
 - X1 clock f_X
 - External main system clock f_{EXCLK}
 - Internal high-speed oscillation clock f_{RH}
- Internal low-speed oscillation clock f_{RL}
- CPU clock f_{CPU}
- Peripheral hardware clock f_{PRS}

The CPU starts operation when the internal high-speed oscillator starts outputting after a reset release in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B, thus enabling the following.

(1) Enhancement of security function

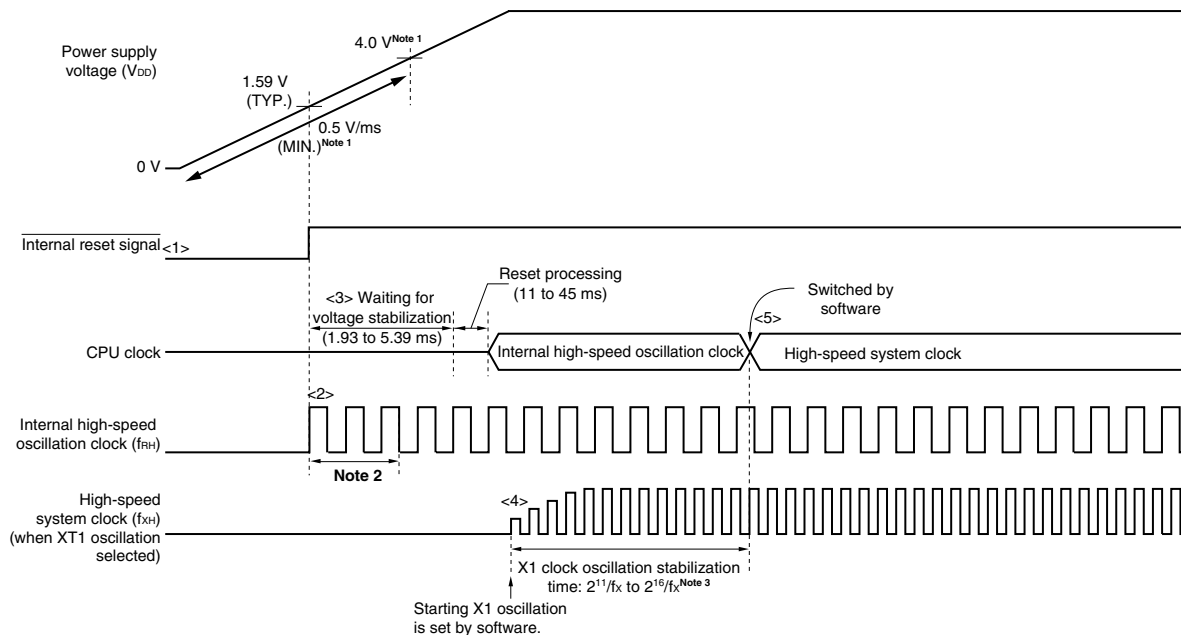
When the X1 clock is set as the CPU clock by the default setting, the device cannot operate if the X1 clock is damaged or badly connected and therefore does not operate after reset is released. However, the start clock of the CPU is the internal high-speed oscillation clock, so the device can be started by the internal high-speed oscillation clock after a reset release. Consequently, the system can be safely shut down by performing a minimum operation, such as acknowledging a reset source by software or performing safety processing when there is a malfunction.

(2) Improvement of performance

Because the CPU can be started without waiting for the X1 clock oscillation stabilization time, the total performance can be improved.

When the power supply voltage is turned on, the clock generator operation is shown in Figure 5-11.

**Figure 5-11. Clock Generator Operation When Power Supply Voltage Is Turned On
(When 1.59 V POC Mode Is Set (Option Byte: POCMODE = 0))**



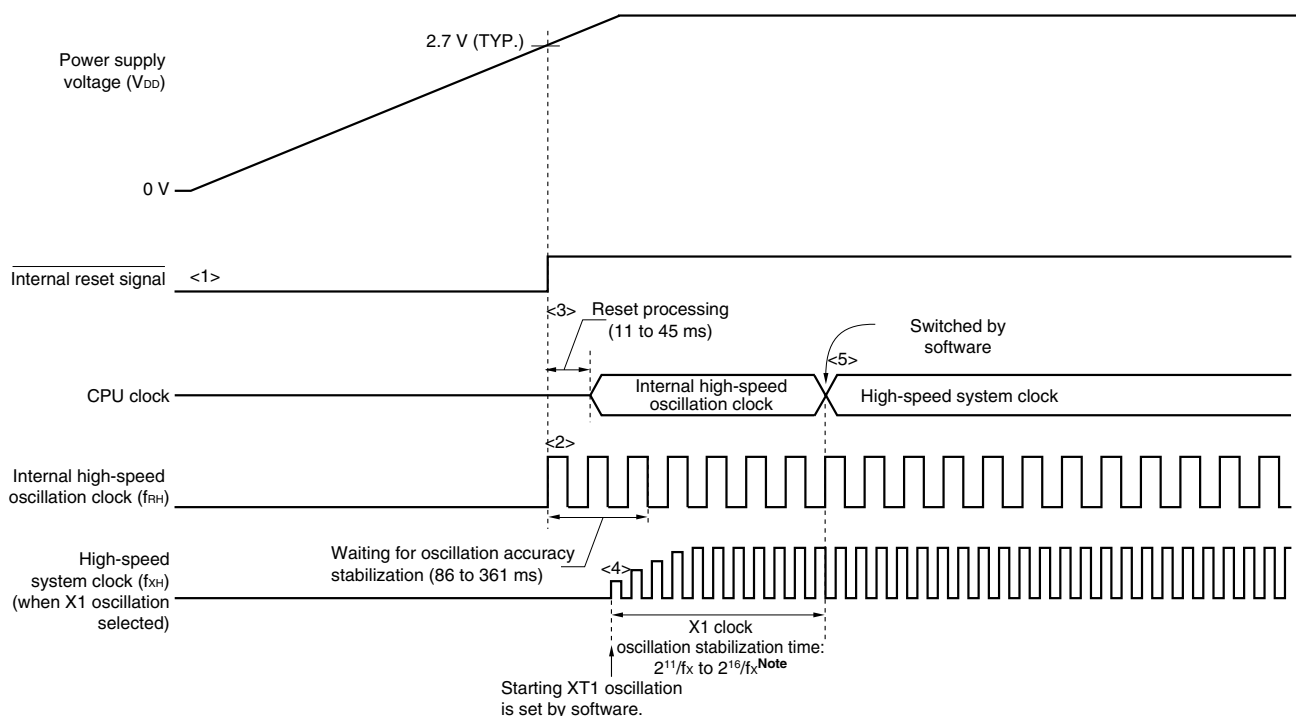
- <1> When the power is turned on, an internal reset signal is generated by the power-on-clear (POC) circuit.
- <2> When the power supply voltage exceeds 1.59 V (TYP.), the reset is released and the internal high-speed oscillator automatically starts oscillation.
- <3> When the power supply voltage rises with a slope of 0.5 V/ms (MIN.), the CPU starts operation on the internal high-speed oscillation clock after the reset is released and after the stabilization times for the voltage of the power supply and regulator have elapsed, and then reset processing is performed.
- <4> Set the start of oscillation of the X1 clock via software (see (1) in 5.6.1 Example of controlling high-speed system clock).
- <5> When switching the CPU clock to the X1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see (3) in 5.6.1 Example of controlling high-speed system clock).

- Notes**
1. If the voltage rises with a slope of less than 0.5 V/ms (MIN.) from power application until the voltage reaches 4.0 V, input a low level to the $\overline{\text{RESET}}$ pin from power application until the voltage reaches 4.0 V, or set the 2.7 V/1.59 V POC mode by using the option byte (POCMODE = 1) (see Figure 5-12). When a low level has been input to the $\overline{\text{RESET}}$ pin until the voltage reaches 4.0 V, the CPU operates with the same timing as <2> and thereafter in Figure 5-11, after the reset has been released by the $\overline{\text{RESET}}$ pin.
 2. The internal voltage stabilization time includes the oscillation accuracy stabilization time of the internal high-speed oscillation clock.
 3. When releasing a reset (above figure) or releasing STOP mode while the CPU is operating on the internal high-speed oscillation clock, confirm the oscillation stabilization time for the X1 clock using the oscillation stabilization time counter status register (OSTC). If the CPU operates on the high-speed system clock (X1 oscillation), set the oscillation stabilization time when releasing STOP mode using the oscillation stabilization time select register (OSTS).

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

Remark While the microcontroller is operating, a clock that is not used as the CPU clock can be stopped via software settings. The internal high-speed oscillation clock and high-speed system clock can be stopped by executing the STOP instruction (see (4) in 5.6.1 Example of controlling high-speed system clock and (3) in 5.6.2 Example of controlling internal high-speed oscillation clock).

**Figure 5-12. Clock Generator Operation When Power Supply Voltage Is Turned On
(When 2.7 V/1.59 V POC Mode Is Set (Option Byte: POCMODE = 1))**



- <1> When the power is turned on, an internal reset signal is generated by the power-on-clear (POC) circuit.
- <2> When the power supply voltage exceeds 2.7 V (TYP.), the reset is released and the internal high-speed oscillator automatically starts oscillation.
- <3> After the reset is released and reset processing is performed, the CPU starts operation on the internal high-speed oscillation clock.
- <4> Set the start of oscillation of the X1 clock via software (see (1) in 5.6.1 **Example of controlling high-speed system clock**).
- <5> When switching the CPU clock to the X1 clock, wait for the clock oscillation to stabilize, and then set switching via software (see (3) in 5.6.1 **Example of controlling high-speed system clock**).

Note When releasing a reset (above figure) or releasing STOP mode while the CPU is operating on the internal high-speed oscillation clock, confirm the oscillation stabilization time for the X1 clock using the oscillation stabilization time counter status register (OSTC). If the CPU operates on the high-speed system clock (X1 oscillation), set the oscillation stabilization time when releasing STOP mode using the oscillation stabilization time select register (OSTS).

- Cautions**
1. A voltage oscillation stabilization time of 1.93 to 5.39 ms is required after the supply voltage reaches 1.59 V (TYP.). If the supply voltage rises from 1.59 V (TYP.) to 2.7 V (TYP.) within 1.93 ms, the power supply oscillation stabilization time of 0 to 5.39 ms is automatically generated before reset processing.
 2. It is not necessary to wait for the oscillation stabilization time when an external clock input from the EXCLK pin is used.

Remark While the microcontroller is operating, a clock that is not used as the CPU clock can be stopped via software settings. The internal high-speed oscillation clock and high-speed system clock can be stopped by executing the STOP instruction (see (4) in 5.6.1 **Example of controlling high-speed system clock** and (3) in 5.6.2 **Example of controlling internal high-speed oscillation clock**).

5.6 Controlling Clock

5.6.1 Example of controlling high-speed system clock

The following two types of high-speed system clocks are available.

- X1 clock: Crystal/ceramic resonator is connected across the X1 and X2 pins.
- External main system clock: External clock is input to the EXCLK pin.

When the high-speed system clock is not used, the X1/P121 and X2/EXCLK/P122 pins can be used as I/O port pins.

Caution The X1/P121 and X2/EXCLK/P122 pins are in the I/O port mode after a reset release.

The following describes examples of setting procedures for the following cases.

- (1) When oscillating X1 clock
- (2) When using external main system clock
- (3) When using high-speed system clock as CPU clock and peripheral hardware clock
- (4) When stopping high-speed system clock

(1) Example of setting procedure when oscillating the X1 clock

- <1> Setting P121/X1 and P122/X2/EXCLK pins and selecting X1 clock or external clock (OSCCTL register)

When EXCLK is cleared to 0 and OSCSEL is set to 1, the mode is switched from port mode to X1 oscillation mode.

| EXCLK | OSCSEL | Operation Mode of High-Speed System Clock Pin | P121/X1 Pin | P122/X2/EXCLK Pin |
|-------|--------|---|--------------------------------------|-------------------|
| 0 | 1 | X1 oscillation mode | Crystal/ceramic resonator connection | |

- <2> Controlling oscillation of X1 clock (MOC register)

If MSTOP is cleared to 0, the X1 oscillator starts oscillating.

- <3> Waiting for the stabilization of the oscillation of X1 clock

Check the OSTC register and wait for the necessary time.

During the wait time, other software processing can be executed with the internal high-speed oscillation clock.

Caution Do not change the value of EXCLK and OSCSEL while the X1 clock is operating.

(2) Example of setting procedure when using the external main system clock

<1> Setting P121/X1 and P122/X2/EXCLK pins and selecting operation mode (OSCCTL register)

When EXCLK and OSCSEL are set to 1, the mode is switched from port mode to external clock input mode.

| EXCLK | OSCSEL | Operation Mode of High-Speed System Clock Pin | P121/X1 Pin | P122/X2/EXCLK Pin |
|-------|--------|---|-------------|----------------------|
| 1 | 1 | External clock input mode | I/O port | External clock input |

<2> Controlling external main system clock input (MOC register)

When MSTOP is cleared to 0, the input of the external main system clock is enabled.

Caution Do not change the value of EXCLK and OSCSEL while the external main system clock is operating.

(3) Example of setting procedure when using high-speed system clock as CPU clock and peripheral hardware clock

<1> Setting high-speed system clock oscillation^{Note}

(See 5.6.1 (1) Example of setting procedure when oscillating the X1 clock and (2) Example of setting procedure when using the external main system clock.)

Note The setting of <1> is not necessary when high-speed system clock is already operating.

<2> Setting the high-speed system clock as the main system clock (MCM register)

When XSEL and MCM0 are set to 1, the high-speed system clock is supplied as the main system clock and peripheral hardware clock.

| XSEL | MCM0 | Selection of Main System Clock and Clock Supplied to Peripheral Hardware | |
|------|------|--|---|
| | | Main System Clock (f_{XP}) | Peripheral Hardware Clock (f_{PRS}) |
| 1 | 1 | High-speed system clock (f_{XH}) | High-speed system clock (f_{XH}) |

Caution If the high-speed system clock is selected as the main system clock, a clock other than the high-speed system clock cannot be set as the peripheral hardware clock.

<3> Selecting the division ratio (PCC register)

To select the CPU clock division ratio, use PCC0, PCC1, and PCC2.

| PCC2 | PCC1 | PCC0 | CPU Clock (f_{CPU}) Selection |
|------------------|------|------|-----------------------------------|
| 0 | 0 | 0 | f_{XP} |
| 0 | 0 | 1 | $f_{XP}/2$ (default) |
| 0 | 1 | 0 | $f_{XP}/2^2$ |
| 0 | 1 | 1 | $f_{XP}/2^3$ |
| 1 | 0 | 0 | $f_{XP}/2^3$ |
| Other than above | | | Setting prohibited |

(4) Example of setting procedure when stopping the high-speed system clock

The high-speed system clock can be stopped in the following two ways.

- Executing the STOP instruction and stopping the X1 oscillation (disabling clock input if the external clock is used)
- Setting MSTOP to 1 and stopping the X1 oscillation (disabling clock input if the external clock is used)

(a) To execute a STOP instruction

<1> Setting to stop peripheral hardware

Stop peripheral hardware that cannot be used in the STOP mode (for peripheral hardware that cannot be used in STOP mode, see **CHAPTER 17 STANDBY FUNCTION**).

<2> Setting the X1 clock oscillation stabilization time after standby release

When the CPU is operating on the X1 clock, set the value of the OSTS register before the STOP instruction is executed.

<3> Executing the STOP instruction

When the STOP instruction is executed, the system is placed in the STOP mode and X1 oscillation is stopped (the input of the external clock is disabled).

(b) To stop X1 oscillation (disabling external clock input) by setting MSTOP to 1

<1> Confirming the CPU clock status (MCM register)

Confirm with MCS that the CPU is operating on a clock other than the high-speed system clock.

When MCS = 1, the high-speed system clock is supplied to the CPU, so change the CPU clock to a clock other than the high-speed system clock.

| MCS | CPU Clock Status |
|-----|---------------------------------------|
| 0 | Internal high-speed oscillation clock |
| 1 | High-speed system clock |

<2> Stopping the high-speed system clock (MOC register)

When MSTOP is set to 1, X1 oscillation is stopped (the input of the external clock is disabled).

Caution Be sure to confirm that MCS = 0 when setting MSTOP to 1. In addition, stop peripheral hardware that is operating on the high-speed system clock.

5.6.2 Example of controlling internal high-speed oscillation clock

The following describes examples of clock setting procedures for the following cases.

- (1) When restarting oscillation of the internal high-speed oscillation clock
- (2) When using internal high-speed oscillation clock as CPU clock, and internal high-speed oscillation clock or high-speed system clock as peripheral hardware clock
- (3) When stopping the internal high-speed oscillation clock

(1) Example of setting procedure when restarting oscillation of the internal high-speed oscillation clock^{Note 1}

<1> Setting restart of oscillation of the internal high-speed oscillation clock (RCM register)

When RSTOP is cleared to 0, the internal high-speed oscillation clock starts operating.

<2> Waiting for the oscillation accuracy stabilization time of internal high-speed oscillation clock (RCM register)

Wait until RSTS is set to 1^{Note 2}.

Notes 1. After a reset release, the internal high-speed oscillator automatically starts oscillating and the internal high-speed oscillation clock is selected as the CPU clock.

2. This wait time is not necessary if high accuracy is not necessary for the CPU clock and peripheral hardware clock.

(2) Example of setting procedure when using internal high-speed oscillation clock as CPU clock, and internal high-speed oscillation clock or high-speed system clock as peripheral hardware clock

- <1> • Restarting oscillation of the internal high-speed oscillation clock^{Note}
 (See 5.6.2 (1) Example of setting procedure when restarting oscillation of the internal high-speed oscillation clock).
- Oscillating the high-speed system clock^{Note}
 (This setting is required when using the high-speed system clock as the peripheral hardware clock. See 5.6.1 (1) Example of setting procedure when oscillating the X1 clock and (2) Example of setting procedure when using the external main system clock.)

Note The setting of <1> is not necessary when the internal high-speed oscillation clock or high-speed system clock is already operating.

- <2> Selecting the clock supplied as the main system clock and peripheral hardware clock (MCM register)
 Set the main system clock and peripheral hardware clock using XSEL and MCM0.

| XSEL | MCM0 | Selection of Main System Clock and Clock Supplied to Peripheral Hardware | |
|------|------|--|--|
| | | Main System Clock (f_{XP}) | Peripheral Hardware Clock (f_{PRS}) |
| 0 | 0 | Internal high-speed oscillation clock (f_{RH}) | Internal high-speed oscillation clock (f_{RH}) |
| 0 | 1 | | Internal high-speed oscillation clock (f_{RH}) |
| 1 | 0 | | High-speed system clock (f_{XH}) |

- <3> Selecting the CPU clock division ratio (PCC register)
 To select the CPU clock division ratio, use PCC0, PCC1, and PCC2.

| PCC2 | PCC1 | PCC0 | CPU Clock (f_{CPU}) Selection |
|------------------|------|------|-----------------------------------|
| 0 | 0 | 0 | f_{XP} |
| 0 | 0 | 1 | $f_{XP}/2$ (default) |
| 0 | 1 | 0 | $f_{XP}/2^2$ |
| 0 | 1 | 1 | $f_{XP}/2^3$ |
| 1 | 0 | 0 | $f_{XP}/2^4$ |
| Other than above | | | Setting prohibited |

(3) Example of setting procedure when stopping the internal high-speed oscillation clock

The internal high-speed oscillation clock can be stopped in the following two ways.

- Executing the STOP instruction to set the STOP mode
- Setting RSTOP to 1 and stopping the internal high-speed oscillation clock

(a) To execute a STOP instruction

<1> Setting of peripheral hardware

Stop peripheral hardware that cannot be used in the STOP mode (for peripheral hardware that cannot be used in STOP mode, see **CHAPTER 17 STANDBY FUNCTION**).

<2> Setting the X1 clock oscillation stabilization time after standby release

When the CPU is operating on the X1 clock, set the value of the OSTS register before the STOP instruction is executed. To operate the CPU immediately after the STOP mode has been released, set MCM0 to 0, switch the CPU clock to the internal high-speed oscillation clock, and check that RSTS is 1.

<3> Executing the STOP instruction

When the STOP instruction is executed, the system is placed in the STOP mode and internal high-speed oscillation clock is stopped.

(b) To stop internal high-speed oscillation clock by setting RSTOP to 1

<1> Confirming the CPU clock status (MCM register)

Confirm with MCS that the CPU is operating on a clock other than the internal high-speed oscillation clock.

When MCS = 0, the internal high-speed oscillation clock is supplied to the CPU, so change the CPU clock to a clock other than the internal high-speed oscillation clock.

| MCS | CPU Clock Status |
|-----|---------------------------------------|
| 0 | Internal high-speed oscillation clock |
| 1 | High-speed system clock |

<2> Stopping the internal high-speed oscillation clock (RCM register)

When RSTOP is set to 1, internal high-speed oscillation clock is stopped.

Caution Be sure to confirm that MCS = 1 when setting RSTOP to 1. In addition, stop peripheral hardware that is operating on the internal high-speed oscillation clock.

5.6.3 Example of controlling internal low-speed oscillation clock

The internal low-speed oscillation clock cannot be used as the CPU clock.

Only the following peripheral hardware can operate with this clock.

- Watchdog timer
- 8-bit timer H1 (if f_{RL} is selected as the count clock)

In addition, the following operation modes can be selected by the option byte.

- Internal low-speed oscillator cannot be stopped
- Internal low-speed oscillator can be stopped by software

The internal low-speed oscillator automatically starts oscillation after a reset release, and the watchdog timer is driven (240 kHz (TYP.)) if the watchdog timer operation has been enabled by the option byte.

(1) Example of setting procedure when stopping the internal low-speed oscillation clock

<1> Setting LSRSTOP to 1 (RCM register)

When LSRSTOP is set to 1, the internal low-speed oscillation clock is stopped.

(2) Example of setting procedure when restarting oscillation of the internal low-speed oscillation clock

<1> Clearing LSRSTOP to 0 (RCM register)

When LSRSTOP is cleared to 0, the internal low-speed oscillation clock is restarted.

Caution If “Internal low-speed oscillator cannot be stopped” is selected by the option byte, oscillation of the internal low-speed oscillation clock cannot be controlled.

5.6.4 Clocks supplied to CPU and peripheral hardware

The following table shows the relation among the clocks supplied to the CPU and peripheral hardware, and setting of registers.

Table 5-3. Clocks Supplied to CPU and Peripheral Hardware, and Register Setting

| Supplied Clock | | XSEL | MCM0 | EXCLK |
|---------------------------------------|---------------------------------------|------|------|-------|
| Clock Supplied to CPU | Clock Supplied to Peripheral Hardware | | | |
| Internal high-speed oscillation clock | | 0 | × | × |
| Internal high-speed oscillation clock | X1 clock | 1 | 0 | 0 |
| | External main system clock | 1 | 0 | 1 |
| X1 clock | | 1 | 1 | 0 |
| External main system clock | | 1 | 1 | 1 |

Remark XSEL: Bit 2 of the main clock mode register (MCM)

MCM0: Bit 0 of MCM

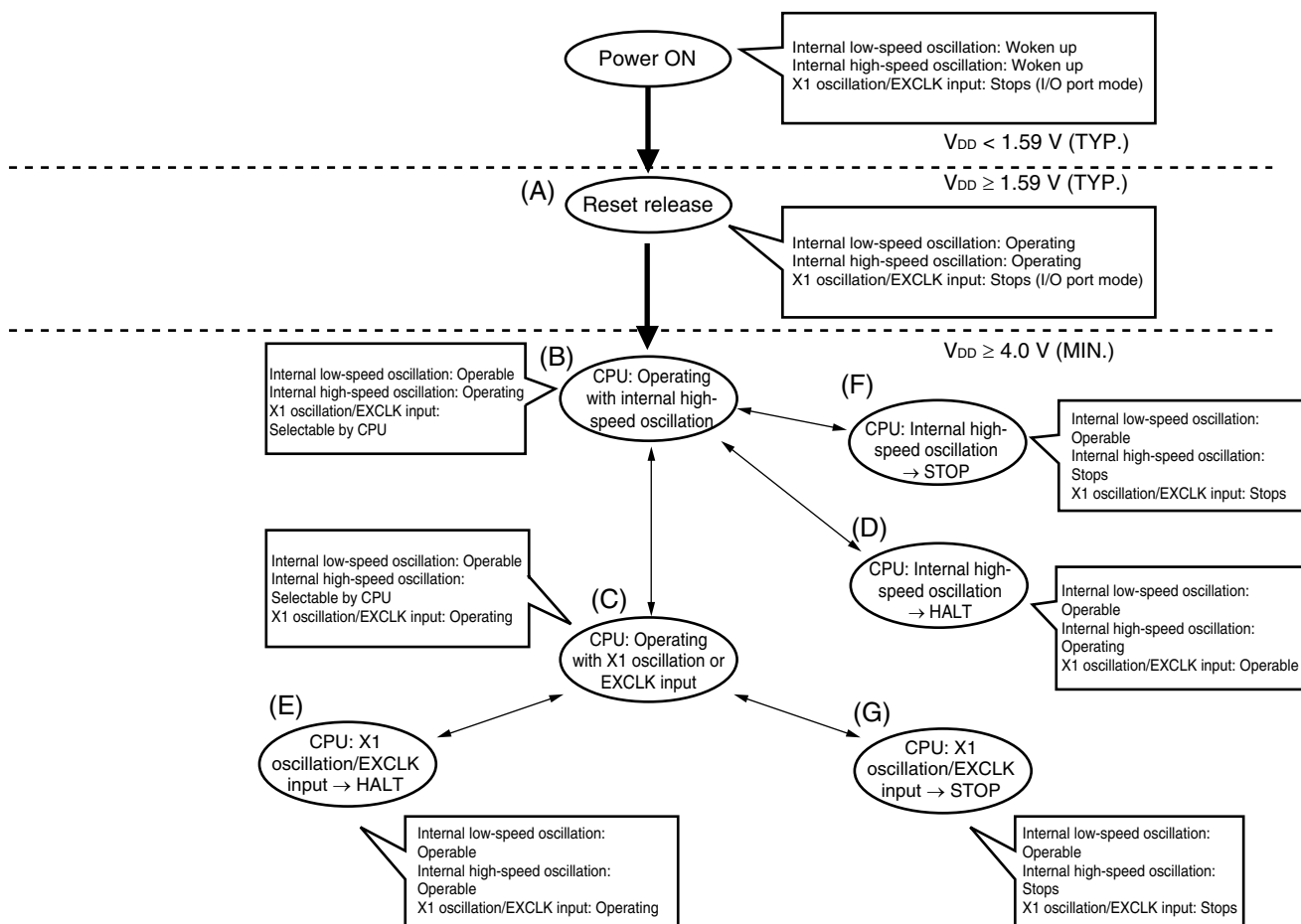
EXCLK: Bit 7 of the clock operation mode select register (OSCCTL)

×: don't care

5.6.5 CPU clock status transition diagram

Figure 5-13 shows the CPU clock status transition diagram of this product.

Figure 5-13. CPU Clock Status Transition Diagram
 (When 1.59 V POC Mode Is Set (Option Byte: POCMODE = 0))



Remark In the 2.7 V/1.59 V POC mode (option byte: POCMODE = 1), the CPU clock status changes to (A) in the above figure when the supply voltage exceeds 2.7 V (TYP.), and to (B) after reset processing (11 to 45 μ s).

Table 5-4 shows transition of the CPU clock and examples of setting the SFR registers.

Table 5-4. CPU Clock Transition and SFR Register Setting Examples (1/2)

(1) CPU operating with internal high-speed oscillation 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 internal high-speed oscillation clock immediately after a reset release (B).)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register Status Transition | EXCLK | OSCSEL | MSTOP | OSTC Register | XSEL | MCM0 |
|--|-------|--------|-------|---------------------|------|------|
| (A) → (B) → (C) (X1 clock: $1 \text{ MHz} \leq f_{XH} \leq 10 \text{ MHz}$) | 0 | 1 | 0 | Must be checked | 1 | 1 |
| (A) → (B) → (C) (external main clock: $1 \text{ MHz} \leq f_{XH} \leq 10 \text{ MHz}$) | 1 | 1 | 0 | Must not be checked | 1 | 1 |

(3) CPU clock changing from internal high-speed oscillation clock (B) to high-speed system clock (C)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register Status Transition | EXCLK | OSCSEL | MSTOP | OSTC Register | XSEL | MCM0 |
|--|-------|--------|-------|---------------------|------|------|
| (B) → (C) (X1 clock: $1 \text{ MHz} \leq f_{XH} \leq 10 \text{ MHz}$) | 0 | 1 | 0 | Must be checked | 1 | 1 |
| (B) → (C) (external main clock: $1 \text{ MHz} \leq f_{XH} \leq 10 \text{ MHz}$) | 1 | 1 | 0 | Must not be checked | 1 | 1 |

Unnecessary if these registers are already set

Unnecessary if the CPU is operating with the high-speed system clock

Remarks 1. (A) to (I) in Table 5-4 correspond to (A) to (I) in Figure 5-13.

2. EXCLK, OSCSEL: Bits 7 and 6 of the clock operation mode select register (OSCCTL)

MSTOP: Bit 7 of the main OSC control register (MOC)

XSEL, MCM0: Bits 2 and 0 of the main clock mode register (MCM)

×: Don't care

Table 5-4. CPU Clock Transition and SFR Register Setting Examples (2/2)

(4) CPU clock changing from high-speed system clock (C) to internal high-speed oscillation clock (B)

(Setting sequence of SFR registers) →

| Setting Flag of SFR Register | RSTOP | RSTS | MCM0 |
|------------------------------|-------|-------------------------|------|
| Status Transition | | | |
| (C) → (B) | 0 | Confirm this flag is 1. | 0 |

Unnecessary if the CPU is operating
with the internal high-speed oscillation clock

(5) • HALT mode (E) set while CPU is operating with internal high-speed oscillation clock (B)

- HALT mode (F) set while CPU is operating with high-speed system clock (C)

| Status Transition | Setting |
|-------------------|----------------------------|
| (B) → (E) | Executing HALT instruction |
| (C) → (F) | |

(6) • STOP mode (H) set while CPU is operating with internal high-speed oscillation clock (B)

- STOP mode (I) set while CPU is operating with high-speed system clock (C)

(Setting sequence) →

| Status Transition | Setting | |
|-------------------|---|----------------------------|
| (B) → (H) | Stopping peripheral functions that cannot operate in STOP mode | Executing STOP instruction |
| (C) → (I) | | |

Remarks 1. (A) to (I) in Table 5-4 correspond to (A) to (I) in Figure 5-13.

- 2.** MCM0: Bit 0 of the main clock mode register (MCM)
 RSTS, RSTOP: Bits 7 and 0 of the internal oscillation mode register (RCM)
 x: Don't care

5.6.6 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 5-5. Changing CPU Clock

| CPU Clock | | Condition Before Change | Processing After Change |
|---------------------------------------|---------------------------------------|---|---|
| Before Change | After Change | | |
| Internal high-speed oscillation clock | X1 clock | Stabilization of X1 oscillation <ul style="list-style-type: none"> • MSTOP = 0, OSCSEL = 1, EXCLK = 0 • After elapse of oscillation stabilization time | Internal high-speed oscillator can be stopped (RSTOP = 1). |
| | External main system clock | Enabling input of external clock from EXCLK pin <ul style="list-style-type: none"> • MSTOP = 0, OSCSEL = 1, EXCLK = 1 | Internal high-speed oscillator can be stopped (RSTOP = 1). |
| X1 clock | Internal high-speed oscillation clock | Oscillation of internal high-speed oscillator <ul style="list-style-type: none"> • RSTOP = 0 | X1 oscillation can be stopped (MSTOP = 1). |
| External main system clock | | | External main system clock input can be disabled (MSTOP = 1). |

5.6.7 Time required for switchover of CPU clock and main system clock

By setting bits 0 to 2 (PCC0 to PCC2) of the processor clock control register (PCC), the CPU clock can be switched the division ratio of the main system clock can be changed.

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

Table 5-6. Time Required for Switchover of CPU Clock and Main System Clock Cycle Division Factor

| Set Value Before Switchover | | | Set Value After Switchover | | | | | | | | | | | | | | |
|-----------------------------|------|------|----------------------------|------|------|-----------|------|------|-----------|------|------|-----------|------|------|-----------|------|------|
| PCC2 | PCC1 | PCC0 | PCC2 | PCC1 | PCC0 | PCC2 | PCC1 | PCC0 | PCC2 | PCC1 | PCC0 | PCC2 | PCC1 | PCC0 | PCC2 | PCC1 | PCC0 |
| | | | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 8 clocks | | | 16 clocks | | | 16 clocks | | | 16 clocks | | | 16 clocks | | |
| 0 | 0 | 1 | | | | 8 clocks | | | 8 clocks | | | 8 clocks | | | 8 clocks | | |
| 0 | 1 | 0 | 4 clocks | | | 4 clocks | | | 4 clocks | | | 4 clocks | | | 4 clocks | | |
| 0 | 1 | 1 | 2 clocks | | | 2 clocks | | | 2 clocks | | | 2 clocks | | | 2 clocks | | |
| 1 | 0 | 0 | 1 clock | | | 1 clock | | | 1 clock | | | 1 clock | | | 1 clock | | |

Remark The number of clocks listed in Table 5-6 is the number of CPU clocks before switchover.

By setting bit 0 (MCM0) of the main clock mode register (MCM), the main system clock can be switched (between the internal high-speed oscillation clock and the high-speed system clock).

The actual switchover operation is not performed immediately after rewriting to MCM0; operation continues on the pre-switchover clock for several clocks (see **Table 5-7**).

Whether the CPU is operating on the internal high-speed oscillation clock or the high-speed system clock can be ascertained using bit 1 (MCS) of MCM.

Table 5-7. Maximum Time Required for Main System Clock Switchover

| Set Value Before Switchover | Set Value After Switchover | |
|-----------------------------|---|---|
| MCM0 | MCM0 | |
| | 0 | 1 |
| 0 | 1 + 2f _{RH} /f _{XH} clock | |
| 1 | | |

Caution When switching the internal high-speed oscillation clock to the high-speed system clock, bit 2 (XSEL) of MCM must be set to 1 in advance. The value of XSEL can be changed only once after a reset release.

Remarks 1. The number of clocks listed in Table 5-7 is the number of main system clocks before switchover.
2. Calculate the number of clocks in Table 5-7 by removing the decimal portion.

Example When switching the main system clock from the internal high-speed oscillation clock to the high-speed system clock (@ oscillation with f_{RH} = 8 MHz, f_{XH} = 10 MHz)

$$1 + 2f_{RH}/f_{XH} = 1 + 2 \times 8/10 = 1 + 2 \times 0.8 = 1 + 1.6 = 2.6 \rightarrow 2 \text{ clocks}$$

5.6.8 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 5-8. 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 |
|---------------------------------------|---|-------------------------------|
| Internal high-speed oscillation clock | MCS = 1 (The CPU is operating on the high-speed system clock) | RSTOP = 1 |
| X1 clock | MCS = 0 (The CPU is operating on the internal high-speed oscillation clock) | MSTOP = 1 |
| External main system clock | | |

5.6.9 Peripheral hardware and source clocks

The following lists peripheral hardware and source clocks incorporated in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B.

Remark The peripheral hardware depends on the product. See 1.6 **Block Diagram** and 1.7 **Outline of Functions**.

Table 5-9. Peripheral Hardware and Source Clocks

| Peripheral Hardware | Source Clock | Peripheral Hardware Clock (f _{PRS}) | Internal Low-Speed Oscillation Clock (f _{RL}) | TM50 Output | External Clock from Peripheral Hardware Pins |
|-------------------------------|--------------------------------|---|---|-------------|--|
| | 16-bit timer/ event counter | 00 | Y | N | N |
| 8-bit timer/ event counter | 50 | Y | N | N | Y (TI50 pin) ^{Note} |
| | 51 | Y | N | N | Y (TI51 pin) ^{Note} |
| 8-Bit timer | H0 | Y | N | Y | N |
| | H1 | Y | Y | N | N |
| Watch timer | | Y | N | N | N |
| Watchdog timer | | N | Y | N | N |
| Buzzer output | | Y | N | N | N |
| Clock output | | Y | N | N | N |
| A/D converter | | Y | N | N | N |
| Serial interface | UART0 | Y | N | Y | N |
| | UART6 | Y | N | Y | N |
| | CSI10 | Y | N | N | Y ($\overline{\text{SCK10}}$ pin) ^{Note} |
| | IIC0 | Y | N | N | Y (SCL0 pin) ^{Note} |

Note Do not start the peripheral hardware operation with the external clock from peripheral hardware pins when the internal high-speed oscillation clock and high-speed system clock are stopped while the CPU operates with the subsystem clock, or when in the STOP mode.

Remark Y: Can be selected, N: Cannot be selected

CHAPTER 6 16-BIT TIMER/EVENT COUNTER 00

6.1 Functions of 16-Bit Timer/Event Counter 00

16-bit timer/event counter 00 has the following functions.

(1) Interval timer

16-bit timer/event counter 00 generates an interrupt request at the preset time interval.

(2) Square-wave output

16-bit timer/event counter 00 can output a square wave with any selected frequency.

(3) External event counter

16-bit timer/event counter 00 can measure the number of pulses of an externally input signal.

(4) One-shot pulse output

16-bit timer event counter 00 can output a one-shot pulse whose output pulse width can be set freely.

(5) PPG output

16-bit timer/event counter 00 can output a rectangular wave whose frequency and output pulse width can be set freely.

(6) Pulse width measurement

16-bit timer/event counter 00 can measure the pulse width of an externally input signal.

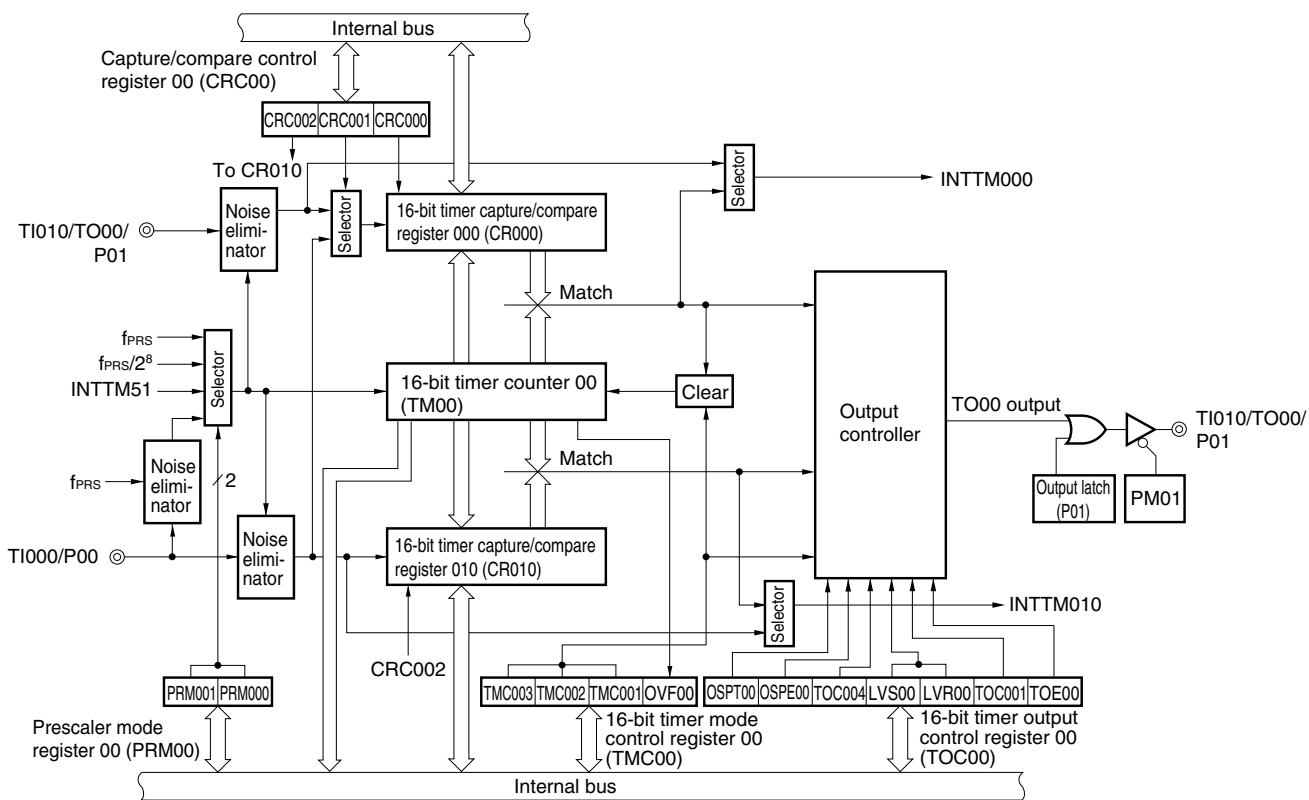
6.2 Configuration of 16-Bit Timer/Event Counter 00

16-bit timer/event counter 00 includes the following hardware.

Table 6-1. Configuration of 16-Bit Timer/Event Counter 00

| Item | Configuration |
|-------------------|--|
| Time/counter | 16-bit timer counter 00 (TM00) |
| Register | 16-bit timer capture/compare registers 000, 010 (CR000, CR010) |
| Timer input | TI000, TI010 |
| Timer output | TO00, output controller |
| Control registers | 16-bit timer mode control register 00 (TMC00) 16-bit timer capture/compare control register 00 (CRC00) 16-bit timer output control register 00 (TOC00) Prescaler mode register 00 (PRM00) Port mode register 0 (PM0) Port register 0 (P0) |

Figure 6-1. Block Diagram of 16-Bit Timer/Event Counter 00



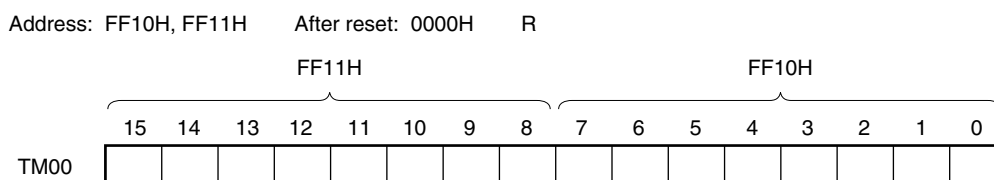
- Cautions**
1. The valid edge of TI010 and timer output (TO00) cannot be used for the P01 pin at the same time. Select either of the functions.
 2. If clearing of bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) to 00 and input of the capture trigger conflict, then the captured data is undefined.
 3. To change the mode from the capture mode to the comparison mode, first clear the TMC003 and TMC002 bits to 00, and then change the setting.
- A value that has been once captured remains stored in CR000 unless the device is reset. If the mode has been changed to the comparison mode, be sure to set a comparison value.

(1) 16-bit timer counter 00 (TM00)

TM00 is a 16-bit read-only register that counts count pulses.

The counter is incremented in synchronization with the rising edge of the count clock.

Figure 6-2. Format of 16-Bit Timer Counter 00 (TM00)



The count value of TM00 can be read by reading TM00 when the value of bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) is other than 00. The value of TM00 is 0000H if it is read when TMC003 and TMC002 = 00.

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

- At reset signal generation
- If TMC003 and TMC002 are cleared to 00
- If the valid edge of the TI000 pin is input in the mode in which the clear & start occurs when inputting the valid edge to the TI000 pin
- If TM00 and CR000 match in the mode in which the clear & start occurs when TM00 and CR000 match
- OSPT00 is set to 1 in one-shot pulse output mode or the valid edge is input to the TI000 pin

Caution Even if TM00 is read, the value is not captured by CR010.

(2) 16-bit timer capture/compare register 000 (CR000), 16-bit timer capture/compare register 010 (CR010)

CR000 and CR010 are 16-bit registers that are used with a capture function or comparison function selected by using CRC00.

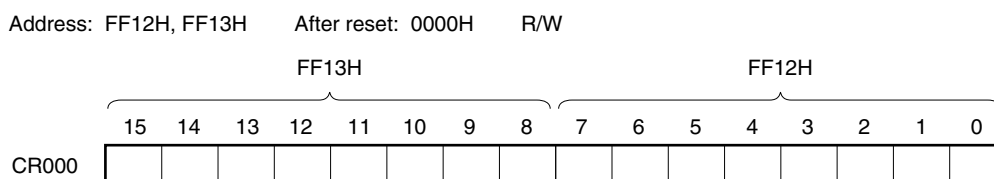
Change the value of CR000 while the timer is stopped (TMC003 and TMC002 = 00).

The value of CR010 can be changed during operation if the value has been set in a specific way. For details, refer to **6.5.1 Rewriting CR010 during TM00 operation.**

These registers can be read or written in 16-bit units.

Reset signal generation clears these registers to 0000H.

Figure 6-3. Format of 16-Bit Timer Capture/Compare Register 000 (CR000)

**(i) When CR000 is used as a compare register**

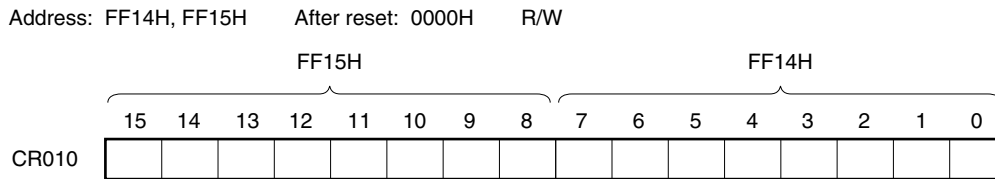
The value set in CR000 is constantly compared with the TM00 count value, and an interrupt request signal (INTTM000) is generated if they match. The value is held until CR000 is rewritten.

Caution CR000 does not perform the capture operation when it is set in the comparison mode, even if a capture trigger is input to it.

(ii) When CR000 is used as a capture register

The count value of TM00 is captured to CR000 when a capture trigger is input.

As the capture trigger, an edge of a phase reverse to that of the TI000 pin or the valid edge of the TI010 pin can be selected by using CRC00 or PRM00.

Figure 6-4. Format of 16-Bit Timer Capture/Compare Register 010 (CR010)**(i) When CR010 is used as a compare register**

The value set in CR010 is constantly compared with the TM00 count value, and an interrupt request signal (INTTM010) is generated if they match.

Caution CR010 does not perform the capture operation when it is set in the comparison mode, even if a capture trigger is input to it.

(ii) When CR010 is used as a capture register

The count value of TM00 is captured to CR010 when a capture trigger is input.

It is possible to select the valid edge of the TI000 pin as the capture trigger. The TI000 pin valid edge is set by PRM00.

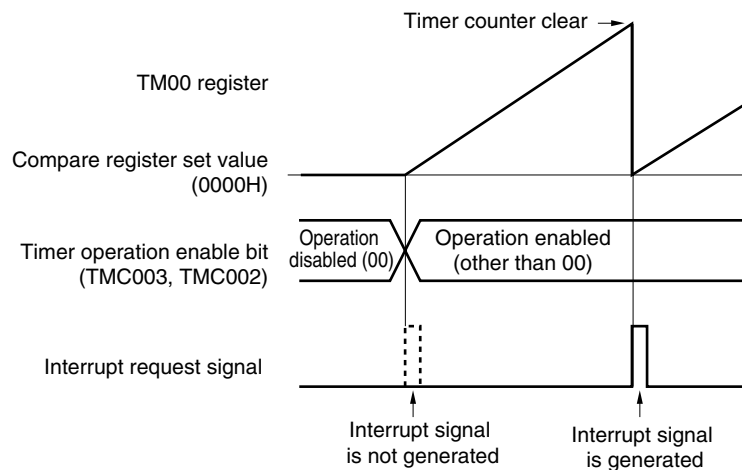
(iii) Setting range when CR000 or CR010 is used as a compare register

When CR000 or CR010 is used as a compare register, set it as shown below.

| Operation | CR000 Register Setting Range | CR010 Register Setting Range |
|---|---|---|
| Operation as interval timer | 0000H < N ≤ FFFFH | 0000H ^{Note} ≤ M ≤ FFFFH |
| Operation as square-wave output | | Normally, this setting is not used. Mask the match interrupt signal (INTTM010). |
| Operation as external event counter | | |
| Operation in the clear & start mode entered by TI000 pin valid edge input | 0000H ^{Note} ≤ N ≤ FFFFH | 0000H ^{Note} ≤ M ≤ FFFFH |
| Operation as free-running timer | | |
| Operation as PPG output | M < N ≤ FFFFH | 0000H ^{Note} ≤ M < N |
| Operation as one-shot pulse output | 0000H ^{Note} ≤ N ≤ FFFFH (N ≠ M) | 0000H ^{Note} ≤ M ≤ FFFFH (M ≠ N) |







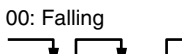
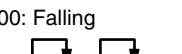
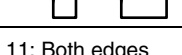
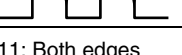

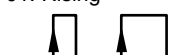
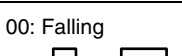
Note When 0000H is set, a match interrupt immediately after the timer operation does not occur and timer output is not changed, and the first match timing is as follows. A match interrupt occurs at the timing when the timer counter (TM00 register) is changed from 0000H to 0001H.

- When the timer counter is cleared due to overflow
- When the timer counter is cleared due to TI000 pin valid edge (when clear & start mode is entered by TI000 pin valid edge input)
- When the timer counter is cleared due to compare match (when clear & start mode is entered by match between TM00 and CR000 (CR000 = other than 0000H, CR010 = 0000H))



- Remarks 1.** N: CR000 register set value, M: CR010 register set value
- 2.** For details of the operation enable bits (bits 3 and 2 (TMC003 and TMC002)), refer to **6.3 (1) 16-bit timer mode control register 00 (TMC00)**.

Table 6-2. Capture Operation of CR000 and CR010

| External Input Signal | TI000 Pin Input  | | TI010 Pin Input  | |
|----------------------------|--|---|---|--|
| | Capture Operation | | | |
| Capture operation of CR000 | CRC001 = 1 TI000 pin input (reverse phase)  | Set values of ES010 and ES000 Position of edge to be captured | CRC001 bit = 0 TI010 pin input  | Set values of ES110 and ES100 Position of edge to be captured |
| | | 01: Rising  | | 01: Rising  |
| | | 00: Falling  | | 00: Falling  |
| | 11: Both edges (cannot be captured) | 11: Both edges  | 11: Both edges  | |
| | Interrupt signal | INTTM000 signal is not generated even if value is captured. | Interrupt signal | INTTM000 signal is generated each time value is captured. |
| Capture operation of CR010 | TI000 pin input ^{Note}  | Set values of ES010 and ES000 Position of edge to be captured | | |
| | | 01: Rising  | | |
| | | 00: Falling  | | |
| | | Interrupt signal | INTTM010 signal is generated each time value is captured. | |

Note The capture operation of CR010 is not affected by the setting of the CRC001 bit.

Caution To capture the count value of the TM00 register to the CR000 register by using the phase reverse to that input to the TI000 pin, the interrupt request signal (INTTM000) is not generated after the value has been captured. If the valid edge is detected on the TI010 pin during this operation, the capture operation is not performed but the INTTM000 signal is generated as an external interrupt signal. To not use the external interrupt, mask the INTTM000 signal.

Remark CRC001: Refer to 6.3 (2) Capture/compare control register 00 (CRC00).
ES110, ES100, ES010, ES000: Refer to 6.3 (4) Prescaler mode register 00 (PRM00).

6.3 Registers Controlling 16-Bit Timer/Event Counter 00

Registers used to control 16-bit timer/event counter 00 are shown below.

- 16-bit timer mode control register 00 (TMC00)
- Capture/compare control register 00 (CRC00)
- 16-bit timer output control register 00 (TOC00)
- Prescaler mode register 00 (PRM00)
- Port mode register 0 (PM0)
- Port register 0 (P0)

(1) 16-bit timer mode control register 00 (TMC00)

TMC00 is an 8-bit register that sets the 16-bit timer/event counter 00 operation mode, TM00 clear mode, and output timing, and detects an overflow.

Rewriting TMC00 is prohibited during operation (when TMC003 and TMC002 = other than 00). However, it can be changed when TMC003 and TMC002 are cleared to 00 (stopping operation) and when OVF00 is cleared to 0.

TMC00 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears TMC00 to 00H.

Caution 16-bit timer/event counter 00 starts operation at the moment TMC003 and TMC002 are set to values other than 00 (operation stop mode), respectively. Set TMC003 and TMC002 to 00 to stop the operation.

Figure 6-5. Format of 16-Bit Timer Mode Control Register 00 (TMC00)

Address: FF86H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|--------|--------|--------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
| TMC00 | 0 | 0 | 0 | 0 | TMC003 | TMC002 | TMC001 | OVF00 |

| TMC003 | TMC002 | Operation enable of 16-bit timer/event counter 00 |
|--------|--------|---|
| 0 | 0 | Disables 16-bit timer/event counter 00 operation. Stops supplying operating clock. Clears 16-bit timer counter 00 (TM00). |
| 0 | 1 | Free-running timer mode |
| 1 | 0 | Clear & start mode entered by TI000 pin valid edge input ^{Note} |
| 1 | 1 | Clear & start mode entered upon a match between TM00 and CR000 |

| TMC001 | Condition to reverse timer output (TO00) |
|--------|---|
| 0 | <ul style="list-style-type: none"> Match between TM00 and CR000 or match between TM00 and CR010 |
| 1 | <ul style="list-style-type: none"> Match between TM00 and CR000 or match between TM00 and CR010 Trigger input of TI000 pin valid edge |

| OVF00 | TM00 overflow flag |
|--|---|
| Clear (0) | Clears OVF00 to 0 or TMC003 and TMC002 = 00 |
| Set (1) | Overflow occurs. |
| OVF00 is set to 1 when the value of TM00 changes from FFFFH to 0000H in all the operation modes (free-running timer mode, clear & start mode entered by TI000 pin valid edge input, and clear & start mode entered upon a match between TM00 and CR000). It can also be set to 1 by writing 1 to OVF00. | |

Note The TI000 pin valid edge is set by bits 5 and 4 (ES010, ES000) of prescaler mode register 00 (PRM00).

(2) Capture/compare control register 00 (CRC00)

CRC00 is the register that controls the operation of CR000 and CR010.

Changing the value of CRC00 is prohibited during operation (when TMC003 and TMC002 = other than 00).

CRC00 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears CRC00 to 00H.

Figure 6-6. Format of Capture/Compare Control Register 00 (CRC00)

Address: FF88H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|--------|--------|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CRC00 | 0 | 0 | 0 | 0 | 0 | CRC002 | CRC001 | CRC000 |

| | |
|--------|--------------------------------|
| CRC002 | CR010 operating mode selection |
| 0 | Operates as compare register |
| 1 | Operates as capture register |

| | |
|--------|--|
| CRC001 | CR000 capture trigger selection |
| 0 | Captures on valid edge of TI010 pin |
| 1 | Captures on valid edge of TI000 pin by reverse phase ^{Note} |

The valid edge of the TI010 and TI000 pin is set by PRM00.
 If ES010 and ES000 are set to 11 (both edges) when CRC001 is 1, the valid edge of the TI000 pin cannot be detected.

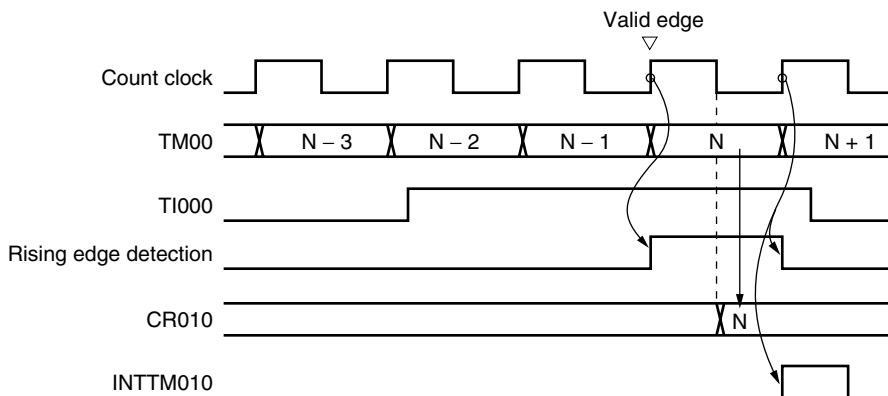
| | |
|--------|--------------------------------|
| CRC000 | CR000 operating mode selection |
| 0 | Operates as compare register |
| 1 | Operates as capture register |

If TMC003 and TMC002 are set to 11 (clear & start mode entered upon a match between TM00 and CR000), be sure to set CRC000 to 0.

Note When the valid edge is detected from the TI010 pin, the capture operation is not performed but the INTTM000 signal is generated as an external interrupt signal.

Caution To ensure that the capture operation is performed properly, the capture trigger requires a pulse two cycles longer than the count clock selected by prescaler mode register 00 (PRM00).

Figure 6-7. Example of CR010 Capture Operation (When Rising Edge Is Specified)



(3) 16-bit timer output control register 00 (TOC00)

TOC00 is an 8-bit register that controls the TO00 output.

TOC00 can be rewritten while only OSPT00 is operating (when TMC003 and TMC002 = other than 00). Rewriting the other bits is prohibited during operation.

However, TOC004 can be rewritten during timer operation as a means to rewrite CR010 (refer to **6.5.1 Rewriting CR010 during TM00 operation**).

TOC00 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears TOC00 to 00H.

Caution Be sure to set TOC00 using the following procedure.

- <1> Set TOC004 and TOC001 to 1.
- <2> Set only TOE00 to 1.
- <3> Set either of LVS00 or LVR00 to 1.

Figure 6-8. Format of 16-Bit Timer Output Control Register 00 (TOC00)

Address: FF89H After reset: 00H R/W

| | | | | | | | | |
|--------|---|--------|--------|--------|-------|-------|--------|-------|
| Symbol | 7 | <6> | <5> | 4 | <3> | <2> | 1 | <0> |
| TOC00 | 0 | OSPT00 | OSPE00 | TOC004 | LVS00 | LVR00 | TOC001 | TOE00 |

| | |
|---|--|
| OSPT00 | One-shot pulse output trigger via software |
| 0 | – |
| 1 | One-shot pulse output |
| The value of this bit is always “0” when it is read. Do not set this bit to 1 in a mode other than the one-shot pulse output mode. If it is set to 1, TM00 is cleared and started. | |

| | |
|--|---|
| OSPE00 | One-shot pulse output operation control |
| 0 | Successive pulse output |
| 1 | One-shot pulse output |
| One-shot pulse output operates correctly in the free-running timer mode or clear & start mode entered by T1000 pin valid edge input. The one-shot pulse cannot be output in the clear & start mode entered upon a match between TM00 and CR000. | |

| | |
|--|---|
| TOC004 | TO00 output control on match between CR010 and TM00 |
| 0 | Disables inversion operation |
| 1 | Enables inversion operation |
| The interrupt signal (INTTM010) is generated even when TOC004 = 0. | |

| | | |
|--|-------|--|
| LVS00 | LVR00 | Setting of TO00 output status |
| 0 | 0 | No change |
| 0 | 1 | Initial value of TO00 output is low level (TO00 output is cleared to 0). |
| 1 | 0 | Initial value of TO00 output is high level (TO00 output is set to 1). |
| 1 | 1 | Setting prohibited |
| <ul style="list-style-type: none"> LVS00 and LVR00 can be used to set the initial value of the TO00 output level. If the initial value does not have to be set, leave LVS00 and LVR00 as 00. Be sure to set LVS00 and LVR00 when TOE00 = 1. LVS00, LVR00, and TOE00 being simultaneously set to 1 is prohibited. LVS00 and LVR00 are trigger bits. By setting these bits to 1, the initial value of the TO00 output level can be set. Even if these bits are cleared to 0, TO00 output is not affected. The values of LVS00 and LVR00 are always 0 when they are read. For how to set LVS00 and LVR00, refer to 6.5.2 Setting LVS00 and LVR00. The actual TO00/TI010/P01 pin output is determined depending on PM01 and P01, besides TO00 output. | | |

| | |
|--|---|
| TOC001 | TO00 output control on match between CR000 and TM00 |
| 0 | Disables inversion operation |
| 1 | Enables inversion operation |
| The interrupt signal (INTTM000) is generated even when TOC001 = 0. | |

| | |
|-------|--|
| TOE00 | TO00 output control |
| 0 | Disables output (TO00 output fixed to low level) |
| 1 | Enables output |

(4) Prescaler mode register 00 (PRM00)

PRM00 is the register that sets the TM00 count clock and TI000 and TI010 pin input valid edges.

Rewriting PRM00 is prohibited during operation (when TMC003 and TMC002 = other than 00).

PRM00 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears PRM00 to 00H.

- Cautions**
- 1. Do not apply the following setting when setting the PRM001 and PRM000 bits to 11 (to specify the valid edge of the TI000 pin as a count clock).**
 - Clear & start mode entered by the TI000 pin valid edge
 - Setting the TI000 pin as a capture trigger
 - 2. If the operation of the 16-bit timer/event counter 00 is enabled when the TI000 or TI010 pin is at high level and when the valid edge of the TI000 or TI010 pin is specified to be the rising edge or both edges, the high level of the TI000 or TI010 pin is detected as a rising edge. Note this when the TI000 or TI010 pin is pulled up. However, the rising edge is not detected when the timer operation has been once stopped and then is enabled again.**
 - 3. The valid edge of TI010 and timer output (TO00) cannot be used for the P34 pin at the same time. Select either of the functions.**

Figure 6-9. Format of Prescaler Mode Register 00 (PRM00)

Address: FF87H After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|-------|-------|-------|-------|---|---|--------|--------|
| PRM00 | ES110 | ES100 | ES010 | ES000 | 0 | 0 | PRM001 | PRM000 |

| ES110 | ES100 | TI010 pin valid edge selection |
|-------|-------|--------------------------------|
| 0 | 0 | Falling edge |
| 0 | 1 | Rising edge |
| 1 | 0 | Setting prohibited |
| 1 | 1 | Both falling and rising edges |

| ES010 | ES000 | TI000 pin valid edge selection |
|-------|-------|--------------------------------|
| 0 | 0 | Falling edge |
| 0 | 1 | Rising edge |
| 1 | 0 | Setting prohibited |
| 1 | 1 | Both falling and rising edges |

| PRM001 | PRM000 | Count clock selection | | |
|--------|--------|--|---------------------------|---------------------------|
| | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ |
| 0 | 0 | f_{PRS} | 2 MHz | 5 MHz |
| 0 | 1 | $f_{PRS}/2^8$ | 7.81 kHz | 19.53 kHz |
| 1 | 0 | INTTM51 | | |
| 1 | 1 | TI000 valid edge ^{Notes 1, 2} | | |

- Notes**
1. The external clock from the TI000 pin requires a pulse longer than twice the cycle of the peripheral hardware clock (f_{PRS}).
 2. Do not start timer operation with the external clock from the TI000 pin when in the STOP mode.

Remark f_{PRS} : Peripheral hardware clock frequency

(5) Port mode register 0 (PM0)

This register sets port 0 input/output in 1-bit units.

When using the P01/TO00/TI010 pin for timer output, set the output latches of PM01 to 0.

When using the P00/TI000 and P01/TI010/TO00 pins for timer input, set PM00 and PM01 to 1. At this time, the output latches of P00 and P01 may be 0 or 1.

PM0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets PM0 to FFH.

Figure 6-10. Format of Port Mode Register 0 (PM0)

Address: FF20H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM0 | 1 | 1 | 1 | 1 | 1 | 1 | PM01 | PM00 |

| | |
|------|---------------------------------------|
| PM0n | P0n pin I/O mode selection (n = 0, 1) |
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

6.4 Operation of 16-Bit Timer/Event Counter 00

6.4.1 Interval timer operation

If bits 3 and 2 (TMC003 and TMC002) of the 16-bit timer mode control register (TMC00) are set to 11 (clear & start mode entered upon a match between TM00 and CR000), the count operation is started in synchronization with the count clock.

When the value of TM00 later matches the value of CR000, TM00 is cleared to 0000H and a match interrupt signal (INTTM000) is generated. This INTTM000 signal enables TM00 to operate as an interval timer.

Remarks 1. For the setting of I/O pins, refer to **6.3 (5) Port mode register 0 (PM0)**.

2. For how to enable the INTTM000 interrupt, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

Figure 6-11. Block Diagram of Interval Timer Operation

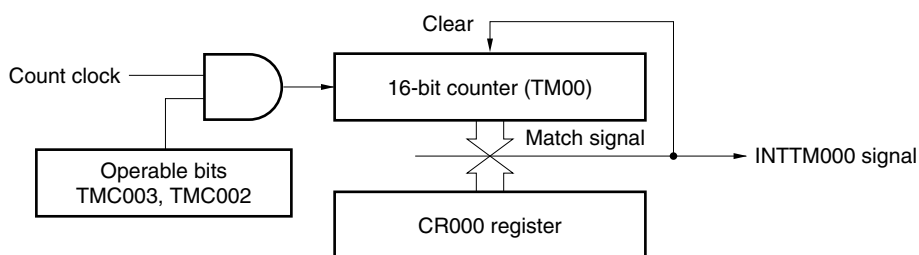


Figure 6-12. Basic Timing Example of Interval Timer Operation

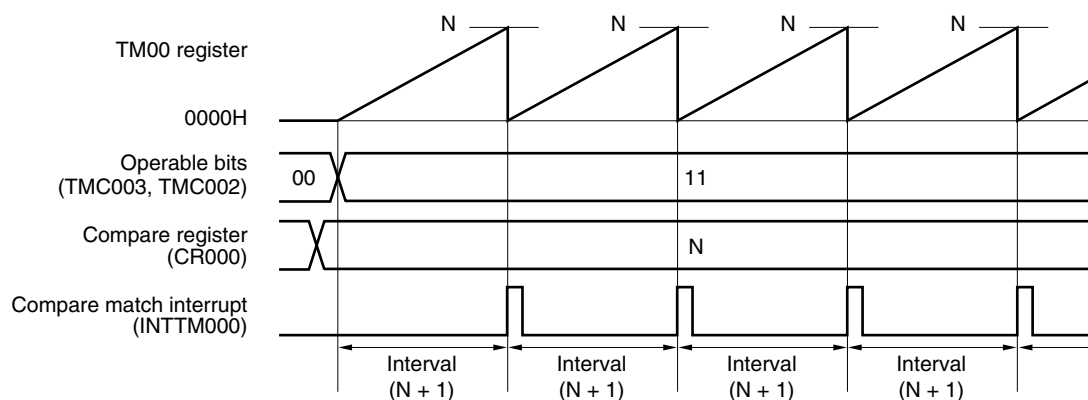
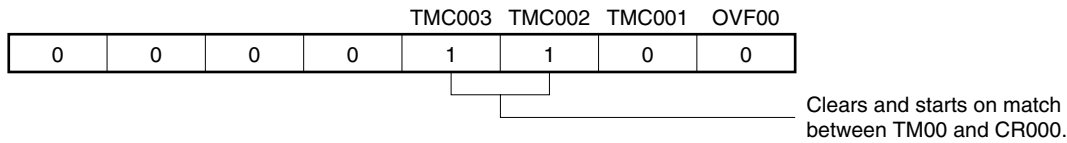
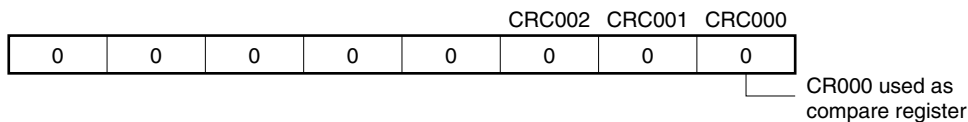


Figure 6-13. Example of Register Settings for Interval Timer Operation

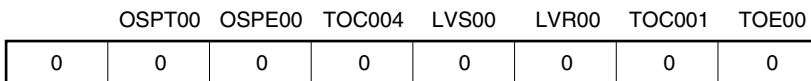
(a) 16-bit timer mode control register 00 (TMC00)



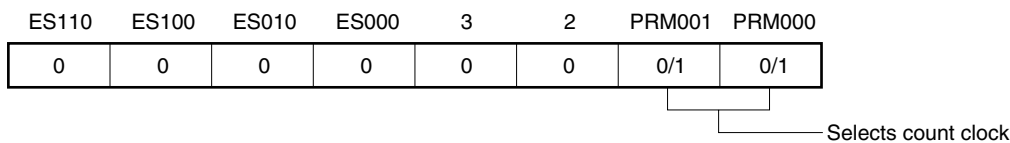
(b) Capture/compare control register 00 (CRC00)



(c) 16-bit timer output control register 00 (TOC00)



(d) Prescaler mode register 00 (PRM00)



(e) 16-bit timer counter 00 (TM00)

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

If M is set to CR000, the interval time is as follows.

- Interval time = (M + 1) × Count clock cycle

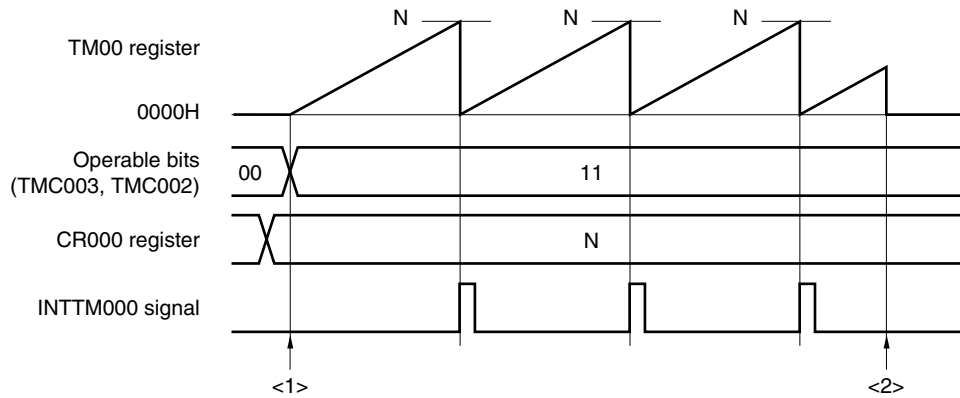
Setting CR000 to 0000H is prohibited.

(g) 16-bit capture/compare register 010 (CR010)

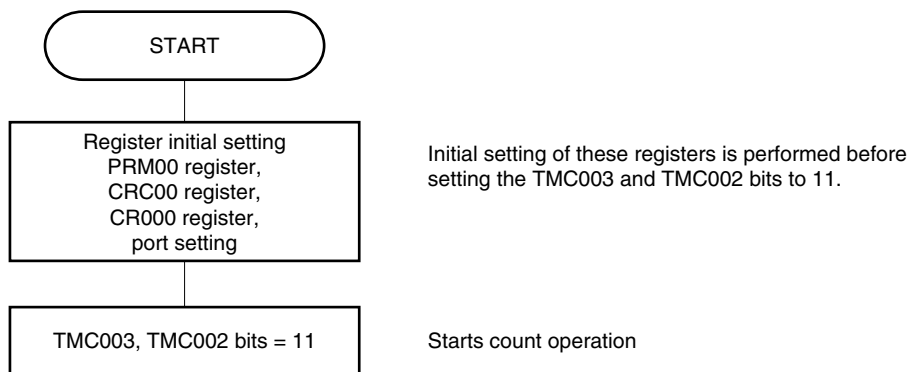
Usually, CR010 is not used for the interval timer function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

Therefore, mask the interrupt request by using the interrupt mask flag (TMMK010).

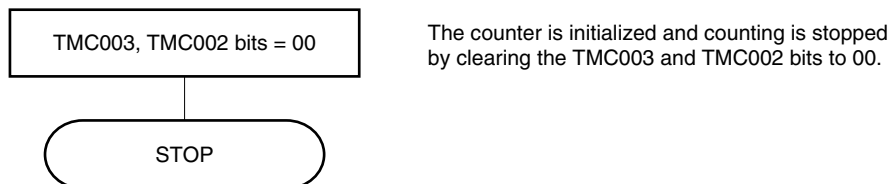
Figure 6-14. Example of Software Processing for Interval Timer Function



<1> Count operation start flow



<2> Count operation stop flow



6.4.2 Square-wave output operation

When 16-bit timer/event counter 00 operates as an interval timer (refer to 6.4.1), a square wave can be output from the TO00 pin by setting the 16-bit timer output control register 00 (TOC00) to 03H.

When TMC003 and TMC002 are set to 11 (count clear & start mode entered upon a match between TM00 and CR000), the counting operation is started in synchronization with the count clock.

When the value of TM00 later matches the value of CR000, TM00 is cleared to 0000H, an interrupt signal (INTTM000) is generated, and TO00 output is inverted. This TO00 output that is inverted at fixed intervals enables TO0n to output a square wave.

Remarks 1. For the setting of I/O pins, refer to 6.3 (5) **Port mode register 0 (PM0)**.

2. For how to enable the INTTM000 signal interrupt, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

Figure 6-15. Block Diagram of Square-Wave Output Operation

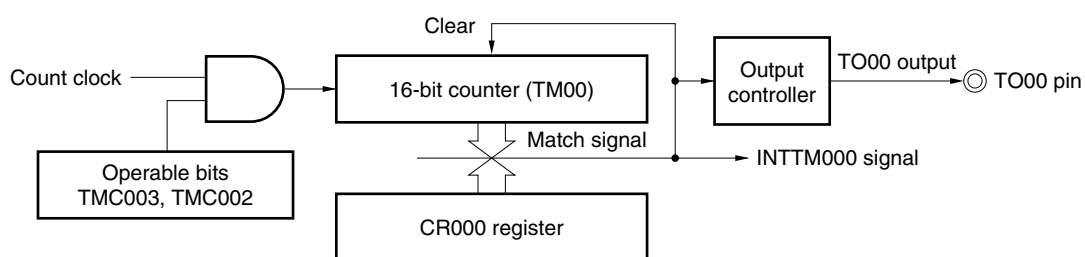


Figure 6-16. Basic Timing Example of Square-Wave Output Operation

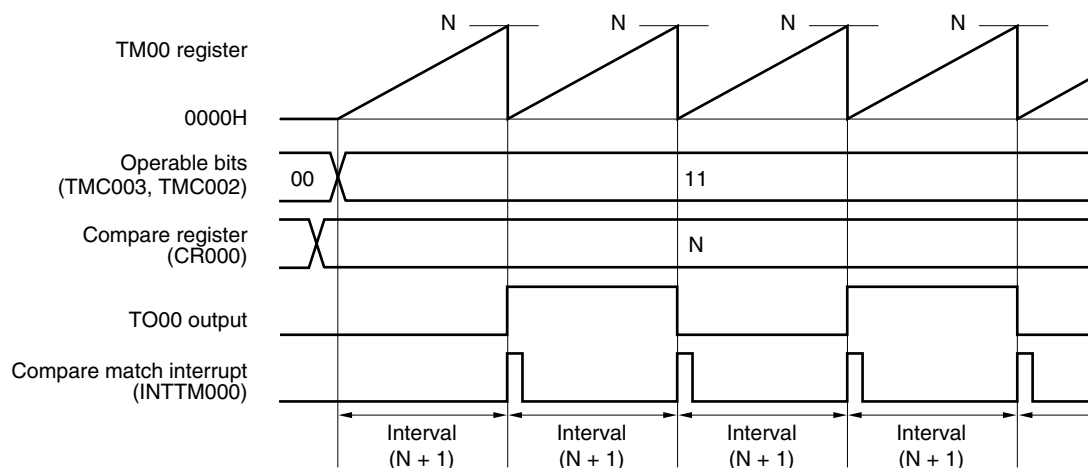


Figure 6-17. Example of Register Settings for Square-Wave Output Operation (1/2)

(a) 16-bit timer mode control register 00 (TMC00)

| | | | | TMC003 | TMC002 | TMC001 | OVF00 |
|---|---|---|---|--------|--------|--------|-------|
| 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |

Clears and starts on match between TM00 and CR000.

(b) Capture/compare control register 00 (CRC00)

| | | | | CRC002 | CRC001 | CRC000 |
|---|---|---|---|--------|--------|--------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

CR000 used as compare register

(c) 16-bit timer output control register 00 (TOC00)

| OSPT00 | OSPE00 | TOC004 | LVS00 | LVR00 | TOC001 | TOE00 |
|--------|--------|--------|-------|-------|--------|-------|
| 0 | 0 | 0 | 0/1 | 0/1 | 1 | 1 |

Enables TO00 output.

Inverts TO00 output on match between TM00 and CR000.

Specifies initial value of TO00 output F/F

(d) Prescaler mode register 00 (PRM00)

| ES110 | ES100 | ES010 | ES000 | 3 | 2 | PRM001 | PRM000 |
|-------|-------|-------|-------|---|---|--------|--------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 |

Selects count clock

Figure 6-17. Example of Register Settings for Square-Wave Output Operation (2/2)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

If M is set to CR000, the interval time is as follows.

- Square wave frequency = $1 / [2 \times (M + 1) \times \text{Count clock cycle}]$

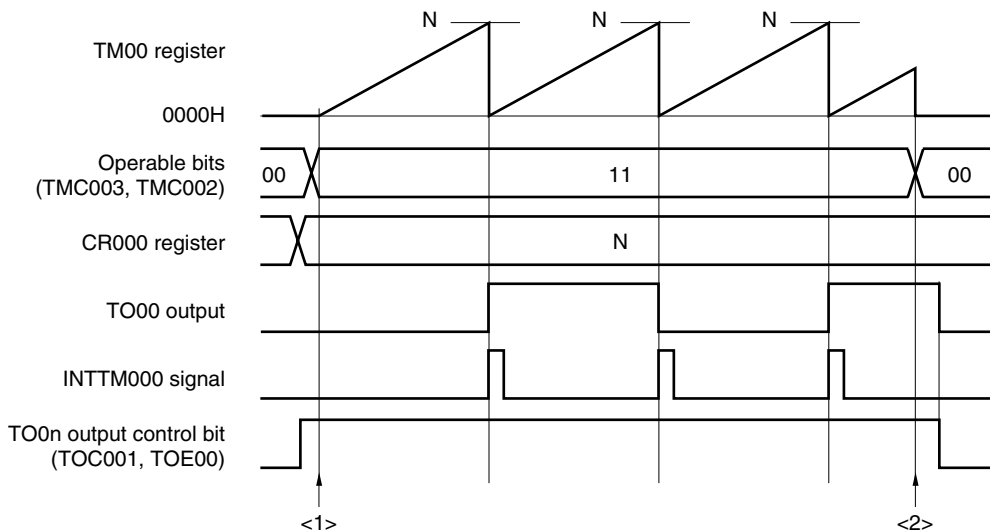
Setting CR000 to 0000H is prohibited.

(g) 16-bit capture/compare register 010 (CR010)

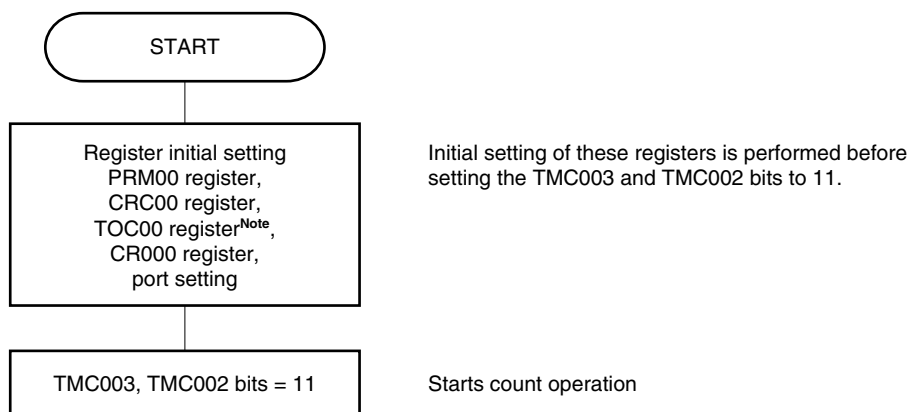
Usually, CR010 is not used for the square-wave output function. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

Therefore, mask the interrupt request by using the interrupt mask flag (TMMK010).

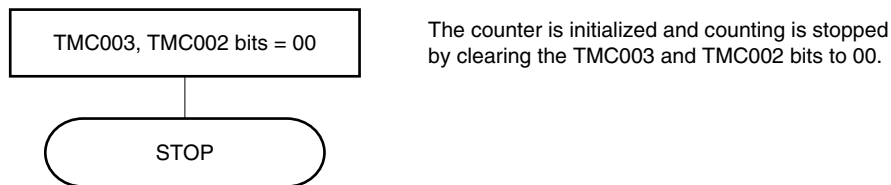
Figure 6-18. Example of Software Processing for Square-Wave Output Function



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, refer to 6.3 (3) 16-bit timer output control register 00 (TOC00).

6.4.3 External event counter operation

When bits 1 and 0 (PRM001 and PRM000) of the prescaler mode register 00 (PRM00) are set to 11 (for counting up with the valid edge of the TI000 pin) and bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) are set to 11, the valid edge of an external event input is counted, and a match interrupt signal indicating matching between TM00 and CR000 (INTTM000) is generated.

To input the external event, the TI000 pin is used. Therefore, the timer/event counter cannot be used as an external event counter in the clear & start mode entered by the TI000 pin valid edge input (when TMC003 and TMC002 = 10).

The INTTM000 signal is generated with the following timing.

- Timing of generation of INTTM000 signal (second time or later)
= Number of times of detection of valid edge of external event × (Set value of CR000 + 1)

However, the first match interrupt immediately after the timer/event counter has started operating is generated with the following timing.

- Timing of generation of INTTM000 signal (first time only)
= Number of times of detection of valid edge of external event input × (Set value of CR000 + 2)

To detect the valid edge, the signal input to the TI000 pin is sampled during the clock cycle of f_{PRS} . The valid edge is not detected until it is detected two times in a row. Therefore, a noise with a short pulse width can be eliminated.

Remarks 1. For the setting of I/O pins, refer to **6.3 (5) Port mode register 0 (PM0)**.

2. For how to enable the INTTM000 signal interrupt, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

Figure 6-19. Block Diagram of External Event Counter Operation

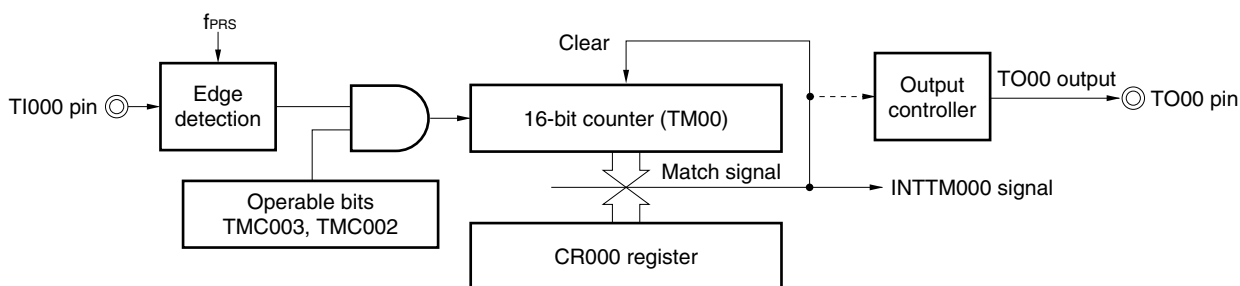
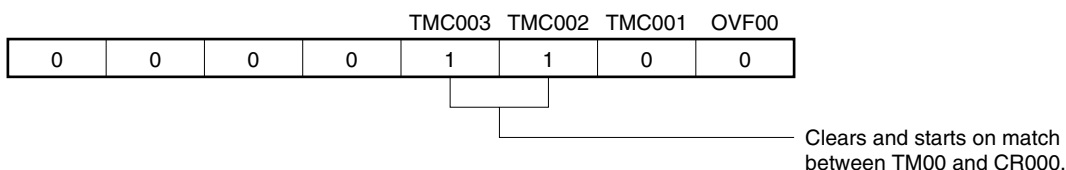
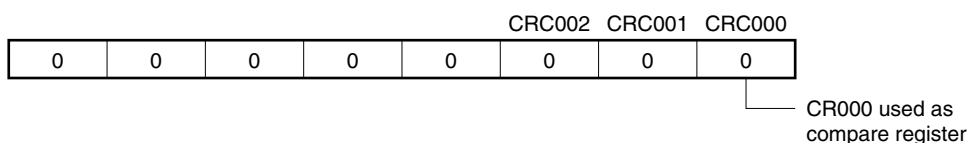


Figure 6-20. Example of Register Settings in External Event Counter Mode (1/2)

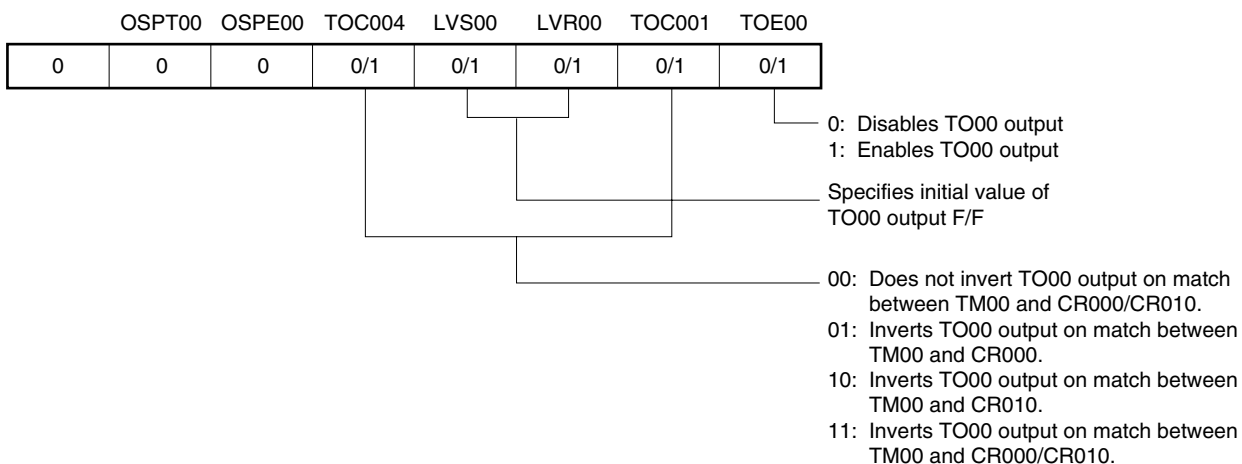
(a) 16-bit timer mode control register 00 (TMC00)



(b) Capture/compare control register 00 (CRC00)



(c) 16-bit timer output control register 00 (TOC00)



(d) Prescaler mode register 00 (PRM00)

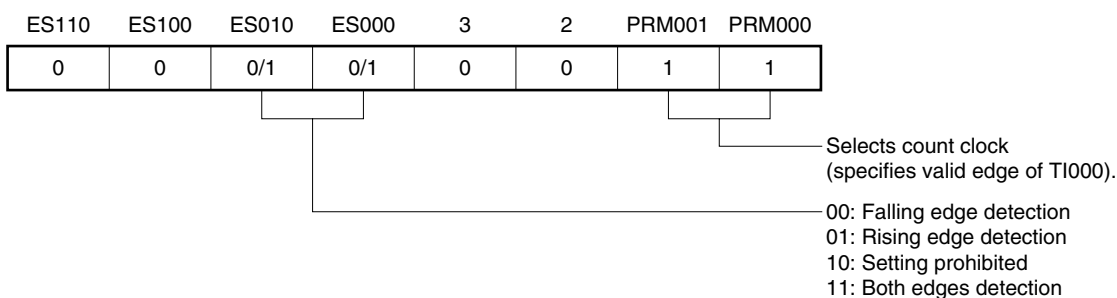


Figure 6-20. Example of Register Settings in External Event Counter Mode (2/2)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

If M is set to CR000, the interrupt signal (INTTM000) is generated when the number of external events reaches (M + 1).

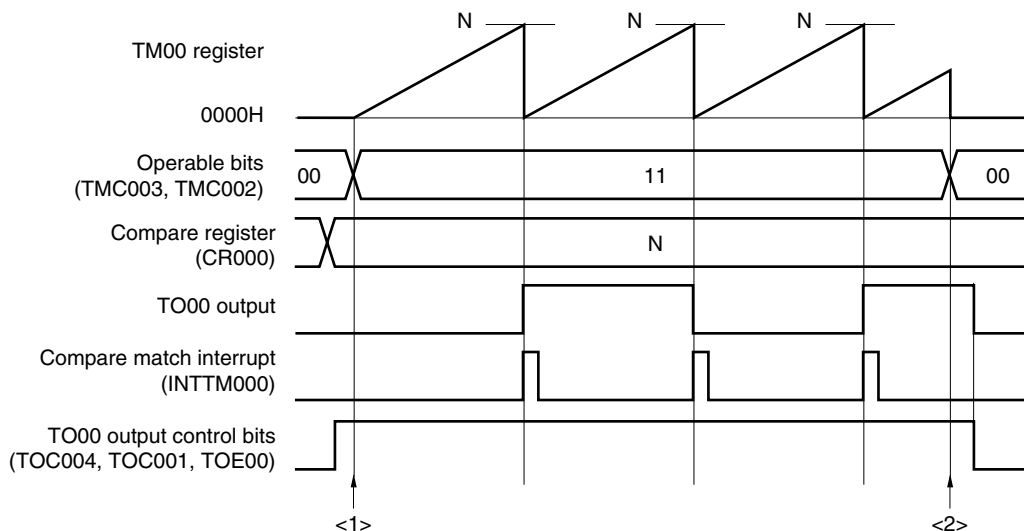
Setting CR000 to 0000H is prohibited.

(g) 16-bit capture/compare register 010 (CR010)

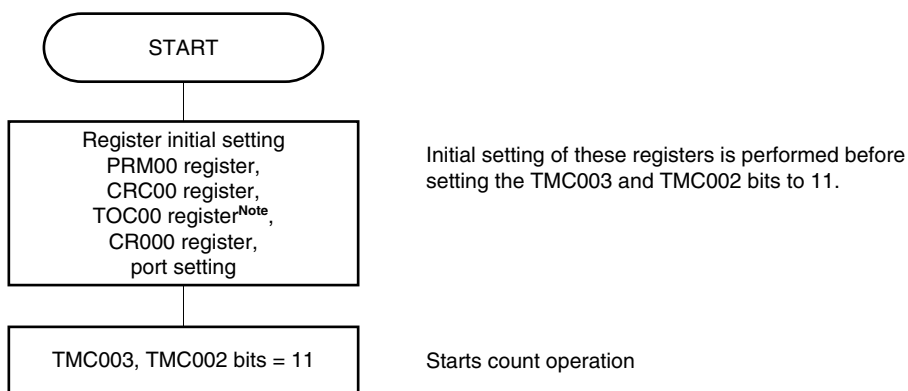
Usually, CR010 is not used in the external event counter mode. However, a compare match interrupt (INTTM010) is generated when the set value of CR010 matches the value of TM00.

Therefore, mask the interrupt request by using the interrupt mask flag (TMMK010).

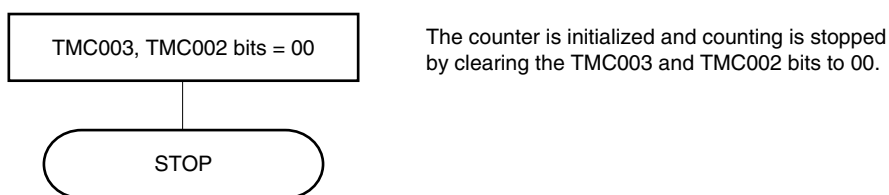
Figure 6-21. Example of Software Processing in External Event Counter Mode



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, refer to 6.3 (3) 16-bit timer output control register 00 (TOC00).

6.4.4 Operation in clear & start mode entered by TI000 pin valid edge input

When bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) are set to 10 (clear & start mode entered by the TI000 pin valid edge input) and the count clock (set by PRM00) is supplied to the timer/event counter, TM00 starts counting up. When the valid edge of the TI000 pin is detected during the counting operation, TM00 is cleared to 0000H and starts counting up again. If the valid edge of the TI000 pin is not detected, TM00 overflows and continues counting.

The valid edge of the TI000 pin is a cause to clear TM00. Starting the counter is not controlled immediately after the start of the operation.

CR000 and CR010 are used as compare registers and capture registers.

(a) When CR000 and CR010 are used as compare registers

Signals INTTM000 and INTTM010 are generated when the value of TM00 matches the value of CR000 and CR010.

(b) When CR000 and CR010 are used as capture registers

The count value of TM00 is captured to CR000 and the INTTM000 signal is generated when the valid edge is input to the TI010 pin (or when the phase reverse to that of the valid edge is input to the TI000 pin).

When the valid edge is input to the TI000 pin, the count value of TM00 is captured to CR010 and the INTTM010 signal is generated. As soon as the count value has been captured, the counter is cleared to 0000H.

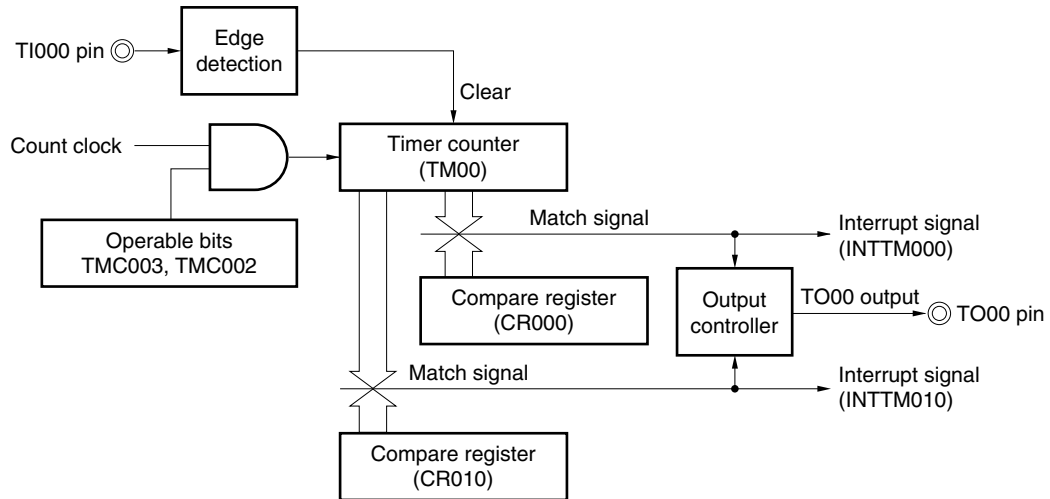
Caution Do not set the count clock as the valid edge of the TI000 pin (PRM001 and PRM000 = 11). When PRM001 and PRM000 = 11, TM00 is cleared.

Remarks 1. For the setting of the I/O pins, refer to 6.3 (5) Port mode register 0 (PM0).

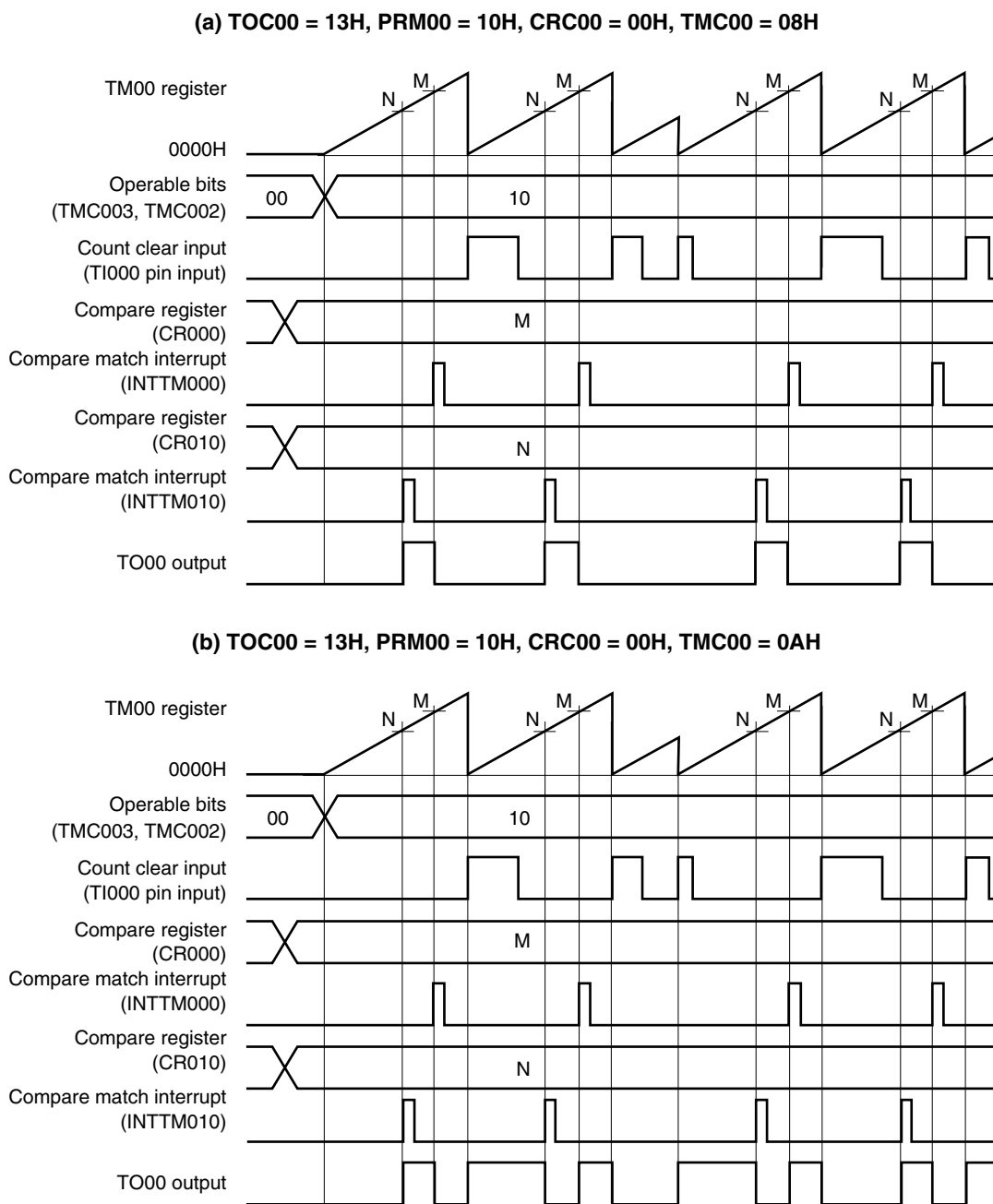
2. For how to enable the INTTM000 signal interrupt, refer to CHAPTER 16 INTERRUPT FUNCTIONS.

(1) Operation in clear & start mode entered by TI000 pin valid edge input
(CR000: compare register, CR010: compare register)

Figure 6-22. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Compare Register, CR010: Compare Register)



**Figure 6-23. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Compare Register, CR010: Compare Register)**

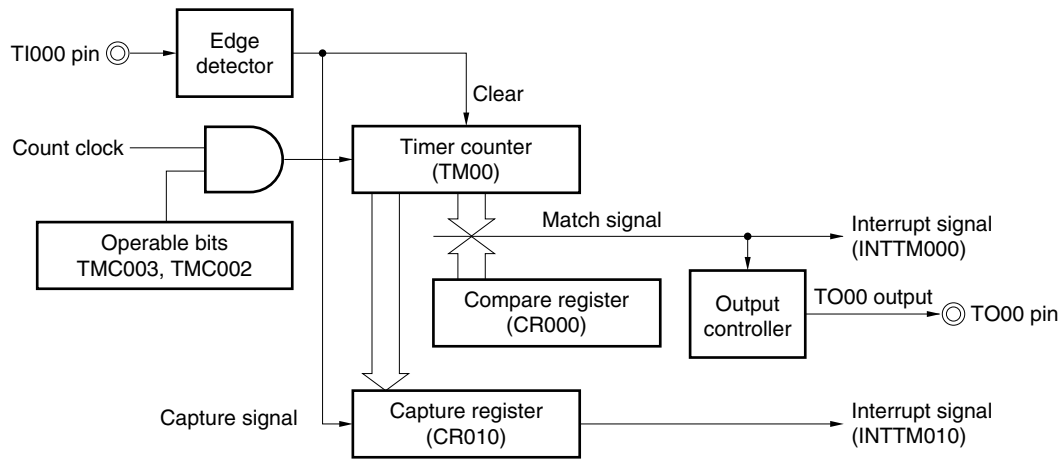


(a) and (b) differ as follows depending on the setting of bit 1 (TMC001) of the 16-bit timer mode control register 00 (TMC00).

- (a) The TO00 output level is inverted when TM00 matches a compare register.
- (b) The TO00 output level is inverted when TM00 matches a compare register or when the valid edge of the TI000 pin is detected.

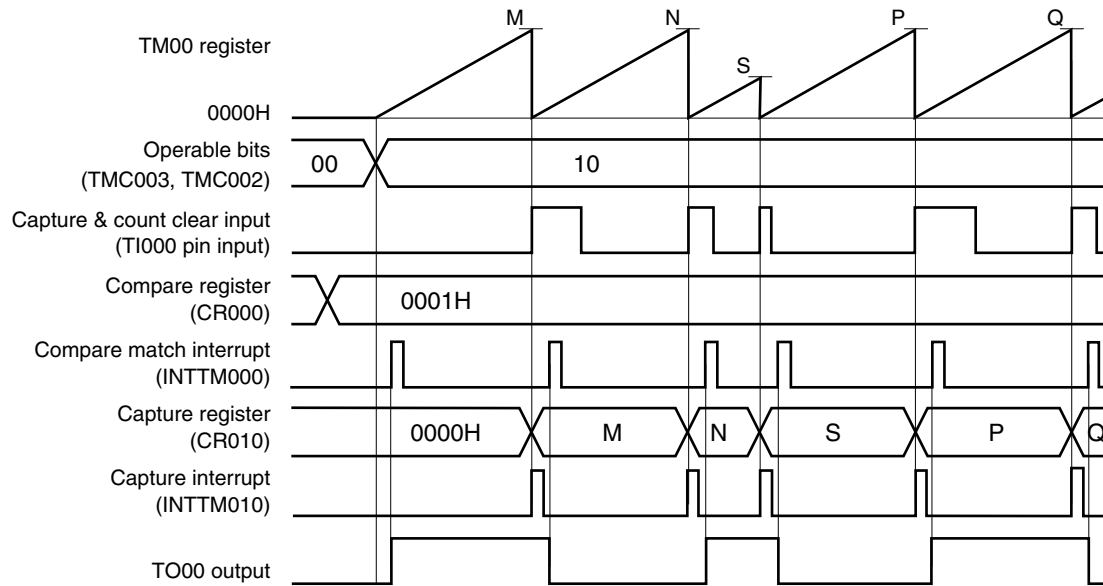
(2) Operation in clear & start mode entered by TI000 pin valid edge input
(CR000: compare register, CR010: capture register)

Figure 6-24. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Compare Register, CR010: Capture Register)



**Figure 6-25. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Compare Register, CR010: Capture Register) (1/2)**

(a) TOC00 = 13H, PRM00 = 10H, CRC00 = 04H, TMC00 = 08H, CR000 = 0001H

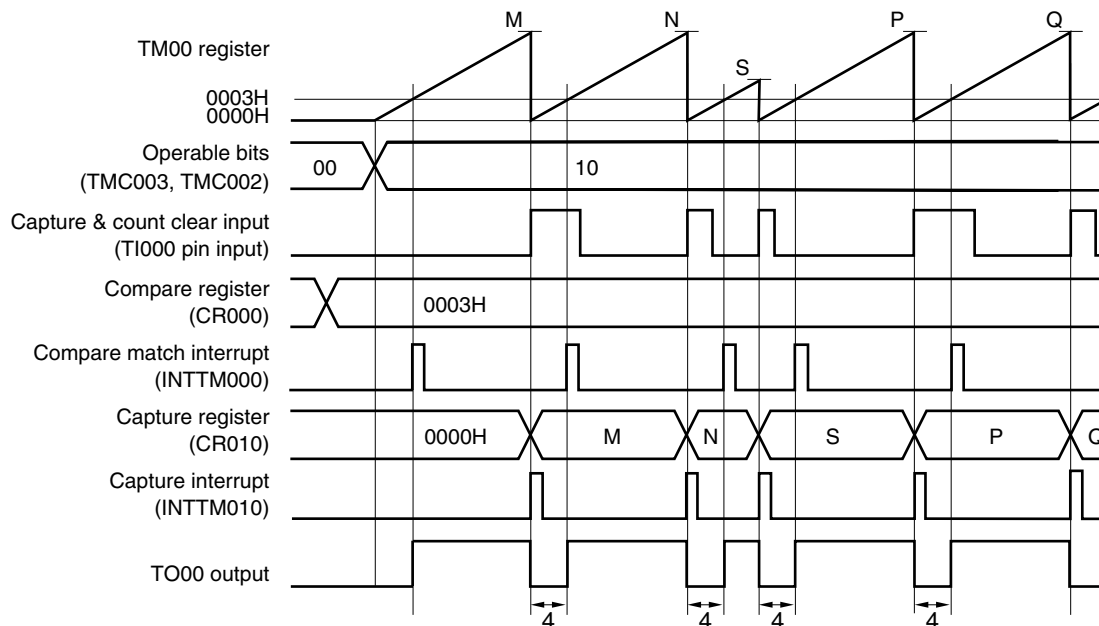


This is an application example where the TO00 output level is inverted when the count value has been captured & cleared.

The count value is captured to CR010 and TM00 is cleared (to 0000H) when the valid edge of the TI000 pin is detected. When the count value of TM00 is 0001H, a compare match interrupt signal (INTTM000) is generated, and the TO00 output level is inverted.

**Figure 6-25. Timing Example of Clear & Start Mode Entered by T1000 Pin Valid Edge Input
(CR000: Compare Register, CR010: Capture Register) (2/2)**

(b) TOC00 = 13H, PRM00 = 10H, CRC00 = 04H, TMC00 = 0AH, CR000 = 0003H

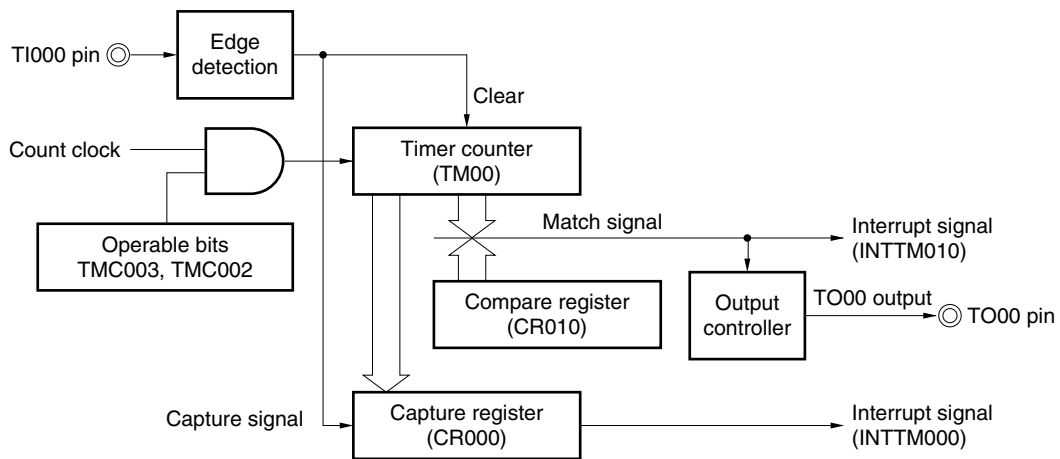


This is an application example where the width set to CR000 (4 clocks in this example) is to be output from the TO00 pin when the count value has been captured & cleared.

The count value is captured to CR010, a capture interrupt signal (INTTM010) is generated, TM00 is cleared (to 0000H), and the TO00 output level is inverted when the valid edge of the T1000 pin is detected. When the count value of TM00 is 0003H (four clocks have been counted), a compare match interrupt signal (INTTM000) is generated and the TO00 output level is inverted.

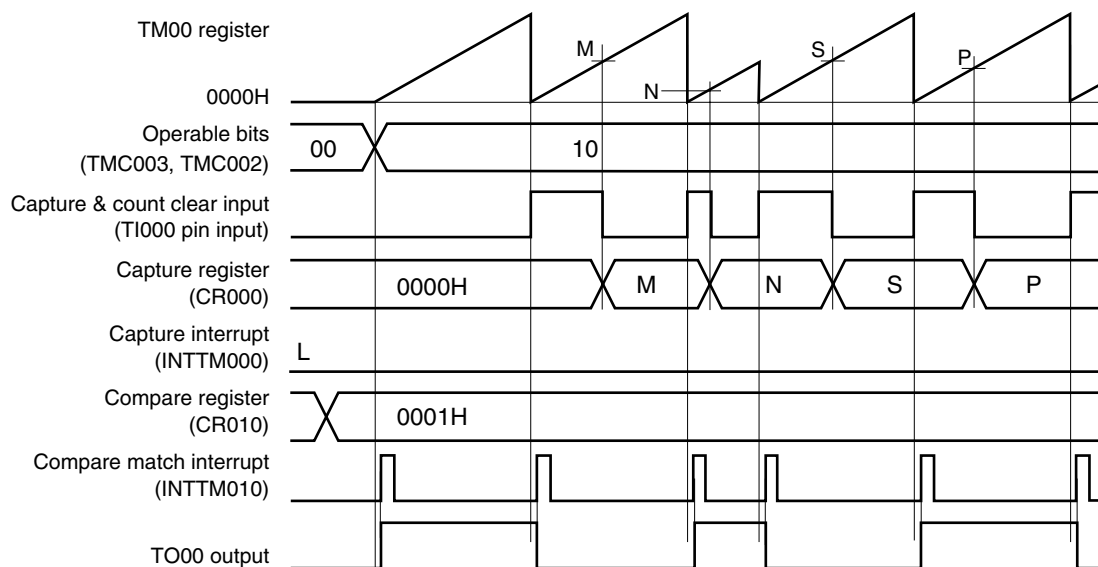
(3) Operation in clear & start mode by entered TI000 pin valid edge input
(CR000: capture register, CR010: compare register)

Figure 6-26. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Compare Register)



**Figure 6-27. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Compare Register) (1/2)**

(a) TOC00 = 13H, PRM00 = 10H, CRC00 = 03H, TMC00 = 08H, CR010 = 0001H



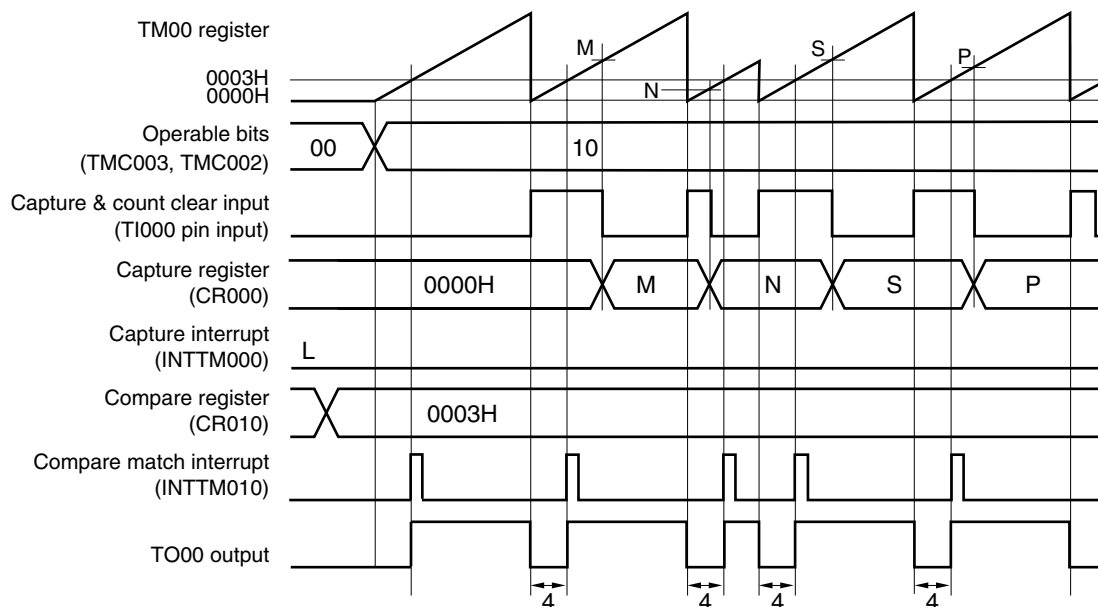
This is an application example where the TO00 output level is to be inverted when the count value has been captured & cleared.

TM00 is cleared at the rising edge detection of the TI000 pin and it is captured to CR000 at the falling edge detection of the TI000 pin.

When bit 1 (CRC001) of capture/compare control register 00 (CRC00) is set to 1, the count value of TM00 is captured to CR000 in the phase reverse to that of the signal input to the TI000 pin, but the capture interrupt signal (INTTM000) is not generated. However, the INTTM000 signal is generated when the valid edge of the TI010 pin is detected. Mask the INTTM000 signal when it is not used.

**Figure 6-27. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Compare Register) (2/2)**

(b) TOC00 = 13H, PRM00 = 10H, CRC00 = 03H, TMC00 = 0AH, CR010 = 0003H



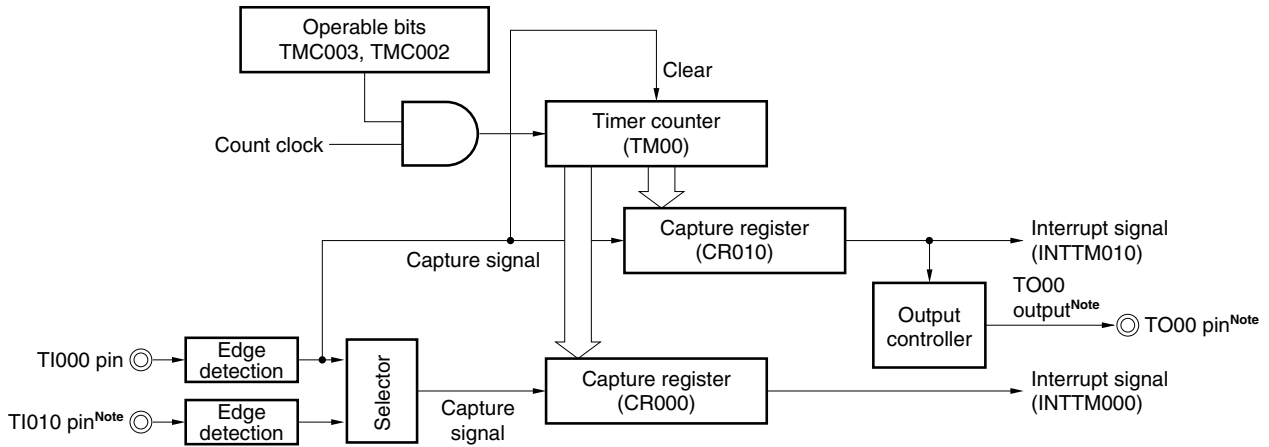
This is an application example where the width set to CR010 (4 clocks in this example) is to be output from the TO00 pin when the count value has been captured & cleared.

TM00 is cleared (to 0000H) at the rising edge detection of the TI000 pin and captured to CR000 at the falling edge detection of the TI000 pin. The TO00 output level is inverted when TM00 is cleared (to 0000H) because the rising edge of the TI000 pin has been detected or when the value of TM00 matches that of a compare register (CR010).

When bit 1 (CRC001) of capture/compare control register 00 (CRC00) is 1, the count value of TM00 is captured to CR000 in the phase reverse to that of the input signal of the TI000 pin, but the capture interrupt signal (INTTM000) is not generated. However, the INTTM000 interrupt is generated when the valid edge of the TI010 pin is detected. Mask the INTTM000 signal when it is not used.

(4) Operation in clear & start mode entered by TI000 pin valid edge input
(CR000: capture register, CR010: capture register)

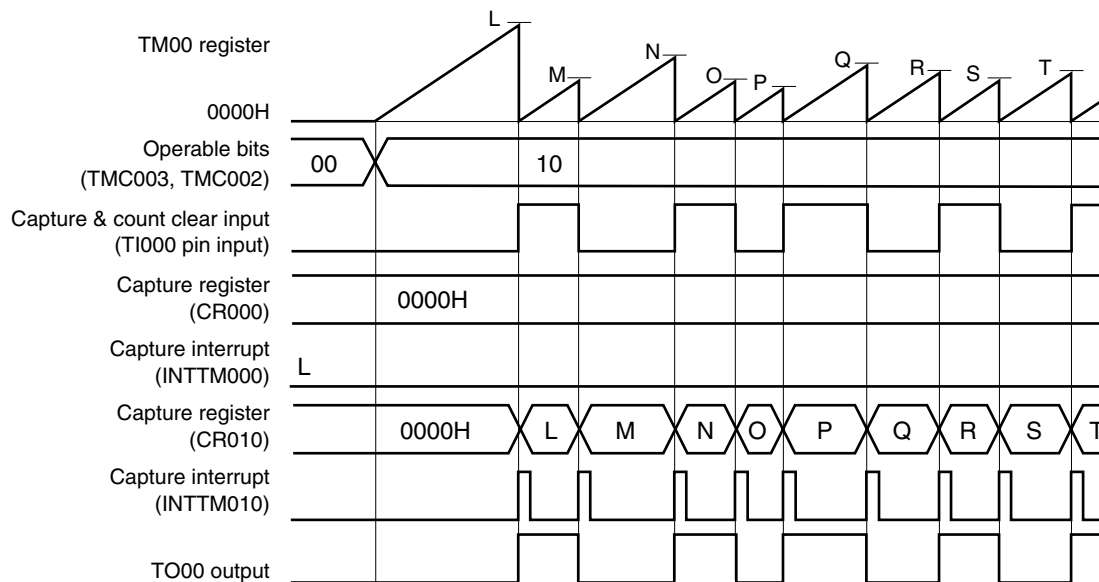
Figure 6-28. Block Diagram of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Capture Register)



Note The timer output (TO00) cannot be used when detecting the valid edge of the TI010 pin is used.

Figure 6-29. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input (CR000: Capture Register, CR010: Capture Register) (1/3)

(a) TOC00 = 13H, PRM00 = 30H, CRC00 = 05H, TMC00 = 0AH

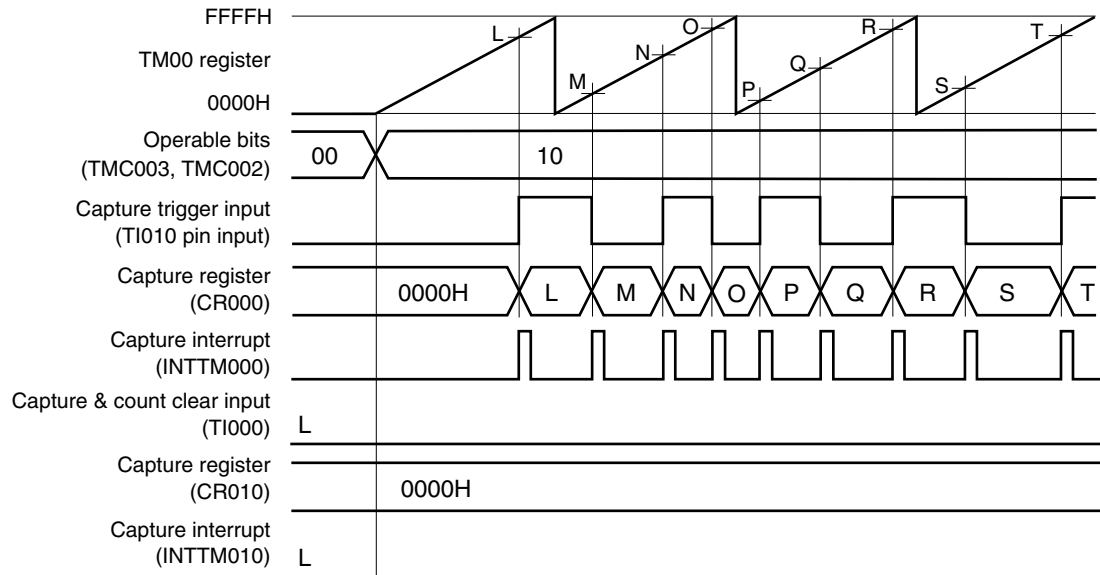


This is an application example where the count value is captured to CR010, TM00 is cleared, and the TO00 output is inverted when the rising or falling edge of the TI000 pin is detected.

When the edge of the TI010 pin is detected, an interrupt signal (INTTM000) is generated. Mask the INTTM000 signal when it is not used.

**Figure 6-29. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Capture Register) (2/3)**

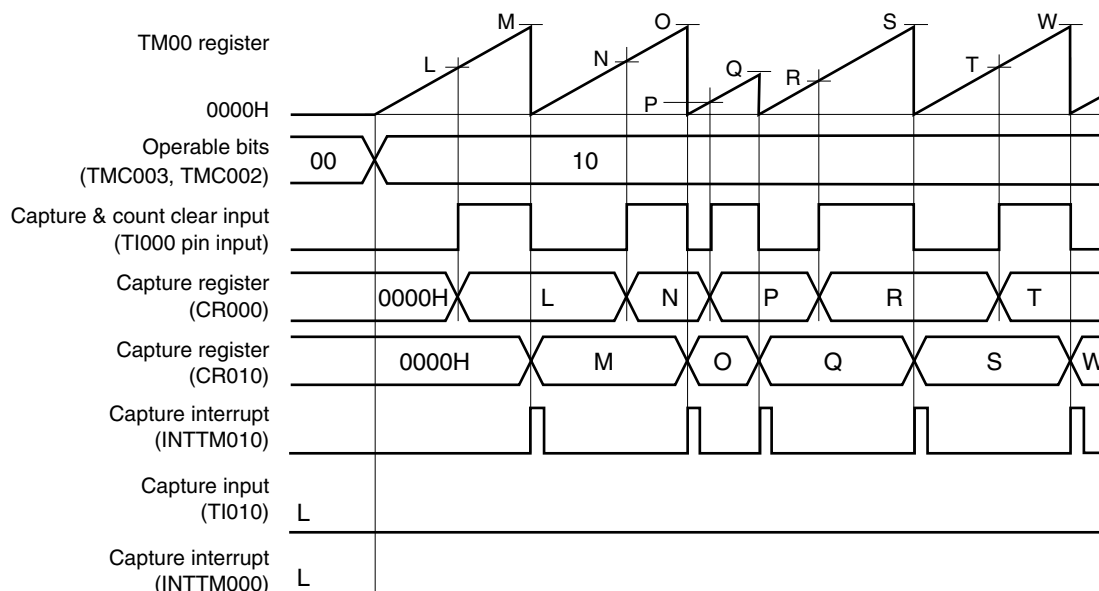
(b) TOC00 = 13H, PRM00 = C0H, CRC00 = 05H, TMC00 = 0AH



This is a timing example where an edge is not input to the TI000 pin, in an application where the count value is captured to CR000 when the rising or falling edge of the TI010 pin is detected.

**Figure 6-29. Timing Example of Clear & Start Mode Entered by TI000 Pin Valid Edge Input
(CR000: Capture Register, CR010: Capture Register) (3/3)**

(c) TOC00 = 13H, PRM00 = 00H, CRC00 = 07H, TMC00 = 0AH



This is an application example where the pulse width of the signal input to the TI000 pin is measured.

By setting CRC00, the count value can be captured to CR000 in the phase reverse to the falling edge of the TI000 pin (i.e., rising edge) and to CR010 at the falling edge of the TI000 pin.

The high- and low-level widths of the input pulse can be calculated by the following expressions.

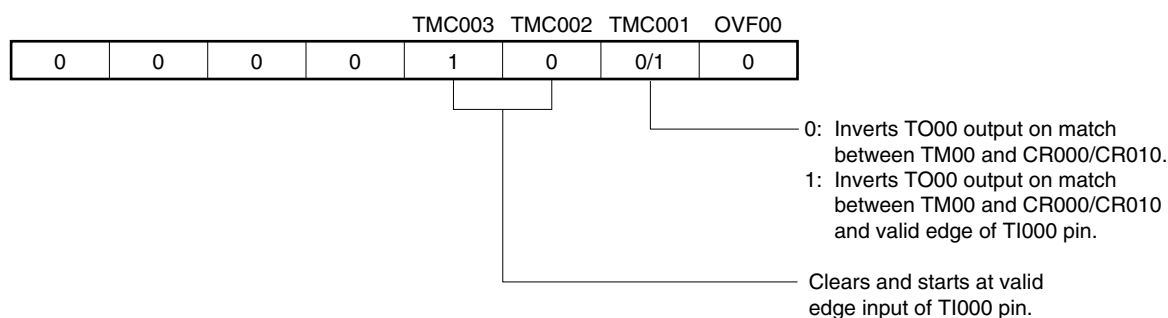
- High-level width = [CR010 value] – [CR000 value] × [Count clock cycle]
- Low-level width = [CR000 value] × [Count clock cycle]

If the reverse phase of the TI000 pin is selected as a trigger to capture the count value to CR000, the INTTM000 signal is not generated. Read the values of CR000 and CR010 to measure the pulse width immediately after the INTTM010 signal is generated.

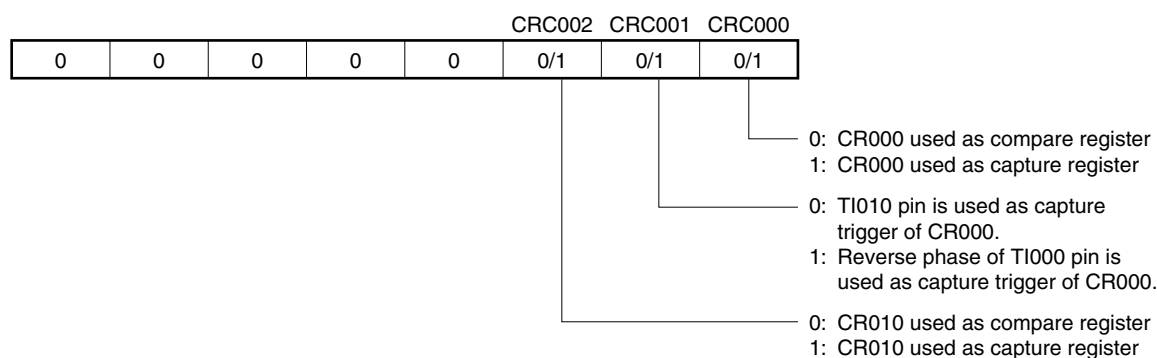
However, if the valid edge specified by bits 6 and 5 (ES110 and ES100) of prescaler mode register 00 (PRM00) is input to the TI010 pin, the count value is not captured but the INTTM000 signal is generated. To measure the pulse width of the TI000 pin, mask the INTTM000 signal when it is not used.

Figure 6-30. Example of Register Settings in Clear & Start Mode Entered by TI000 Pin Valid Edge Input (1/2)

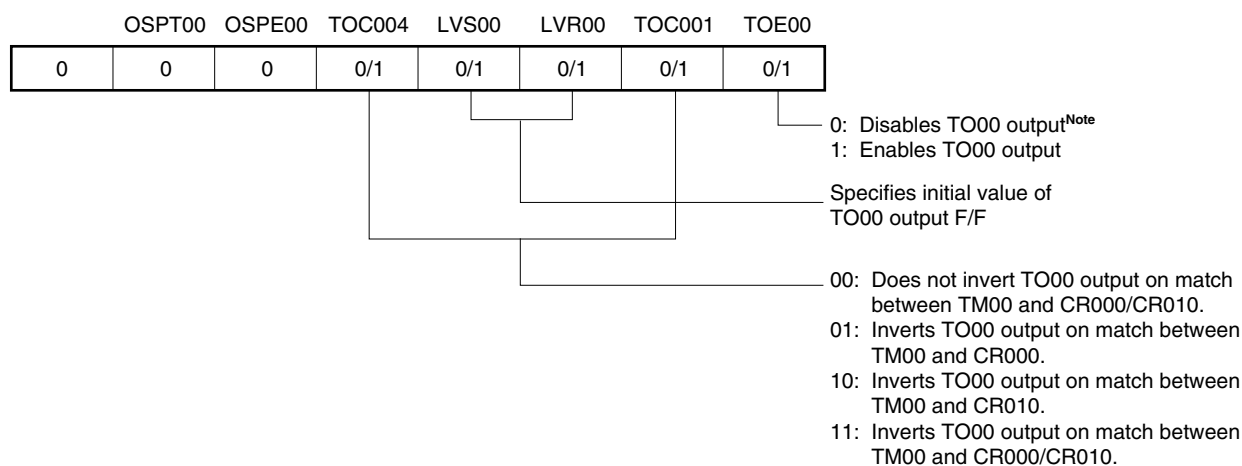
(a) 16-bit timer mode control register 00 (TMC00)



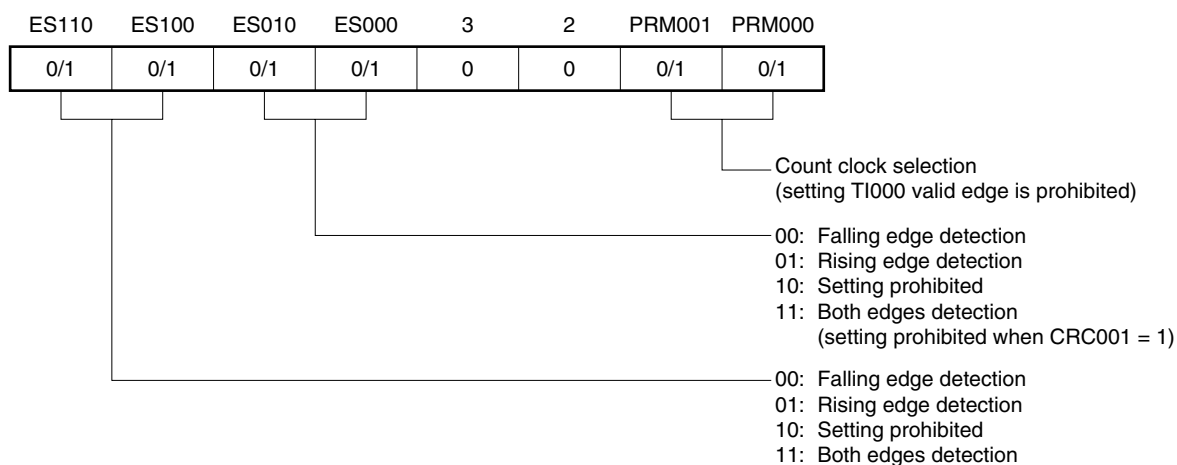
(b) Capture/compare control register 00 (CRC00)



(c) 16-bit timer output control register 00 (TOC00)



Note The timer output (TO00) cannot be used when detecting the valid edge of the TI010 pin is used.

Figure 6-30. Example of Register Settings in Clear & Start Mode Entered by TI000 Pin Valid Edge Input (2/2)**(d) Prescaler mode register 00 (PRM00)****(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM000) is generated. The count value of TM00 is not cleared.

To use this register as a capture register, select either the TI000 or TI010 pin^{Note} input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

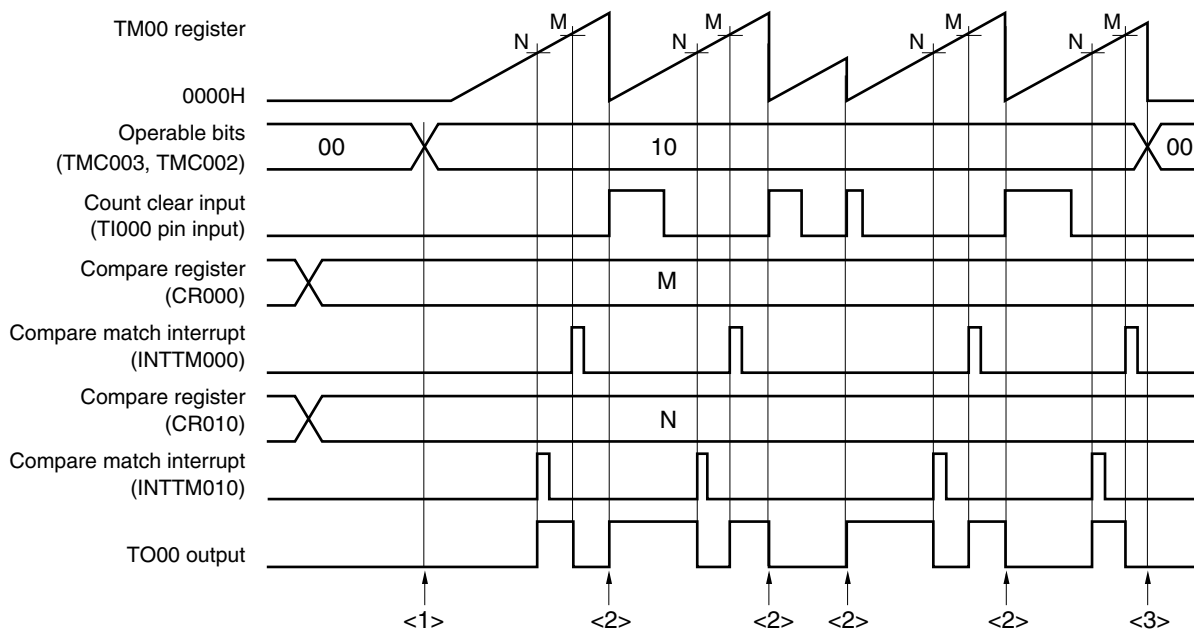
Note The timer output (TO00) cannot be used when detection of the valid edge of the TI010 pin is used.

(g) 16-bit capture/compare register 010 (CR010)

When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM010) is generated. The count value of TM00 is not cleared.

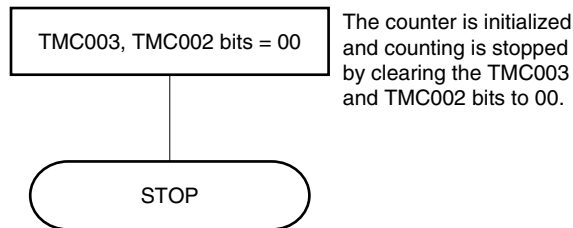
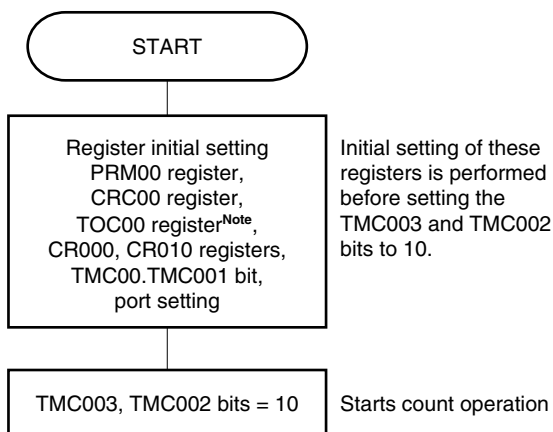
When this register is used as a capture register, the TI000 pin input is used as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR010.

Figure 6-31. Example of Software Processing in Clear & Start Mode Entered by TI000 Pin Valid Edge Input

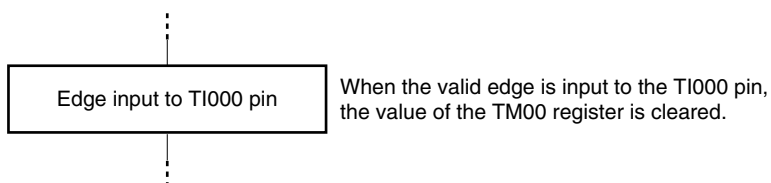


<1> Count operation start flow

<3> Count operation stop flow



<2> TM00 register clear & start flow



Note Care must be exercised when setting TOC00. For details, refer to 6.3 (3) 16-bit timer output control register 00 (TOC00).

6.4.5 Free-running timer operation

When bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) are set to 01 (free-running timer mode), 16-bit timer/event counter 00 continues counting up in synchronization with the count clock. When it has counted up to FFFFH, the overflow flag (OVF00) is set to 1 at the next clock, and TM00 is cleared (to 0000H) and continues counting. Clear OVF00 to 0 by executing the CLR instruction via software.

The following three types of free-running timer operations are available.

- Both CR000 and CR010 are used as compare registers.
- One of CR000 or CR010 is used as a compare register and the other is used as a capture register.
- Both CR000 and CR010 are used as capture registers.

Remarks 1. For the setting of the I/O pins, refer to **6.3 (5) Port mode register 0 (PM0)**.

2. For how to enable the INTTM000 signal interrupt, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

(1) Free-running timer mode operation

(CR000: compare register, CR010: compare register)

**Figure 6-32. Block Diagram of Free-Running Timer Mode
(CR000: Compare Register, CR010: Compare Register)**

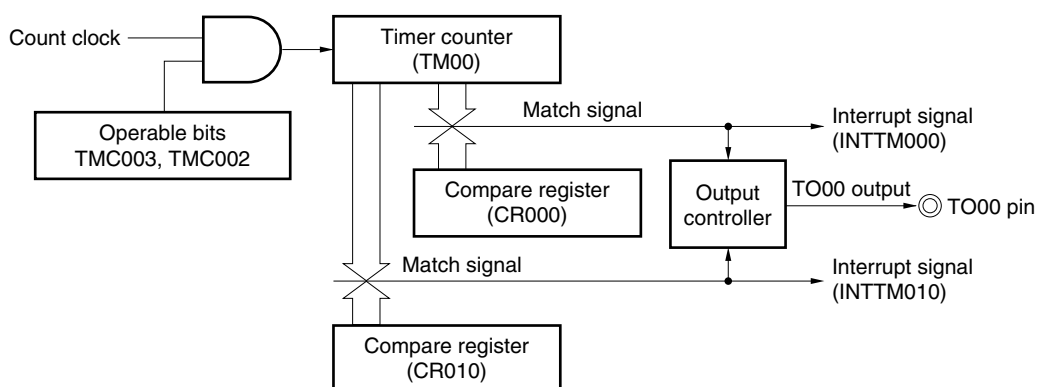
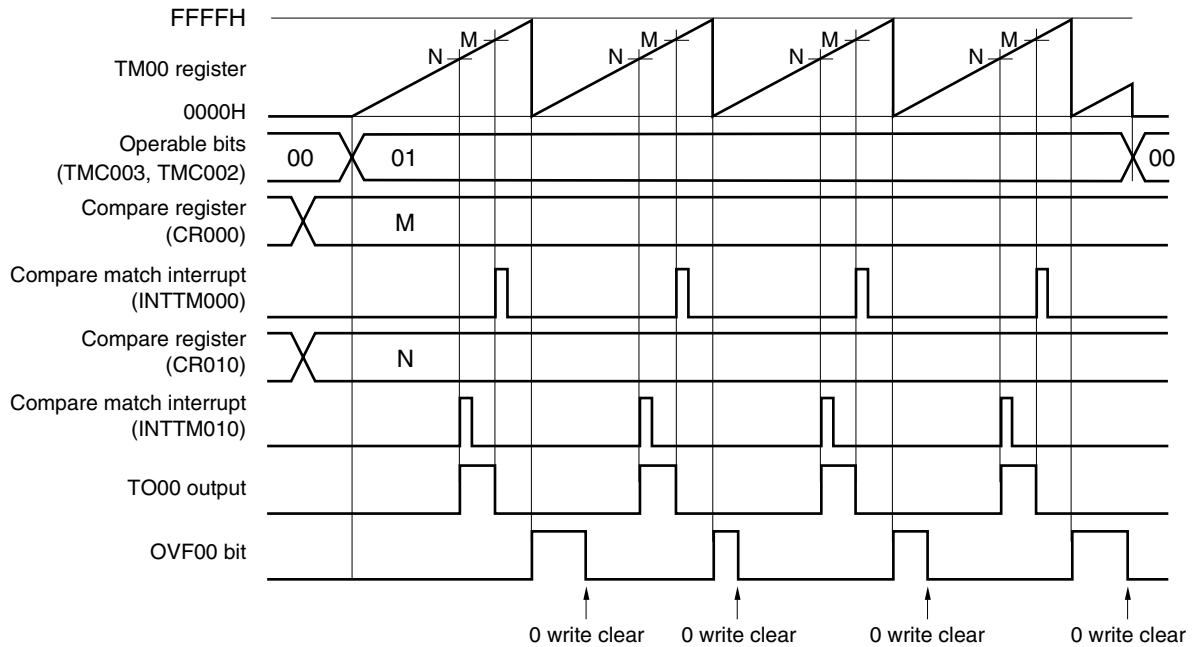


Figure 6-33. Timing Example of Free-Running Timer Mode
(CR000: Compare Register, CR010: Compare Register)

• TOC00 = 13H, PRM00 = 00H, CRC00 = 00H, TMC00 = 04H



This is an application example where two compare registers are used in the free-running timer mode. The TO00 output level is reversed each time the count value of TM00 matches the set value of CR000 or CR010. When the count value matches the register value, the INTTM000 or INTTM010 signal is generated.

(2) Free-running timer mode operation
(CR000: compare register, CR010: capture register)

Figure 6-34. Block Diagram of Free-Running Timer Mode
(CR000: Compare Register, CR010: Capture Register)

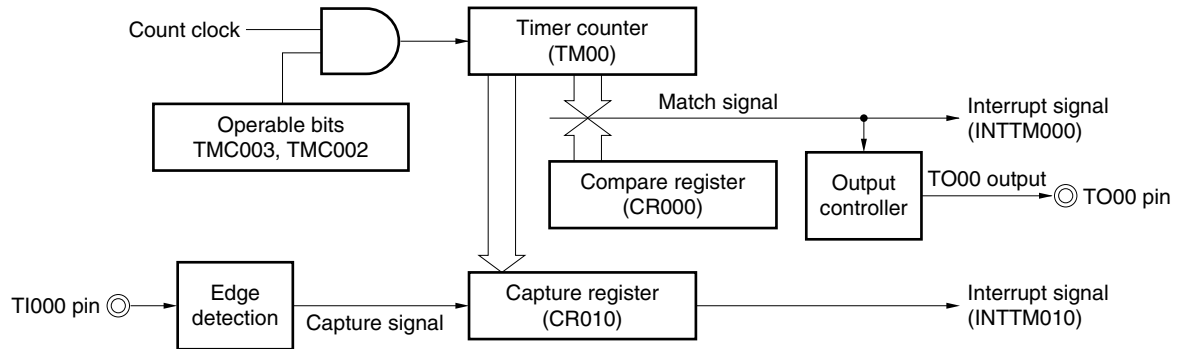
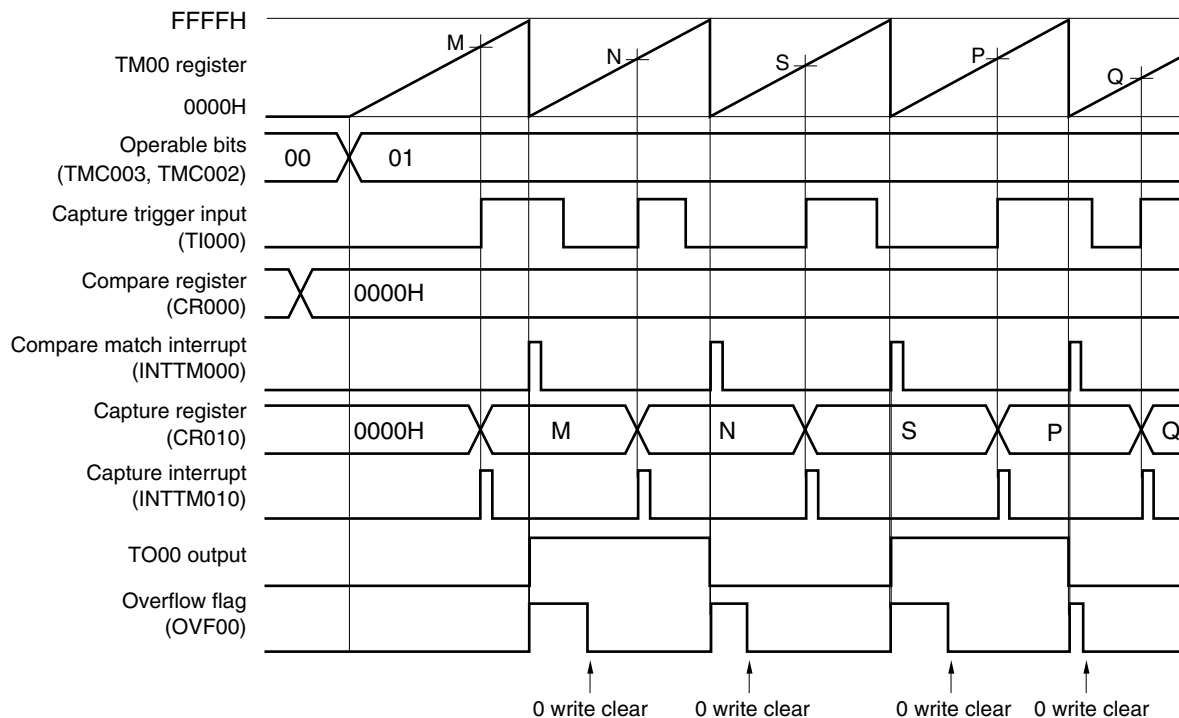


Figure 6-35. Timing Example of Free-Running Timer Mode
(CR000: Compare Register, CR010: Capture Register)

• TOC00 = 13H, PRM00 = 10H, CRC00 = 04H, TMC00 = 04H

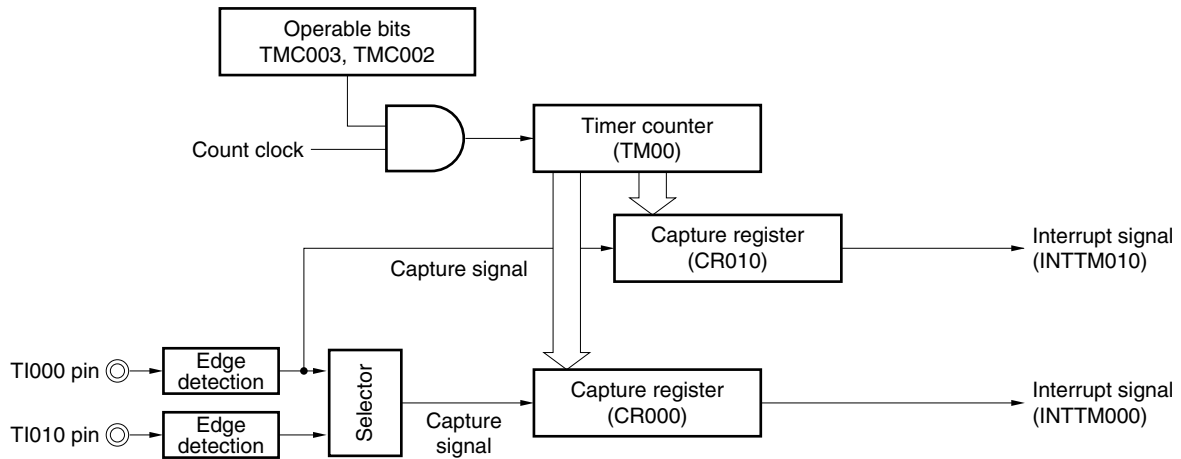


This is an application example where a compare register and a capture register are used at the same time in the free-running timer mode.

In this example, the INTTM000 signal is generated and the TO00 output level is reversed each time the count value of TM00 matches the set value of CR000 (compare register). In addition, the INTTM010 signal is generated and the count value of TM00 is captured to CR010 each time the valid edge of the TI000 pin is detected.

(3) Free-running timer mode operation
(CR000: capture register, CR010: capture register)

Figure 6-36. Block Diagram of Free-Running Timer Mode
(CR000: Capture Register, CR010: Capture Register)

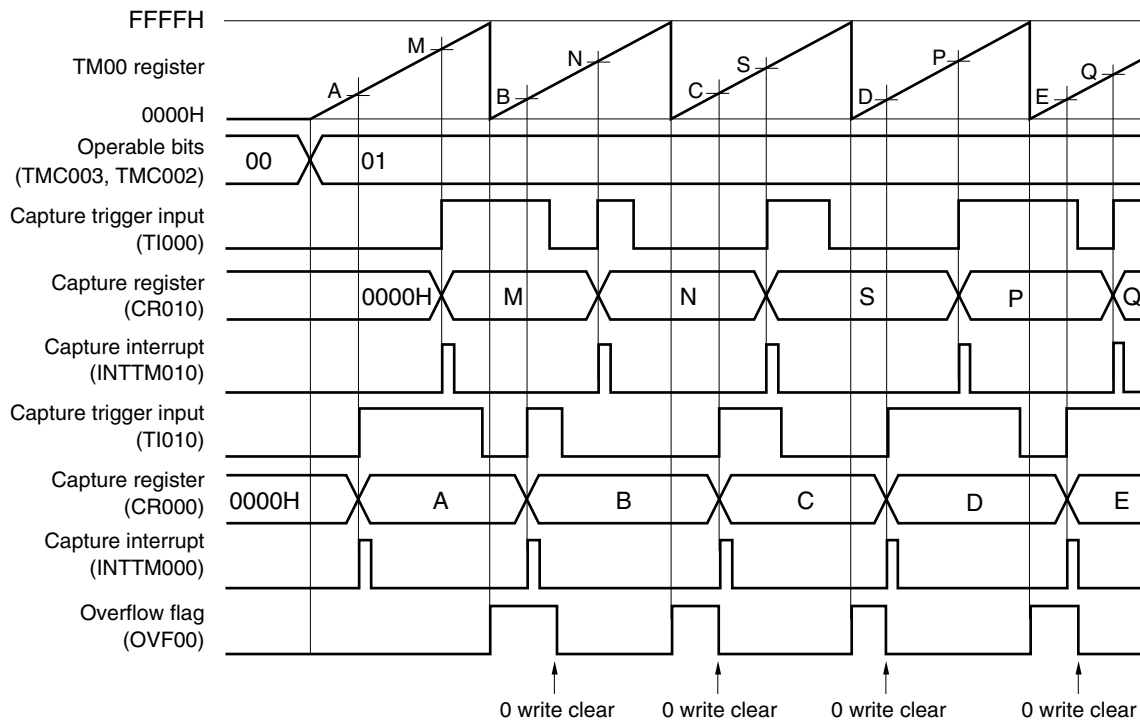


Remark If both CR000 and CR010 are used as capture registers in the free-running timer mode, the TO00 output level is not inverted.

However, it can be inverted each time the valid edge of the TI000 pin is detected if bit 1 (TMC001) of 16-bit timer mode control register 00 (TMC00) is set to 1.

**Figure 6-37. Timing Example of Free-Running Timer Mode
(CR000: Capture Register, CR010: Capture Register) (1/2)**

(a) TOC00 = 13H, PRM00 = 50H, CRC00 = 05H, TMC00 = 04H

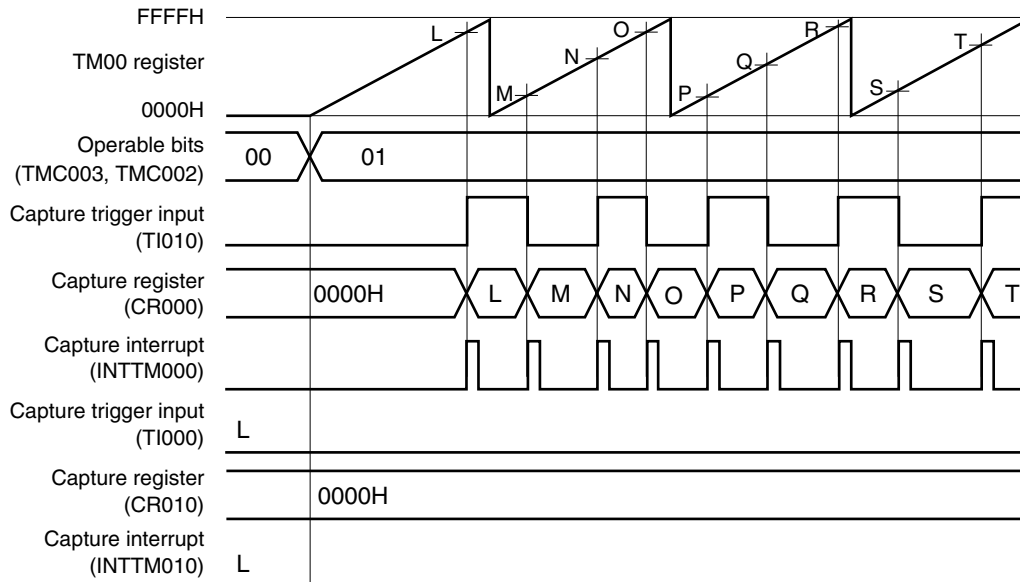


This is an application example where the count values that have been captured at the valid edges of separate capture trigger signals are stored in separate capture registers in the free-running timer mode.

The count value is captured to CR010 when the valid edge of the TI000 pin input is detected and to CR000 when the valid edge of the TI010 pin input is detected.

**Figure 6-37. Timing Example of Free-Running Timer Mode
(CR000: Capture Register, CR010: Capture Register) (2/2)**

(b) TOC00 = 13H, PRM00 = C0H, CRC00 = 05H, TMC00 = 04H

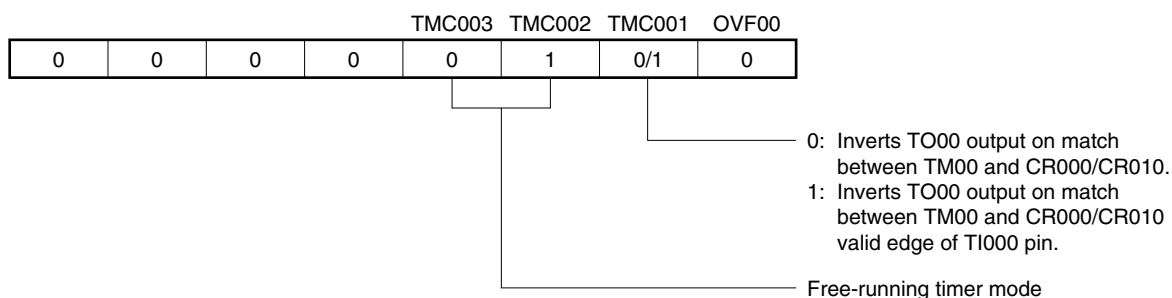


This is an application example where both the edges of the TI010 pin are detected and the count value is captured to CR000 in the free-running timer mode.

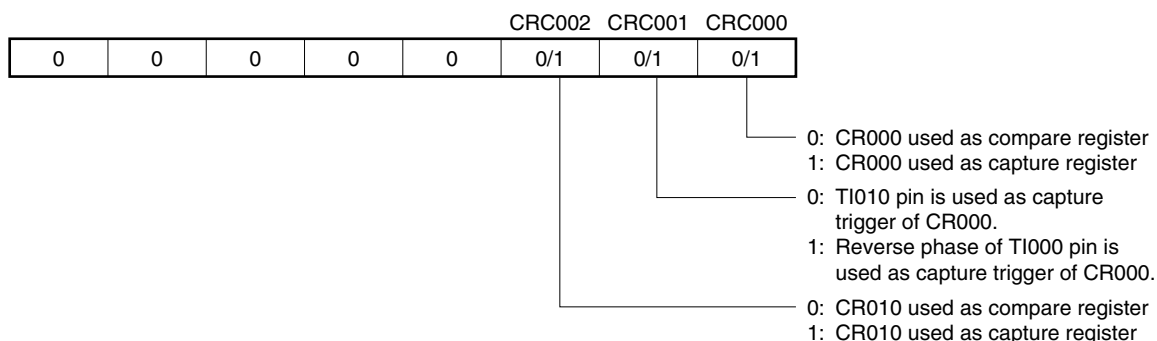
When both CR000 and CR010 are used as capture registers and when the valid edge of only the TI010 pin is to be detected, the count value cannot be captured to CR010.

Figure 6-38. Example of Register Settings in Free-Running Timer Mode (1/2)

(a) 16-bit timer mode control register 00 (TMC00)



(b) Capture/compare control register 00 (CRC00)



(c) 16-bit timer output control register 00 (TOC00)

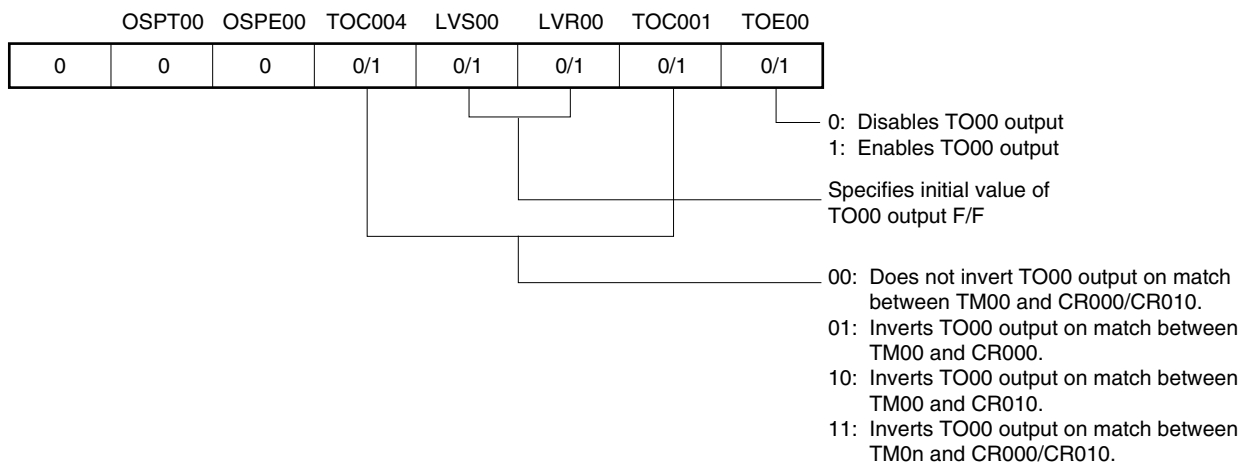
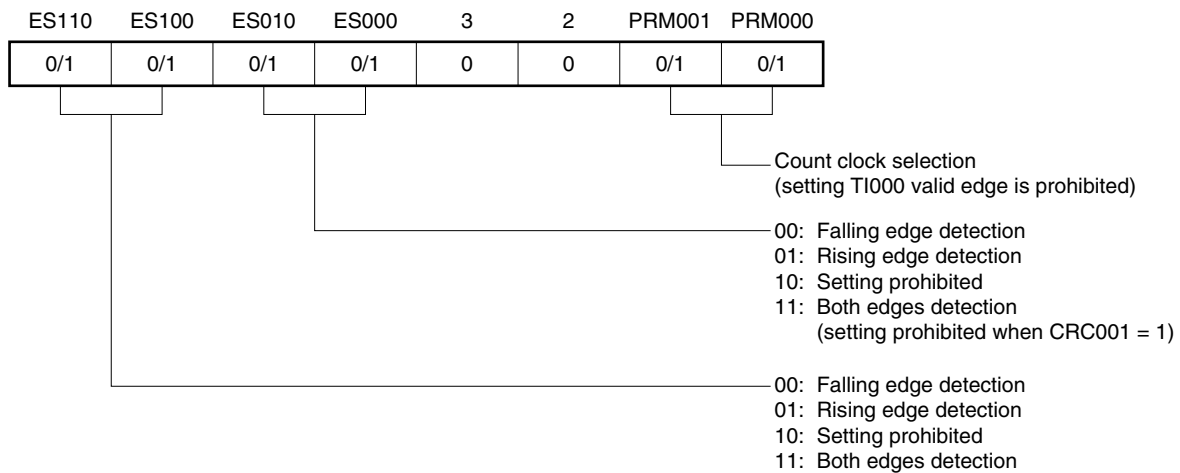


Figure 6-38. Example of Register Settings in Free-Running Timer Mode (2/2)

(d) Prescaler mode register 00 (PRM00)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM000) is generated. The count value of TM00 is not cleared.

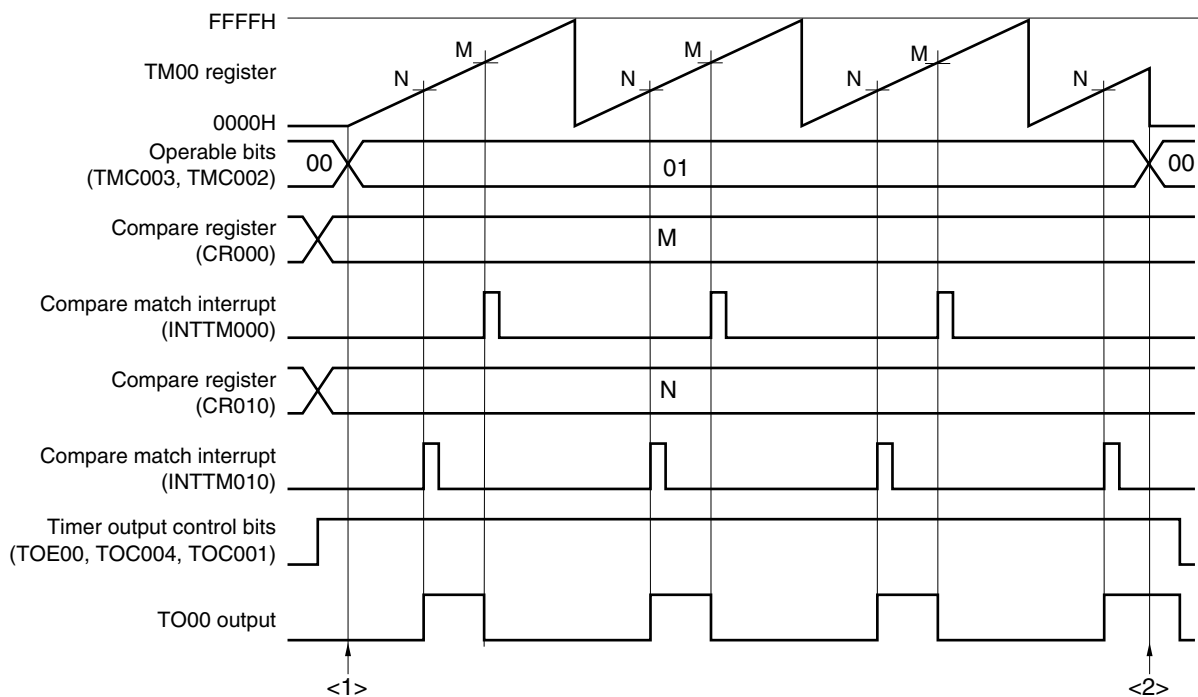
To use this register as a capture register, select either the TI000 or TI010 pin input as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

(g) 16-bit capture/compare register 010 (CR010)

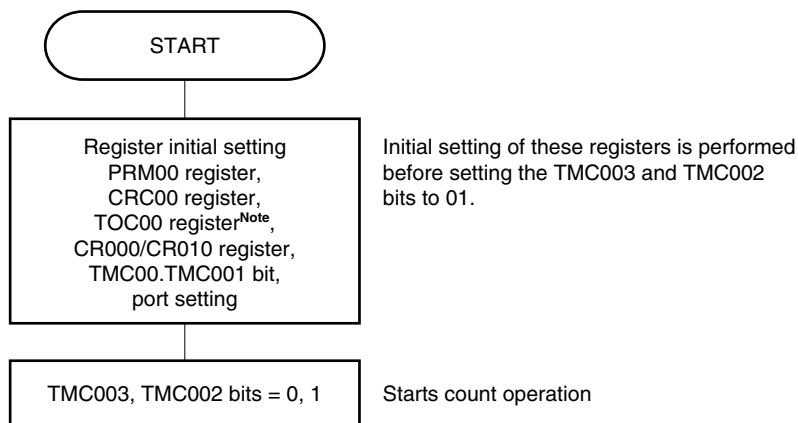
When this register is used as a compare register and when its value matches the count value of TM00, an interrupt signal (INTTM010) is generated. The count value of TM00 is not cleared.

When this register is used as a capture register, the TI000 pin input is used as a capture trigger. When the valid edge of the capture trigger is detected, the count value of TM00 is stored in CR010.

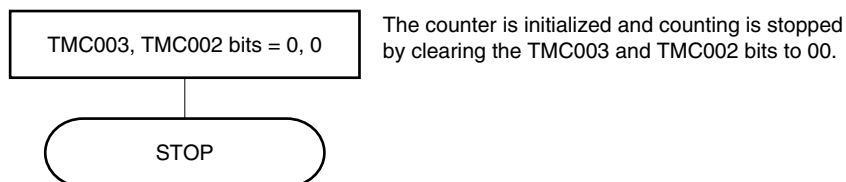
Figure 6-39. Example of Software Processing in Free-Running Timer Mode



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, refer to 6.3 (3) 16-bit timer output control register 00 (TOC00).

6.4.6 PPG output operation

A square wave having a pulse width set in advance by CR010 is output from the TO00 pin as a PPG (Programmable Pulse Generator) signal during a cycle set by CR000 when bits 3 and 2 (TMC003 and TMC002) of 16-bit timer mode control register 00 (TMC00) are set to 11 (clear & start upon a match between TM00 and CR000).

The pulse cycle and duty factor of the pulse generated as the PPG output are as follows.

- Pulse cycle = (Set value of CR000 + 1) × Count clock cycle
- Duty = (Set value of CR010 + 1) / (Set value of CR000 + 1)

Caution To change the duty factor (value of CR010) during operation, refer to 6.5.1 Rewriting CR010 during TM00 operation.

- Remarks**
1. For the setting of I/O pins, refer to 6.3 (5) Port mode register 0 (PM0).
 2. For how to enable the INTTM000 signal interrupt, refer to CHAPTER 16 INTERRUPT FUNCTIONS.

Figure 6-40. Block Diagram of PPG Output Operation

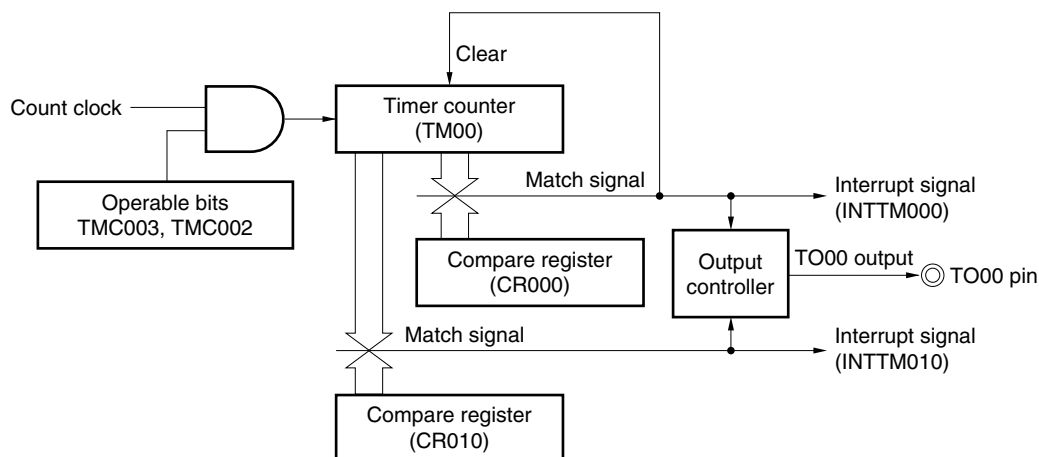
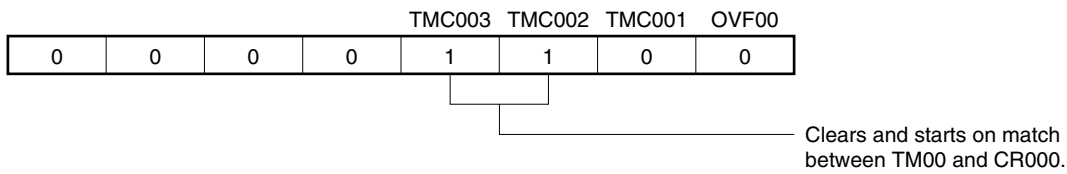
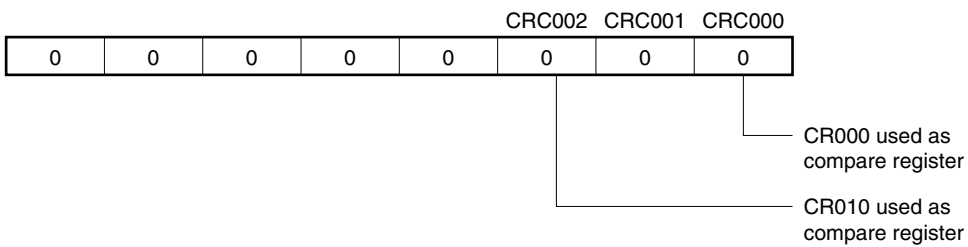


Figure 6-41. Example of Register Settings for PPG Output Operation (1/2)

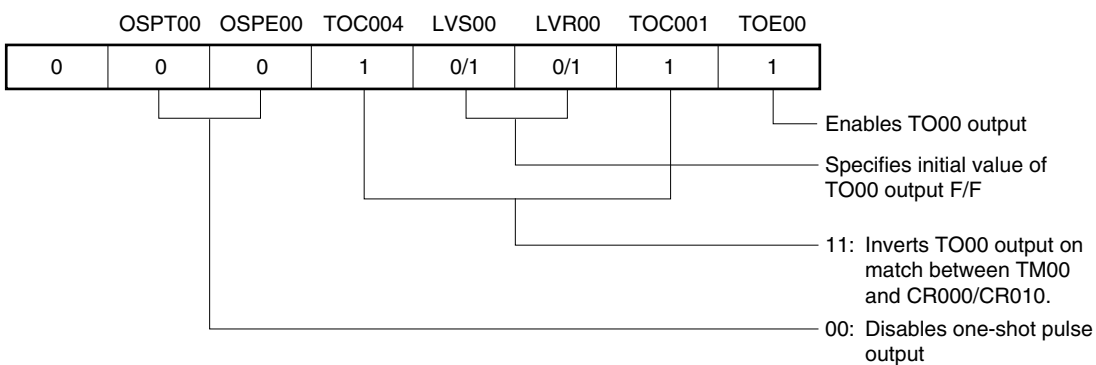
(a) 16-bit timer mode control register 00 (TMC00)



(b) Capture/compare control register 00 (CRC00)



(c) 16-bit timer output control register 00 (TOC00)



(d) Prescaler mode register 00 (PRM00)

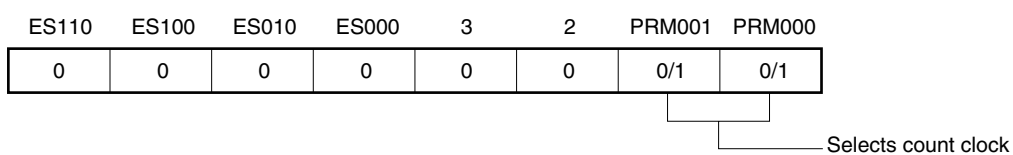


Figure 6-41. Example of Register Settings for PPG Output Operation (2/2)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

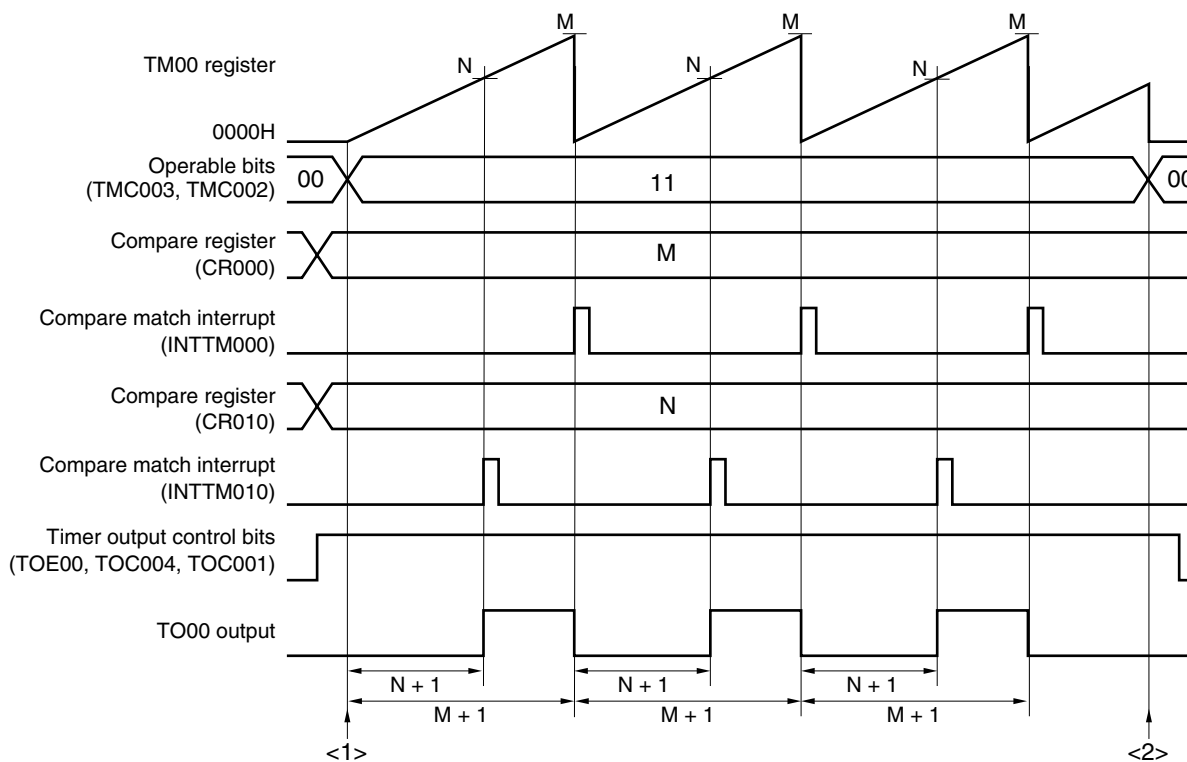
An interrupt signal (INTTM000) is generated when the value of this register matches the count value of TM00.
The count value of TM00 is cleared.

(g) 16-bit capture/compare register 010 (CR010)

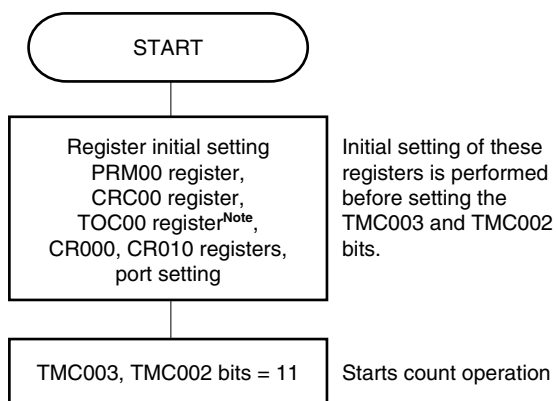
An interrupt signal (INTTM010) is generated when the value of this register matches the count value of TM00.
The count value of TM00 is not cleared.

Caution Set values to CR000 and CR010 such that the condition $0000H \leq CR010 < CR000 \leq FFFFH$ is satisfied.

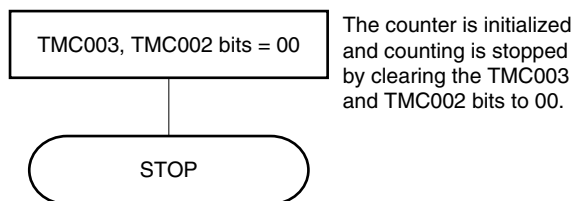
Figure 6-42. Example of Software Processing for PPG Output Operation



<1> Count operation start flow



<2> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, refer to 6.3 (3) 16-bit timer output control register 00 (TOC00).

Remark PPG pulse cycle = (M + 1) × Count clock cycle
PPG duty = (N + 1)/(M + 1)

6.4.7 One-shot pulse output operation

A one-shot pulse can be output by setting bits 3 and 2 (TMC003 and TMC002) of the 16-bit timer mode control register 00 (TMC00) to 01 (free-running timer mode) or to 10 (clear & start mode entered by the TI000 pin valid edge) and setting bit 5 (OSPE00) of 16-bit timer output control register 00 (TOC00) to 1.

When bit 6 (OSPT00) of TOC00 is set to 1 or when the valid edge is input to the TI000 pin during timer operation, clearing & starting of TM00 is triggered, and a pulse of the difference between the values of CR000 and CR010 is output only once from the TO00 pin.

- Cautions**
1. Do not input the trigger again (setting OSPT00 to 1 or detecting the valid edge of the TI000 pin) while the one-shot pulse is output. To output the one-shot pulse again, generate the trigger after the current one-shot pulse output has completed.
 2. To use only the setting of OSPT00 to 1 as the trigger of one-shot pulse output, do not change the level of the TI000 pin or its alternate function port pin. Otherwise, the pulse will be unexpectedly output.

- Remarks**
1. For the setting of the I/O pins, refer to 6.3 (5) Port mode register 0 (PM0).
 2. For how to enable the INTTM000 signal interrupt, refer to CHAPTER 16 INTERRUPT FUNCTIONS.

Figure 6-43. Block Diagram of One-Shot Pulse Output Operation

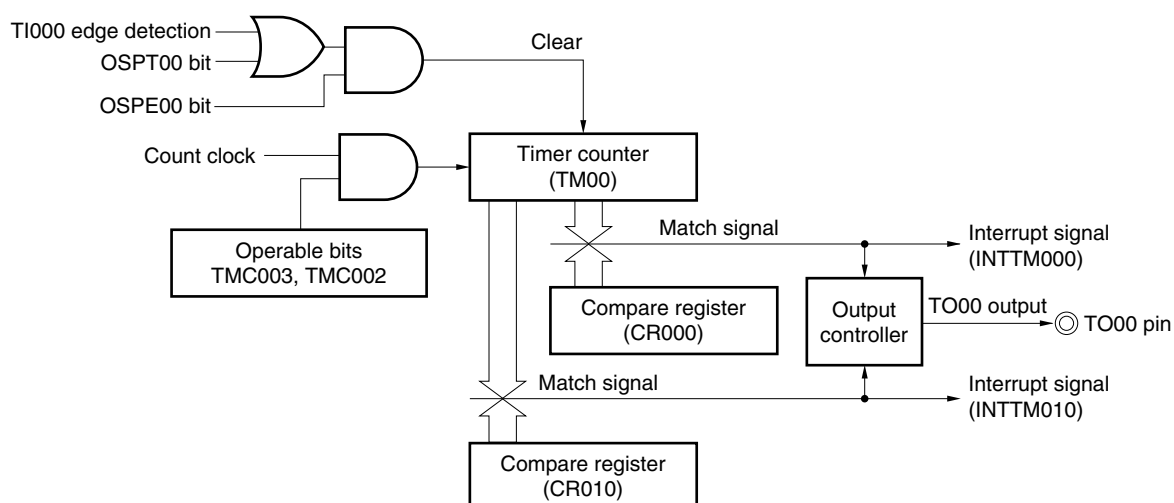
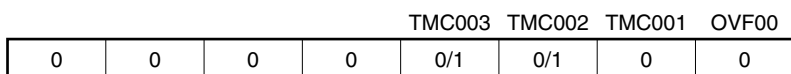


Figure 6-44. Example of Register Settings for One-Shot Pulse Output Operation (1/2)

(a) 16-bit timer mode control register 00 (TMC00)



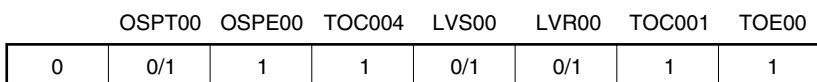
01: Free running timer mode
10: Clear and start mode by valid edge of T1000 pin.

(b) Capture/compare control register 00 (CRC00)



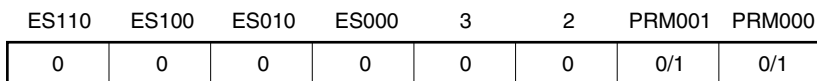
CR000 used as compare register
CR010 used as compare register

(c) 16-bit timer output control register 00 (TOC00)



Enables TO00 output
Specifies initial value of TO00 output
Inverts TO00 output on match between TM00 and CR000/CR010.
Enables one-shot pulse output
Software trigger is generated by writing 1 to this bit (operation is not affected even if 0 is written to it).

(d) Prescaler mode register 00 (PRM00)



Selects count clock

Figure 6-44. Example of Register Settings for One-Shot Pulse Output Operation (2/2)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

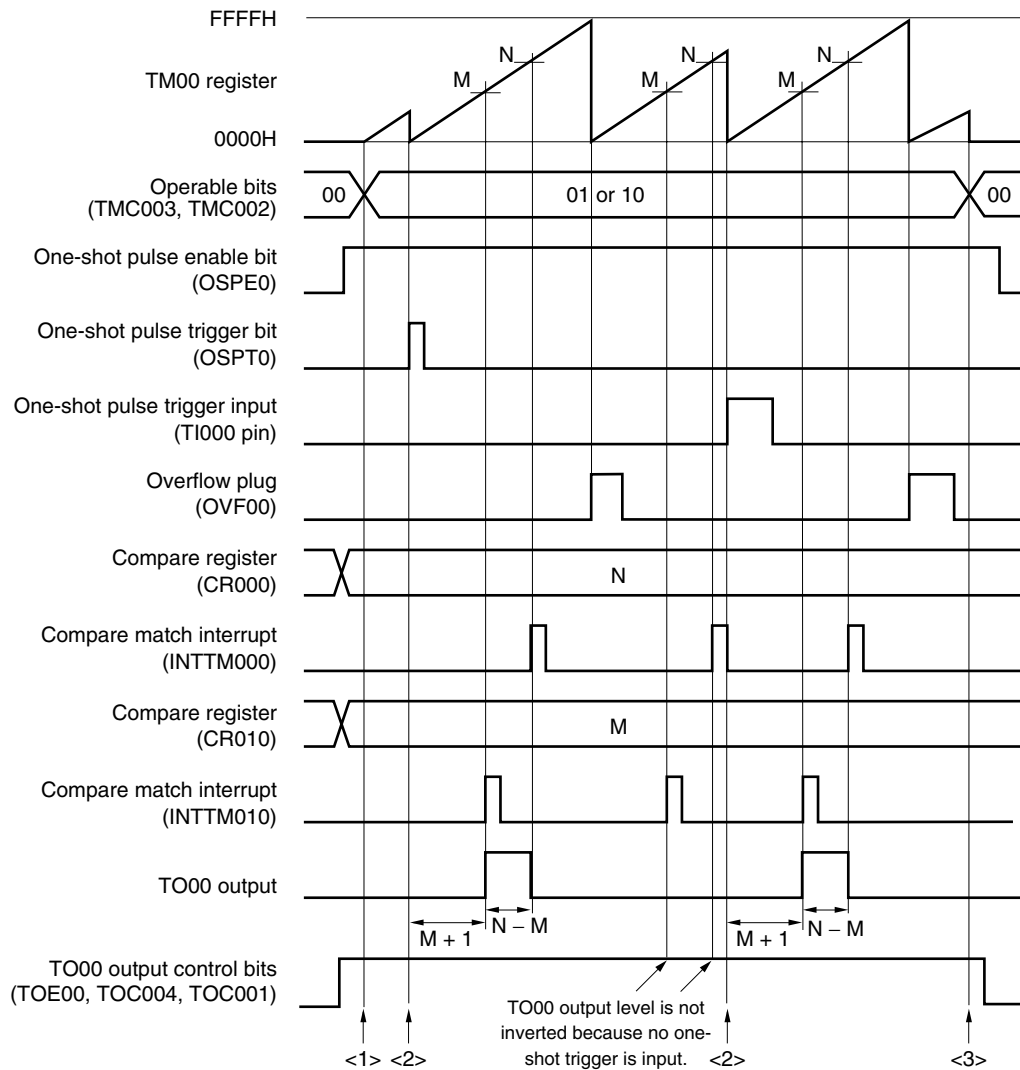
This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR000, an interrupt signal (INTTM000) is generated and the TO00 output level is inverted.

(g) 16-bit capture/compare register 010 (CR010)

This register is used as a compare register when a one-shot pulse is output. When the value of TM00 matches that of CR010, an interrupt signal (INTTM010) is generated and the TO00 output level is inverted.

Caution Do not set the same value to CR000 and CR010.

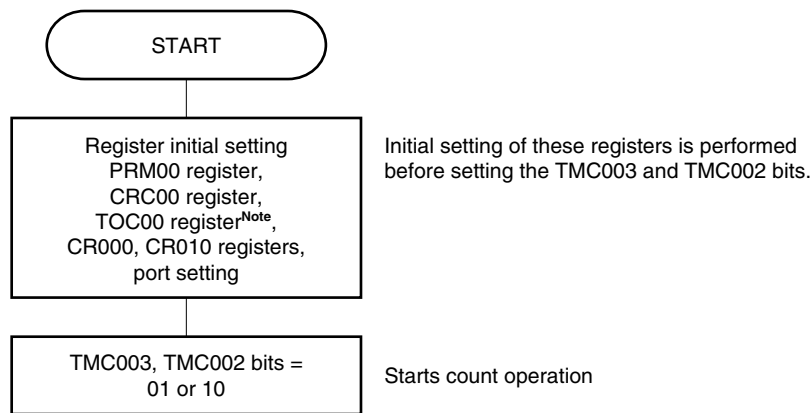
Figure 6-45. Example of Software Processing for One-Shot Pulse Output Operation (1/2)



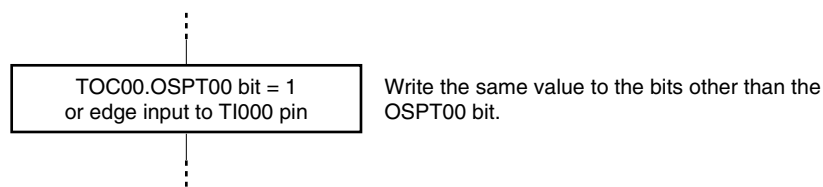
- Time from when the one-shot pulse trigger is input until the one-shot pulse is output
= $(M + 1) \times$ Count clock cycle
- One-shot pulse output active level width
= $(N - M) \times$ Count clock cycle

Figure 6-45. Example of Software Processing for One-Shot Pulse Output Operation (2/2)

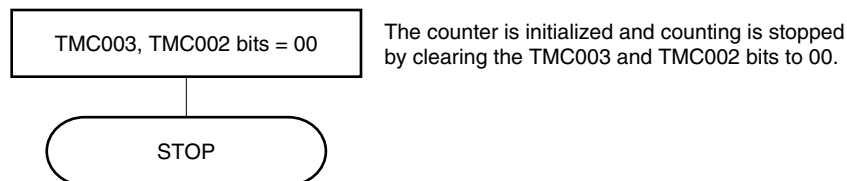
<1> Count operation start flow



<2> One-shot trigger input flow



<3> Count operation stop flow



Note Care must be exercised when setting TOC00. For details, refer to **6.3 (3) 16-bit timer output control register 00 (TOC00)**.

6.4.8 Pulse width measurement operation

TM00 can be used to measure the pulse width of the signal input to the TI000 and TI010 pins.

Measurement can be accomplished by operating the 16-bit timer/event counter 00 in the free-running timer mode or by restarting the timer in synchronization with the signal input to the TI000 pin.

When an interrupt is generated, read the value of the valid capture register and measure the pulse width. Check bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00). If it is set (to 1), clear it to 0 by software.

Figure 6-46. Block Diagram of Pulse Width Measurement (Free-Running Timer Mode)

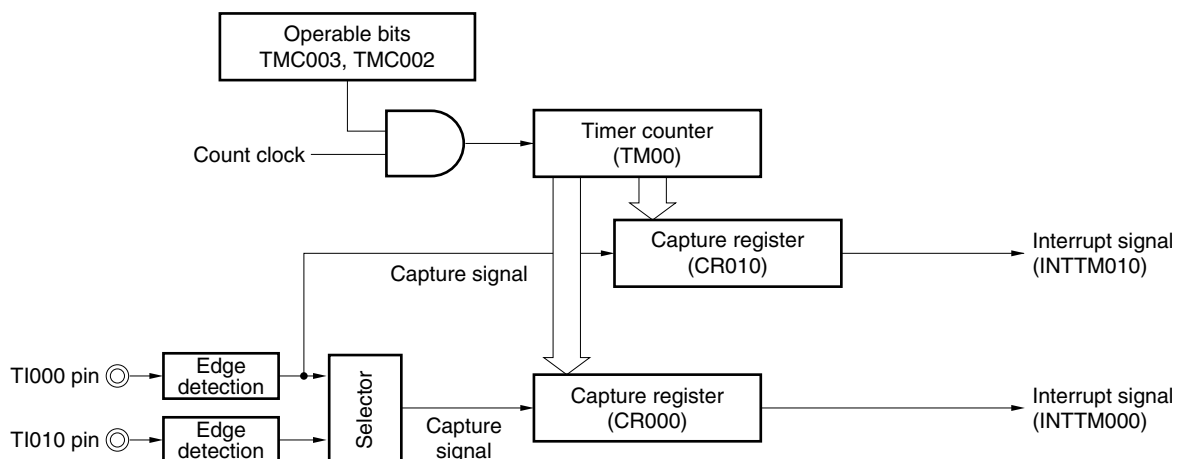
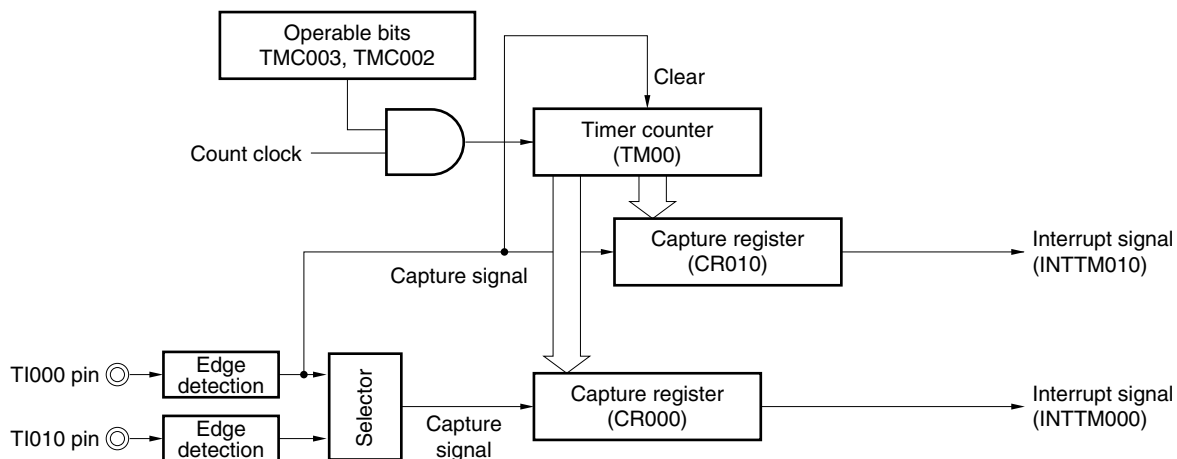


Figure 6-47. Block Diagram of Pulse Width Measurement (Clear & Start Mode Entered by TI000 Pin Valid Edge Input)



A pulse width can be measured in the following three ways.

- Measuring the pulse width by using two input signals of the TI000 and TI010 pins (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI000 pin (free-running timer mode)
- Measuring the pulse width by using one input signal of the TI000 pin (clear & start mode entered by the TI000 pin valid edge input)

Remarks 1. For the setting of the I/O pins, refer to **6.3 (5) Port mode register 0 (PM0)**.

2. For how to enable the INTTM000 signal interrupt, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

(1) Measuring the pulse width by using two input signals of the TI000 and TI010 pins (free-running timer mode)

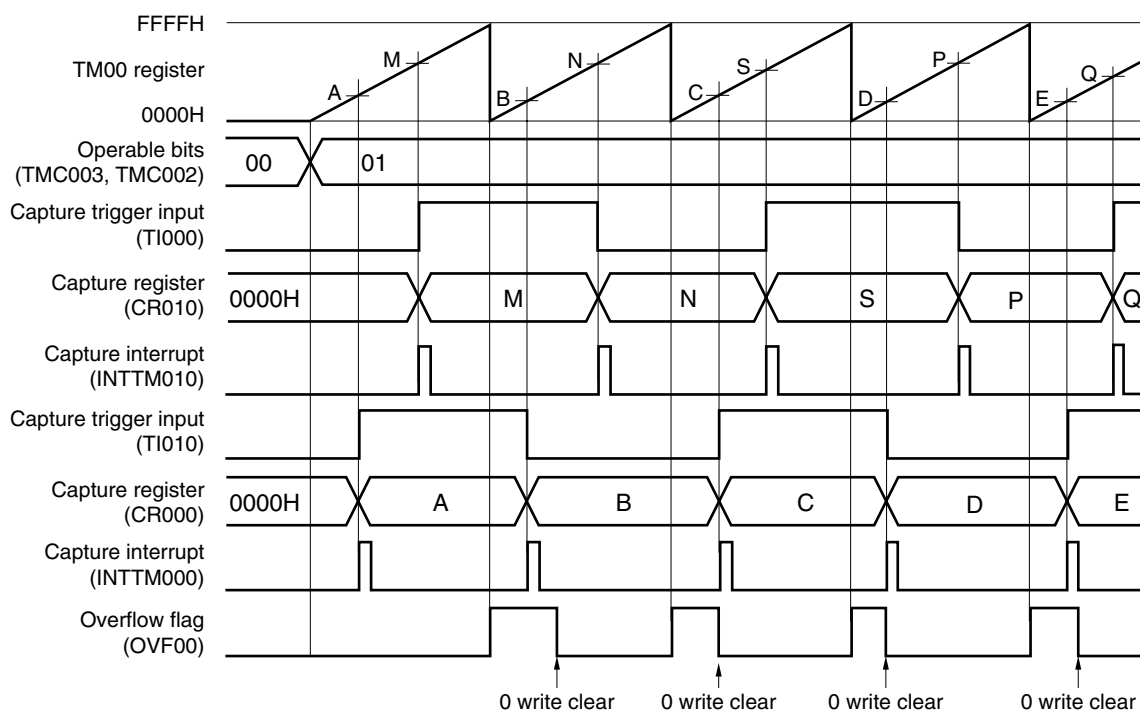
Set the free-running timer mode (TMC003 and TMC002 = 01). When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010. When the valid edge of the TI010 pin is detected, the count value of TM00 is captured to CR000. Specify detection of both the edges of the TI000 and TI010 pins.

By this measurement method, the previous count value is subtracted from the count value captured by the edge of each input signal. Therefore, save the previously captured value to a separate register in advance.

If an overflow occurs, the value becomes negative if the previously captured value is simply subtracted from the current captured value and, therefore, a borrow occurs (bit 0 (CY) of the program status word (PSW) is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-48. Timing Example of Pulse Width Measurement (1)

• TMC00 = 04H, PRM00 = F0H, CRC00 = 05H



(2) Measuring the pulse width by using one input signal of the TI000 pin (free-running timer mode)

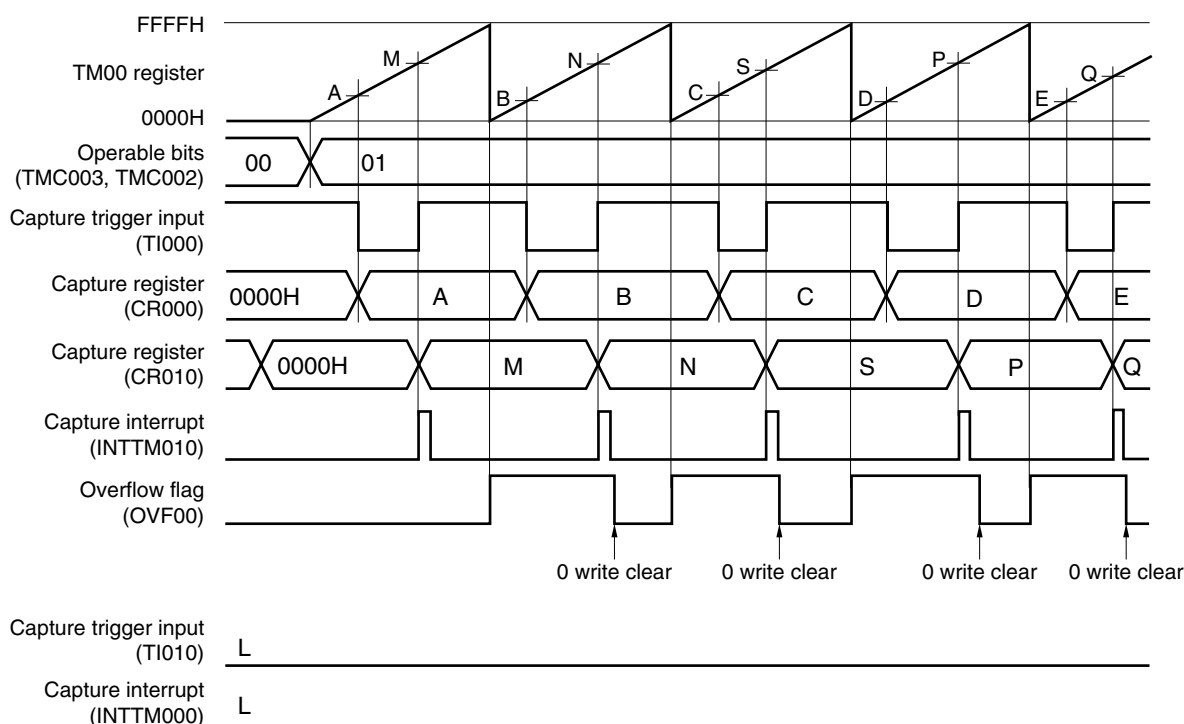
Set the free-running timer mode (TMC003 and TMC002 = 01). The count value of TM00 is captured to CR000 in the phase reverse to the valid edge detected on the TI000 pin. When the valid edge of the TI000 pin is detected, the count value of TM00 is captured to CR010.

By this measurement method, values are stored in separate capture registers when a width from one edge to another is measured. Therefore, the capture values do not have to be saved. By subtracting the value of one capture register from that of another, a high-level width, low-level width, and cycle are calculated.

If an overflow occurs, the value becomes negative if one captured value is simply subtracted from another and, therefore, a borrow occurs (bit 0 (CY) of the program status word (PSW) is set to 1). If this happens, ignore CY and take the calculated value as the pulse width. In addition, clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-49. Timing Example of Pulse Width Measurement (2)

• TMC00 = 04H, PRM00 = 10H, CRC00 = 07H

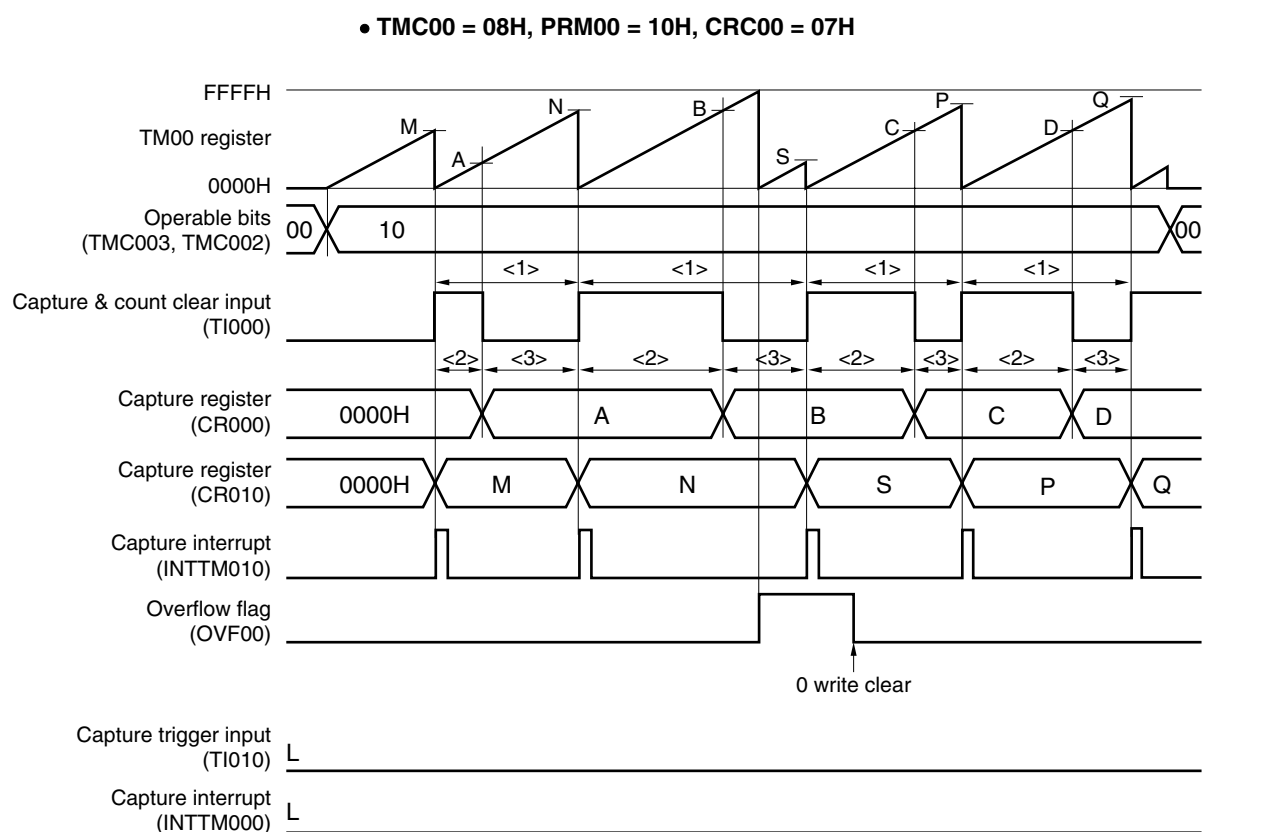


(3) Measuring the pulse width by using one input signal of the TI000 pin (clear & start mode entered by the TI000 pin valid edge input)

Set the clear & start mode entered by the TI000 pin valid edge (TMC003 and TMC002 = 10). The count value of TM00 is captured to CR000 in the phase reverse to the valid edge of the TI000 pin, and the count value of TM00 is captured to CR010 and TM00 is cleared (0000H) when the valid edge of the TI000 pin is detected. Therefore, a cycle is stored in CR010 if TM00 does not overflow.

If an overflow occurs, take the value that results from adding 10000H to the value stored in CR010 as a cycle. Clear bit 0 (OVF00) of 16-bit timer mode control register 00 (TMC00) to 0.

Figure 6-50. Timing Example of Pulse Width Measurement (3)



<1> Pulse cycle = $(10000H \times \text{Number of times OVF00 bit is set to 1} + \text{Captured value of CR010}) \times \text{Count clock cycle}$

<2> High-level pulse width = $(10000H \times \text{Number of times OVF00 bit is set to 1} + \text{Captured value of CR000}) \times \text{Count clock cycle}$

<3> Low-level pulse width = (Pulse cycle – High-level pulse width)

Figure 6-51. Example of Register Settings for Pulse Width Measurement (1/2)

(a) 16-bit timer mode control register 00 (TMC00)

| | | | | TMC003 | TMC002 | TMC001 | OVF00 |
|---|---|---|---|--------|--------|--------|-------|
| 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0 | 0 |

01: Free running timer mode
10: Clear and start mode entered by valid edge of T1000 pin.

(b) Capture/compare control register 00 (CRC00)

| | | | | CRC002 | | CRC001 | CRC000 |
|---|---|---|---|--------|---|--------|--------|
| 0 | 0 | 0 | 0 | 0 | 1 | 0/1 | 1 |

1: CR000 used as capture register
0: T1010 pin is used as capture trigger of CR000.
1: Reverse phase of T1000 pin is used as capture trigger of CR000.
1: CR010 used as capture register

(c) 16-bit timer output control register 00 (TOC00)

| OSPT00 | OSPE00 | TOC004 | LVS00 | LVR00 | TOC001 | TOE00 |
|--------|--------|--------|-------|-------|--------|-------|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(d) Prescaler mode register 00 (PRM00)

| ES110 | ES100 | ES010 | ES000 | 3 | 2 | PRM001 | PRM000 |
|-------|-------|-------|-------|---|---|--------|--------|
| 0/1 | 0/1 | 0/1 | 0/1 | 0 | 0 | 0/1 | 0/1 |

Selects count clock (setting valid edge of T1000 is prohibited)
00: Falling edge detection
01: Rising edge detection
10: Setting prohibited
11: Both edges detection (setting when CRC001 = 1 is prohibited)
00: Falling edge detection
01: Rising edge detection
10: Setting prohibited
11: Both edges detection

Figure 6-51. Example of Register Settings for Pulse Width Measurement (2/2)**(e) 16-bit timer counter 00 (TM00)**

By reading TM00, the count value can be read.

(f) 16-bit capture/compare register 000 (CR000)

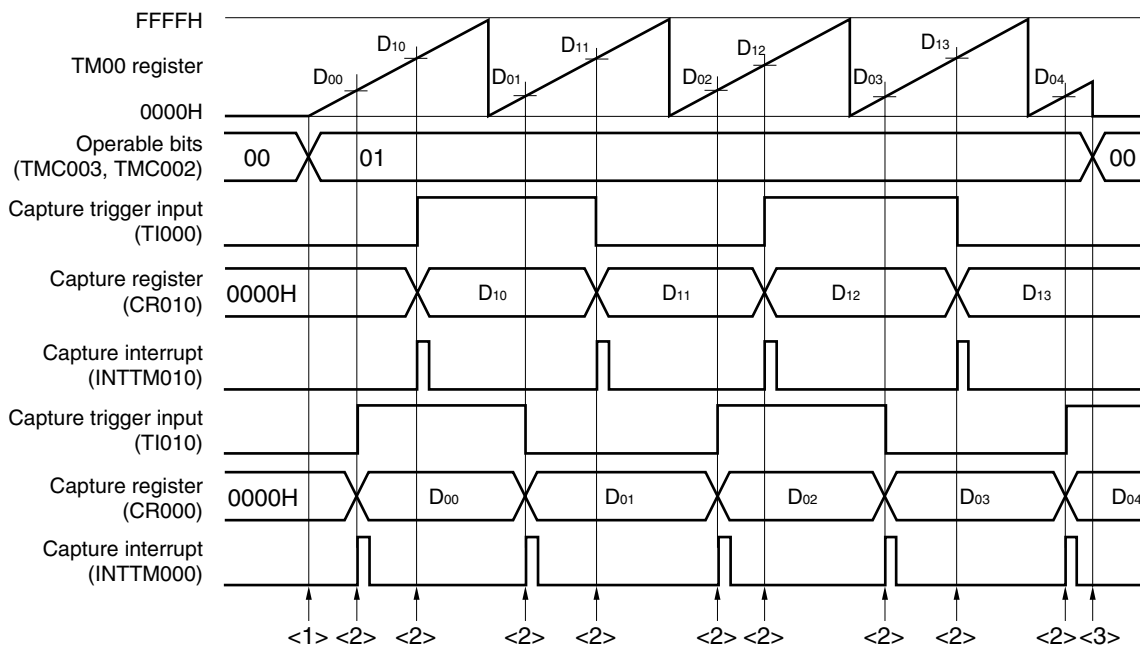
This register is used as a capture register. Either the TI000 or TI010 pin is selected as a capture trigger. When a specified edge of the capture trigger is detected, the count value of TM00 is stored in CR000.

(g) 16-bit capture/compare register 010 (CR010)

This register is used as a capture register. The signal input to the TI000 pin is used as a capture trigger. When the capture trigger is detected, the count value of TM00 is stored in CR010.

Figure 6-52. Example of Software Processing for Pulse Width Measurement (1/2)

(a) Example of free-running timer mode



(b) Example of clear & start mode entered by TI000 pin valid edge

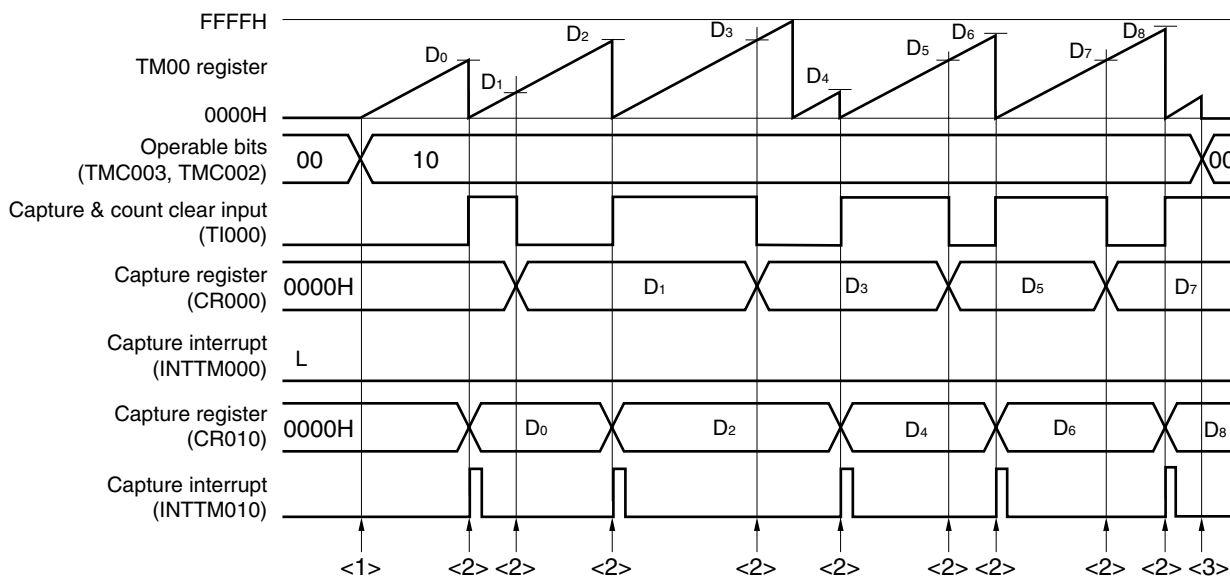
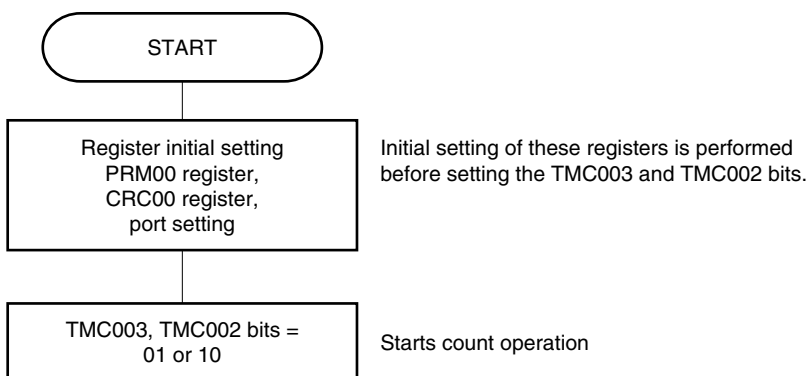
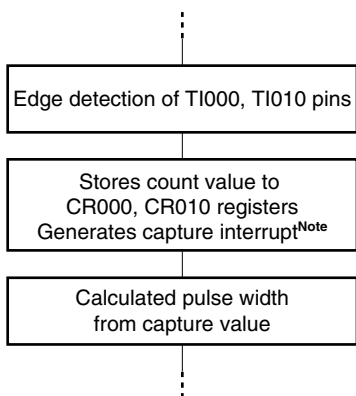


Figure 6-52. Example of Software Processing for Pulse Width Measurement (2/2)

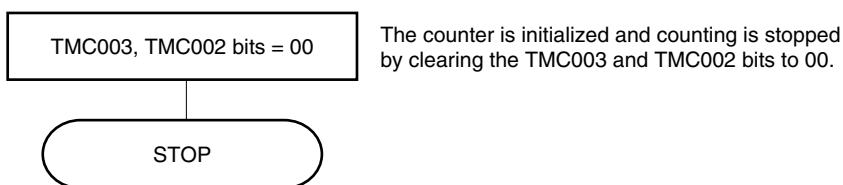
<1> Count operation start flow



<2> Capture trigger input flow



<3> Count operation stop flow



Note The capture interrupt signal (INTTM000) is not generated when the reverse-phase edge of the TI000 pin input is selected to the valid edge of CR000.

6.5 Special Use of TM00

6.5.1 Rewriting CR010 during TM00 operation

In principle, rewriting CR000 and CR010 of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B when they are used as compare registers is prohibited while TM00 is operating (TMC003 and TMC002 = other than 00).

However, the value of CR010 can be changed, even while TM00 is operating, using the following procedure if CR010 is used for PPG output and the duty factor is changed. (When changing the value of CR010 to a smaller value than the current one, rewrite it immediately after its value matches the value of TM00. When changing the value of CR010 to a larger value than the current one, rewrite it immediately after the values of CR000 and TM00 match. If the value of CR010 is rewritten immediately before a match between CR010 and TM00, or between CR000 and TM00, an unexpected operation may be performed.).

Procedure for changing value of CR010

- <1> Disable interrupt INTTM010 (TMMK010 = 1).
- <2> Disable reversal of the timer output when the value of TM00 matches that of CR010 (TOC004 = 0).
- <3> Change the value of CR010.
- <4> Wait for one cycle of the count clock of TM00.
- <5> Enable reversal of the timer output when the value of TM00 matches that of CR010 (TOC004 = 1).
- <6> Clear the interrupt flag of INTTM010 (TMIF010 = 0) to 0.
- <7> Enable interrupt INTTM010 (TMMK010 = 0).

Remark For TMIF010 and TMMK010, refer to **CHAPTER 16 INTERRUPT FUNCTIONS**.

6.5.2 Setting LVS00 and LVR00

(1) Usage of LVS00 and LVR00

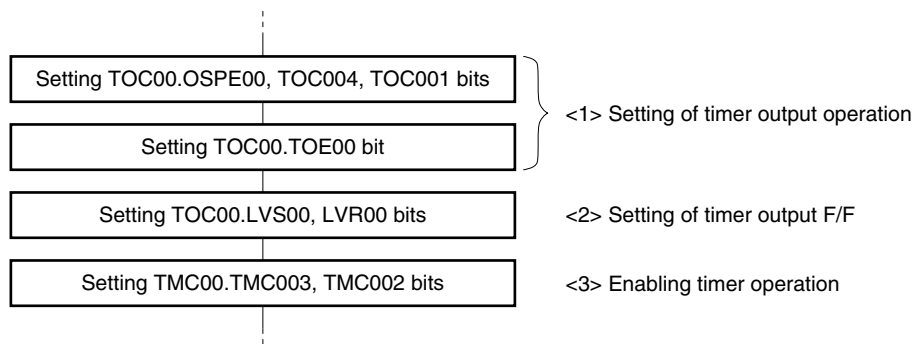
LVS00 and LVR00 are used to set the default value of the TO00 output and to invert the timer output without enabling the timer operation (TMC003 and TMC002 = 00). Clear LVS00 and LVR00 to 00 (default value: low-level output) when software control is unnecessary.

| LVS00 | LVR00 | Timer Output Status |
|-------|-------|--------------------------------|
| 0 | 0 | Not changed (low-level output) |
| 0 | 1 | Cleared (low-level output) |
| 1 | 0 | Set (high-level output) |
| 1 | 1 | Setting prohibited |

(2) Setting LVS00 and LVR00

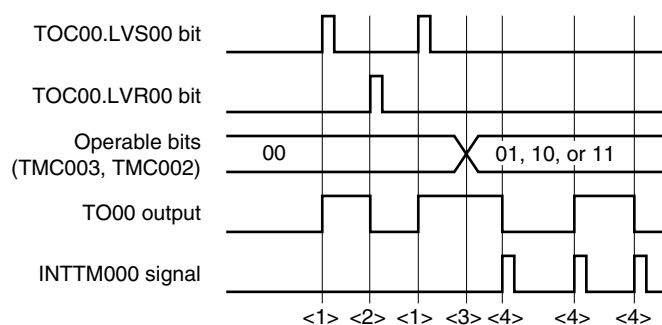
Set LVS00 and LVR00 using the following procedure.

Figure 6-53. Example of Flow for Setting LVS00 and LVR00 Bits



Caution Be sure to set LVS00 and LVR00 following steps <1>, <2>, and <3> above. Step <2> can be performed after <1> and before <3>.

Figure 6-54. Timing Example of LVR00 and LVS00



- <1> The TO00 output goes high when LVS00 and LVR00 = 10.
- <2> The TO00 output goes low when LVS00 and LVR00 = 01 (the pin output remains unchanged from the high level even if LVS00 and LVR00 are cleared to 00).
- <3> The timer starts operating when TMC003 and TMC002 are set to 01, 10, or 11. Because LVS00 and LVR00 were set to 10 before the operation was started, the TO00 output starts from the high level. After the timer starts operating, setting LVS00 and LVR00 is prohibited until TMC003 and TMC002 = 00 (disabling the timer operation).
- <4> The TO00 output level is inverted each time an interrupt signal (INTTM000) is generated.

6.6 Cautions for 16-Bit Timer/Event Counter 00

(1) Restrictions for each channel of 16-bit timer/event counter 00

Table 6-3 shows the restrictions for each channel.

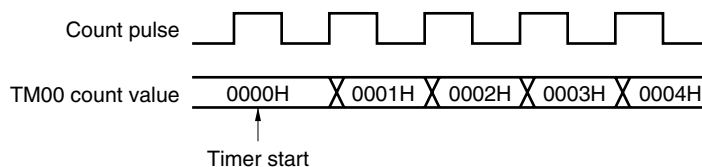
Table 6-3. Restrictions for Each Channel of 16-Bit Timer/Event Counter 00

| Operation | Restriction |
|---|--|
| As interval timer | - |
| As square-wave output | |
| As external event counter | |
| As clear & start mode entered by TI000 pin valid edge input | Using timer output (TO00) is prohibited when detection of the valid edge of the TI010 pin is used. (TOC00 = 00H) |
| As free-running timer | - |
| As PPG output | $0000H \leq CP010 < CR000 \leq FFFFH$ |
| As one-shot pulse output | Setting the same value to CR000 and CP010 is prohibited. |
| As pulse width measurement | Using timer output (TO00) is prohibited (TOC00 = 00H) |

(2) Timer start errors

An error of up to one clock may occur in the time required for a match signal to be generated after timer start. This is because counting TM00 is started asynchronously to the count pulse.

Figure 6-55. Start Timing of TM00 Count



(3) Setting of CR000 and CR010 (clear & start mode entered upon a match between TM00 and CR000)

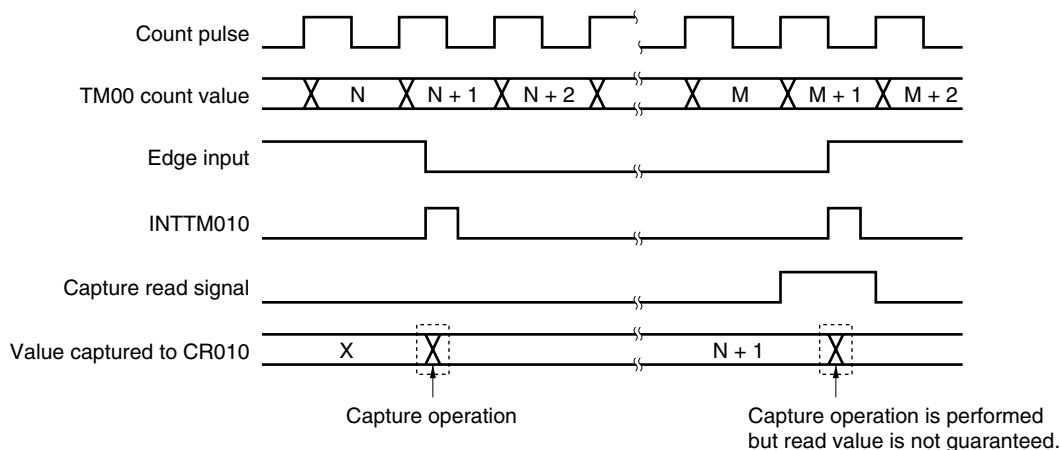
Set a value other than 0000H to CR000 and CR010 (TM00 cannot count one pulse when it is used as an external event counter).

(4) Timing of holding data by capture register

- (a) When the valid edge is input to the TI000/TI010 pin and the reverse phase of the TI000 pin is detected while CR000/CR010 is read, CR010 performs a capture operation but the read value of CR000/CR010 is not guaranteed. At this time, an interrupt signal (INTTM000/INTTM010) is generated when the valid edge of the TI000/TI010 pin is detected (the interrupt signal is not generated when the reverse-phase edge of the TI000 pin is detected).

When the count value is captured because the valid edge of the TI000/TI010 pin was detected, read the value of CR000/CR010 after INTTM000/INTTM010 is generated.

Figure 6-56. Timing of Holding Data by Capture Register



- (b) The values of CR000 and CR010 are not guaranteed after 16-bit timer/event counter 00 stops.

(5) Setting valid edge

Set the valid edge of the TI000 pin while the timer operation is stopped (TMC003 and TMC002 = 00). Set the valid edge by using ES000 and ES010.

(6) Re-triggering one-shot pulse

Make sure that the trigger is not generated while an active level is being output in the one-shot pulse output mode. Be sure to input the next trigger after the current active level is output.

(7) Operation of OVF00 flag**(a) Setting OVF00 flag (1)**

The OVF00 flag is set to 1 in the following case, as well as when TM00 overflows.

Select the clear & start mode entered upon a match between TM00 and CR000.

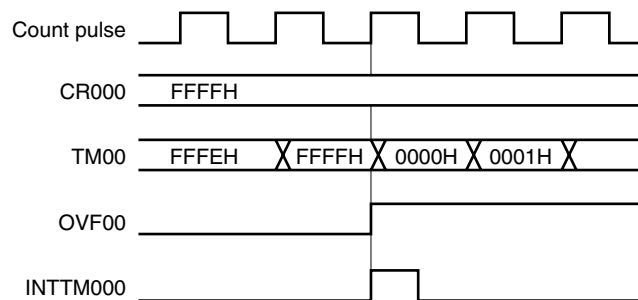
↓

Set CR000 to FFFFH.

↓

When TM00 matches CR000 and TM00 is cleared from FFFFH to 0000H

Figure 6-57. Operation Timing of OVF00 Flag

**(b) Clearing OVF00 flag**

Even if the OVF00 flag is cleared to 0 after TM00 overflows and before the next count clock is counted (before the value of TM00 becomes 0001H), it is set to 1 again and clearing is invalid.

(8) One-shot pulse output

One-shot pulse output operates correctly in the free-running timer mode or the clear & start mode entered by the TI000 pin valid edge. The one-shot pulse cannot be output in the clear & start mode entered upon a match between TM00 and CR000.

(9) Capture operation

(a) When valid edge of TI000 is specified as count clock

When the valid edge of TI000 is specified as the count clock, the capture register for which TI000 is specified as a trigger does not operate correctly.

(b) Pulse width to accurately capture value by signals input to TI010 and TI000 pins

To accurately capture the count value, the pulse input to the TI000 and TI010 pins as a capture trigger must be wider than two count clocks selected by PRM00 (refer to **Figure 6-7**).

(c) Generation of interrupt signal

The capture operation is performed at the falling edge of the count clock but the interrupt signals (INTTM000 and INTTM010) are generated at the rising edge of the next count clock (refer to **Figure 6-7**).

(d) Note when CRC001 (bit 1 of capture/compare control register 00 (CRC00)) is set to 1

When the count value of the TM00 register is captured to the CR000 register in the phase reverse to the signal input to the TI000 pin, the interrupt signal (INTTM000) is not generated after the count value is captured. If the valid edge is detected on the TI010 pin during this operation, the capture operation is not performed but the INTTM000 signal is generated as an external interrupt signal. Mask the INTTM000 signal when the external interrupt is not used.

(10) Edge detection

(a) Specifying valid edge after reset

If the operation of the 16-bit timer/event counter 00 is enabled after reset and while the TI000 or TI010 pin is at high level and when the rising edge or both the edges are specified as the valid edge of the TI000 or TI010 pin, then the high level of the TI000 or TI010 pin is detected as the rising edge. Note this when the TI000 or TI010 pin is pulled up. However, the rising edge is not detected when the operation is once stopped and then enabled again.

(b) Sampling clock for eliminating noise

The sampling clock for eliminating noise differs depending on whether the valid edge of TI000 is used as the count clock or capture trigger. In the former case, the sampling clock is fixed to f_{PRS} . In the latter, the count clock selected by PRM00 is used for sampling.

When the signal input to the TI000 pin is sampled and the valid level is detected two times in a row, the valid edge is detected. Therefore, noise having a short pulse width can be eliminated (refer to **Figure 6-7**).

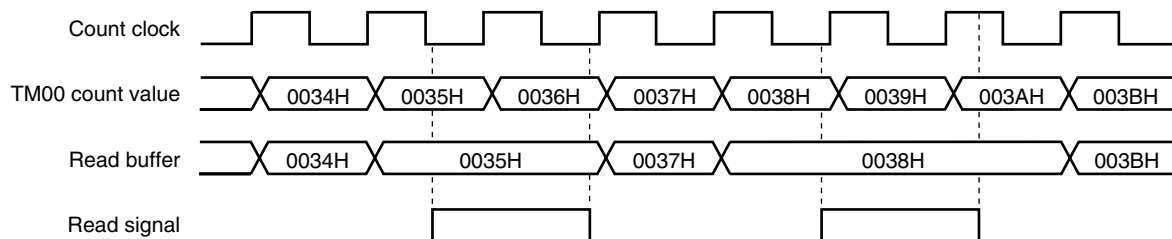
(11) Timer operation

The signal input to the TI000/TI010 pin is not acknowledged while the timer is stopped, regardless of the operation mode of the CPU.

Remark f_{PRS} : Peripheral hardware clock frequency

(12) Reading of 16-bit timer counter 00 (TM00)

TM00 can be read without stopping the actual counter, because the count values captured to the buffer are fixed when it is read. The buffer, however, may not be updated when it is read immediately before the counter counts up, because the buffer is updated at the timing the counter counts up.

Figure 6-58. 16-bit Timer Counter 00 (TM00) Read Timing

CHAPTER 7 8-BIT TIMER/EVENT COUNTERS 50 AND 51

7.1 Functions of 8-Bit Timer/Event Counters 50 and 51

8-bit timer/event counters 50 and 51 have the following functions.

- Interval timer
- External event counter
- Square-wave output
- PWM output

7.2 Configuration of 8-Bit Timer/Event Counters 50 and 51

8-bit timer/event counters 50 and 51 include the following hardware.

Table 7-1. Configuration of 8-Bit Timer/Event Counters 50 and 51

| Item | Configuration |
|-------------------|---|
| Timer register | 8-bit timer counter 5n (TM5n) |
| Register | 8-bit timer compare register 5n (CR5n) |
| Timer input | TI5n |
| Timer output | TO5n |
| Control registers | Timer clock selection register 5n (TCL5n) 8-bit timer mode control register 5n (TMC5n) Port mode register 1 (PM1) or port mode register 3 (PM3) Port register 1 (P1) or port register 3 (P3) |

Figures 7-1 and 7-2 show the block diagrams of 8-bit timer/event counters 50 and 51.

Figure 7-1. Block Diagram of 8-Bit Timer/Event Counter 50

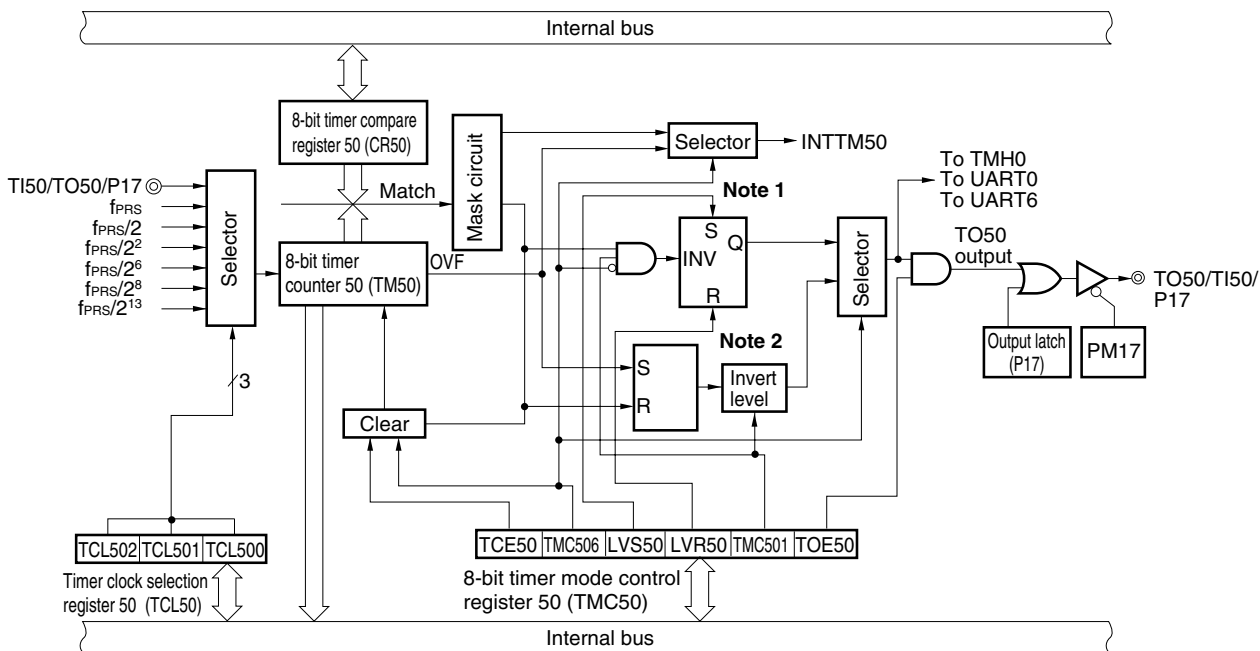
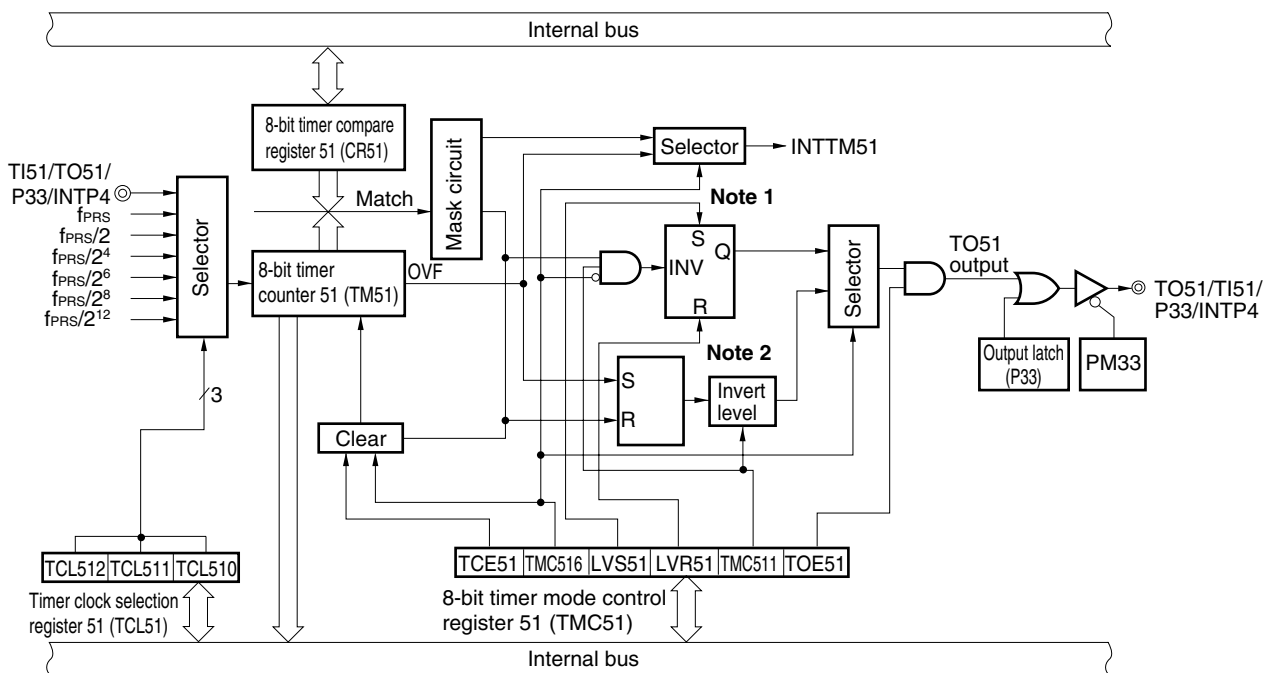


Figure 7-2. Block Diagram of 8-Bit Timer/Event Counter 51



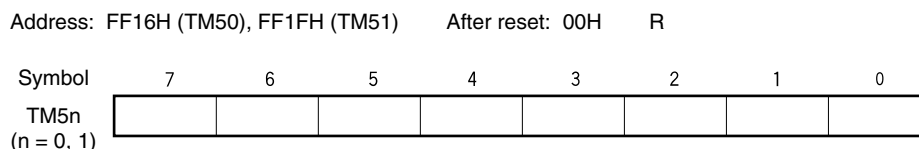
- Notes**
1. Timer output F/F
 2. PWM output F/F

(1) 8-bit timer counter 5n (TM5n)

TM5n is an 8-bit register that counts the count pulses and is read-only.

The counter is incremented in synchronization with the rising edge of the count clock.

Figure 7-3. Format of 8-Bit Timer Counter 5n (TM5n)



In the following situations, the count value is cleared to 00H.

<1> Reset signal generation

<2> When TCE5n is cleared

<3> When TM5n and CR5n match in the mode in which clear & start occurs upon a match of the TM5n and CR5n.

(2) 8-bit timer compare register 5n (CR5n)

CR5n can be read and written by an 8-bit memory manipulation instruction.

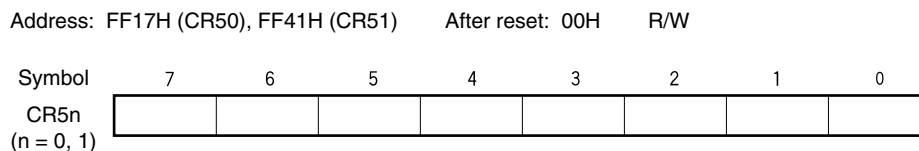
Except in PWM mode, the value set in CR5n is constantly compared with the 8-bit timer counter 5n (TM5n) count value, and an interrupt request (INTTM5n) is generated if they match.

In the PWM mode, TO5n output becomes inactive when the values of TM5n and CR5n match, but no interrupt is generated.

The value of CR5n can be set within 00H to FFH.

Reset signal generation clears CR5n to 00H.

Figure 7-4. Format of 8-Bit Timer Compare Register 5n (CR5n)



Cautions 1. In the mode in which clear & start occurs on a match of TM5n and CR5n (TMC5n6 = 0), do not write other values to CR5n during operation.

2. In PWM mode, make the CR5n rewrite period 3 count clocks of the count clock (clock selected by TCL5n) or more.

Remark n = 0, 1

7.3 Registers Controlling 8-Bit Timer/Event Counters 50 and 51

The following four registers are used to control 8-bit timer/event counters 50 and 51.

- Timer clock selection register 5n (TCL5n)
- 8-bit timer mode control register 5n (TMC5n)
- Port mode register 1 (PM1) or port mode register 3 (PM3)
- Port register 1 (P1) or port register 3 (P3)

(1) Timer clock selection register 5n (TCL5n)

This register sets the count clock of 8-bit timer/event counter 5n and the valid edge of the TI5n pin input.

TCL5n can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears TCL5n to 00H.

Remark n = 0, 1

Figure 7-5. Format of Timer Clock Selection Register 50 (TCL50)

Address: FF6AH After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|--------|--------|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TCL50 | 0 | 0 | 0 | 0 | 0 | TCL502 | TCL501 | TCL500 |

| TCL502 | TCL501 | TCL500 | Count clock selection | | | |
|--------|--------|--------|---------------------------------------|---------------------------|----------------------------|------------|
| | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ | |
| 0 | 0 | 0 | TI50 pin falling edge ^{Note} | | | |
| 0 | 0 | 1 | TI50 pin rising edge ^{Note} | | | |
| 0 | 1 | 0 | f_{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 1 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 1 | 0 | 0 | $f_{PRS}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz |
| 1 | 0 | 1 | $f_{PRS}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 1 | 1 | 0 | $f_{PRS}/2^8$ | 7.81 kHz | 19.53 kHz | 39.06 kHz |
| 1 | 1 | 1 | $f_{PRS}/2^{13}$ | 0.24 kHz | 0.61 kHz | 1.22 kHz |

Note Do not start timer operation with the external clock from the TI50 pin when in the STOP mode.

- Cautions**
1. When rewriting TCL50 to other data, stop the timer operation beforehand.
 2. Be sure to clear bits 3 to 7 to "0".

Remark f_{PRS} : Peripheral hardware clock frequency

Figure 7-6. Format of Timer Clock Selection Register 51 (TCL51)

Address: FF8CH After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|--------|--------|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TCL51 | 0 | 0 | 0 | 0 | 0 | TCL512 | TCL511 | TCL510 |

| TCL512 | TCL511 | TCL510 | Count clock selection | | | |
|--------|--------|--------|---------------------------------------|---------------------------|----------------------------|------------|
| | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ | |
| 0 | 0 | 0 | TI51 pin falling edge ^{Note} | | | |
| 0 | 0 | 1 | TI51 pin rising edge ^{Note} | | | |
| 0 | 1 | 0 | f_{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 1 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 1 | 0 | 0 | $f_{PRS}/2^4$ | 125 kHz | 312.5 kHz | 625 kHz |
| 1 | 0 | 1 | $f_{PRS}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 1 | 1 | 0 | $f_{PRS}/2^8$ | 7.81 kHz | 19.53 kHz | 39.06 kHz |
| 1 | 1 | 1 | $f_{PRS}/2^{12}$ | 0.49 kHz | 1.22 kHz | 2.44 kHz |

Note Do not start timer operation with the external clock from the TI51 pin when in the STOP mode.

Cautions 1. When rewriting TCL51 to other data, stop the timer operation beforehand.

2. Be sure to clear bits 3 to 7 to "0".

Remark f_{PRS} : Peripheral hardware clock frequency

(2) 8-bit timer mode control register 5n (TMC5n)

TMC5n is a register that performs the following five types of settings.

- <1> 8-bit timer counter 5n (TM5n) count operation control
- <2> 8-bit timer counter 5n (TM5n) operating mode selection
- <3> Timer output F/F (flip flop) status setting
- <4> Active level selection in timer F/F control or PWM (free-running) mode.
- <5> Timer output control

TMC5n can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Remark n = 0, 1

Figure 7-7. Format of 8-Bit Timer Mode Control Register 50 (TMC50)

Address: FF6BH After reset: 00H R/W^{Note}

| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | 1 | <0> |
|--------|---|--|---|---|--------------------------|-------|--------|-------|
| TMC50 | TCE50 | TMC506 | 0 | 0 | LVS50 | LVR50 | TMC501 | TOE50 |
| TCE50 | TM50 count operation control | | | | | | | |
| 0 | After clearing to 0, count operation disabled (counter stopped) | | | | | | | |
| 1 | Count operation start | | | | | | | |
| TMC506 | TM50 operating mode selection | | | | | | | |
| 0 | Mode in which clear & start occurs on a match between TM50 and CR50 | | | | | | | |
| 1 | PWM (free-running) mode | | | | | | | |
| LVS50 | LVR50 | Timer output F/F status setting | | | | | | |
| 0 | 0 | No change | | | | | | |
| 0 | 1 | Timer output F/F clear (0) (default value of TO50 output: low level) | | | | | | |
| 1 | 0 | Timer output F/F set (1) (default value of TO50 output: high level) | | | | | | |
| 1 | 1 | Setting prohibited | | | | | | |
| TMC501 | In other modes (TMC506 = 0) | | | | In PWM mode (TMC506 = 1) | | | |
| | Timer F/F control | | | | Active level selection | | | |
| 0 | Inversion operation disabled | | | | Active-high | | | |
| 1 | Inversion operation enabled | | | | Active-low | | | |
| TOE50 | Timer output control | | | | | | | |
| 0 | Output disabled (TO50 output is low level) | | | | | | | |
| 1 | Output enabled | | | | | | | |

Note Bits 2 and 3 are write-only.

(Cautions and Remarks are listed on the next page.)

Figure 7-8. Format of 8-Bit Timer Mode Control Register 51 (TMC51)

Address: FF43H After reset: 00H R/W^{Note}

| | | | | | | | | |
|--------|-------|--------|---|---|-------|-------|--------|-------|
| Symbol | <7> | 6 | 5 | 4 | <3> | <2> | 1 | <0> |
| TMC51 | TCE51 | TMC516 | 0 | 0 | LVS51 | LVR51 | TMC511 | TOE51 |

| | |
|-------|---|
| TCE51 | TM51 count operation control |
| 0 | After clearing to 0, count operation disabled (counter stopped) |
| 1 | Count operation start |

| | |
|--------|---|
| TMC516 | TM51 operating mode selection |
| 0 | Mode in which clear & start occurs on a match between TM51 and CR51 |
| 1 | PWM (free-running) mode |

| | | |
|-------|-------|--|
| LVS51 | LVR51 | Timer output F/F status setting |
| 0 | 0 | No change |
| 0 | 1 | Timer output F/F clear (0) (default value of TO51 output: low) |
| 1 | 0 | Timer output F/F set (1) (default value of TO51 output: high) |
| 1 | 1 | Setting prohibited |

| | | |
|--------|------------------------------|--------------------------|
| TMC511 | In other modes (TMC516 = 0) | In PWM mode (TMC516 = 1) |
| | Timer F/F control | |
| 0 | Inversion operation disabled | Active-high |
| 1 | Inversion operation enabled | Active-low |

| | |
|-------|--|
| TOE51 | Timer output control |
| 0 | Output disabled (TO51 output is low level) |
| 1 | Output enabled |

Note Bits 2 and 3 are write-only.

- Cautions**
- The settings of LVS5n and LVR5n are valid in other than PWM mode.
 - Perform <1> to <4> below in the following order, not at the same time.
 - <1> Set TMC5n1, TMC5n6: Operation mode setting
 - <2> Set TOE5n to enable output: Timer output enable
 - <3> Set LVS5n, LVR5n (see Caution 1): Timer F/F setting
 - <4> Set TCE5n
 - When TCE5n = 1, setting the other bits of TMC5n is prohibited.
 - The actual TO50/TI50/P17 and TO51/TI51/P33/INTP4 pin outputs are determined depending on PM17 and P17, and PM33 and P33, besides TO5n output.

- Remarks**
- In PWM mode, PWM output is made inactive by clearing TCE5n to 0.
 - If LVS5n and LVR5n are read, the value is 0.
 - The values of the TMC5n6, LVS5n, LVR5n, TMC5n1, and TOE5n bits are reflected at the TO5n output regardless of the value of TCE5n.
 - n = 0, 1

(3) Port mode registers 1 and 3 (PM1, PM3)

These registers set port 1 and 3 input/output in 1-bit units.

When using the P17/TO50/TI50 and P33/TO51/TI51/INTP4 pins for timer output, clear PM17 and PM33 and the output latches of P17 and P33 to 0.

When using the P17/TO50/TI50 and P33/TO51/TI51/INTP4 pins for timer input, set PM17 and PM33 to 1. The output latches of P17 and P33 at this time may be 0 or 1.

PM1 and PM3 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 7-9. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

| | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |

| | |
|------|---|
| PM1n | P1n pin I/O mode selection (n = 0 to 7) |
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

Figure 7-10. Format of Port Mode Register 3 (PM3)

Address: FF23H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|---|---|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM3 | 1 | 1 | 1 | 1 | PM33 | PM32 | PM31 | PM30 |

| | |
|------|---|
| PM3n | P3n pin I/O mode selection (n = 0 to 3) |
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

7.4 Operations of 8-Bit Timer/Event Counters 50 and 51

7.4.1 Operation as interval timer

8-bit timer/event counter 5n operates as an interval timer that generates interrupt requests repeatedly at intervals of the count value preset to 8-bit timer compare register 5n (CR5n).

When the count value of 8-bit timer counter 5n (TM5n) matches the value set to CR5n, counting continues with the TM5n value cleared to 0 and an interrupt request signal (INTTM5n) is generated.

The count clock of TM5n can be selected with bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock selection register 5n (TCL5n).

Setting

<1> Set the registers.

- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on a match of TM5n and CR5n.
(TMC5n = 0000xxx0B x = Don't care)

<2> After TCE5n = 1 is set, the count operation starts.

<3> If the values of TM5n and CR5n match, INTTM5n is generated (TM5n is cleared to 00H).

<4> INTTM5n is generated repeatedly at the same interval.

Set TCE5n to 0 to stop the count operation.

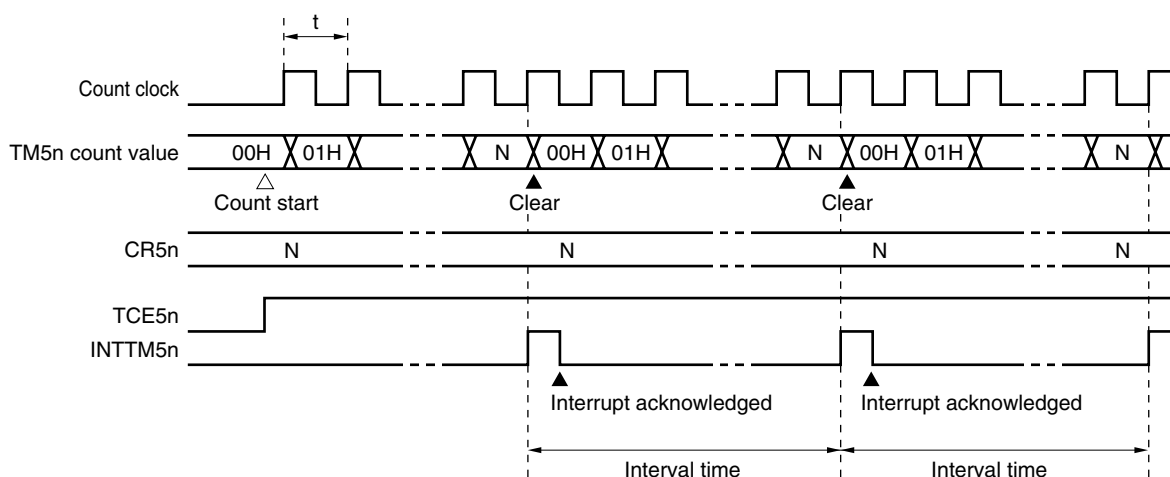
Caution Do not write other values to CR5n during operation.

Remarks 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 16 INTERRUPT FUNCTIONS**.

2. n = 0, 1

Figure 7-11. Interval Timer Operation Timing (1/2)

(a) Basic operation



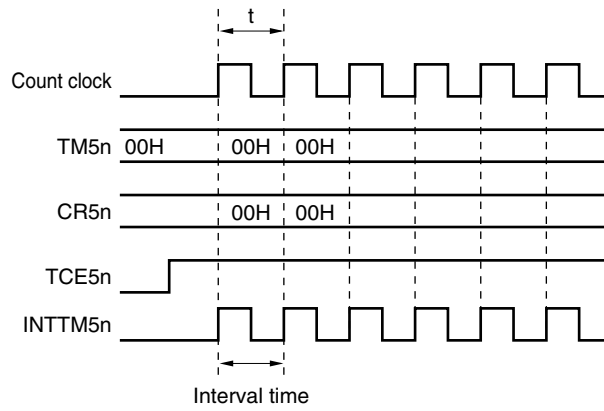
Remark Interval time = $(N + 1) \times t$

N = 01H to FFH

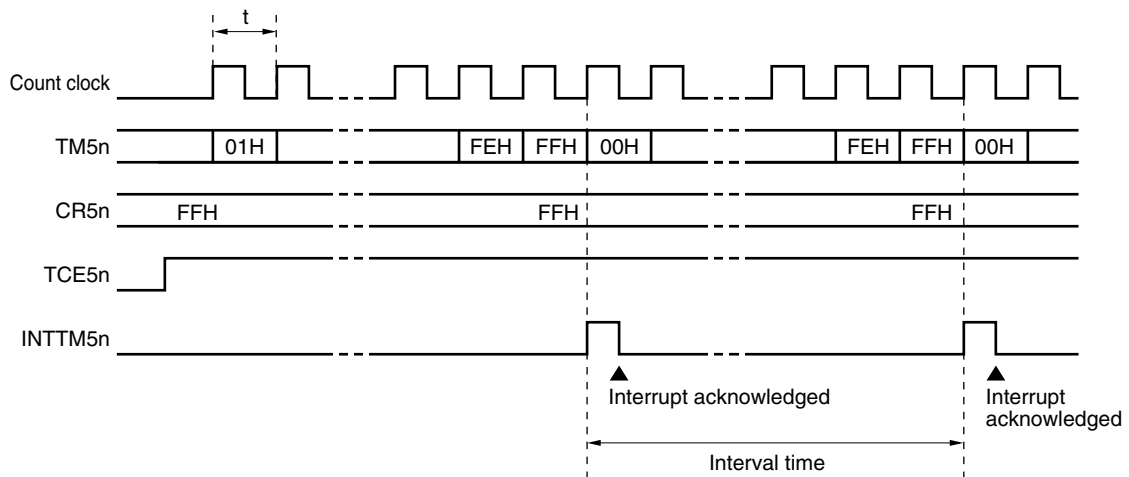
n = 0, 1

Figure 7-11. Interval Timer Operation Timing (2/2)

(b) When CR5n = 00H



(c) When CR5n = FFH



Remark n = 0, 1

7.4.2 Operation as external event counter

The external event counter counts the number of external clock pulses to be input to the TI5n pin by 8-bit timer counter 5n (TM5n).

TM5n is incremented each time the valid edge specified by timer clock selection register 5n (TCL5n) is input. Either the rising or falling edge can be selected.

When the TM5n count value matches the value of 8-bit timer compare register 5n (CR5n), TM5n is cleared to 0 and an interrupt request signal (INTTM5n) is generated.

Whenever the TM5n value matches the value of CR5n, INTTM5n is generated.

Setting

<1> Set each register.

- Set the port mode register (PM17 or PM33)^{Note} to 1.
- TCL5n: Select TI5n pin input edge.
TI5n pin falling edge → TCL5n = 00H
TI5n pin rising edge → TCL5n = 01H
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on match of TM5n and CR5n, disable the timer F/F inversion operation, disable timer output.
(TMC5n = 0000000B)

<2> When TCE5n = 1 is set, the number of pulses input from the TI5n pin is counted.

<3> When the values of TM5n and CR5n match, INTTM5n is generated (TM5n is cleared to 00H).

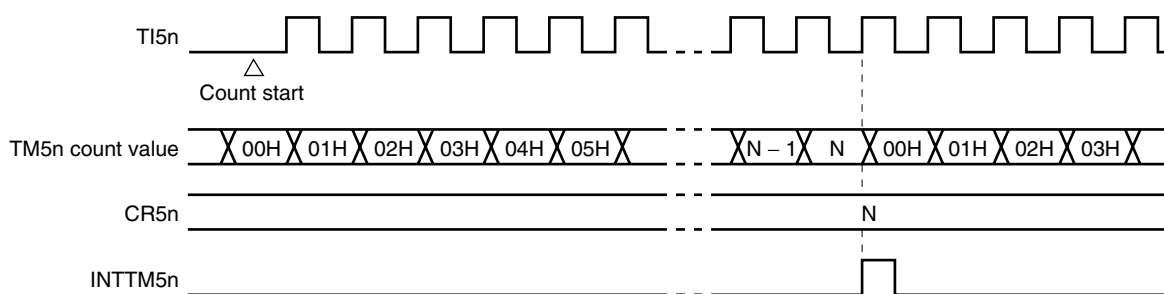
<4> After these settings, INTTM5n is generated each time the values of TM5n and CR5n match.

Note 8-bit timer/event counter 50: PM17

8-bit timer/event counter 51: PM33

Remark For how to enable the INTTM5n signal interrupt, see **CHAPTER 16 INTERRUPT FUNCTIONS**.

Figure 7-12. External Event Counter Operation Timing (with Rising Edge Specified)



Remark N = 00H to FFH

n = 0, 1

7.4.3 Square-wave output operation

A square wave with any selected frequency is output at intervals determined by the value preset to 8-bit timer compare register 5n (CR5n).

The TO5n pin output status is inverted at intervals determined by the count value preset to CR5n by setting bit 0 (TOE5n) of 8-bit timer mode control register 5n (TMC5n) to 1. This enables a square wave with any selected frequency to be output (duty = 50%).

Setting

<1> Set each register.

- Clear the port output latch (P17 or P33)^{Note} and port mode register (PM17 or PM33)^{Note} to 0.
- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select the mode in which clear & start occurs on a match of TM5n and CR5n.

| LVS5n | LVR5n | Timer Output F/F Status Setting |
|-------|-------|--|
| 0 | 1 | Timer output F/F clear (0) (default value of TO5n output: low level) |
| 1 | 0 | Timer output F/F set (1) (default value of TO5n output: high level) |

Timer output enabled

(TMC5n = 00001011B or 00000111B)

<2> After TCE5n = 1 is set, the count operation starts.

<3> The timer output F/F is inverted by a match of TM5n and CR5n. After INTTM5n is generated, TM5n is cleared to 00H.

<4> After these settings, the timer output F/F is inverted at the same interval and a square wave is output from TO5n. The frequency is as follows.

- Frequency = $1/2t(N + 1)$
(N: 00H to FFH)

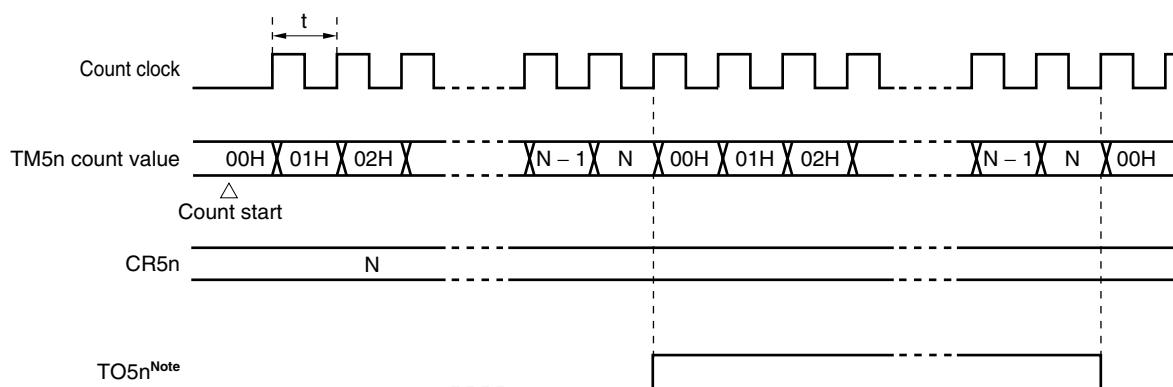
Note 8-bit timer/event counter 50: P17, PM17
8-bit timer/event counter 51: P33, PM33

Caution Do not write other values to CR5n during operation.

Remarks 1. For how to enable the INTTM5n signal interrupt, see **CHAPTER 16 INTERRUPT FUNCTIONS**.

2. n = 0, 1

Figure 7-13. Square-Wave Output Operation Timing



Note The initial value of TO5n output can be set by bits 2 and 3 (LVR5n, LVS5n) of 8-bit timer mode control register 5n (TMC5n).

7.4.4 PWM output operation

8-bit timer/event counter 5n operates as a PWM output when bit 6 (TMC5n6) of 8-bit timer mode control register 5n (TMC5n) is set to 1.

The duty pulse determined by the value set to 8-bit timer compare register 5n (CR5n) is output from TO5n.

Set the active level width of the PWM pulse to CR5n; the active level can be selected with bit 1 (TMC5n1) of TMC5n.

The count clock can be selected with bits 0 to 2 (TCL5n0 to TCL5n2) of timer clock selection register 5n (TCL5n).

PWM output can be enabled/disabled with bit 0 (TOE5n) of TMC5n.

Caution In PWM mode, make the CR5n rewrite period 3 count clocks of the count clock (clock selected by TCL5n) or more.

Remark n = 0, 1

(1) PWM output basic operation**Setting**

<1> Set each register.

- Clear the port output latch (P17 or P33)^{Note} and port mode register (PM17 or PM33)^{Note} to 0.
- TCL5n: Select the count clock.
- CR5n: Compare value
- TMC5n: Stop the count operation, select PWM mode.

The timer output F/F is not changed.

| TMC5n1 | Active Level Selection |
|--------|------------------------|
| 0 | Active-high |
| 1 | Active-low |

Timer output enabled

(TMC5n = 0100001B or 0100011B)

<2> The count operation starts when TCE5n = 1.
Clear TCE5n to 0 to stop the count operation.

Note 8-bit timer/event counter 50: P17, PM17
8-bit timer/event counter 51: P33, PM33

PWM output operation

- <1> PWM output (TO5n output) outputs an inactive level until an overflow occurs.
- <2> When an overflow occurs, the active level is output. The active level is output until CR5n matches the count value of 8-bit timer counter 5n (TM5n).
- <3> After the CR5n matches the count value, the inactive level is output until an overflow occurs again.
- <4> Operations <2> and <3> are repeated until the count operation stops.
- <5> When the count operation is stopped with TCE5n = 0, PWM output becomes inactive.

For details of timing, see **Figures 7-14** and **7-15**.

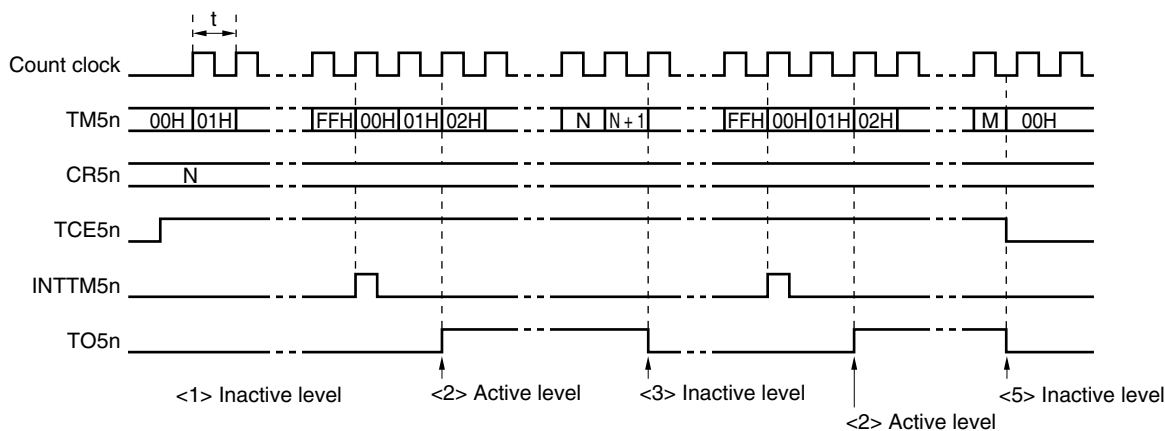
The cycle, active-level width, and duty are as follows.

- Cycle = $2^8 t$
- Active-level width = Nt
- Duty = $N/2^8$
(N = 00H to FFH)

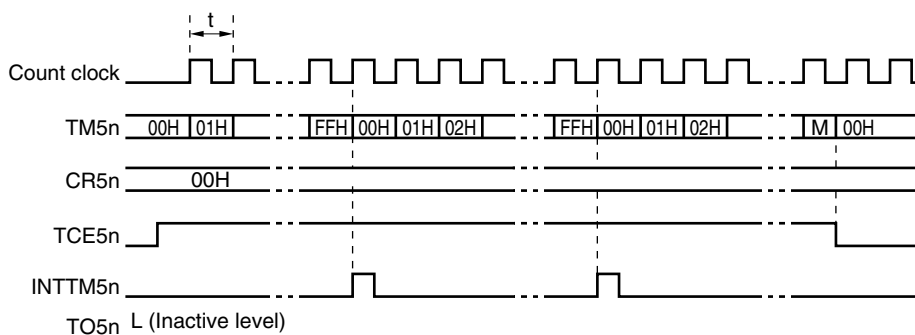
Remark n = 0, 1

Figure 7-14. PWM Output Operation Timing

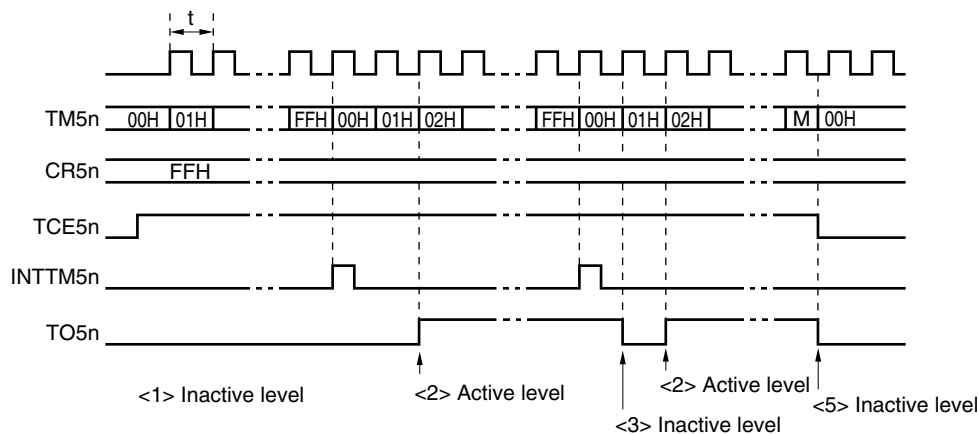
(a) Basic operation (active level = H)



(b) CR5n = 00H



(c) CR5n = FFH



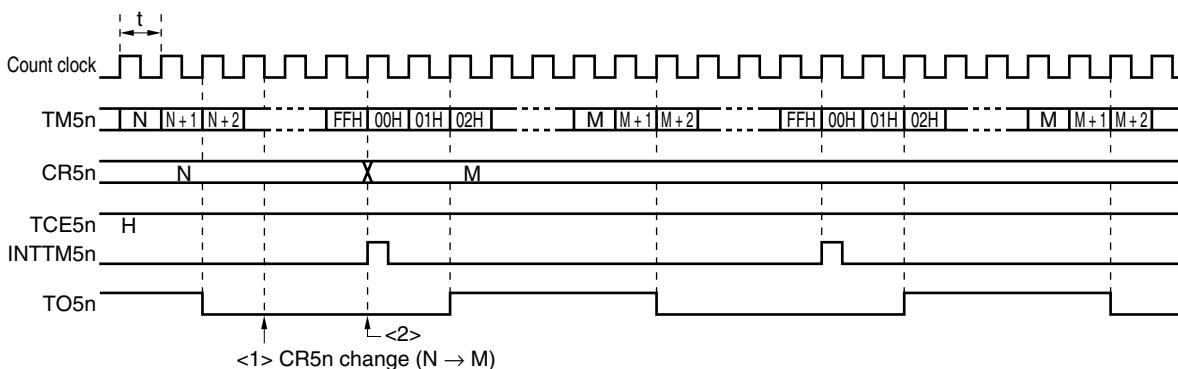
Remarks 1. <1> to <3> and <5> in Figure 7-14 (a) and (c) correspond to <1> to <3> and <5> in PWM output operation in 7.4.4 (1) PWM output basic operation.

2. n = 0, 1

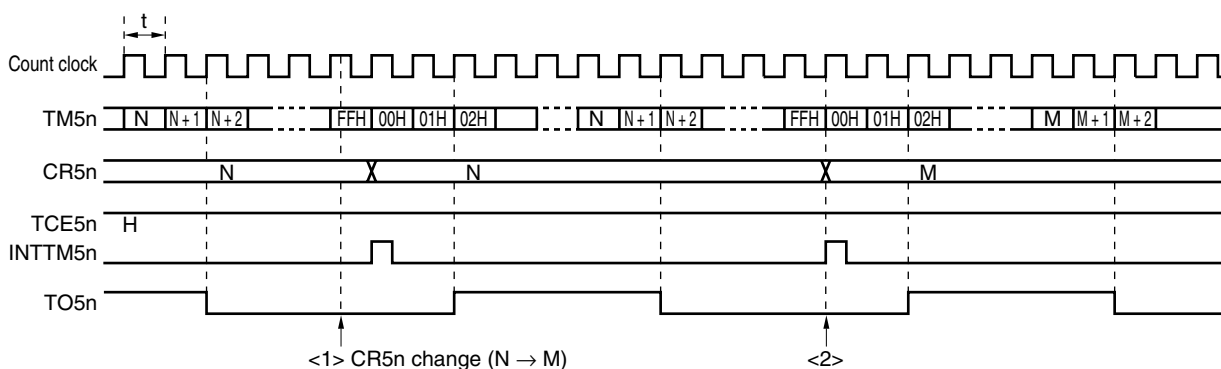
(2) Operation with CR5n changed

Figure 7-15. Timing of Operation with CR5n Changed

- (a) CR5n value is changed from N to M before clock rising edge of FFH
 → Value is transferred to CR5n at overflow immediately after change.



- (b) CR5n value is changed from N to M after clock rising edge of FFH
 → Value is transferred to CR5n at second overflow.



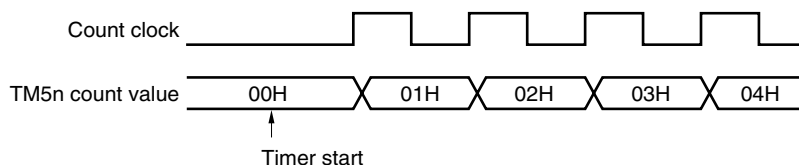
Caution When reading from CR5n between <1> and <2> in Figure 7-15, the value read differs from the actual value (read value: M, actual value of CR5n: N).

7.5 Cautions for 8-Bit Timer/Event Counters 50 and 51

(1) Timer start error

An error of up to one clock may occur in the time required for a match signal to be generated after timer start. This is because 8-bit timer counters 50 and 51 (TM50, TM51) are started asynchronously to the count clock.

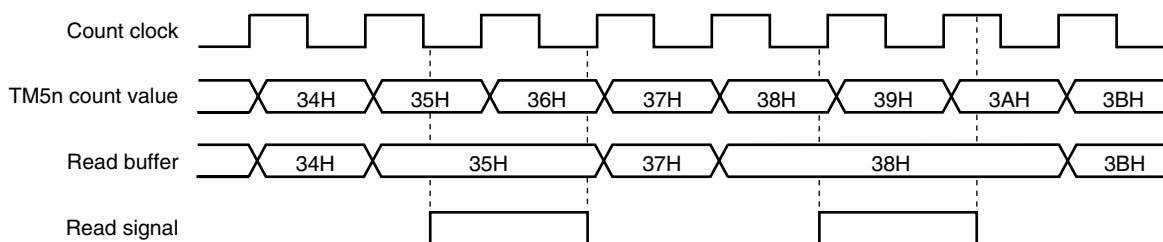
Figure 7-16. 8-Bit Timer Counter 5n (TM5n) Start Timing



(2) Reading of 8-bit timer counter 5n (TM5n)

TM5n can be read without stopping the actual counter, because the count values captured to the buffer are fixed when it is read. The buffer, however, may not be updated when it is read immediately before the counter counts up, because the buffer is updated at the timing the counter counts up.

Figure 7-17. 8-bit Timer Counter 5n (TM5n) Read Timing



Remark n = 0, 1

CHAPTER 8 8-BIT TIMERS H0 AND H1

8.1 Functions of 8-Bit Timers H0 and H1

8-bit timers H0 and H1 have the following functions.

- Interval timer
- Square-wave output
- PWM output
- Carrier generator (8-bit timer H1 only)

8.2 Configuration of 8-Bit Timers H0 and H1

8-bit timers H0 and H1 include the following hardware.

Table 8-1. Configuration of 8-Bit Timers H0 and H1

| Item | Configuration |
|-------------------|---|
| Timer register | 8-bit timer counter Hn |
| Registers | 8-bit timer H compare register 0n (CMP0n) 8-bit timer H compare register 1n (CMP1n) |
| Timer output | TOHn, output controller |
| Control registers | 8-bit timer H mode register n (TMHMDn) 8-bit timer H carrier control register 1 (TMCYC1) ^{Note} Port mode register 1 (PM1) Port register 1 (P1) |

Note 8-bit timer H1 only

Remark n = 0, 1

Figures 8-1 and 8-2 show the block diagrams.

Figure 8-1. Block Diagram of 8-Bit Timer H0

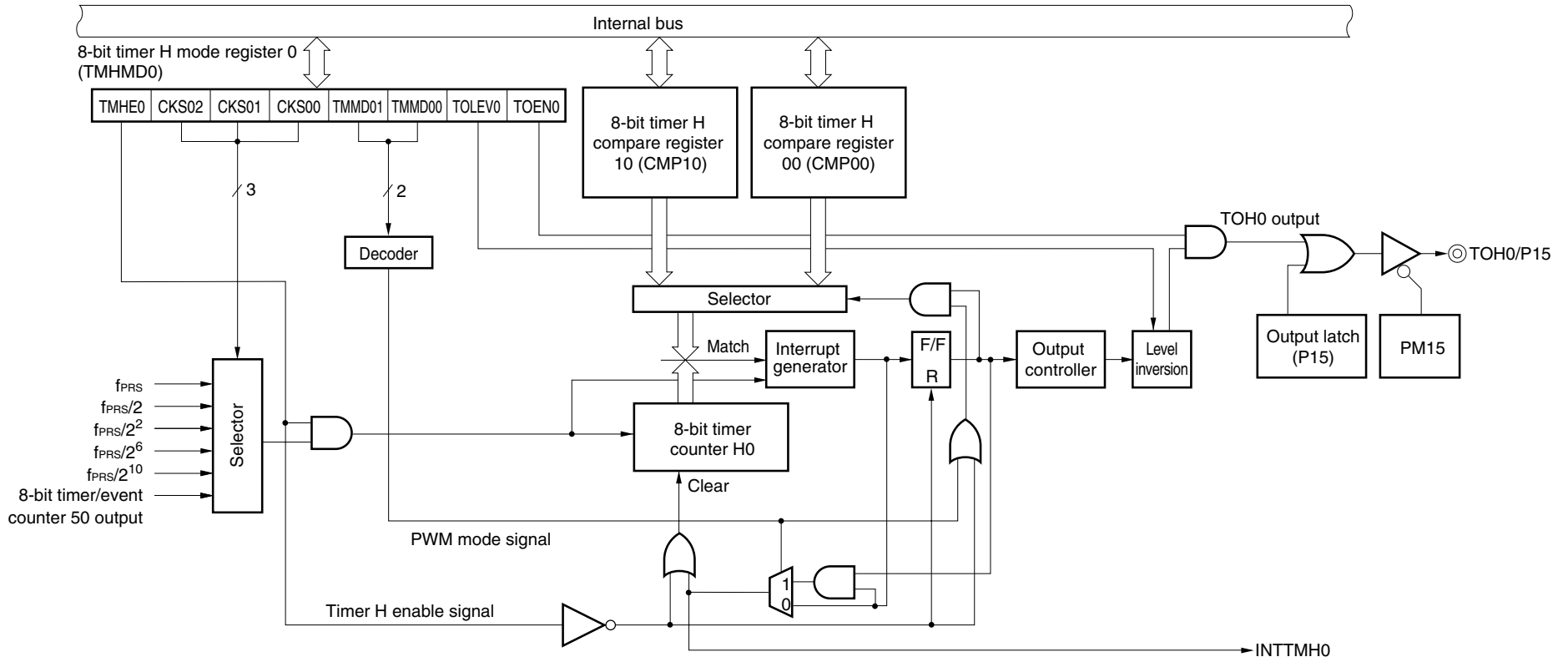
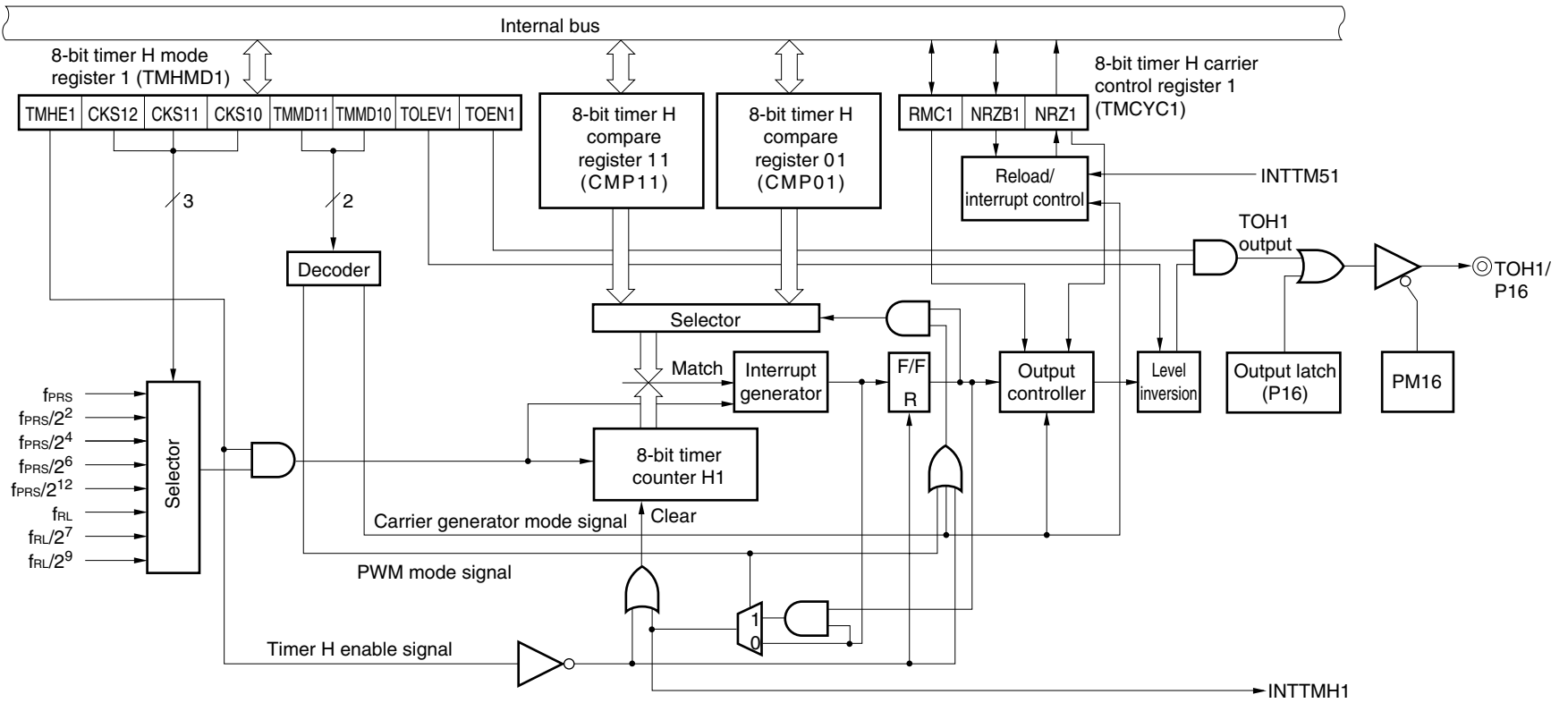


Figure 8-2. Block Diagram of 8-Bit Timer H1



(1) 8-bit timer H compare register 0n (CMP0n)

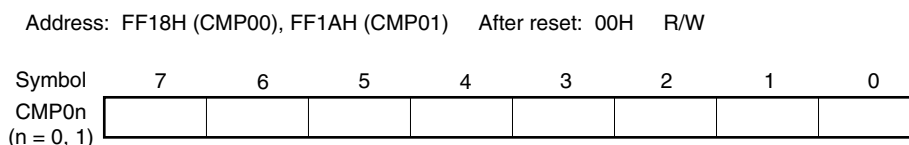
This register can be read or written by an 8-bit memory manipulation instruction. This register is used in all of the timer operation modes.

This register constantly compares the value set to CMP0n with the count value of the 8-bit timer counter Hn and, when the two values match, generates an interrupt request signal (INTTMHn) and inverts the output level of TOHn.

Rewrite the value of CMP0n while the timer is stopped (TMHEn = 0).

A reset signal generation clears this register to 00H.

Figure 8-3. Format of 8-Bit Timer H Compare Register 0n (CMP0n)



Caution CMP0n cannot be rewritten during timer count operation. CMP0n can be refreshed (the same value is written) during timer count operation.

(2) 8-bit timer H compare register 1n (CMP1n)

This register can be read or written by an 8-bit memory manipulation instruction. This register is used in the PWM output mode and carrier generator mode.

In the PWM output mode, this register constantly compares the value set to CMP1n with the count value of the 8-bit timer counter Hn and, when the two values match, inverts the output level of TOHn. No interrupt request signal is generated.

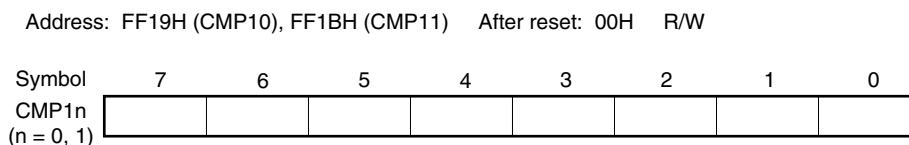
In the carrier generator mode, the CMP1n register always compares the value set to CMP1n with the count value of the 8-bit timer counter Hn and, when the two values match, generates an interrupt request signal (INTTMHn). At the same time, the count value is cleared.

CMP1n can be refreshed (the same value is written) and rewritten during timer count operation.

If the value of CMP1n is rewritten while the timer is operating, the new value is latched and transferred to CMP1n when the count value of the timer matches the old value of CMP1n, and then the value of CMP1n is changed to the new value. If matching of the count value and the CMP1n value and writing a value to CMP1n conflict, the value of CMP1n is not changed.

A reset signal generation clears this register to 00H.

Figure 8-4. Format of 8-Bit Timer H Compare Register 1n (CMP1n)



Caution In the PWM output mode and carrier generator mode, be sure to set CMP1n when starting the timer count operation (TMHEn = 1) after the timer count operation was stopped (TMHEn = 0) (be sure to set again even if setting the same value to CMP1n).

Remark n = 0, 1

8.3 Registers Controlling 8-Bit Timers H0 and H1

The following four registers are used to control 8-bit timers H0 and H1.

- 8-bit timer H mode register n (TMHMDn)
- 8-bit timer H carrier control register 1 (TMCYC1)^{Note}
- Port mode register 1 (PM1)
- Port register 1 (P1)

Note 8-bit timer H1 only

(1) 8-bit timer H mode register n (TMHMDn)

This register controls the mode of timer H.

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

Reset signal generation clears this register to 00H.

Remark n = 0, 1

Figure 8-5. Format of 8-Bit Timer H Mode Register 0 (TMHMD0)

Address: FF69H After reset: 00H R/W

| | | | | | | | | |
|--------|-------|-------|-------|-------|--------|--------|--------|-------|
| | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
| TMHMD0 | TMHE0 | CKS02 | CKS01 | CKS00 | TMMD01 | TMMD00 | TOLEV0 | TOEN0 |

| | |
|-------|--|
| TMHE0 | Timer operation enable |
| 0 | Stops timer count operation (counter is cleared to 0) |
| 1 | Enables timer count operation (count operation started by inputting clock) |

| | | | | | | |
|------------------|-------|-------|-----------------------------|---------------------------|----------------------------|------------|
| CKS02 | CKS01 | CKS00 | Count clock selection | | | |
| | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ | |
| 0 | 0 | 0 | f_{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 0 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 0 | 1 | 0 | $f_{PRS}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz |
| 0 | 1 | 1 | $f_{PRS}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 1 | 0 | 0 | $f_{PRS}/2^{10}$ | 1.95 kHz | 4.88 kHz | 9.77 kHz |
| 1 | 0 | 1 | TM50 output ^{Note} | | | |
| Other than above | | | Setting prohibited | | | |

| | | |
|------------------|--------|----------------------|
| TMMD01 | TMMD00 | Timer operation mode |
| 0 | 0 | Interval timer mode |
| 1 | 0 | PWM output mode |
| Other than above | | Setting prohibited |

| | |
|--------|--|
| TOLEV0 | Timer output level control (in default mode) |
| 0 | Low level |
| 1 | High level |

| | |
|-------|----------------------|
| TOEN0 | Timer output control |
| 0 | Disables output |
| 1 | Enables output |

Note Note the following points when selecting the TM50 output as the count clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.
It is not necessary to enable (TOE50 = 1) TO50 output in any mode.

- Cautions**
1. When $TMHE0 = 1$, setting the other bits of $TMHMD0$ is prohibited. However, $TMHMD0$ can be refreshed (the same value is written).
 2. In the PWM output mode, be sure to set the 8-bit timer H compare register 10 ($CMP10$) when starting the timer count operation ($TMHE0 = 1$) after the timer count operation was stopped ($TMHE0 = 0$) (be sure to set again even if setting the same value to $CMP10$).
 3. The actual $TOH0/P15$ pin output is determined depending on $PM15$ and $P15$, besides $TOH0$ output.

- Remarks**
1. f_{PRS} : Peripheral hardware clock frequency
 2. $TMC506$: Bit 6 of 8-bit timer mode control register 50 ($TMC50$)
 $TMC501$: Bit 1 of $TMC50$

Figure 8-6. Format of 8-Bit Timer H Mode Register 1 ($TMHMD1$)

Address: FF6CH After reset: 00H R/W

| | | | | | | | | |
|----------|-------|-------|-------|-------|--------|--------|--------|-------|
| | <7> | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
| $TMHMD1$ | TMHE1 | CKS12 | CKS11 | CKS10 | TMMD11 | TMMD10 | TOLEV1 | TOEN1 |

| | |
|-------|--|
| TMHE1 | Timer operation enable |
| 0 | Stops timer count operation (counter is cleared to 0) |
| 1 | Enables timer count operation (count operation started by inputting clock) |

| | | | | | | |
|-------|-------|-------|-----------------------|---------------------------|---------------------------|----------------------------|
| CKS12 | CKS11 | CKS10 | Count clock selection | | | |
| | | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ |
| 0 | 0 | 0 | f_{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 0 | 1 | $f_{PRS}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz |
| 0 | 1 | 0 | $f_{PRS}/2^4$ | 125 kHz | 312.5 kHz | 625 kHz |
| 0 | 1 | 1 | $f_{PRS}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 1 | 0 | 0 | $f_{PRS}/2^{12}$ | 0.49 kHz | 1.22 kHz | 2.44 kHz |
| 1 | 0 | 1 | $f_{RL}/2^7$ | 1.88 kHz (TYP.) | | |
| 1 | 1 | 0 | $f_{RL}/2^9$ | 0.47 kHz (TYP.) | | |
| 1 | 1 | 1 | f_{RL} | 240 kHz (TYP.) | | |

| | | |
|--------|--------|------------------------|
| TMMD11 | TMMD10 | Timer operation mode |
| 0 | 0 | Interval timer mode |
| 0 | 1 | Carrier generator mode |
| 1 | 0 | PWM output mode |
| 1 | 1 | Setting prohibited |

| | |
|--------|--|
| TOLEV1 | Timer output level control (in default mode) |
| 0 | Low level |
| 1 | High level |

| | |
|-------|----------------------|
| TOEN1 | Timer output control |
| 0 | Disables output |
| 1 | Enables output |

- Cautions**
1. When $TMHE1 = 1$, setting the other bits of $TMHMD1$ is prohibited. However, $TMHMD1$ can be refreshed (the same value is written).
 2. In the PWM output mode and carrier generator mode, be sure to set the 8-bit timer H compare register 11 ($CMP11$) when starting the timer count operation ($TMHE1 = 1$) after the timer count operation was stopped ($TMHE1 = 0$) (be sure to set again even if setting the same value to $CMP11$).
 3. When the carrier generator mode is used, set so that the count clock frequency of $TMH1$ becomes more than 6 times the count clock frequency of $TM51$.
 4. The actual $TOH1/P16$ pin output is determined depending on $PM16$ and $P16$, besides $TOH1$ output.

- Remarks**
1. f_{PRS} : Peripheral hardware clock frequency
 2. f_{RL} : Internal low-speed oscillation clock frequency

(2) 8-bit timer H carrier control register 1 (TMCYC1)

This register controls the remote control output and carrier pulse output status of 8-bit timer H1. This register can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

Figure 8-7. Format of 8-Bit Timer H Carrier Control Register 1 (TMCYC1)

Address: FF6DH After reset: 00H R/W^{Note}

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|--------|---|---|---|---|---|------|-------|------|
| TMCYC1 | 0 | 0 | 0 | 0 | 0 | RMC1 | NRZB1 | NRZ1 |

| RMC1 | NRZB1 | Remote control output |
|------|-------|---|
| 0 | 0 | Low-level output |
| 0 | 1 | High-level output at rising edge of INTTM51 signal input |
| 1 | 0 | Low-level output |
| 1 | 1 | Carrier pulse output at rising edge of INTTM51 signal input |

| NRZ1 | Carrier pulse output status flag |
|------|--|
| 0 | Carrier output disabled status (low-level status) |
| 1 | Carrier output enabled status (RMC1 = 1: Carrier pulse output, RMC1 = 0: High-level status) |

Note Bit 0 is read-only.

Caution Do not rewrite RMC1 when $TMHE = 1$. However, TMCYC1 can be refreshed (the same value is written).

(3) Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.

When using the P15/TOH0 pins for timer output, clear PM15 and the output latches of P15 to 0.

When using the P16/TOH1 pins for timer output, clear PM16 and the output latches of P16 to 0.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 8-8. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

| | | | | | | | | |
|--------|---|------|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |
| PM1n | P1n pin I/O mode selection (n = 0 to 7) | | | | | | | |
| 0 | Output mode (output buffer on) | | | | | | | |
| 1 | Input mode (output buffer off) | | | | | | | |

8.4 Operation of 8-Bit Timers H0 and H1

8.4.1 Operation as interval timer/square-wave output

When the 8-bit timer counter Hn and compare register 0n (CMP0n) match, an interrupt request signal (INTTMHn) is generated and the 8-bit timer counter Hn is cleared to 00H.

Compare register 1n (CMP1n) is not used in interval timer mode. Since a match of the 8-bit timer counter Hn and the CMP1n register is not detected even if the CMP1n register is set, timer output is not affected.

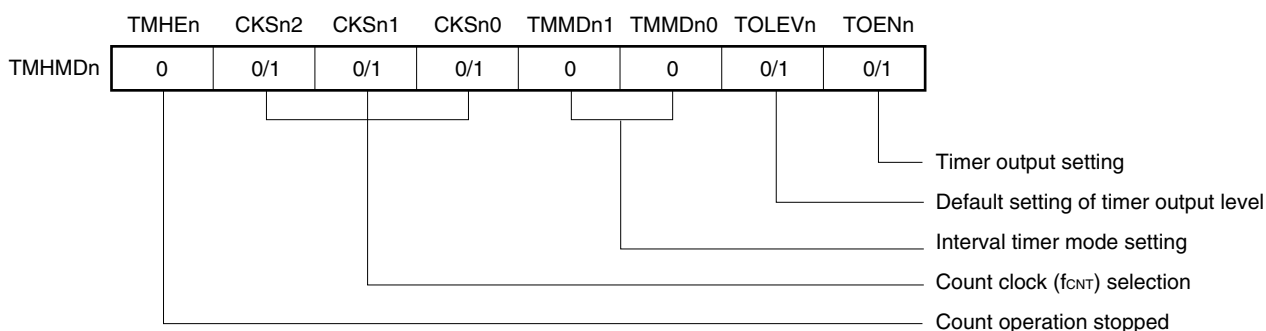
By setting bit 0 (TOEN1) of timer H mode register 1 (TMHMD1) to 1, a square wave of any frequency (duty = 50%) is output from TOH1.

Setting

<1> Set each register.

Figure 8-9. Register Setting During Interval Timer/Square-Wave Output Operation

(i) Setting timer H mode register n (TMHMDn)



(ii) CMP0n register setting

The interval time is as follows if N is set as a comparison value.

- Interval time = $(N + 1)/f_{CNT}$

<2> Count operation starts when TMHEn = 1.

<3> When the values of the 8-bit timer counter Hn and the CMP0n register match, the INTTMHn signal is generated and the 8-bit timer counter Hn is cleared to 00H.

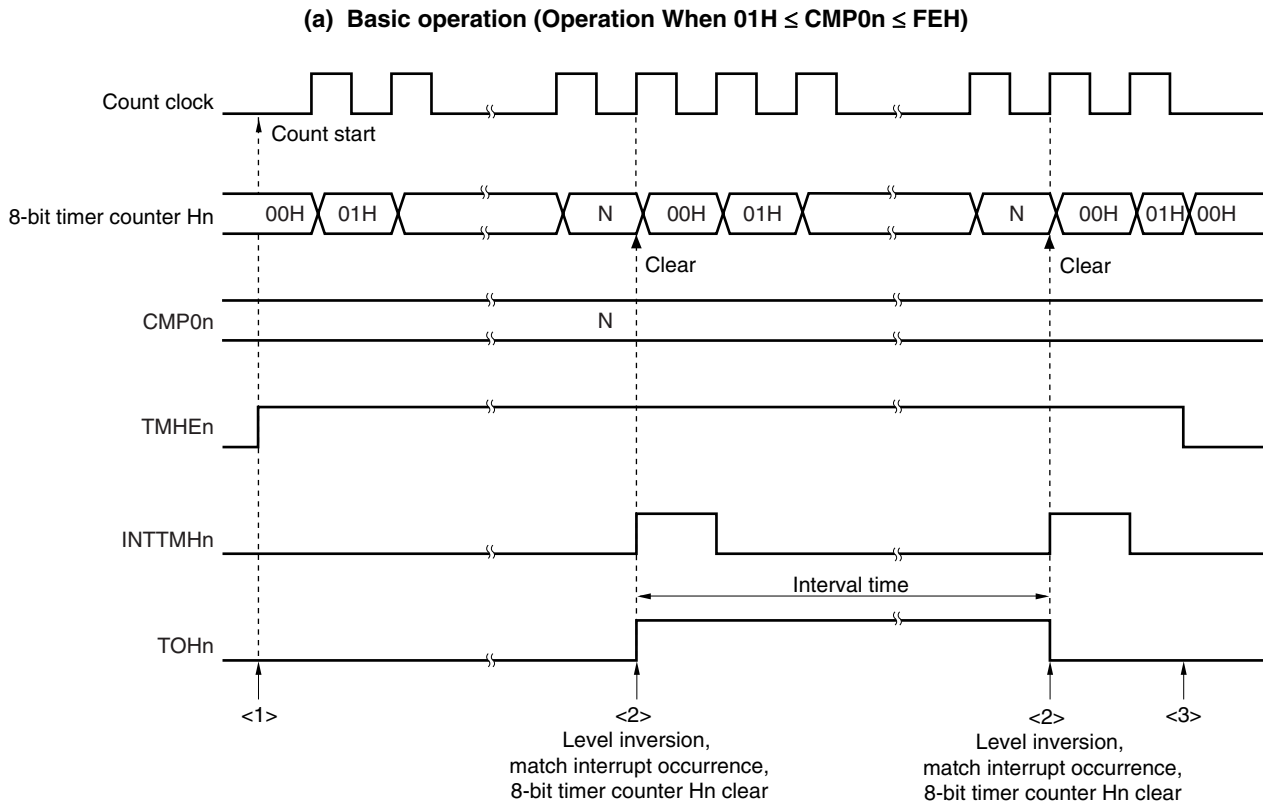
<4> Subsequently, the INTTMHn signal is generated at the same interval. To stop the count operation, clear TMHEn to 0.

Remarks 1. For the setting of the output pin, see **8.3 (3) Port mode register 1 (PM1)**.

2. For how to enable the INTTMHn signal interrupt, see **CHAPTER 16 INTERRUPT FUNCTIONS**.

3. n = 0, 1

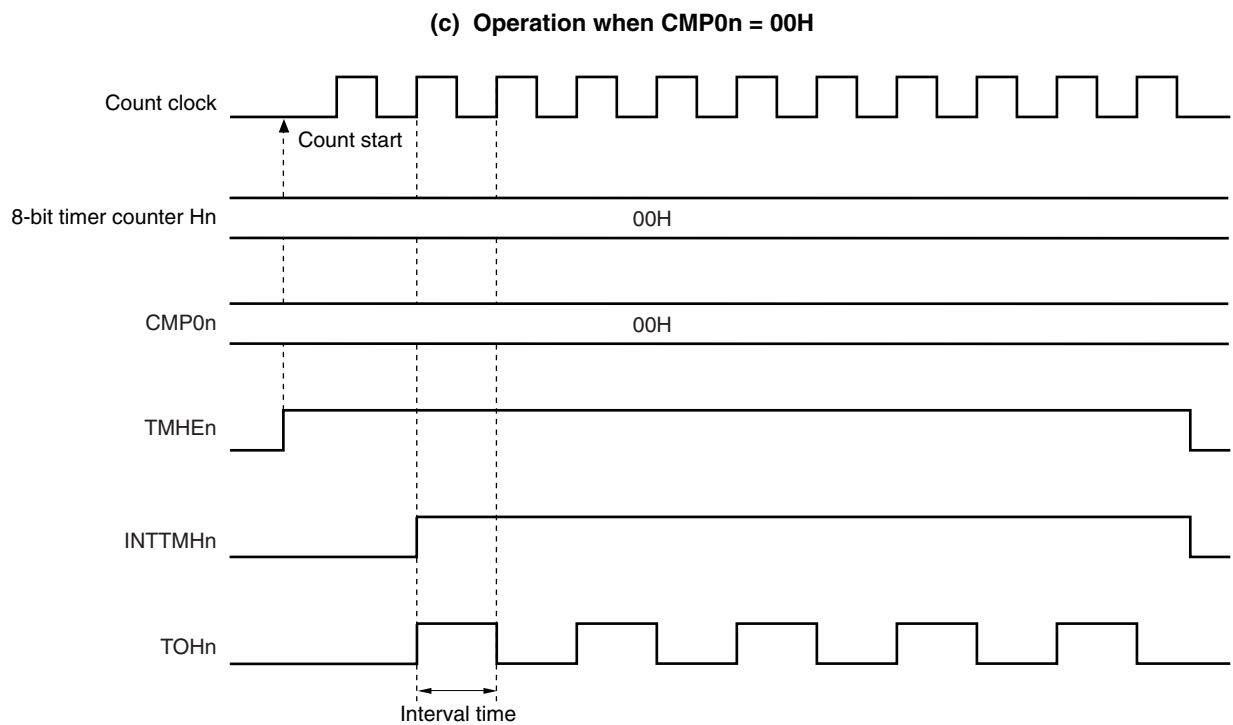
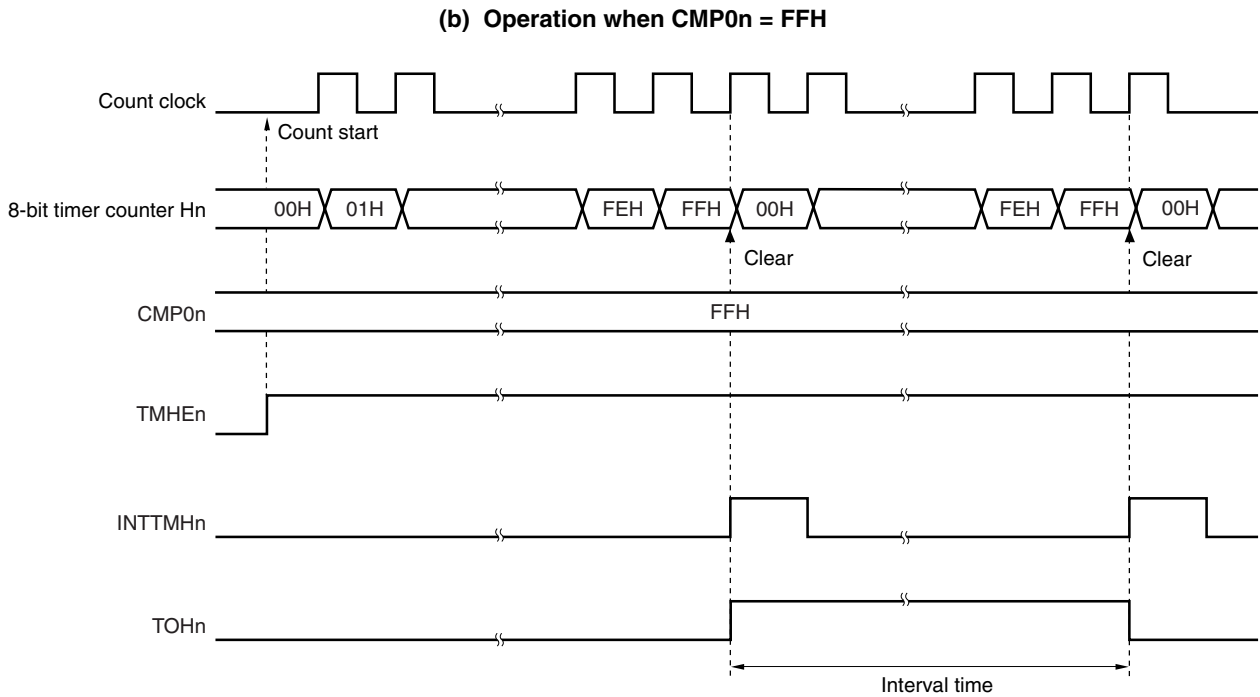
Figure 8-10. Timing of Interval Timer/Square-Wave Output Operation (1/2)



- <1> The count operation is enabled by setting the TMHEn bit to 1. The count clock starts counting no more than 1 clock after the operation is enabled.
- <2> When the value of the 8-bit timer counter Hn matches the value of the CMP0n register, the value of the timer counter is cleared, and the level of the TOHn output is inverted. In addition, the INTTMHn signal is output at the rising edge of the count clock.
- <3> If the TMHEn bit is cleared to 0 while timer H is operating, the INTTMHn signal and TOHn output are set to the default level. If they are already at the default level before the TMHEn bit is cleared to 0, then that level is maintained.

Remark n = 0, 1
 $01H \leq N \leq FEH$

Figure 8-10. Timing of Interval Timer/Square-Wave Output Operation (2/2)



Remark n = 0, 1

8.4.2 Operation as PWM output

In PWM output mode, a pulse with an arbitrary duty and arbitrary cycle can be output.

The 8-bit timer compare register 0n (CMP0n) controls the cycle of timer output (TOHn). Rewriting the CMP0n register during timer operation is prohibited.

The 8-bit timer compare register 1n (CMP1n) controls the duty of timer output (TOHn). Rewriting the CMP1n register during timer operation is possible.

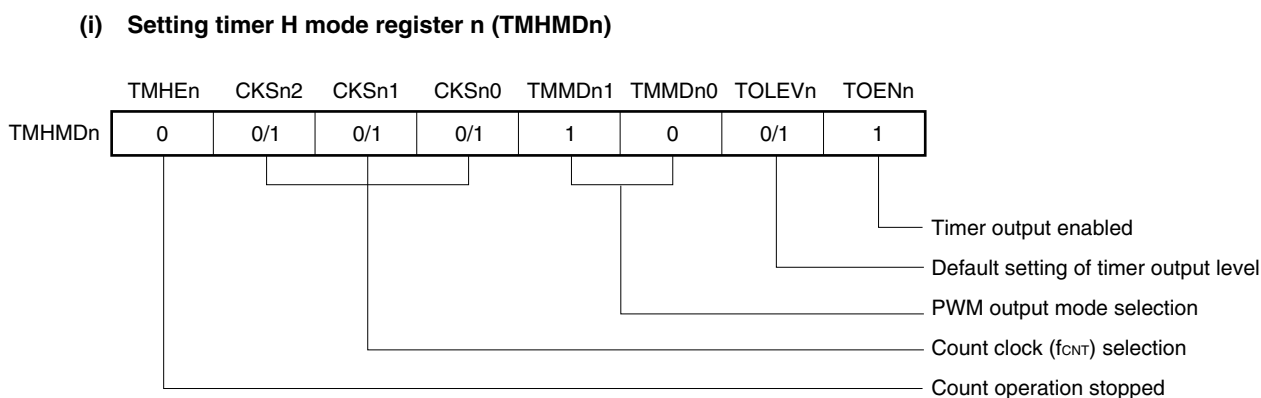
The operation in PWM output mode is as follows.

PWM output (TOHn output) outputs an active level and 8-bit timer counter Hn is cleared to 0 when 8-bit timer counter Hn and the CMP0n register match after the timer count is started. PWM output (TOHn output) outputs an inactive level when 8-bit timer counter Hn and the CMP1n register match.

Setting

<1> Set each register.

Figure 8-11. Register Setting in PWM Output Mode



(ii) Setting CMP0n register

- Compare value (N): Cycle setting

(iii) Setting CMP1n register

- Compare value (M): Duty setting

Remarks 1. $n = 0, 1$

2. $00H \leq \text{CMP1n (M)} < \text{CMP0n (N)} \leq \text{FFH}$

<2> The count operation starts when TMHEn = 1.

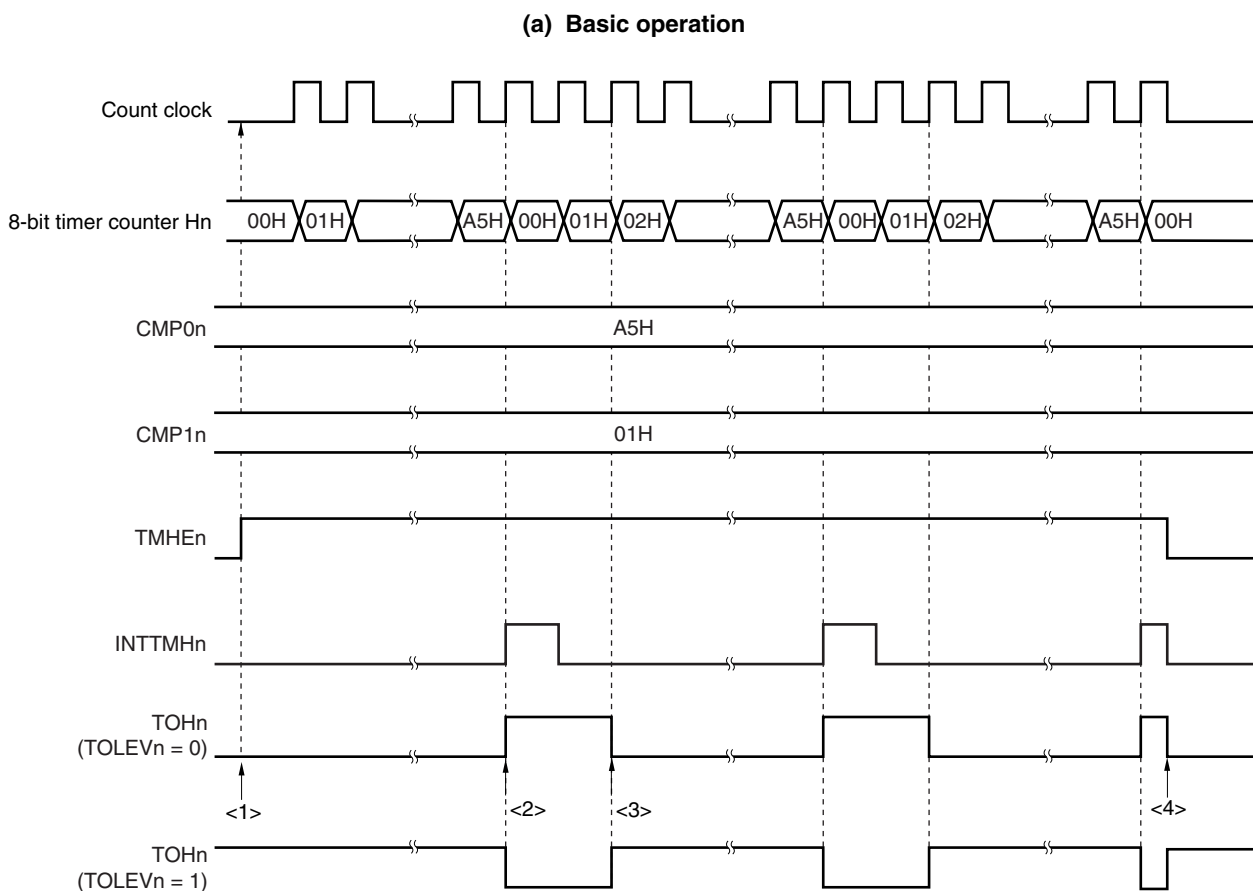
- <3> The CMP0n register is the compare register that is to be compared first after counter operation is enabled. When the values of the 8-bit timer counter Hn and the CMP0n register match, the 8-bit timer counter Hn is cleared, an interrupt request signal (INTTMHn) is generated, and an active level is output. At the same time, the compare register to be compared with the 8-bit timer counter Hn is changed from the CMP0n register to the CMP1n register.
- <4> When the 8-bit timer counter Hn and the CMP1n register match, an inactive level is output and the compare register to be compared with the 8-bit timer counter Hn is changed from the CMP1n register to the CMP0n register. At this time, the 8-bit timer counter Hn is not cleared and the INTTMHn signal is not generated.
- <5> By performing procedures <3> and <4> repeatedly, a pulse with an arbitrary duty can be obtained.
- <6> To stop the count operation, set TMHEn = 0.
If the setting value of the CMP0n register is N, the setting value of the CMP1n register is M, and the count clock frequency is f_{CNT} , the PWM pulse output cycle and duty are as follows.

- PWM pulse output cycle = $(N + 1)/f_{CNT}$
- Duty = $(M + 1)/(N + 1)$

- Cautions**
1. The set value of the CMP1n register can be changed while the timer counter is operating. However, this takes a duration of three operating clocks (signal selected by the CKSn2 to CKSn0 bits of the TMHMDn register) from when the value of the CMP1n register is changed until the value is transferred to the register.
 2. Be sure to set the CMP1n register when starting the timer count operation (TMHEn = 1) after the timer count operation was stopped (TMHEn = 0) (be sure to set again even if setting the same value to the CMP1n register).
 3. Make sure that the CMP1n register setting value (M) and CMP0n register setting value (N) are within the following range.
 $00H \leq \text{CMP1n (M)} < \text{CMP0n (N)} \leq FFH$

- Remarks**
1. For the setting of the output pin, see 8.3 (3) **Port mode register 1 (PM1)**.
 2. For details on how to enable the INTTMHn signal interrupt, see **CHAPTER 16 INTERRUPT FUNCTIONS**.
 3. $n = 0, 1$

Figure 8-12. Operation Timing in PWM Output Mode (1/4)

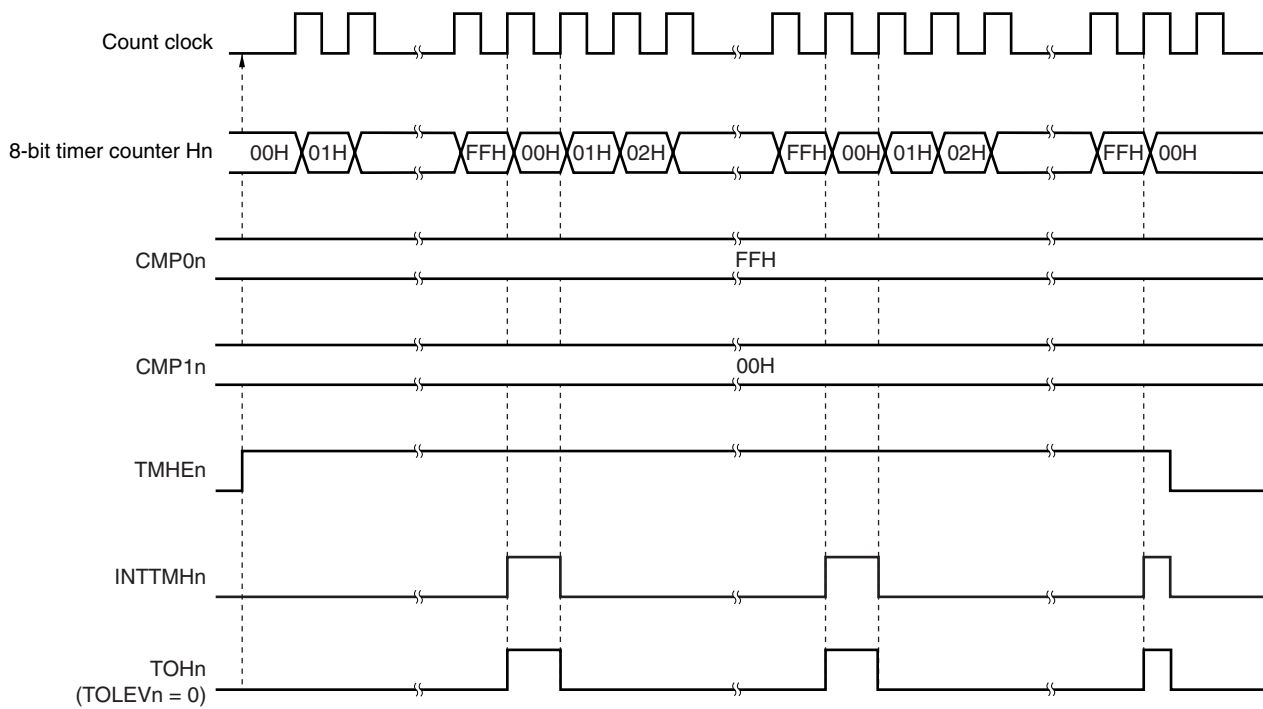


- <1> The count operation is enabled by setting the TMHEn bit to 1. Start the 8-bit timer counter Hn by masking one count clock to count up. At this time, PWM output outputs an inactive level.
- <2> When the values of the 8-bit timer counter Hn and the CMP0n register match, an active level is output. At this time, the value of the 8-bit timer counter Hn is cleared, and the INTTMHn signal is output.
- <3> When the values of the 8-bit timer counter Hn and the CMP1n register match, an inactive level is output. At this time, the 8-bit timer counter value is not cleared and the INTTMHn signal is not output.
- <4> Clearing the TMHEn bit to 0 during timer Hn operation sets the INTTMHn signal to the default and PWM output to an inactive level.

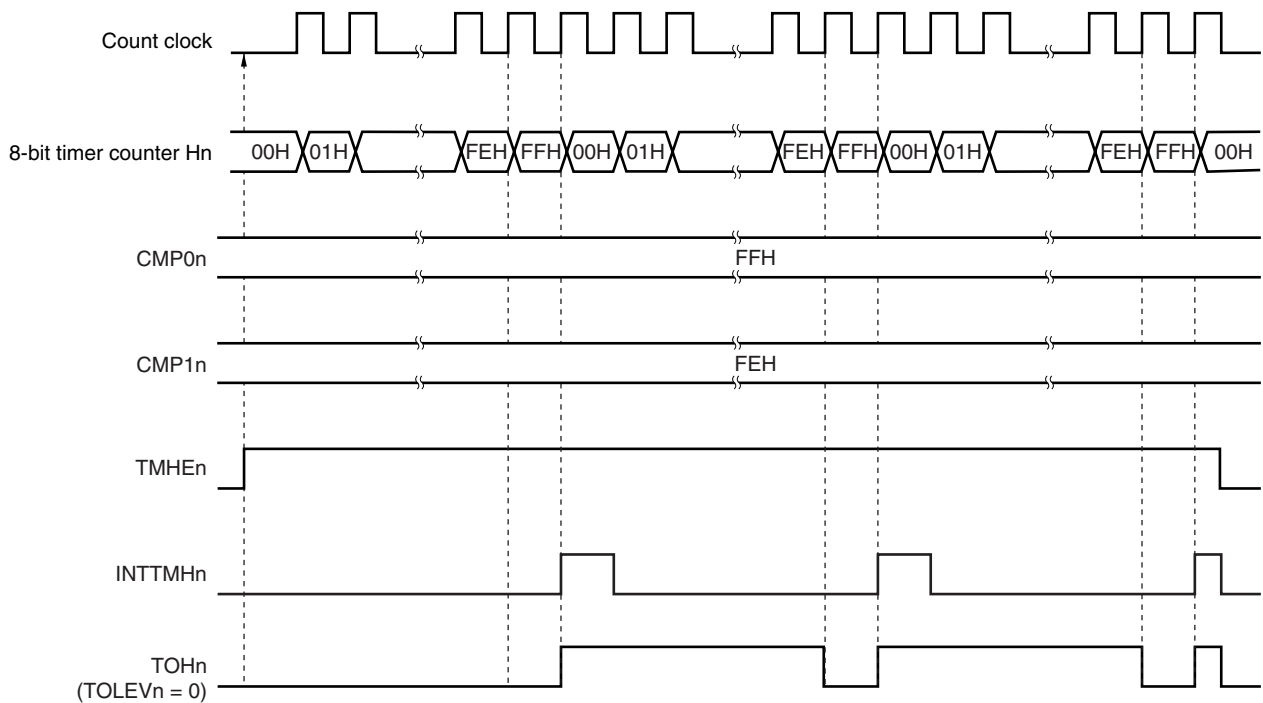
Remark n = 0, 1

Figure 8-12. Operation Timing in PWM Output Mode (2/4)

(b) Operation when $CMP0n = FFH$, $CMP1n = 00H$



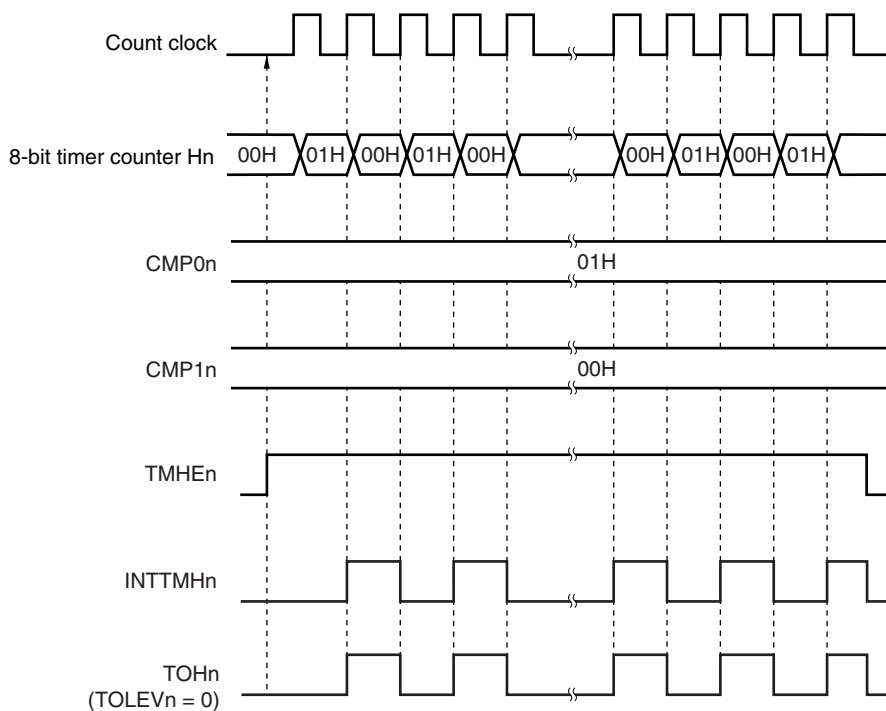
(c) Operation when $CMP0n = FFH$, $CMP1n = FEH$



Remark n = 0, 1

Figure 8-12. Operation Timing in PWM Output Mode (3/4)

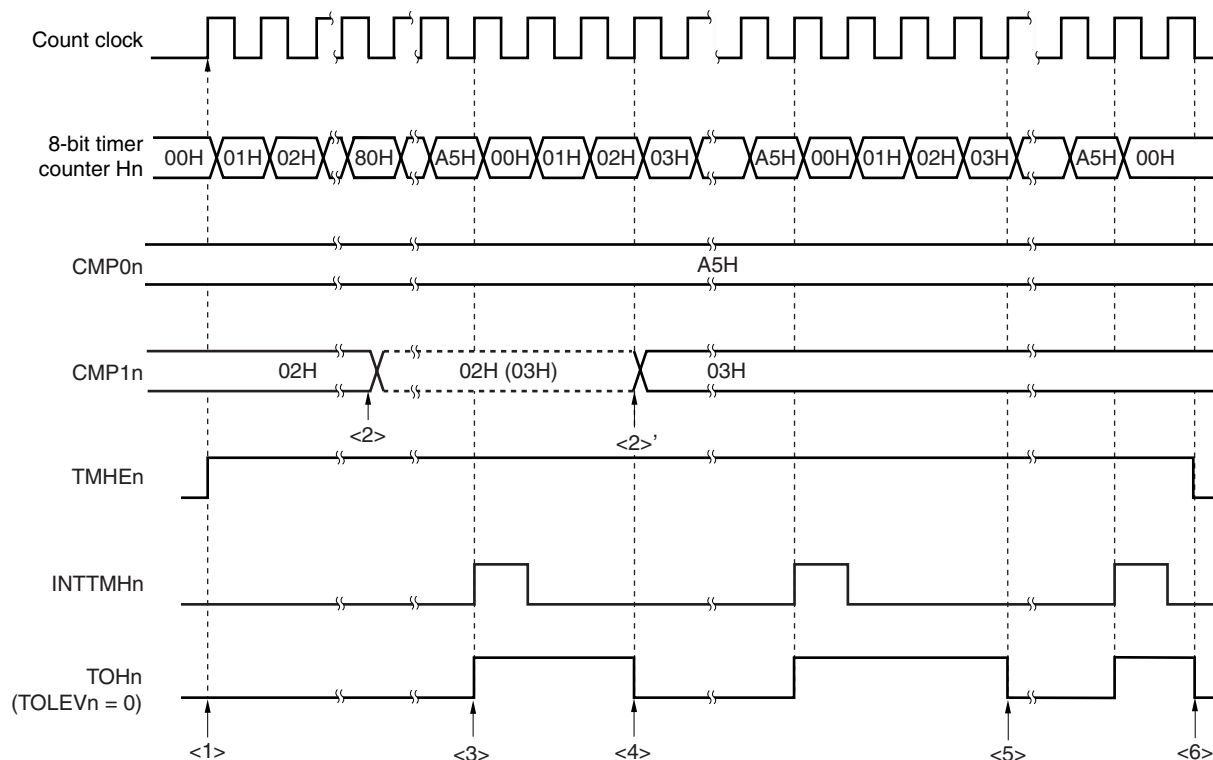
(d) Operation when $CMP0n = 01H$, $CMP1n = 00H$



Remark n = 0, 1

Figure 8-12. Operation Timing in PWM Output Mode (4/4)

(e) Operation by changing CMP1n (CMP1n = 02H → 03H, CMP0n = A5H)



- <1> The count operation is enabled by setting TMHEn = 1. Start the 8-bit timer counter Hn by masking one count clock to count up. At this time, PWM output outputs an inactive level.
- <2> The CMP1n register value can be changed during timer counter operation. This operation is asynchronous to the count clock.
- <3> When the values of the 8-bit timer counter Hn and the CMP0n register match, the value of the 8-bit timer counter Hn is cleared, an active level is output, and the INTTMHn signal is output.
- <4> If the CMP1n register value is changed, the value is latched and not transferred to the register. When the values of the 8-bit timer counter Hn and the CMP1n register before the change match, the value is transferred to the CMP1n register and the CMP1n register value is changed (<2>'). However, three count clocks or more are required from when the CMP1n register value is changed to when the value is transferred to the register. If a match signal is generated within three count clocks, the changed value cannot be transferred to the register.
- <5> When the values of the 8-bit timer counter Hn and the CMP1n register after the change match, an inactive level is output. The 8-bit timer counter Hn is not cleared and the INTTMHn signal is not generated.
- <6> Clearing the TMHEn bit to 0 during timer Hn operation sets the INTTMHn signal to the default and PWM output to an inactive level.

Remark n = 0, 1

8.4.3 Carrier generator operation (8-bit timer H1 only)

In the carrier generator mode, the 8-bit timer H1 is used to generate the carrier signal of an infrared remote controller, and the 8-bit timer/event counter 51 is used to generate an infrared remote control signal (time count).

The carrier clock generated by the 8-bit timer H1 is output in the cycle set by the 8-bit timer/event counter 51.

In carrier generator mode, the output of the 8-bit timer H1 carrier pulse is controlled by the 8-bit timer/event counter 51, and the carrier pulse is output from the TOH1 output.

(1) Carrier generation

In carrier generator mode, the 8-bit timer H compare register 01 (CMP01) generates a low-level width carrier pulse waveform and the 8-bit timer H compare register 11 (CMP11) generates a high-level width carrier pulse waveform.

Rewriting the CMP11 register during the 8-bit timer H1 operation is possible but rewriting the CMP01 register is prohibited.

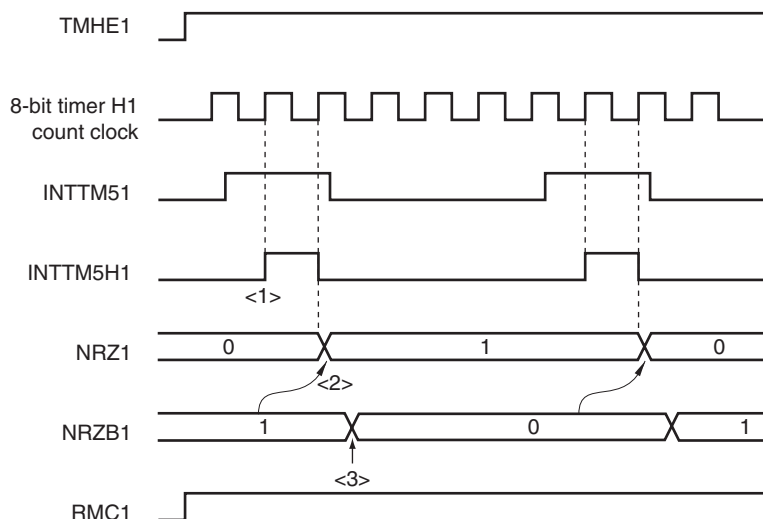
(2) Carrier output control

Carrier output is controlled by the interrupt request signal (INTTM51) of the 8-bit timer/event counter 51 and the NRZB1 and RMC1 bits of the 8-bit timer H carrier control register (TMCYC1). The relationship between the outputs is shown below.

| RMC1 Bit | NRZB1 Bit | Output |
|----------|-----------|---|
| 0 | 0 | Low-level output |
| 0 | 1 | High-level output at rising edge of INTTM51 signal input |
| 1 | 0 | Low-level output |
| 1 | 1 | Carrier pulse output at rising edge of INTTM51 signal input |

To control the carrier pulse output during a count operation, the NRZ1 and NRZB1 bits of the TMCYC1 register have a master and slave bit configuration. The NRZ1 bit is read-only but the NRZB1 bit can be read and written. The INTTM51 signal is synchronized with the 8-bit timer H1 count clock and is output as the INTTM5H1 signal. The INTTM5H1 signal becomes the data transfer signal of the NRZ1 bit, and the NRZB1 bit value is transferred to the NRZ1 bit. The timing for transfer from the NRZB1 bit to the NRZ1 bit is as shown below.

Figure 8-13. Transfer Timing



- <1> The INTTM51 signal is synchronized with the count clock of the 8-bit timer H1 and is output as the INTTM5H1 signal.
- <2> The value of the NRZB1 bit is transferred to the NRZ1 bit at the second clock from the rising edge of the INTTM5H1 signal.
- <3> Write the next value to the NRZB1 bit in the interrupt servicing program that has been started by the INTTM5H1 interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR51 register.

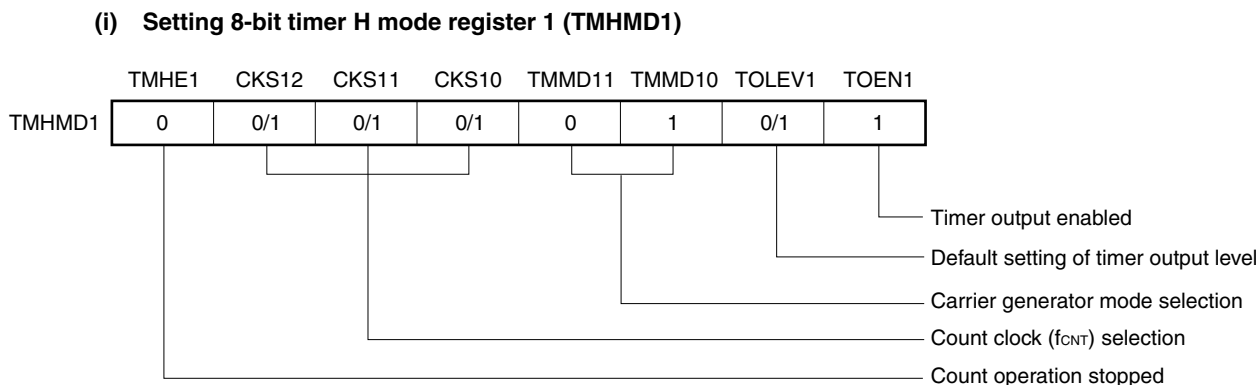
- Cautions**
1. Do not rewrite the NRZB1 bit again until at least the second clock after it has been rewritten, or else the transfer from the NRZB1 bit to the NRZ1 bit is not guaranteed.
 2. When the 8-bit timer/event counter 51 is used in the carrier generator mode, an interrupt is generated at the timing of <1>. When the 8-bit timer/event counter 51 is used in a mode other than the carrier generator mode, the timing of the interrupt generation differs.

Remark INTTM5H1 is an internal signal and not an interrupt source.

Setting

<1> Set each register.

Figure 8-14. Register Setting in Carrier Generator Mode

**(ii) CMP01 register setting**

- Compare value

(iii) CMP11 register setting

- Compare value

(iv) TMCYC1 register setting

- RMC1 = 1 ... Remote control output enable bit
- NRZB1 = 0/1 ... carrier output enable bit

(v) TCL51 and TMC51 register setting

- See 7.3 **Registers Controlling 8-Bit Timer/Event Counters 50 and 51.**

<2> When TMHE1 = 1, the 8-bit timer H1 starts counting.

<3> When TCE51 of the 8-bit timer mode control register 51 (TMC51) is set to 1, the 8-bit timer/event counter 51 starts counting.

<4> After the count operation is enabled, the first compare register to be compared is the CMP01 register. When the count value of the 8-bit timer counter H1 and the CMP01 register value match, the INTTMH1 signal is generated, the 8-bit timer counter H1 is cleared. At the same time, the compare register to be compared with the 8-bit timer counter H1 is switched from the CMP01 register to the CMP11 register.

<5> When the count value of the 8-bit timer counter H1 and the CMP11 register value match, the INTTMH1 signal is generated, the 8-bit timer counter H1 is cleared. At the same time, the compare register to be compared with the 8-bit timer counter H1 is switched from the CMP11 register to the CMP01 register.

<6> By performing procedures <4> and <5> repeatedly, a carrier clock is generated.

- <7> The INTTM51 signal is synchronized with count clock of the 8-bit timer H1 and output as the INTTM5H1 signal. The INTTM5H1 signal becomes the data transfer signal for the NRZB1 bit, and the NRZB1 bit value is transferred to the NRZ1 bit.
- <8> Write the next value to the NRZB1 bit in the interrupt servicing program that has been started by the INTTM5H1 interrupt or after timing has been checked by polling the interrupt request flag. Write data to count the next time to the CR51 register.
- <9> When the NRZ1 bit is high level, a carrier clock is output by TOH1 output.
- <10> By performing the procedures above, an arbitrary carrier clock is obtained. To stop the count operation, clear TMHE1 to 0.

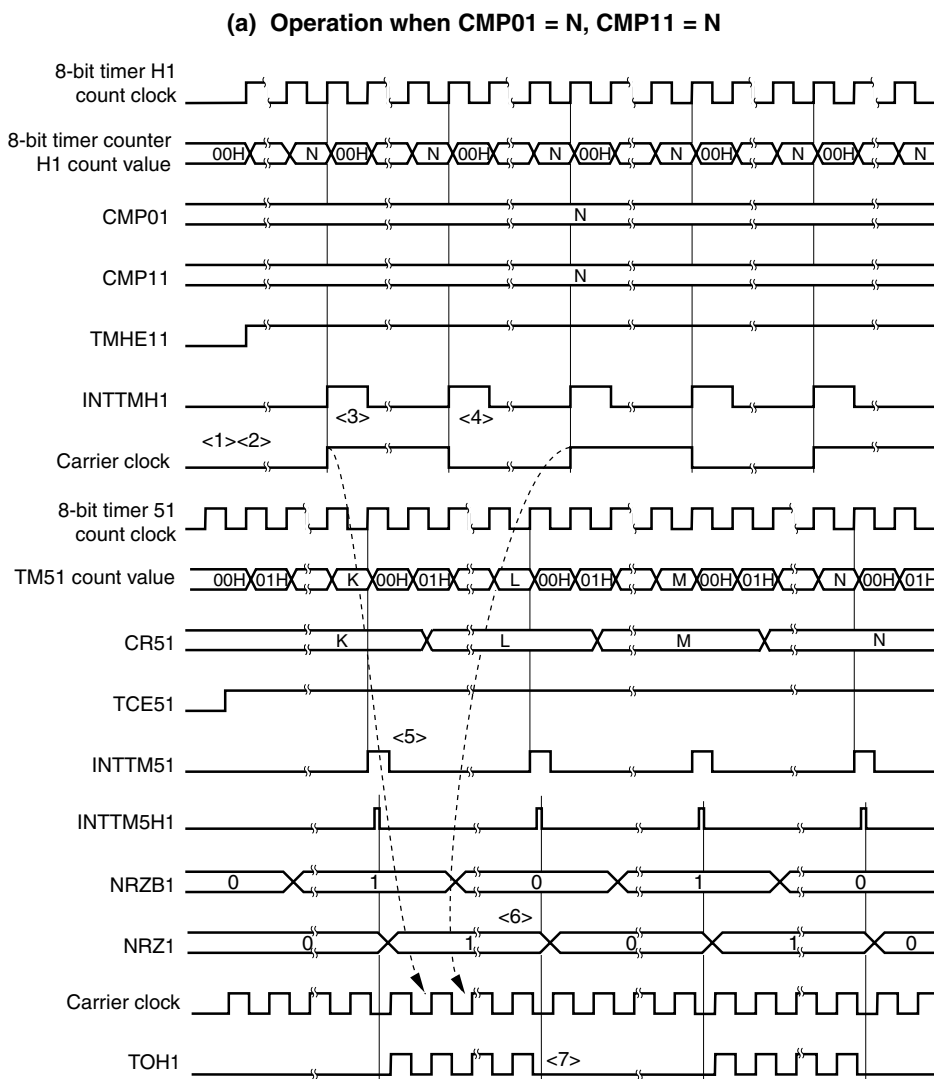
If the setting value of the CMP01 register is N, the setting value of the CMP11 register is M, and the count clock frequency is f_{CNT} , the carrier clock output cycle and duty are as follows.

- Carrier clock output cycle = $(N + M + 2)/f_{CNT}$
- Duty = High-level width/carrier clock output width = $(M + 1)/(N + M + 2)$

- Cautions**
1. Be sure to set the CMP11 register when starting the timer count operation (TMHE1 = 1) after the timer count operation was stopped (TMHE1 = 0) (be sure to set again even if setting the same value to the CMP11 register).
 2. Set so that the count clock frequency of TMH1 becomes more than 6 times the count clock frequency of TM51.
 3. Set the values of the CMP01 and CMP11 registers in a range of 01H to FFH.
 4. The set value of the CMP11 register can be changed while the timer counter is operating. However, it takes the duration of three operating clocks (signal selected by the CKS12 to CKS10 bits of the TMHMD1 register) since the value of the CMP11 register has been changed until the value is transferred to the register.
 5. Be sure to set the RMC1 bit before the count operation is started.

- Remarks**
1. For the setting of the output pin, see 8.3 (3) Port mode register 1 (PM1).
 2. For how to enable the INTTMH1 signal interrupt, see CHAPTER 16 INTERRUPT FUNCTIONS.

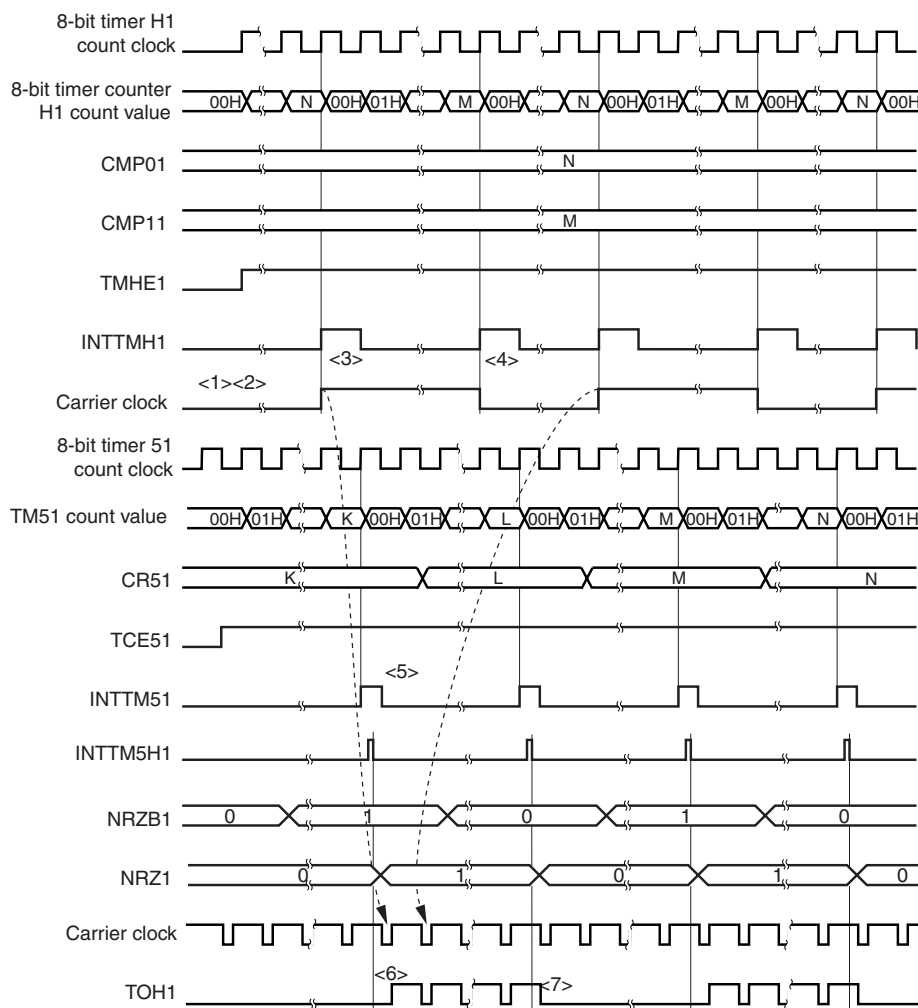
Figure 8-15. Carrier Generator Mode Operation Timing (1/3)



- <1> When TMHE1 = 0 and TCE51 = 0, the 8-bit timer counter H1 operation is stopped.
- <2> When TMHE1 = 1 is set, the 8-bit timer counter H1 starts a count operation. At that time, the carrier clock remains default.
- <3> When the count value of the 8-bit timer counter H1 matches the CMP01 register value, the first INTTMH1 signal is generated, the carrier clock signal is inverted, and the compare register to be compared with the 8-bit timer counter H1 is switched from the CMP01 register to the CMP11 register. The 8-bit timer counter H1 is cleared to 00H.
- <4> When the count value of the 8-bit timer counter H1 matches the CMP11 register value, the INTTMH1 signal is generated, the carrier clock signal is inverted, and the compare register to be compared with the 8-bit timer counter H1 is switched from the CMP11 register to the CMP01 register. The 8-bit timer counter H1 is cleared to 00H. By performing procedures <3> and <4> repeatedly, a carrier clock with duty fixed to 50% is generated.
- <5> When the INTTM51 signal is generated, it is synchronized with the 8-bit timer H1 count clock and is output as the INTTM5H1 signal.
- <6> The INTTM5H1 signal becomes the data transfer signal for the NRZB1 bit, and the NRZB1 bit value is transferred to the NRZ1 bit.
- <7> When NRZ1 = 0 is set, the TOH1 output becomes low level.

Remark INTTM5H1 is an internal signal and not an interrupt source.

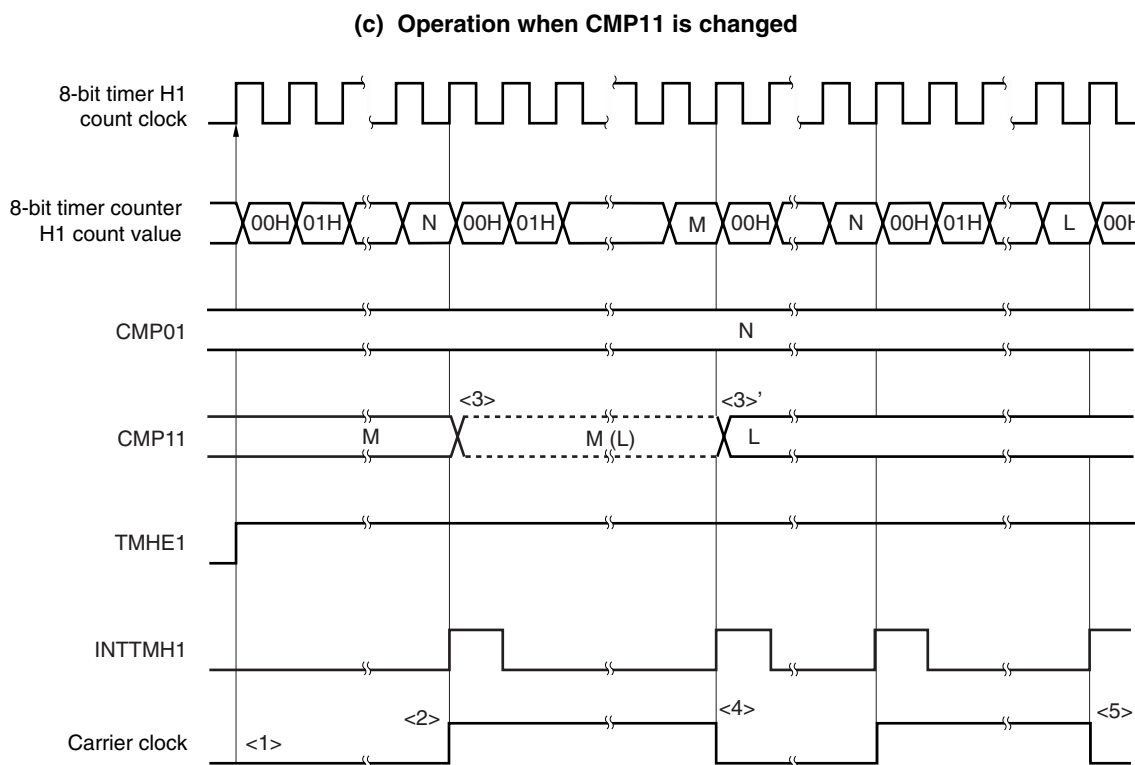
Figure 8-15. Carrier Generator Mode Operation Timing (2/3)

(b) Operation when $CMP01 = N$, $CMP11 = M$ 

- <1> When $TMHE1 = 0$ and $TCE51 = 0$, the 8-bit timer counter H1 operation is stopped.
- <2> When $TMHE1 = 1$ is set, the 8-bit timer counter H1 starts a count operation. At that time, the carrier clock remains default.
- <3> When the count value of the 8-bit timer counter H1 matches the $CMP01$ register value, the first $INTTMH1$ signal is generated, the carrier clock signal is inverted, and the compare register to be compared with the 8-bit timer counter H1 is switched from the $CMP01$ register to the $CMP11$ register. The 8-bit timer counter H1 is cleared to 00H.
- <4> When the count value of the 8-bit timer counter H1 matches the $CMP11$ register value, the $INTTMH1$ signal is generated, the carrier clock signal is inverted, and the compare register to be compared with the 8-bit timer counter H1 is switched from the $CMP11$ register to the $CMP01$ register. The 8-bit timer counter H1 is cleared to 00H. By performing procedures <3> and <4> repeatedly, a carrier clock with duty fixed to other than 50% is generated.
- <5> When the $INTTM51$ signal is generated, it is synchronized with the 8-bit timer H1 count clock and is output as the $INTTM5H1$ signal.
- <6> A carrier signal is output at the first rising edge of the carrier clock if $NRZ1$ is set to 1.
- <7> When $NRZ1 = 0$, the $TOH1$ output is held at the high level and is not changed to low level while the carrier clock is high level (from <6> and <7>, the high-level width of the carrier clock waveform is guaranteed).

Remark $INTTM5H1$ is an internal signal and not an interrupt source.

Figure 8-15. Carrier Generator Mode Operation Timing (3/3)



- <1> When $TMHE1 = 1$ is set, the 8-bit timer H1 starts a count operation. At that time, the carrier clock remains default.
- <2> When the count value of the 8-bit timer counter H1 matches the value of the CMP01 register, the INTTMH1 signal is output, the carrier signal is inverted, and the timer counter is cleared to 00H. At the same time, the compare register whose value is to be compared with that of the 8-bit timer counter H1 is changed from the CMP01 register to the CMP11 register.
- <3> The CMP11 register is asynchronous to the count clock, and its value can be changed while the 8-bit timer H1 is operating. The new value (L) to which the value of the register is to be changed is latched. When the count value of the 8-bit timer counter H1 matches the value (M) of the CMP11 register before the change, the CMP11 register is changed (<3>').
- However, it takes three count clocks or more since the value of the CMP11 register has been changed until the value is transferred to the register. Even if a match signal is generated before the duration of three count clocks elapses, the new value is not transferred to the register.
- <4> When the count value of 8-bit timer counter H1 matches the value (M) of the CMP1 register before the change, the INTTMH1 signal is output, the carrier signal is inverted, and the timer counter is cleared to 00H. At the same time, the compare register whose value is to be compared with that of the 8-bit timer counter H1 is changed from the CMP11 register to the CMP01 register.
- <5> The timing at which the count value of the 8-bit timer counter H1 and the CMP11 register value match again is indicated by the value after the change (L).

CHAPTER 9 WATCH TIMER

9.1 Functions of Watch Timer

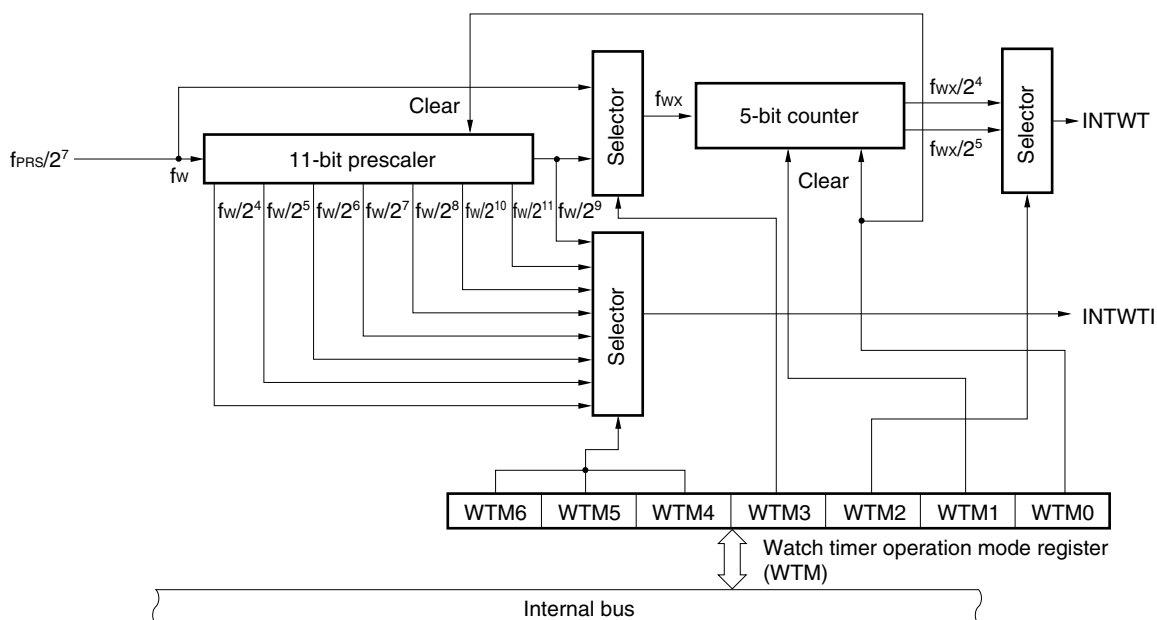
The watch timer has the following functions.

- Watch timer
- Interval timer

The watch timer and the interval timer can be used simultaneously.

Figure 9-1 shows the watch timer block diagram.

Figure 9-1. Block Diagram of Watch Timer



Remark f_{PRS} : Peripheral hardware clock frequency
 f_w : Watch timer clock frequency ($f_{PRS}/2^7$)
 f_{wx} : f_w or $f_w/2^9$

(1) Watch timer

When the peripheral hardware clock is used, interrupt request signals (INTWT) are generated at preset intervals.

Table 9-1. Watch Timer Interrupt Time

| Interrupt Time | When Operated at $f_{PRS} = 2 \text{ MHz}$ | When Operated at $f_{PRS} = 5 \text{ MHz}$ | When Operated at $f_{PRS} = 10 \text{ MHz}$ |
|----------------|---|---|--|
| $2^4/f_w$ | 1.02 ms | 410 μs | 205 μs |
| $2^5/f_w$ | 2.05 ms | 819 μs | 410 μs |
| $2^{13}/f_w$ | 0.52 s | 0.210 s | 0.105 s |
| $2^{14}/f_w$ | 1.05 s | 0.419 s | 0.210 s |

Remark f_{PRS} : Peripheral hardware clock frequency
 f_w : Watch timer clock frequency ($f_{PRS}/2^7$)

(2) Interval timer

Interrupt request signals (INTWTI) are generated at preset time intervals.

Table 9-2. Interval Timer Interval Time

| Interval Time | When Operated at $f_{PRS} = 2 \text{ MHz}$ | When Operated at $f_{PRS} = 5 \text{ MHz}$ | When Operated at $f_{PRS} = 10 \text{ MHz}$ |
|---------------|---|---|--|
| $2^4/f_w$ | 1.02 ms | 410 μs | 205 μs |
| $2^5/f_w$ | 2.05 ms | 820 μs | 410 μs |
| $2^6/f_w$ | 4.10 ms | 1.64 ms | 820 μs |
| $2^7/f_w$ | 8.20 ms | 3.28 ms | 1.64 ms |
| $2^8/f_w$ | 16.4 ms | 6.55 ms | 3.28 ms |
| $2^9/f_w$ | 32.8 ms | 13.1 ms | 6.55 ms |
| $2^{10}/f_w$ | 65.5 ms | 26.2 ms | 13.1 ms |
| $2^{11}/f_w$ | 131.1 ms | 52.4 ms | 26.2 ms |

Remark f_{PRS} : Peripheral hardware clock frequency
 f_w : Watch timer clock frequency ($f_{PRS}/2^7$)

9.2 Configuration of Watch Timer

The watch timer includes the following hardware.

Table 9-3. Watch Timer Configuration

| Item | Configuration |
|------------------|---|
| Counter | 5 bits \times 1 |
| Prescaler | 11 bits \times 1 |
| Control register | Watch timer operation mode register (WTM) |

9.3 Register Controlling Watch Timer

The watch timer is controlled by the watch timer operation mode register (WTM).

- **Watch timer operation mode register (WTM)**

This register sets the watch timer count clock, enables/disables operation, prescaler interval time, and 5-bit counter operation control.

WTM is set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears WTM to 00H.

Figure 9-2. Format of Watch Timer Operation Mode Register (WTM)

Address: FF6FH After reset: 00H R/W

| | | | | | | | | |
|--------|---|------|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | <1> | <0> |
| WTM | 0 | WTM6 | WTM5 | WTM4 | WTM3 | WTM2 | WTM1 | WTM0 |

| WTM6 | WTM5 | WTM4 | Prescaler interval time selection |
|------|------|------|-----------------------------------|
| 0 | 0 | 0 | $2^4/f_w$ |
| 0 | 0 | 1 | $2^5/f_w$ |
| 0 | 1 | 0 | $2^6/f_w$ |
| 0 | 1 | 1 | $2^7/f_w$ |
| 1 | 0 | 0 | $2^8/f_w$ |
| 1 | 0 | 1 | $2^9/f_w$ |
| 1 | 1 | 0 | $2^{10}/f_w$ |
| 1 | 1 | 1 | $2^{11}/f_w$ |

| WTM3 | WTM2 | Selection of watch timer interrupt time |
|------|------|---|
| 0 | 0 | $2^{14}/f_w$ |
| 0 | 1 | $2^{13}/f_w$ |
| 1 | 0 | $2^5/f_w$ |
| 1 | 1 | $2^4/f_w$ |

| WTM1 | 5-bit counter operation control |
|------|---------------------------------|
| 0 | Clear after operation stop |
| 1 | Start |

| WTM0 | Watch timer operation enable |
|------|---|
| 0 | Operation stop (clear both prescaler and 5-bit counter) |
| 1 | Operation enable |

- Cautions**
1. Do not change the count clock and interval time (by setting bits 4 to 6 (WTM4 to WTM6) of WTM) during watch timer operation.
 2. Be sure to set bit 7 to "0".

- Remarks**
1. f_w : Watch timer clock frequency ($f_{PRS}/2^7$)
 2. f_{PRS} : Peripheral hardware clock frequency

9.4 Watch Timer Operations

9.4.1 Watch timer operation

The watch timer generates an interrupt request signal (INTWT) at a specific time interval by using the peripheral hardware clock.

When bit 0 (WTM0) and bit 1 (WTM1) of the watch timer operation mode register (WTM) are set to 1, the count operation starts. When these bits are cleared to 0, the 5-bit counter is cleared and the count operation stops.

When the interval timer is simultaneously operated, zero-second start can be achieved only for the watch timer by clearing WTM1 to 0. In this case, however, the 11-bit prescaler is not cleared. Therefore, an error up to $2^9 \times 1/f_w$ seconds occurs in the first overflow (INTWT) after zero-second start.

The interrupt request is generated at the following time intervals.

Table 9-4. Watch Timer Interrupt Time

| WTM3 | WTM2 | Interrupt Time Selection | When Operated at $f_{PRS} = 2 \text{ MHz}$ | When Operated at $f_{PRS} = 5 \text{ MHz}$ | When Operated at $f_{PRS} = 10 \text{ MHz}$ |
|------|------|--------------------------|--|--|---|
| 0 | 0 | $2^{14}/f_w$ | 1.05 s | 0.419 s | 0.210 s |
| 0 | 1 | $2^{13}/f_w$ | 0.52 s | 0.210 s | 0.105 s |
| 1 | 0 | $2^5/f_w$ | 2.05 ms | 819 μs | 410 μs |
| 1 | 1 | $2^4/f_w$ | 1.02 ms | 410 μs | 205 μs |

- Remarks**
1. f_w : Watch timer clock frequency ($f_{PRS}/2^7$)
 2. f_{PRS} : Peripheral hardware clock frequency

9.4.2 Interval timer operation

The watch timer operates as interval timer which generates interrupt request signals (INTWTI) repeatedly at an interval of the preset count value.

The interval time can be selected with bits 4 to 6 (WTM4 to WTM6) of the watch timer operation mode register (WTM).

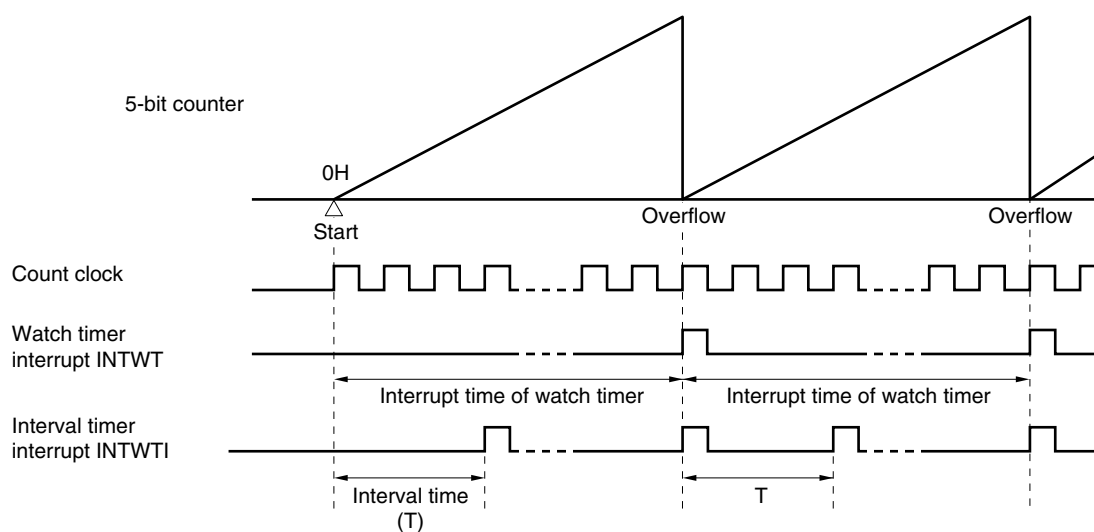
When bit 0 (WTM0) of the WTM is set to 1, the count operation starts. When this bit is set to 0, the count operation stops.

Table 9-5. Interval Timer Interval Time

| WTM6 | WTM5 | WTM4 | Interval Time | When Operated at $f_{PRS} = 2 \text{ MHz}$ | When Operated at $f_{PRS} = 5 \text{ MHz}$ | When Operated at $f_{PRS} = 10 \text{ MHz}$ |
|------|------|------|---------------|---|---|--|
| 0 | 0 | 0 | $2^4/f_w$ | 1.02 ms | 410 μs | 205 μs |
| 0 | 0 | 1 | $2^5/f_w$ | 2.05 ms | 820 μs | 410 μs |
| 0 | 1 | 0 | $2^6/f_w$ | 4.10 ms | 1.64 ms | 820 μs |
| 0 | 1 | 1 | $2^7/f_w$ | 8.20 ms | 3.28 ms | 1.64 ms |
| 1 | 0 | 0 | $2^8/f_w$ | 16.4 ms | 6.55 ms | 3.28 ms |
| 1 | 0 | 1 | $2^9/f_w$ | 32.8 ms | 13.1 ms | 6.55 ms |
| 1 | 1 | 0 | $2^{10}/f_w$ | 65.5 ms | 26.2 ms | 13.1 ms |
| 1 | 1 | 1 | $2^{11}/f_w$ | 131.1 ms | 52.4 ms | 26.2 ms |

- Remarks**
1. f_w : Watch timer clock frequency ($f_{PRS}/2^7$)
 2. f_{PRS} : Peripheral hardware clock frequency

Figure 9-3. Operation Timing of Watch Timer/Interval Timer



9.5 Cautions for Watch Timer

When operation of the watch timer and 5-bit counter is enabled by the watch timer mode control register (WTM) (by setting bits 0 (WTM0) and 1 (WTM1) of WTM to 1), the interval until the first interrupt request signal (INTWT) is generated after the register is set does not exactly match the specification made with bits 2 and 3 (WTM2, WTM3) of WTM. Subsequently, however, the INTWT signal is generated at the specified intervals.

CHAPTER 10 WATCHDOG TIMER

10.1 Functions of Watchdog Timer

The watchdog timer operates on the internal low-speed oscillation 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 WDTE
- If data is written to WDTE during a window close period
- If the instruction is fetched from an area not set by the IMS register (detection of an invalid check while the CPU hangs up)
- If the CPU accesses an area that is not set by the IMS register (excluding FB00H to FFCFH and FFE0H to FFFFH) by executing a read/write instruction (detection of an abnormal access during a CPU program loop)

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 RESF, see **CHAPTER 18 RESET FUNCTION**.

10.2 Configuration of Watchdog Timer

The watchdog timer includes the following hardware.

Table 10-1. Configuration of Watchdog Timer

| Item | Configuration |
|------------------|---------------------------------------|
| Control register | Watchdog timer enable register (WDTE) |

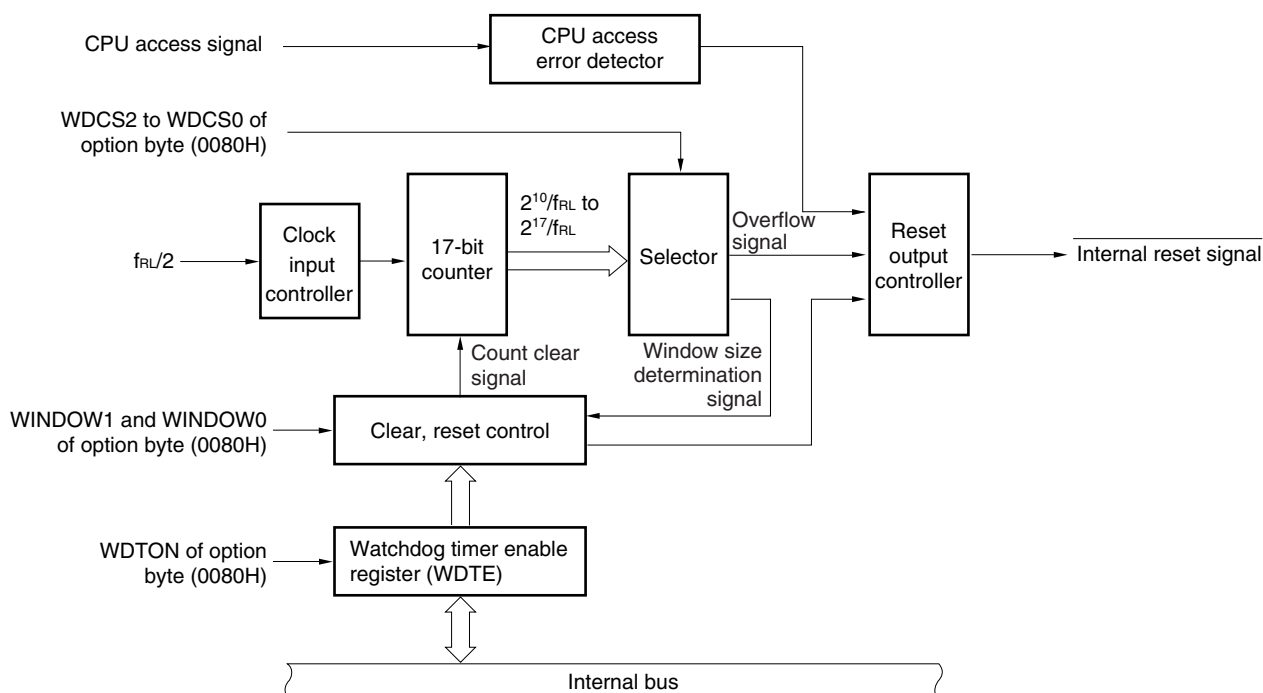
How the counter operation is controlled, overflow time, and window open period are set by the option byte.

Table 10-2. Setting of Option Bytes and Watchdog Timer

| Setting of Watchdog Timer | Option Byte (0080H) |
|---|---------------------------------|
| 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) |

Remark For the option byte, see **CHAPTER 21 OPTION BYTE**.

Figure 10-1. Block Diagram of Watchdog Timer



10.3 Register Controlling Watchdog Timer

The watchdog timer is controlled by the watchdog timer enable register (WDTE).

(1) Watchdog timer enable register (WDTE)

Writing ACH to WDTE 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^{Note}.

Figure 10-2. Format of Watchdog Timer Enable Register (WDTE)

| | | | | | | | | |
|----------------|--------------------------------------|-----|---|---|---|---|---|---|
| Address: FF99H | After reset: 9AH/1AH ^{Note} | R/W | | | | | | |
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| WDTE | | | | | | | | |

Note The WDTE reset value differs depending on the WDTON setting value of the option byte (0080H). To operate watchdog timer, set WDTON to 1.

| WDTON Setting Value | WDTE 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 WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.
 2. If a 1-bit memory manipulation instruction is executed for WDTE, an internal reset signal is generated. If the source clock to the watchdog timer is stopped, however, an internal reset signal is generated when the source clock to the watchdog timer resumes operation.
 3. The value read from WDTE is 9AH/1AH (this differs from the written value (ACH)).

10.4 Operation of Watchdog Timer

10.4.1 Controlling operation of watchdog timer

1. When the watchdog timer is used, its operation is specified by the option byte (0080H).
 - Enable counting operation of the watchdog timer by setting bit 4 (WDTON) of the option byte (0080H) to 1 (the counter starts operating after a reset release) (for details, see **CHAPTER 21**).

| WDTON | Operation Control of Watchdog Timer Counter/Illegal Access Detection |
|-------|--|
| 0 | Counter operation disabled (counting stopped after reset), illegal access detection operation disabled |
| 1 | Counter operation enabled (counting started after reset), illegal access detection operation enabled |

- Set an overflow time by using bits 3 to 1 (WDCS2 to WDCS0) of the option byte (0080H) (for details, see **10.4.2** and **CHAPTER 21**).
 - Set a window open period by using bits 6 and 5 (WINDOW1 and WINDOW0) of the option byte (0080H) (for details, see **10.4.3** and **CHAPTER 21**).
2. After a reset release, the watchdog timer starts counting.
 3. By writing “ACH” to 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.
 4. After that, write WDTE the second time or later after a reset release during the window open period. If WDTE is written during a window close period, an internal reset signal is generated.
 5. If the overflow time expires without “ACH” written to WDTE, an internal reset signal is generated. A internal reset signal is generated in the following cases.
 - If a 1-bit manipulation instruction is executed on the watchdog timer enable register (WDTE)
 - If data other than “ACH” is written to WDTE
 - If the instruction is fetched from an area not set by the IMS and IXS registers (detection of an invalid check during a CPU program loop)
 - If the CPU accesses an area not set by the IMS and IXS registers (excluding FB00H to FFCFH and FFE0H to FFFFH) by executing a read/write instruction (detection of an abnormal access during a CPU program loop)

- Cautions**
1. The first writing to WDTE after a reset release clears the watchdog timer, if it is made before the overflow time regardless of the timing of the writing, and the watchdog timer starts counting again.
 2. If the watchdog timer is cleared by writing “ACH” to WDTE, the actual overflow time may be different from the overflow time set by the option byte by up to 2/f_{RL} seconds.
 3. The watchdog timer can be cleared immediately before the count value overflows (FFFFH).

Cautions 4. The operation of the watchdog timer in the HALT and STOP modes differs as follows depending on the set value of bit 0 (LSROSC) of the option byte.

| | | |
|--------------|---|--|
| | LSROSC = 0 (Internal Low-Speed Oscillator Can Be Stopped by Software) | LSROSC = 1 (Internal Low-Speed Oscillator Cannot Be Stopped) |
| In HALT mode | Watchdog timer operation stops. | Watchdog timer operation continues. |
| In STOP mode | | |

If LSROSC = 0, the watchdog timer resumes counting after the HALT or STOP mode is released. At this time, the counter is not cleared to 0 but starts counting from the value at which it was stopped.

If oscillation of the internal low-speed oscillator is stopped by setting LSRSTOP (bit 1 of the internal oscillation mode register (RCM) = 1) when LSROSC = 0, the watchdog timer stops operating. At this time, the counter is not cleared to 0.

10.4.2 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 (0080H).

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 WDTE during the window open period before the overflow time.

The following overflow time is set.

Table 10-3. Setting of Overflow Time of Watchdog Timer

| WDCS2 | WDCS1 | WDCS0 | Overflow Time of Watchdog Timer |
|-------|-------|-------|---------------------------------|
| 0 | 0 | 0 | $2^{10}/f_{RL}$ (3.88 ms) |
| 0 | 0 | 1 | $2^{11}/f_{RL}$ (7.76 ms) |
| 0 | 1 | 0 | $2^{12}/f_{RL}$ (15.52 ms) |
| 0 | 1 | 1 | $2^{13}/f_{RL}$ (31.03 ms) |
| 1 | 0 | 0 | $2^{14}/f_{RL}$ (62.06 ms) |
| 1 | 0 | 1 | $2^{15}/f_{RL}$ (124.12 ms) |
| 1 | 1 | 0 | $2^{16}/f_{RL}$ (248.24 ms) |
| 1 | 1 | 1 | $2^{17}/f_{RL}$ (496.48 ms) |

Caution The combination of WDCS2 = WDCS1 = WDCS0 = 0 and WINDOW1 = WINDOW0 = 0 is prohibited.

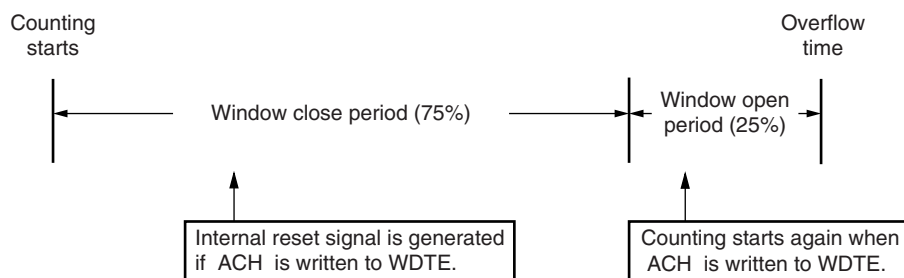
Remarks 1. f_{RL} : Internal low-speed oscillation clock frequency
 2. (): f_{RL} = 264 kHz (MAX.)

10.4.3 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 (0080H). The outline of the window is as follows.

- If “ACH” is written to WDTE during the window open period, the watchdog timer is cleared and starts counting again.
- Even if “ACH” is written to WDTE during the window close period, an abnormality is detected and an internal reset signal is generated.

Example: If the window open period is 25%



Caution The first writing to WDTE after a reset release clears the watchdog timer, if it is made before the overflow time regardless of the timing of the writing, and the watchdog timer starts counting again.

The window open period to be set is as follows.

Table 10-4. Setting Window Open Period of Watchdog Timer

| WINDOW1 | WINDOW0 | Window Open Period of Watchdog Timer |
|---------|---------|--------------------------------------|
| 0 | 0 | 25% |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

Caution The combination of WDCS2 = WDCS1 = WDCS0 = 0 and WINDOW1 = WINDOW0 = 0 is prohibited.

Remark If the overflow time is set to $2^{11}/f_{RL}$, the window close time and open time are as follows.

| | Setting of Window Open Period | | | |
|-------------------|-------------------------------|-----------------|-----------------|--------------|
| | 25% | 50% | 75% | 100% |
| Window close time | 0 to 7.11 ms | 0 to 4.74 ms | 0 to 2.37 ms | None |
| Window open time | 7.11 to 7.76 ms | 4.74 to 7.76 ms | 2.37 to 7.76 ms | 0 to 7.76 ms |

<When window open period is 25%>

- Overflow time:
 $2^{11}/f_{RL} \text{ (MAX.)} = 2^{11}/264 \text{ kHz (MAX.)} = 7.76 \text{ ms}$
- Window close time:
 $0 \text{ to } 2^{11}/f_{RL} \text{ (MIN.)} \times (1 - 0.25) = 0 \text{ to } 2^{11}/216 \text{ kHz (MIN.)} \times 0.75 = 0 \text{ to } 7.11 \text{ ms}$
- Window open time:
 $2^{11}/f_{RL} \text{ (MIN.)} \times (1 - 0.25) \text{ to } 2^{11}/f_{RL} \text{ (MAX.)} = 2^{11}/216 \text{ kHz (MIN.)} \times 0.75 \text{ to } 2^{11}/264 \text{ kHz (MAX.)}$
 $= 7.11 \text{ to } 7.76 \text{ ms}$

CHAPTER 11 A/D CONVERTER

11.1 Function of A/D Converter

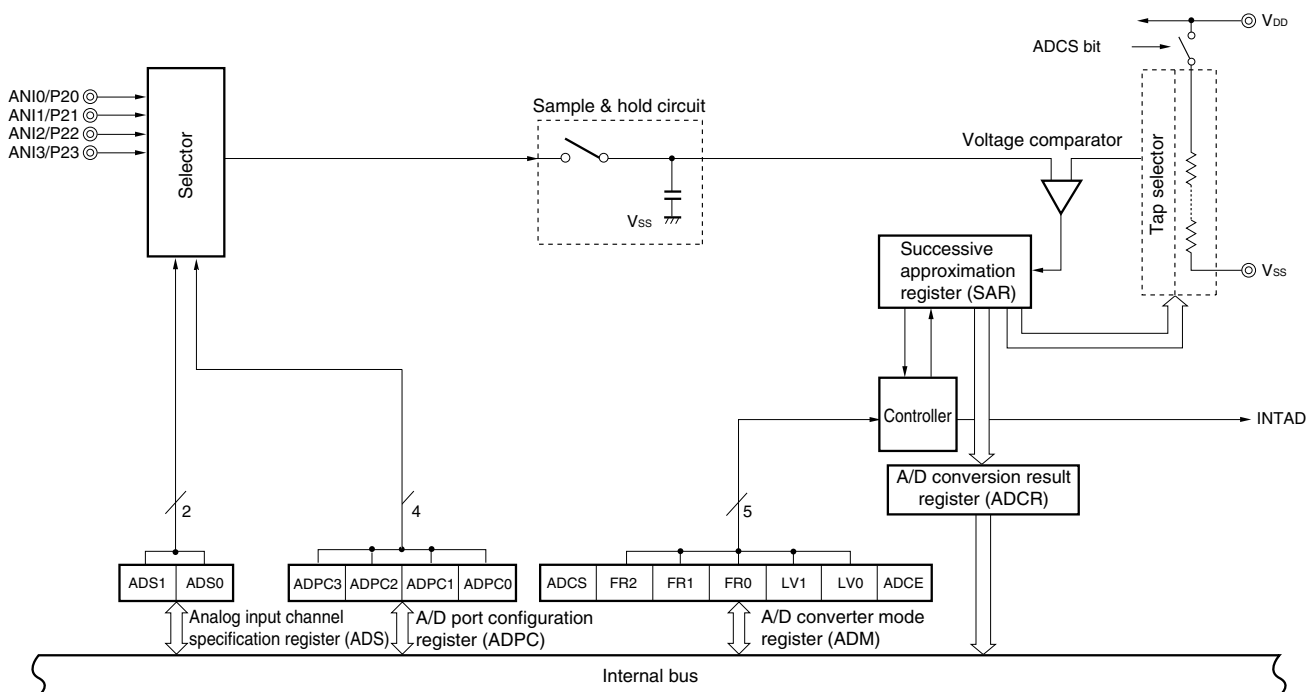
The A/D converter converts an analog input signal into a digital value, and consists of up to four channels (ANI0 to ANI3) with a resolution of 10 bits.

The A/D converter has the following function.

- 10-bit resolution A/D conversion

10-bit resolution A/D conversion is carried out repeatedly for one analog input channel selected from ANI0 to ANI3. Each time an A/D conversion operation ends, an interrupt request (INTAD) is generated.

Figure 11-1. Block Diagram of A/D Converter



11.2 Configuration of A/D Converter

The A/D converter includes the following hardware.

(1) ANI0 to ANI3 pins

These are the analog input pins of the 4-channel 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.

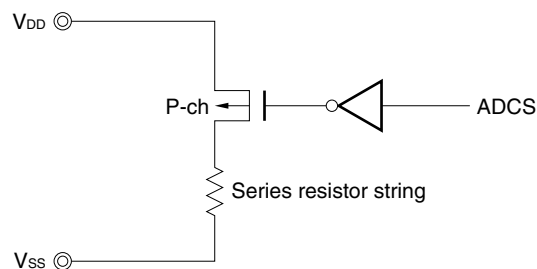
(2) Sample & hold circuit

The sample & hold circuit samples the input voltage of the analog input pin selected by the selector when A/D conversion is started, and holds the sampled voltage value during A/D conversion.

(3) Series resistor string

The series resistor string is connected between V_{DD} and V_{SS} , and generates a voltage to be compared with the sampled voltage value.

Figure 11-2. Circuit Configuration of Series Resistor String



(4) Voltage comparator

The voltage comparator compares the sampled voltage value and the output voltage of the series resistor string.

(5) Successive approximation register (SAR)

This register converts the result of comparison by the voltage comparator, starting from the most significant bit (MSB). When the voltage value is converted into a digital value down to the least significant bit (LSB) (end of A/D conversion), the contents of the SAR register are transferred to the A/D conversion result register (ADCR).

(6) 10-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 higher 10 bits (the lower 6 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.

Caution When data is read from ADCR and ADCRH, a wait cycle is generated. Do not read data from ADCR and ADCRH when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(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.

(9) V_{DD} pin

This pin inputs an analog power/reference voltage to the A/D converter. Make this pin the same potential as the V_{DD} pin when port 2 is used as a digital port.

The signal input to ANI0 to ANI3 is converted into a digital signal, based on the voltage applied across V_{DD} and V_{SS}.

(10) V_{SS} pin

This is the ground potential pin of the A/D converter.

(11) A/D converter mode register (ADM)

This register is used to set the conversion time of the analog input signal to be converted, and to start or stop the conversion operation.

(12) A/D port configuration register (ADPC)

This register switches the ANI0/P20 to ANI3/P23 pins to analog input of A/D converter or digital I/O of port.

(13) Analog input channel specification register (ADS)

This register is used to specify the port that inputs the analog voltage to be converted into a digital signal.

(14) Port mode register 2 (PM2)

This register switches the ANI0/P20 to ANI3/P23 pins to input or output.

11.3 Registers Used in A/D Converter

The A/D converter uses the following six registers.

- A/D converter mode register (ADM)
- A/D port configuration register (ADPC)
- Analog input channel specification register (ADS)
- Port mode register 2 (PM2)
- 10-bit A/D conversion result register (ADCR)
- 8-bit A/D conversion result register (ADCRH)

(1) A/D converter mode register (ADM)

This register sets the conversion time for analog input to be A/D converted, and starts/stops conversion. ADM can be set by a 1-bit or 8-bit memory manipulation instruction. Reset signal generation clears this register to 00H.

Figure 11-3. Format of A/D Converter Mode Register (ADM)

Address: FF28H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|--------|------|---|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|------|
| ADM | ADCS | 0 | FR2 ^{Note 1} | FR1 ^{Note 1} | FR0 ^{Note 1} | LV1 ^{Note 1} | LV0 ^{Note 1} | ADCE |

| ADCS | A/D conversion operation control |
|------|----------------------------------|
| 0 | Stops conversion operation |
| 1 | Enables conversion operation |

| ADCE | Comparator operation control ^{Note 2} |
|------|--|
| 0 | Stops comparator operation |
| 1 | Enables comparator operation |

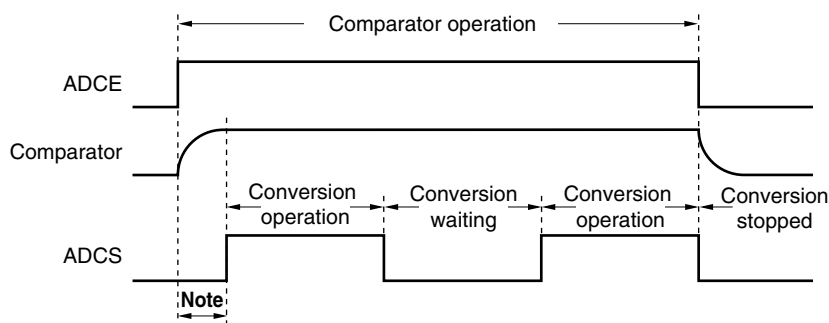
- Notes**
1. For details of FR2 to FR0, LV1, LV0, and A/D conversion, see **Table 11-2 A/D Conversion Time Selection**.
 2. The operation of the comparator is controlled by ADCS and ADCE, and it takes 1 μ s from operation start to operation stabilization. Therefore, when ADCS is set to 1 after 1 μ s or more has elapsed from the time ADCE is set to 1, the conversion result at that time has priority over the first conversion result. Otherwise, ignore data of the first conversion.

Table 11-1. Settings of ADCS and ADCE

| ADCS | ADCE | A/D Conversion Operation |
|------|------|--|
| 0 | 0 | Stop status (DC power consumption path does not exist) |
| 0 | 1 | Conversion waiting mode (comparator operation, only comparator consumes power) |
| 1 | 0 | Conversion mode (comparator operation stopped ^{Note}) |
| 1 | 1 | Conversion mode (comparator operation) |

Note Ignore the first conversion data.

Figure 11-4. Timing Chart When Comparator Is Used



Note To stabilize the internal circuit, the time from the rising of the ADCE bit to the falling of the ADCS bit must be 1 μs or longer.

- Cautions**
1. A/D conversion must be stopped before rewriting bits FR0 to FR2, LV1, and LV0 to values other than the identical data.
 2. If data is written to ADM, a wait cycle is generated. Do not write data to ADM when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

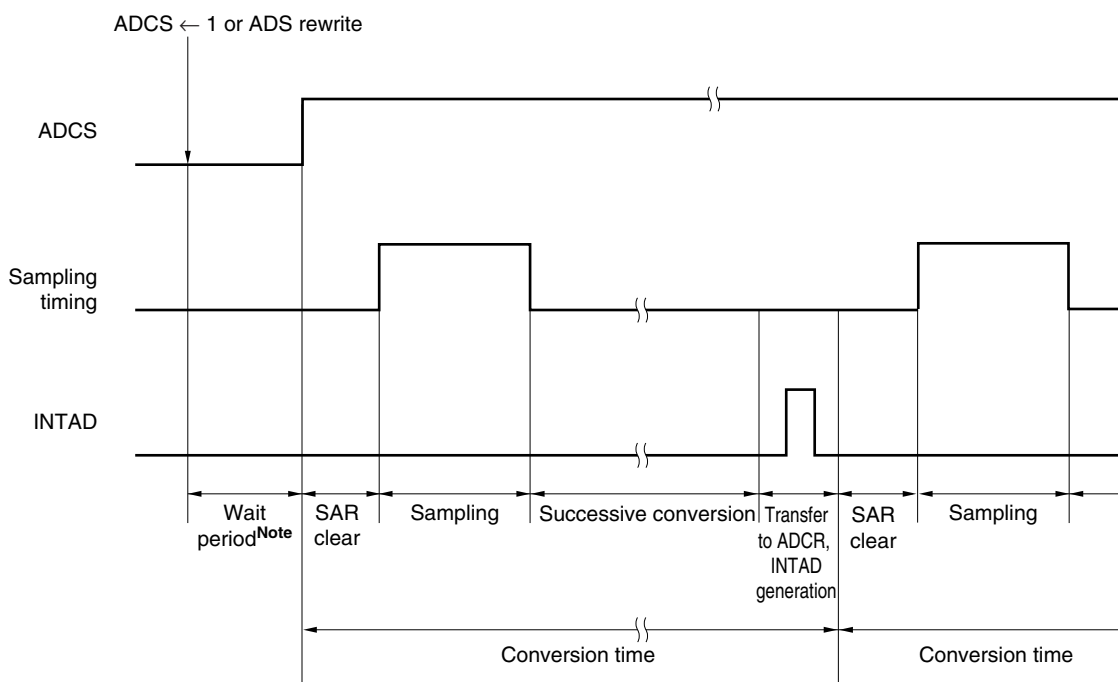
Table 11-2. A/D Conversion Time Selection

| A/D Converter Mode Register (ADM) | | | | | Conversion Time Selection | | | Conversion Clock (f_{AD}) |
|-----------------------------------|-----|-----|-----|-----|----------------------------------|-----------------------------------|---------------------------------|--------------------------------------|
| FR2 | FR1 | FR0 | LV1 | LV0 | $f_{\text{PRS}} = 2 \text{ MHz}$ | $f_{\text{PRS}} = 10 \text{ MHz}$ | | |
| 0 | 0 | 0 | 0 | 1 | $264/f_{\text{PRS}}$ | Setting prohibited | $26.4 \mu\text{s}$ | $f_{\text{PRS}}/12$ |
| 0 | 0 | 1 | 0 | 1 | $176/f_{\text{PRS}}$ | | $17.6 \mu\text{s}$ | $f_{\text{PRS}}/8$ |
| 0 | 1 | 0 | 0 | 1 | $132/f_{\text{PRS}}$ | $66.0 \mu\text{s}$ | $13.2 \mu\text{s}$ | $f_{\text{PRS}}/6$ |
| 0 | 1 | 1 | 0 | 1 | $88/f_{\text{PRS}}$ | $44.0 \mu\text{s}$ | $8.8 \mu\text{s}^{\text{Note}}$ | $f_{\text{PRS}}/4$ |
| 1 | 0 | 0 | 0 | 1 | $66/f_{\text{PRS}}$ | $33.0 \mu\text{s}$ | $6.6 \mu\text{s}^{\text{Note}}$ | $f_{\text{PRS}}/3$ |
| 1 | 0 | 1 | 0 | 1 | $44/f_{\text{PRS}}$ | $22.0 \mu\text{s}$ | Setting prohibited | $f_{\text{PRS}}/2$ |
| Other than above | | | | | Setting prohibited | | | |

- Cautions**
1. Set the conversion times with the following conditions.
 - $4.0 \text{ V} \leq V_{\text{DD}} \leq 5.5 \text{ V}$: $f_{\text{AD}} = 0.33 \text{ to } 3.6 \text{ MHz}$
 2. When rewriting FR2 to FR0, LV1, and LV0 to other than the same data, stop A/D conversion once (ADCS = 0) beforehand.
 3. The above conversion time does not include clock frequency errors. Select conversion time, taking clock frequency errors into consideration.

Remark f_{PRS} : Peripheral hardware clock frequency

Figure 11-5. A/D Converter Sampling and A/D Conversion Timing



Note For details of wait period, see **CHAPTER 27 CAUTIONS FOR WAIT**.

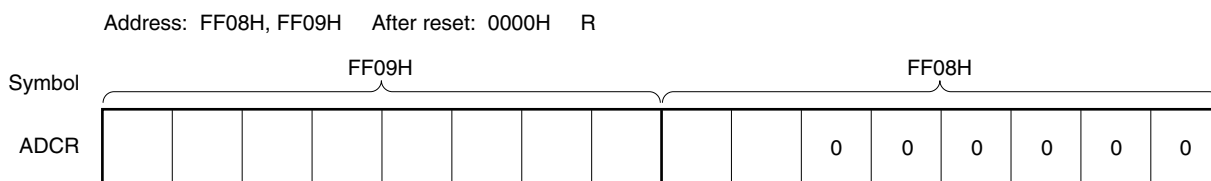
(2) 10-bit A/D conversion result register (ADCR)

This register is a 16-bit register that stores the A/D conversion result. The lower 6 bits are fixed to 0. Each time A/D conversion ends, the conversion result is loaded from the successive approximation register. The higher 8 bits of the conversion result are stored in FF09H and the lower 2 bits are stored in the higher 2 bits of FF08H.

ADCR can be read by a 16-bit memory manipulation instruction.

Reset signal generation clears this register to 0000H.

Figure 11-6. Format of 10-Bit A/D Conversion Result Register (ADCR)



- Cautions**
1. When writing to the A/D converter mode register (ADM), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of ADCR may become undefined. Read the conversion result following conversion completion before writing to ADM, ADS, and ADPC. Using timing other than the above may cause an incorrect conversion result to be read.
 2. If data is read from ADCR, a wait cycle is generated. Do not read data from ADCR when the peripheral hardware clock (f_{PRS}) is stopped. For details, see **CHAPTER 27 CAUTIONS FOR WAIT**.

(3) 8-bit A/D conversion result register (ADCRH)

This register is an 8-bit register that stores the A/D conversion result. The higher 8 bits of 10-bit resolution are stored. ADCRH can be read by an 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 11-7. Format of 8-Bit A/D Conversion Result Register (ADCRH)

Address: FF09H After reset: 00H R

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|---|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADCRH | | | | | | | | |

- Cautions**
1. When writing to the A/D converter mode register (ADM), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of ADCRH may become undefined. Read the conversion result following conversion completion before writing to ADM, ADS, and ADPC. Using timing other than the above may cause an incorrect conversion result to be read.
 2. If data is read from ADCRH, a wait cycle is generated. Do not read data from ADCRH when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(4) Analog input channel specification register (ADS)

This register specifies the input channel of the analog voltage to be A/D converted.

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

Reset signal generation clears this register to 00H.

Figure 11-8. Format of Analog Input Channel Specification Register (ADS)

Address: FF29H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADS | 0 | 0 | 0 | 0 | 0 | 0 | ADS1 | ADS0 |

| | | |
|------|------|------------------------------------|
| ADS1 | ADS0 | Analog input channel specification |
| 0 | 0 | ANI0 |
| 0 | 1 | ANI1 |
| 1 | 0 | ANI2 |
| 1 | 1 | ANI3 |

- Cautions**
1. Be sure to clear bits 2 to 7 to "0".
 2. Set a channel to be used for A/D conversion in the input mode by using port mode register 2 (PM2).
 3. If data is written to ADS, a wait cycle is generated. Do not write data to ADS when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(5) A/D port configuration register (ADPC)

This register switches the ANI0/P20 to ANI3/P23 pins to analog input of A/D converter or digital I/O of port.

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

Reset signal generation clears this register to 00H.

Figure 11-9. Format of A/D Port Configuration Register (ADPC)

Address: FF2FH After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|-------|-------|-------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADPC | 0 | 0 | 0 | 0 | ADPC3 | ADPC2 | ADPC1 | ADPC0 |

| ADPC3 | ADPC2 | ADPC1 | ADPC0 | Digital I/O(D)/analog input (A) switching | | | |
|------------------|-------|-------|-------|---|----------|----------|----------|
| | | | | P23/ANI3 | P22/ANI2 | P21/ANI1 | P20/ANI0 |
| 0 | 0 | 0 | 0 | A | A | A | A |
| 0 | 0 | 0 | 1 | A | A | A | D |
| 0 | 0 | 1 | 0 | A | A | D | D |
| 0 | 0 | 1 | 1 | A | D | D | D |
| 0 | 1 | 0 | 0 | D | D | D | D |
| 1 | 0 | 0 | 0 | D | D | D | D |
| Other than above | | | | Setting prohibited | | | |

- Cautions**
1. Set a channel to be used for A/D conversion in the input mode by using port mode register 2 (PM2).
 2. If data is written to ADPC, a wait cycle is generated. Do not write data to ADPC when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(6) Port mode register 2 (PM2)

When using the ANI0/P20 to ANI3/P23 pins for analog input port, set PM20 to PM23 to 1. The output latches of P20 to P23 at this time may be 0 or 1.

If PM20 to PM23 are set to 0, they cannot be used as analog input port pins.

PM2 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 11-10. Format of Port Mode Register 2 (PM2)

Address: FF22H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|---|---|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM2 | 1 | 1 | 1 | 1 | PM23 | PM22 | PM21 | PM20 |

| | |
|------|---|
| PM2n | P2n pin I/O mode selection (n = 0 to 3) |
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

Caution Be sure to set bits 4 to 7 to “1”.

ANI0/P20 to ANI3/P23 pins are as shown below depending on the settings of ADPC, ADS, and PM2.

Table 11-4. Setting Functions of ANI0/P20 to ANI3/P23 Pins

| ADPC | PM2 | ADS | ANI0/P20 to ANI3/P23 Pins |
|------------------------|-------------|----------------------|------------------------------------|
| 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. | |
| Digital I/O selection | Input mode | – | Digital input |
| | Output mode | – | Digital output |

11.4 A/D Converter Operations

11.4.1 Basic operations of A/D converter

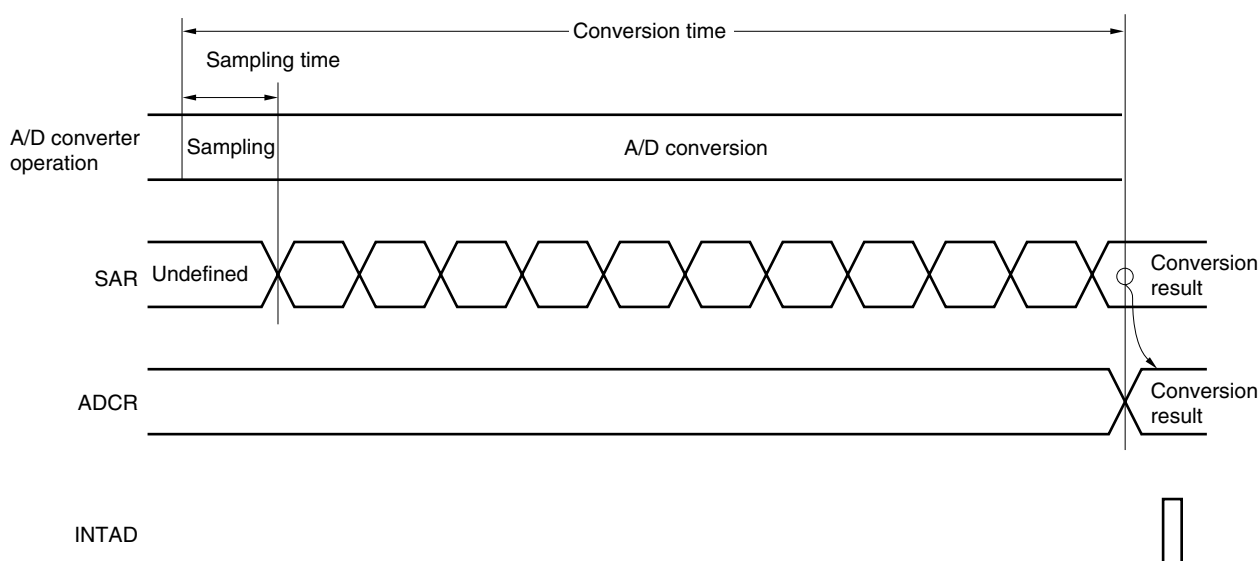
- <1> Set bit 0 (ADCE) of the A/D converter mode register (ADM) to 1 to start the operation of the comparator.
- <2> Set channels for A/D conversion to analog input by using the A/D port configuration register (ADPC) and set to input mode by using port mode register 2 (PM2).
- <3> Set A/D conversion time by using bits 5 to 1 (FR2 to FR0, LV1, and LV0) of ADM.
- <4> Select one channel for A/D conversion using the analog input channel specification register (ADS).
- <5> Start the conversion operation by setting bit 7 (ADCS) of ADM to 1.
(<6> to <12> are operations performed by hardware.)
- <6> The voltage input to the selected analog input channel is sampled by the sample & hold circuit.
- <7> 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.
- <8> Bit 9 of the successive approximation register (SAR) is set. The series resistor string voltage tap is set to $(1/2) V_{DD}$ by the tap selector.
- <9> 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 $(1/2) V_{DD}$, the MSB of SAR remains set to 1. If the analog input is smaller than $(1/2) V_{DD}$, the MSB is reset to 0.
- <10> Next, bit 8 of SAR 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 9, as described below.
 - Bit 9 = 1: $(3/4) V_{DD}$
 - Bit 9 = 0: $(1/4) V_{DD}$
 The voltage tap and sampled voltage are compared and bit 8 of SAR is manipulated as follows.
 - Analog input voltage \geq Voltage tap: Bit 8 = 1
 - Analog input voltage $<$ Voltage tap: Bit 8 = 0
- <11> Comparison is continued in this way up to bit 0 of SAR.
- <12> Upon completion of the comparison of 10 bits, an effective digital result value remains in SAR, and the result value is transferred to the A/D conversion result register (ADCR, ADCRH) and then latched.
At the same time, the A/D conversion end interrupt request (INTAD) can also be generated.
- <13> Repeat steps <6> to <12>, until ADCS is cleared to 0.
To stop the A/D converter, clear ADCS to 0.
To restart A/D conversion from the status of ADCE = 1, start from <5>. To start A/D conversion again when ADCE = 0, set ADCE to 1, wait for 1 μ s or longer, and start <5>. To change a channel of A/D conversion, start from <4>.

Caution Make sure the period of <1> to <5> is 1 μ s or more.

Remark Two types of A/D conversion result registers are available.

- ADCR (16 bits): Store 10-bit A/D conversion value
- ADCRH (8 bits): Store 8-bit A/D conversion value

Figure 11-11. Basic Operation of A/D Converter



A/D conversion operations are performed continuously until bit 7 (ADCS) of the A/D converter mode register (ADM) is reset (0) by software.

If a write operation is performed to the analog input channel specification register (ADS) during an A/D conversion operation, the conversion operation is initialized, and if the ADCS bit is set (1), conversion starts again from the beginning.

Reset signal generation clears the A/D conversion result register (ADCR, ADCRH) to 0000H or 00H.

11.4.2 Input voltage and conversion results

The relationship between the analog input voltage input to the analog input pins (ANI0 to ANI3) and the theoretical A/D conversion result (stored in the 10-bit A/D conversion result register (ADCR)) is shown by the following expression.

$$\text{SAR} = \text{INT} \left(\frac{V_{\text{AIN}}}{V_{\text{DD}}} \times 1024 + 0.5 \right)$$

$$\text{ADCR} = \text{SAR} \times 64$$

or

$$\left(\frac{\text{ADCR}}{64} - 0.5 \right) \times \frac{V_{\text{DD}}}{1024} \leq V_{\text{AIN}} < \left(\frac{\text{ADCR}}{64} + 0.5 \right) \times \frac{V_{\text{DD}}}{1024}$$

where, INT(): Function which returns integer part of value in parentheses

V_{AIN} : Analog input voltage

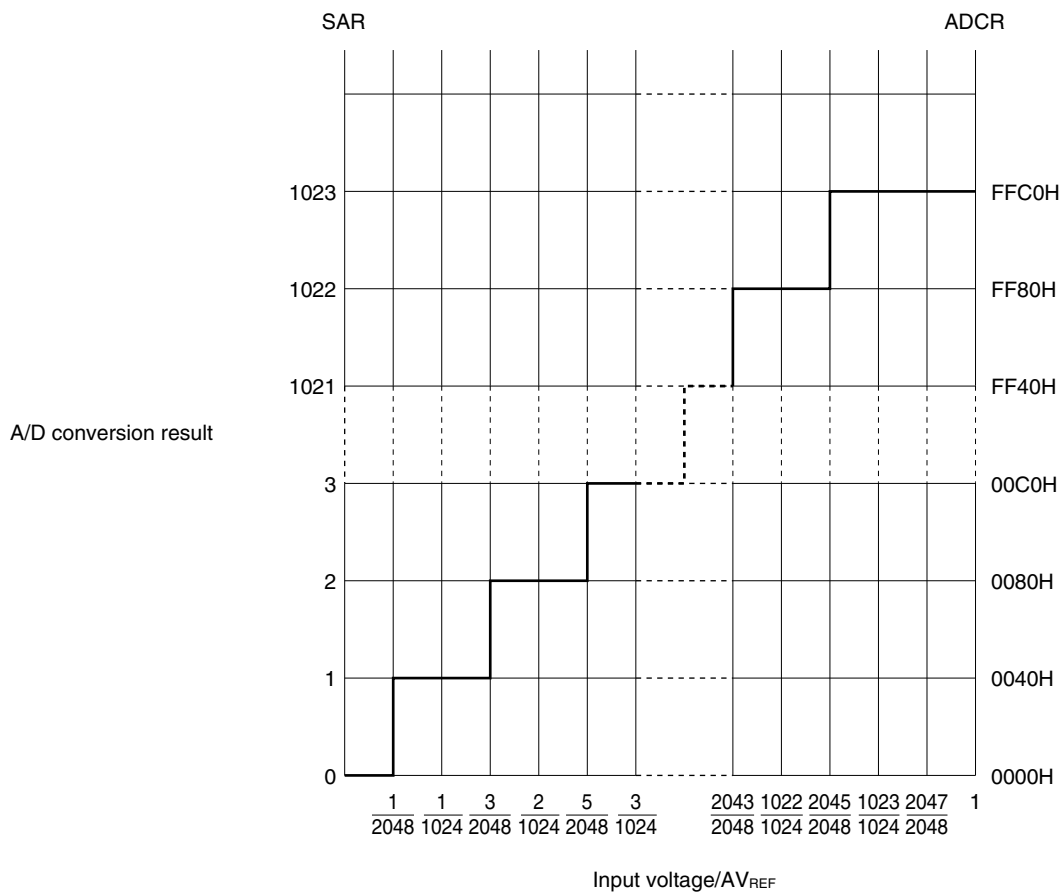
V_{DD} : V_{DD} pin voltage

ADCR: A/D conversion result register (ADCR) value

SAR: Successive approximation register

Figure 11-12 shows the relationship between the analog input voltage and the A/D conversion result.

Figure 11-12. Relationship Between Analog Input Voltage and A/D Conversion Result



11.4.3 A/D converter operation mode

The operation mode of the A/D converter is the select mode. One channel of analog input is selected from ANI0 to ANI3 by the analog input channel specification register (ADS) and A/D conversion is executed.

(1) A/D conversion operation

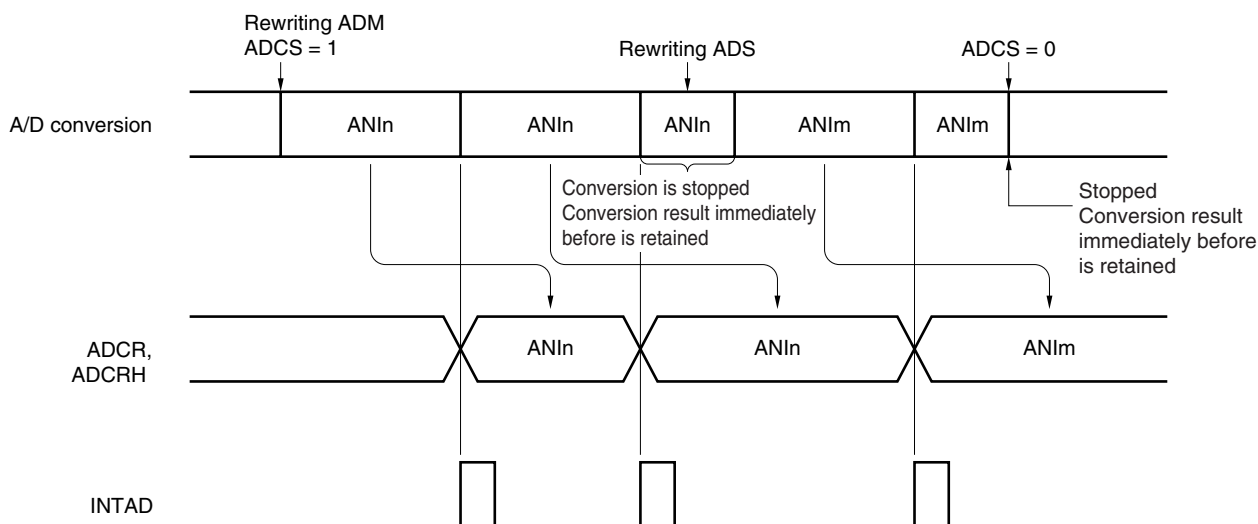
By setting bit 7 (ADCS) of the A/D converter mode register (ADM) to 1, the A/D conversion operation of the voltage, which is applied to the analog input pin specified by the analog input channel specification register (ADS), is started.

When A/D conversion has been completed, the result of the A/D conversion is stored in the A/D conversion result register (ADCR), and an interrupt request signal (INTAD) is generated. When one A/D conversion has been completed, the next A/D conversion operation is immediately started.

If ADS is rewritten during A/D conversion, the A/D conversion operation under execution is stopped and restarted from the beginning.

If 0 is written to ADCS during A/D conversion, A/D conversion is immediately stopped. At this time, the conversion result immediately before is retained.

Figure 11-13. A/D Conversion Operation



Remarks 1. n = 0 to 3

2. m = 0 to 3

The setting methods are described below.

- <1> Set bit 0 (ADCE) of the A/D converter mode register (ADM) to 1.
 - <2> Set the channel to be used in the analog input mode by using bits 3 to 0 (ADPC3 to ADPC0) of the A/D port configuration register (ADPC) and bits 3 to 0 (PM23 to PM20) of port mode register 2 (PM2).
 - <3> Select conversion time by using bits 5 to 1 (FR2 to FR0, LV1, and LV0) of ADM.
 - <4> Select a channel to be used by using bits 1 and 0 (ADS1 and ADS0) of the analog input channel specification register (ADS).
 - <5> Set bit 7 (ADCS) of ADM to 1 to start A/D conversion.
 - <6> When one A/D conversion has been completed, an interrupt request signal (INTAD) is generated.
 - <7> Transfer the A/D conversion data to the A/D conversion result register (ADCR, ADCRH).
- <Change the channel>
- <8> Change the channel using bits 1 and 0 (ADS1 and ADS0) of ADS to start A/D conversion.
 - <9> When one A/D conversion has been completed, an interrupt request signal (INTAD) is generated.
 - <10> Transfer the A/D conversion data to the A/D conversion result register (ADCR, ADCRH).
- <Complete A/D conversion>
- <11> Clear ADCS to 0.
 - <12> Clear ADCE to 0.

- Cautions**
1. Make sure the period of <1> to <5> is 1 μ s or more.
 2. <1> may be done between <2> and <4>.
 3. <1> can be omitted. However, ignore data of the first conversion after <5> in this case.
 4. The period from <6> to <9> differs from the conversion time set using bits 5 to 1 (FR2 to FR0, LV1, LV0) of ADM. The period from <8> to <9> is the conversion time set using FR2 to FR0, LV1, and LV0.

11.5 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 10 bits.

$$\begin{aligned} 1\text{LSB} &= 1/2^{10} = 1/1024 \\ &= 0.098\%\text{FSR} \end{aligned}$$

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 11-14. Overall Error

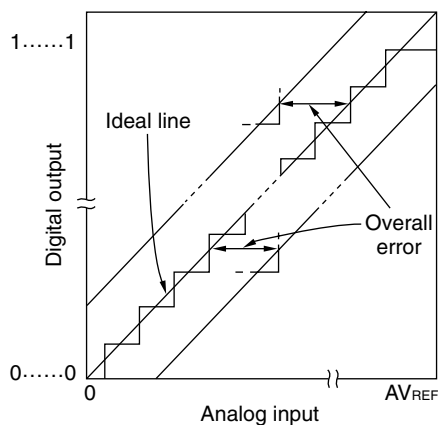
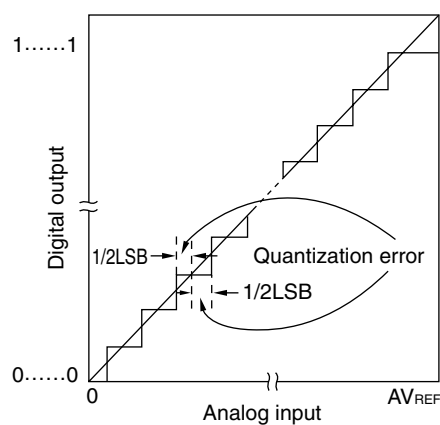


Figure 11-15. Quantization Error



(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 – $3/2\text{LSB}$) 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.

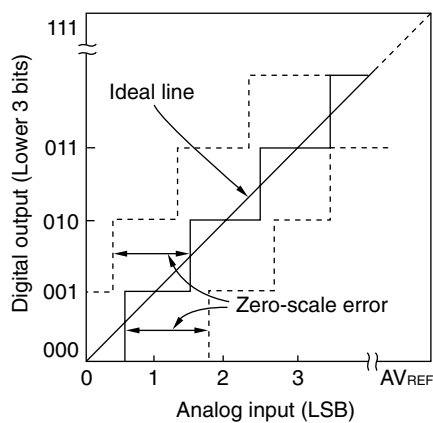
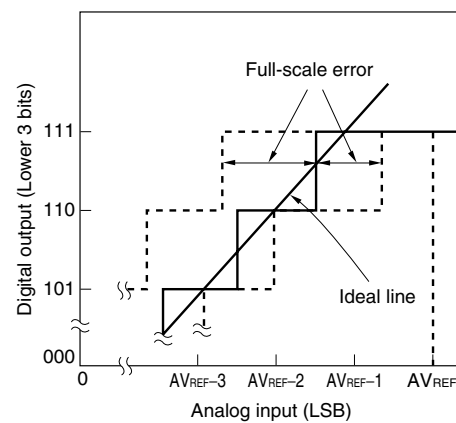
Figure 11-16. Zero-Scale Error**Figure 11-17. Full-Scale Error**

Figure 11-18. Integral Linearity Error

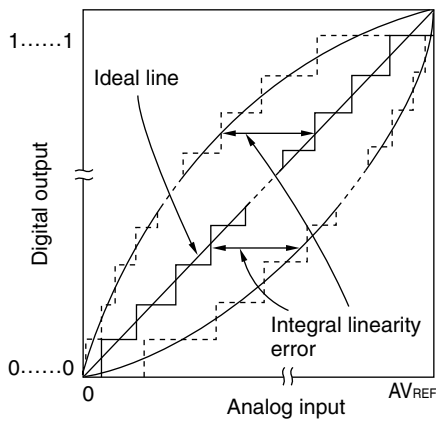
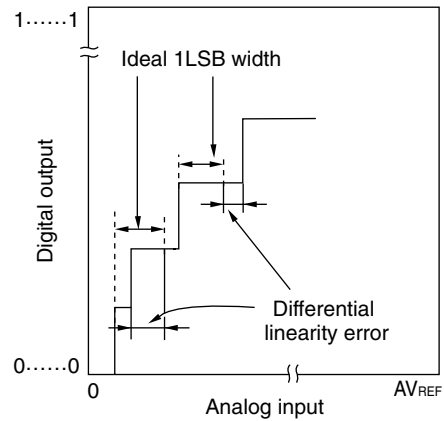


Figure 11-19. Differential Linearity Error

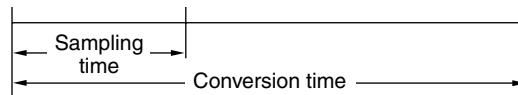
**(8) Conversion time**

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.



11.6 Cautions for A/D Converter

(1) Operating current in STOP mode

The A/D converter stops operating in the STOP mode. At this time, the operating current can be reduced by clearing bit 7 (ADCS) and bit 0 (ADCE) of the A/D converter mode register (ADM) to 0.

To restart from the standby status, clear bit 0 (ADIF) of interrupt request flag register 1L (IF1L) to 0 and start operation.

(2) Input range of ANI0 to ANI3

Observe the rated range of the ANI0 to ANI3 input voltage. If a voltage of V_{DD} or higher and V_{SS} 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.

(3) Conflicting operations

<1> Conflict between A/D conversion result register (ADCR, ADCRH) write and ADCR or ADCRH read by instruction upon the end of conversion

ADCR or ADCRH read has priority. After the read operation, the new conversion result is written to ADCR or ADCRH.

<2> Conflict between ADCR or ADCRH write and A/D converter mode register (ADM) write, analog input channel specification register (ADS), or A/D port configuration register (ADPC) write upon the end of conversion

ADM, ADS, or ADPC write has priority. ADCR or ADCRH write is not performed, nor is the conversion end interrupt signal (INTAD) generated.

(4) Noise countermeasures

To maintain the 10-bit resolution, attention must be paid to noise input to the V_{DD} pin and pins ANI0 to ANI3.

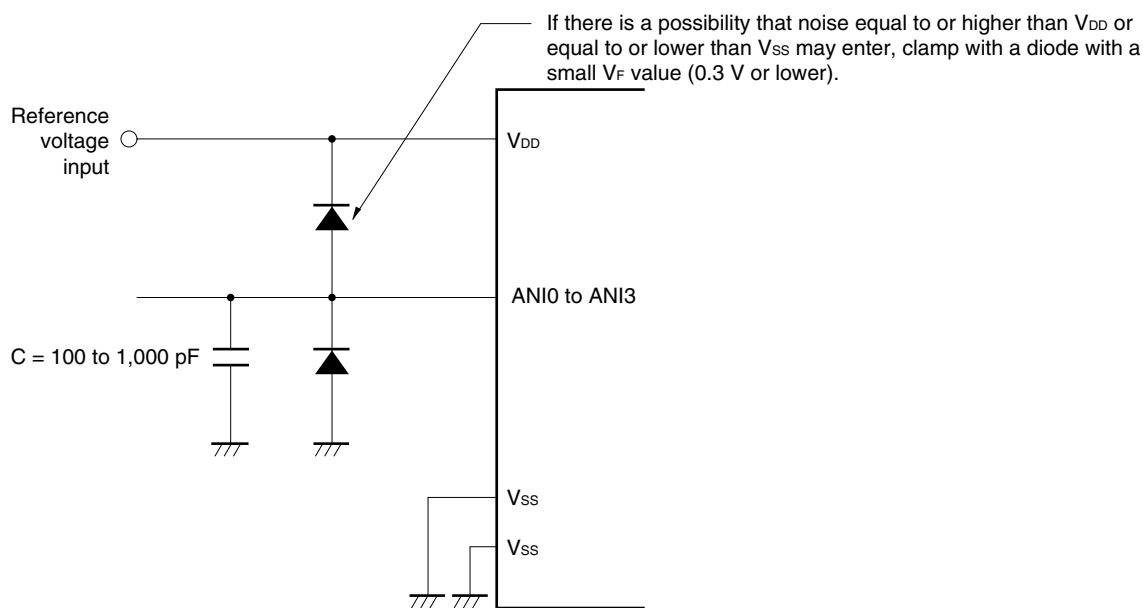
<1> Connect a capacitor with a low equivalent resistance and a good frequency response to the power supply.

<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 11-20 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.

Figure 11-20. Analog Input Pin Connection

**(5) ANI0/P20 to ANI3/P23**

<1> The analog input pins (ANI0 to ANI3) are also used as I/O port pins (P20 to P23).

When A/D conversion is performed with any of ANI0 to ANI3 selected, do not access P20 to P23 while conversion is in progress; otherwise the conversion resolution may be degraded. It is recommended to select pins used as P20 to P23 starting with the ANI0/P20 that is the furthest from V_{DD} .

<2> If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the pins adjacent to the pin undergoing A/D conversion.

(6) Input impedance of ANI0 to ANI3 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, it is recommended to keep the output impedance of the analog input source to within 10 k Ω , and to connect a capacitor of about 100 pF to the ANI0 to ANI3 pins (see **Figure 11-20**).

(7) V_{DD} pin input impedance

A series resistor string of several tens of k Ω is connected between the V_{DD} and V_{SS} pins.

Therefore, if the output impedance of the reference voltage source is high, this will result in a series connection to the series resistor string between the V_{DD} and V_{SS} pins, resulting in a large reference voltage error.

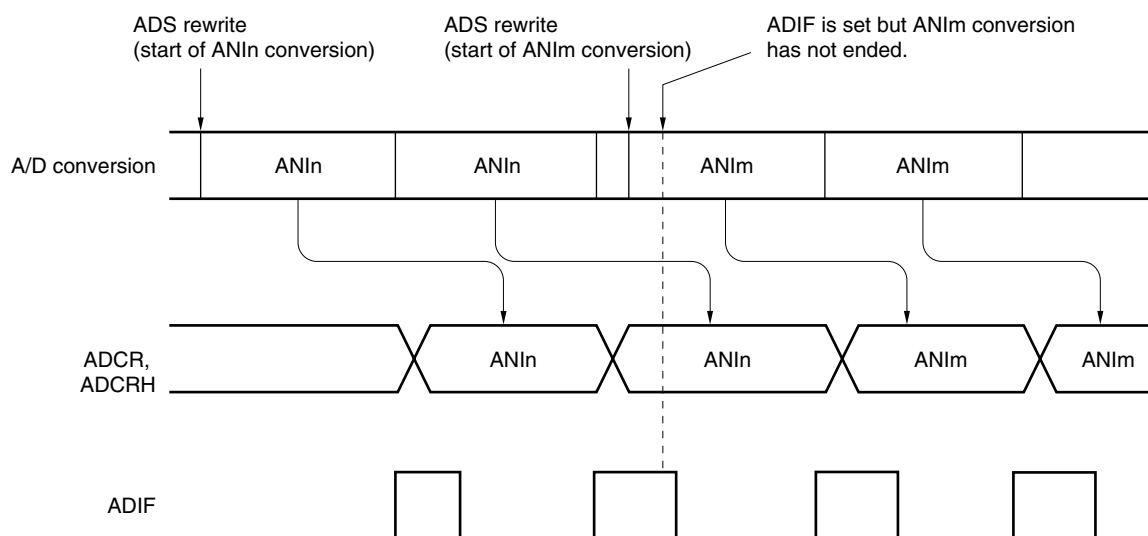
(8) Interrupt request flag (ADIF)

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 for the pre-change analog input may be set just before the ADS rewrite. Caution is therefore required since, at this time, when ADIF is read immediately after the ADS rewrite, ADIF 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 before the A/D conversion operation is resumed.

Figure 11-21. Timing of A/D Conversion End Interrupt Request Generation



Remarks 1. $n = 0$ to 3

2. $m = 0$ to 3

(9) Conversion results just after A/D conversion start

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 μ s after the ADCE bit was set to 1, or if the ADCS bit is set to 1 with the ADCE bit = 0. Take measures such as polling the A/D conversion end interrupt request (INTAD) and removing the first conversion result.

(10) A/D conversion result register (ADCR, ADCRH) read operation

When a write operation is performed to the A/D converter mode register (ADM), analog input channel specification register (ADS), and A/D port configuration register (ADPC), the contents of ADCR and ADCRH may become undefined. Read the conversion result following conversion completion before writing to ADM, ADS, and ADPC. Using a timing other than the above may cause an incorrect conversion result to be read.

(11) Internal equivalent circuit

The equivalent circuit of the analog input block is shown below.

Figure 11-22. Internal Equivalent Circuit of ANIn Pin

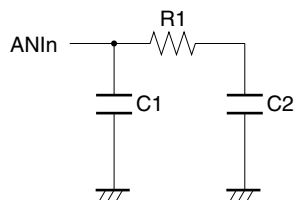


Table 11-5. Resistance and Capacitance Values of Equivalent Circuit (Reference Values)

| V_{DD} | R1 | C1 | C2 |
|--|----------------|------|------|
| $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ | 8.1 k Ω | 8 pF | 5 pF |

- Remarks**
1. The resistance and capacitance values shown in Table 11-5 are not guaranteed values.
 2. n = 0 to 3

CHAPTER 12 SERIAL INTERFACE UART0

12.1 Functions of Serial Interface UART0

Serial interface UART0 has the following two modes.

(1) Operation stop mode

This mode is used when serial communication is not executed and can enable a reduction in the power consumption. For details, see **12.4.1 Operation stop mode**.

(2) Asynchronous serial interface (UART) mode

The functions of this mode are outlined below.

For details, see **12.4.2 Asynchronous serial interface (UART) mode** and **12.4.3 Dedicated baud rate generator**.

- Maximum transfer rate: 625 kbps
- Two-pin configuration TxD0: Transmit data output pin
RxD0: Receive data input pin
- Length of communication data can be selected from 7 or 8 bits.
- Dedicated on-chip 5-bit baud rate generator allowing any baud rate to be set
- Transmission and reception can be performed independently (full-duplex operation).
- Fixed to LSB-first communication

- Cautions**
1. If clock supply to serial interface UART0 is not stopped (e.g., in the HALT mode), normal operation continues. If clock supply to serial interface UART0 is stopped (e.g., in the STOP mode), each register stops operating, and holds the value immediately before clock supply was stopped. The TxD0 pin also holds the value immediately before clock supply was stopped and outputs it. However, the operation is not guaranteed after clock supply is resumed. Therefore, reset the circuit so that POWER0 = 0, RXE0 = 0, and TXE0 = 0.
 2. Set POWER0 = 1 and then set TXE0 = 1 (transmission) or RXE0 = 1 (reception) to start communication.
 3. TXE0 and RXE0 are synchronized by the base clock (f_{CLK0}) set by BRGC0. To enable transmission or reception again, set TXE0 or RXE0 to 1 at least two clocks of base clock after TXE0 or RXE0 has been cleared to 0. If TXE0 or RXE0 is set within two clocks of base clock, the transmission circuit or reception circuit may not be initialized.
 4. Set transmit data to TXS0 at least one base clock (f_{CLK0}) after setting TXE0 = 1.

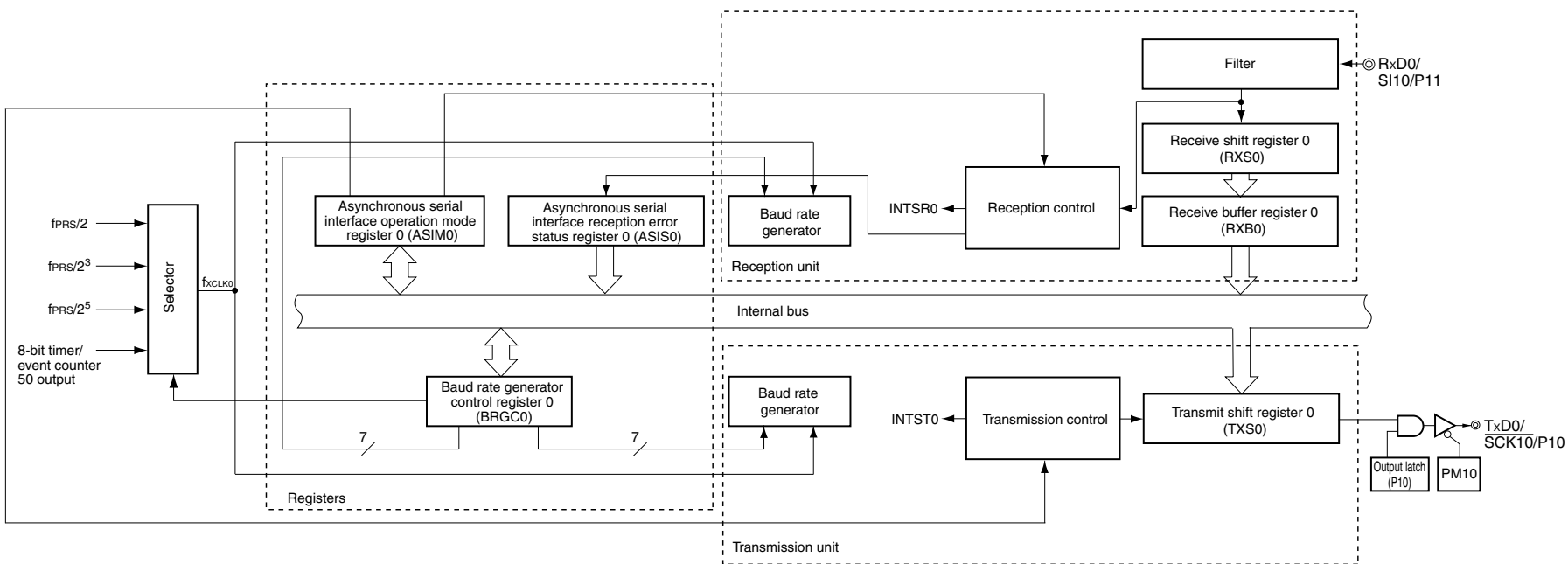
12.2 Configuration of Serial Interface UART0

Serial interface UART0 includes the following hardware.

Table 12-1. Configuration of Serial Interface UART0

| Item | Configuration |
|-------------------|--|
| Registers | Receive buffer register 0 (RXB0) Receive shift register 0 (RXS0) Transmit shift register 0 (TXS0) |
| Control registers | Asynchronous serial interface operation mode register 0 (ASIM0) Asynchronous serial interface reception error status register 0 (ASIS0) Baud rate generator control register 0 (BRGC0) Port mode register 1 (PM1) Port register 1 (P1) |

Figure 12-1. Block Diagram of Serial Interface UART0



(1) Receive buffer register 0 (RXB0)

This 8-bit register stores parallel data converted by receive shift register 0 (RXS0).

Each time 1 byte of data has been received, new receive data is transferred to this register from receive shift register 0 (RXS0).

If the data length is set to 7 bits the receive data is transferred to bits 0 to 6 of RXB0 and the MSB of RXB0 is always 0.

If an overrun error (OVE0) occurs, the receive data is not transferred to RXB0.

RXB0 can be read by an 8-bit memory manipulation instruction. No data can be written to this register.

Reset signal generation and POWER0 = 0 set this register to FFH.

(2) Receive shift register 0 (RXS0)

This register converts the serial data input to the RxD0 pin into parallel data.

RXS0 cannot be directly manipulated by a program.

(3) Transmit shift register 0 (TXS0)

This register is used to set transmit data. Transmission is started when data is written to TXS0, and serial data is transmitted from the TxD0 pins.

TXS0 can be written by an 8-bit memory manipulation instruction. This register cannot be read.

Reset signal generation, POWER0 = 0, and TXE0 = 0 set this register to FFH.

- Cautions**
1. **Set transmit data to TXS0 at least one base clock (f_{XCLK0}) after setting TXE0 = 1.**
 2. **Do not write the next transmit data to TXS0 before the transmission completion interrupt signal (INTST0) is generated.**

12.3 Registers Controlling Serial Interface UART0

Serial interface UART0 is controlled by the following five registers.

- Asynchronous serial interface operation mode register 0 (ASIM0)
- Asynchronous serial interface reception error status register 0 (ASIS0)
- Baud rate generator control register 0 (BRGC0)
- Port mode register 1 (PM1)
- Port register 1 (P1)

(1) Asynchronous serial interface operation mode register 0 (ASIM0)

This 8-bit register controls the serial communication operations of serial interface UART0.

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

Reset signal generation sets this register to 01H.

Figure 12-2. Format of Asynchronous Serial Interface Operation Mode Register 0 (ASIM0) (1/2)

Address: FF70H After reset: 01H R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
|--------|--------|------|------|------|------|-----|-----|---|
| ASIM0 | POWER0 | TXE0 | RXE0 | PS01 | PS00 | CL0 | SL0 | 1 |

| POWER0 | Enables/disables operation of internal operation clock |
|---------------------|--|
| 0 ^{Note 1} | Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit ^{Note 2} . |
| 1 | Enables operation of the internal operation clock. |

| TXE0 | Enables/disables transmission |
|------|--|
| 0 | Disables transmission (synchronously resets the transmission circuit). |
| 1 | Enables transmission. |

| RXE0 | Enables/disables reception |
|------|--|
| 0 | Disables reception (synchronously resets the reception circuit). |
| 1 | Enables reception. |

- Notes**
1. The input from the RxD0 pin is fixed to high level when POWER0 = 0.
 2. Asynchronous serial interface reception error status register 0 (ASIS0), transmit shift register 0 (TXS0), and receive buffer register 0 (RXB0) are reset.

Figure 12-2. Format of Asynchronous Serial Interface Operation Mode Register 0 (ASIM0) (2/2)

| PS01 | PS00 | Transmission operation | Reception operation |
|------|------|-----------------------------|---------------------------------------|
| 0 | 0 | Does not output parity bit. | Reception without parity |
| 0 | 1 | Outputs 0 parity. | Reception as 0 parity ^{Note} |
| 1 | 0 | Outputs odd parity. | Judges as odd parity. |
| 1 | 1 | Outputs even parity. | Judges as even parity. |

| CL0 | Specifies character length of transmit/receive data |
|-----|---|
| 0 | Character length of data = 7 bits |
| 1 | Character length of data = 8 bits |

| SL0 | Specifies number of stop bits of transmit data |
|-----|--|
| 0 | Number of stop bits = 1 |
| 1 | Number of stop bits = 2 |

Note If “reception as 0 parity” is selected, the parity is not judged. Therefore, bit 2 (PE0) of asynchronous serial interface reception error status register 0 (ASIS0) is not set and the error interrupt does not occur.

- Cautions**
1. To start the transmission, set POWER0 to 1 and then set TXE0 to 1. To stop the transmission, clear TXE0 to 0, and then clear POWER0 to 0.
 2. To start the reception, set POWER0 to 1 and then set RXE0 to 1. To stop the reception, clear RXE0 to 0, and then clear POWER0 to 0.
 3. Set POWER0 to 1 and then set RXE0 to 1 while a high level is input to the RxD0 pin. If POWER0 is set to 1 and RXE0 is set to 1 while a low level is input, reception is started.
 4. TXE0 and RXE0 are synchronized by the base clock (f_{CLK0}) set by BRGC0. To enable transmission or reception again, set TXE0 or RXE0 to 1 at least two clocks of base clock after TXE0 or RXE0 has been cleared to 0. If TXE0 or RXE0 is set within two clocks of base clock, the transmission circuit or reception circuit may not be initialized.
 5. Set transmit data to TXS0 at least one base clock (f_{CLK0}) after setting TXE0 = 1.
 6. Clear the TXE0 and RXE0 bits to 0 before rewriting the PS01, PS00, and CL0 bits.
 7. Make sure that TXE0 = 0 when rewriting the SL0 bit. Reception is always performed with “number of stop bits = 1”, and therefore, is not affected by the set value of the SL0 bit.
 8. Be sure to set bit 0 to 1.

(2) Asynchronous serial interface reception error status register 0 (ASIS0)

This register indicates an error status on completion of reception by serial interface UART0. It includes three error flag bits (PE0, FE0, OVE0).

This register is read-only by an 8-bit memory manipulation instruction.

Reset signal generation, or clearing bit 7 (POWER0) or bit 5 (RXE0) of ASIM0 to 0 clears this register to 00H. 00H is read when this register is read. If a reception error occurs, read ASIS0 and then read receive buffer register 0 (RXB0) to clear the error flag.

Figure 12-3. Format of Asynchronous Serial Interface Reception Error Status Register 0 (ASIS0)

Address: FF73H After reset: 00H R

| | | | | | | | | |
|--------|---|---|---|---|---|-----|-----|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ASIS0 | 0 | 0 | 0 | 0 | 0 | PE0 | FE0 | OVE0 |

| | |
|-----|--|
| PE0 | Status flag indicating parity error |
| 0 | If POWER0 = 0 or RXE0 = 0, or if ASIS0 register is read. |
| 1 | If the parity of transmit data does not match the parity bit on completion of reception. |

| | |
|-----|---|
| FE0 | Status flag indicating framing error |
| 0 | If POWER0 = 0 or RXE0 = 0, or if ASIS0 register is read. |
| 1 | If the stop bit is not detected on completion of reception. |

| | |
|------|--|
| OVE0 | Status flag indicating overrun error |
| 0 | If POWER0 = 0 and RXE0 = 0, or if ASIS0 register is read. |
| 1 | If receive data is set to the RXB0 register and the next reception operation is completed before the data is read. |

- Cautions**
1. The operation of the PE0 bit differs depending on the set values of the PS01 and PS00 bits of asynchronous serial interface operation mode register 0 (ASIM0).
 2. Only the first bit of the receive data is checked as the stop bit, regardless of the number of stop bits.
 3. If an overrun error occurs, the next receive data is not written to receive buffer register 0 (RXB0) but discarded.
 4. If data is read from ASIS0, a wait cycle is generated. Do not read data from ASIS0 when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(3) Baud rate generator control register 0 (BRGC0)

This register selects the base clock of serial interface UART0 and the division value of the 5-bit counter.

BRGC0 can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 1FH.

Figure 12-4. Format of Baud Rate Generator Control Register 0 (BRGC0)

Address: FF71H After reset: 1FH R/W

| | | | | | | | | |
|--------|-------|-------|---|-------|-------|-------|-------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BRGC0 | TPS01 | TPS00 | 0 | MDL04 | MDL03 | MDL02 | MDL01 | MDL00 |

| TPS01 | TPS00 | Base clock (f_{XCLK0}) selection | | | |
|-------|-------|--------------------------------------|---------------------------|----------------------------|-----------|
| | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ | |
| 0 | 0 | TM50 output ^{Note} | | | |
| 0 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 1 | 0 | $f_{PRS}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz |
| 1 | 1 | $f_{PRS}/2^5$ | 62.5 kHz | 156.25 kHz | 312.5 kHz |

| MDL04 | MDL03 | MDL02 | MDL01 | MDL00 | k | Selection of 5-bit counter output clock |
|-------|-------|-------|-------|-------|----|---|
| 0 | 0 | × | × | × | × | Setting prohibited |
| 0 | 1 | 0 | 0 | 0 | 8 | $f_{XCLK0}/8$ |
| 0 | 1 | 0 | 0 | 1 | 9 | $f_{XCLK0}/9$ |
| 0 | 1 | 0 | 1 | 0 | 10 | $f_{XCLK0}/10$ |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| • | • | • | • | • | • | • |
| 1 | 1 | 0 | 1 | 0 | 26 | $f_{XCLK0}/26$ |
| 1 | 1 | 0 | 1 | 1 | 27 | $f_{XCLK0}/27$ |
| 1 | 1 | 1 | 0 | 0 | 28 | $f_{XCLK0}/28$ |
| 1 | 1 | 1 | 0 | 1 | 29 | $f_{XCLK0}/29$ |
| 1 | 1 | 1 | 1 | 0 | 30 | $f_{XCLK0}/30$ |
| 1 | 1 | 1 | 1 | 1 | 31 | $f_{XCLK0}/31$ |

Note Note the following points when selecting the TM50 output as the base clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.
It is not necessary to enable (TOE50 = 1) TO50 output in any mode.

- Cautions**
1. Make sure that bit 6 (TXE0) and bit 5 (RXE0) of the ASIM0 register = 0 when rewriting the MDL04 to MDL00 bits.
 2. Make sure that bit 7 (POWER0) of the ASIM0 register = 0 when rewriting the TPS01 and TPS00 bits.
 3. The baud rate value is the output clock of the 5-bit counter divided by 2.

- Remarks**
1. f_{CLK0} : Frequency of base clock selected by the TPS01 and TPS00 bits
 2. f_{PRS} : Peripheral hardware clock frequency
 3. k : Value set by the MDL04 to MDL00 bits ($k = 8, 9, 10, \dots, 31$)
 4. \times : Don't care
 5. TMC506: Bit 6 of 8-bit timer mode control register 50 (TMC50)
TMC501: Bit 1 of TMC50

(4) Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.

When using the P10/TxD0/ $\overline{\text{SCK10}}$ pin for serial interface data output, clear PM10 to 0 and set the output latch of P10 to 1.

When using the P11/RxD0/SI10 pin for serial interface data input, set PM11 to 1. The output latch of P11 at this time may be 0 or 1.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 12-5. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|------|------|------|------|------|------|------|
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |
| PM1n | P1n pin I/O mode selection (n = 0 to 7) | | | | | | | |
| 0 | Output mode (output buffer on) | | | | | | | |
| 1 | Input mode (output buffer off) | | | | | | | |

12.4 Operation of Serial Interface UART0

Serial interface UART0 has the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

12.4.1 Operation stop mode

In this mode, serial communication cannot be executed, thus reducing the power consumption. In addition, the pins can be used as ordinary port pins in this mode. To set the operation stop mode, clear bits 7, 6, and 5 (POWER0, TXE0, and RXE0) of ASIM0 to 0.

(1) Register used

The operation stop mode is set by asynchronous serial interface operation mode register 0 (ASIM0).

ASIM0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 01H.

Address: FF70H After reset: 01H R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
|--------|--------|------|------|------|------|-----|-----|---|
| ASIM0 | POWER0 | TXE0 | RXE0 | PS01 | PS00 | CL0 | SL0 | 1 |

| | |
|---------------------|--|
| POWER0 | Enables/disables operation of internal operation clock |
| 0 ^{Note 1} | Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit ^{Note 2} . |

| | |
|------|--|
| TXE0 | Enables/disables transmission |
| 0 | Disables transmission (synchronously resets the transmission circuit). |

| | |
|------|--|
| RXE0 | Enables/disables reception |
| 0 | Disables reception (synchronously resets the reception circuit). |

- Notes**
1. The input from the RxD0 pin is fixed to high level when POWER0 = 0.
 2. Asynchronous serial interface reception error status register 0 (ASIS0), transmit shift register 0 (TXS0), and receive buffer register 0 (RXB0) are reset.

Caution Clear POWER0 to 0 after clearing TXE0 and RXE0 to 0 to set the operation stop mode. To start the communication, set POWER0 to 1, and then set TXE0 or RXE0 to 1.

Remark To use the RxD0/SI10/P11 and TxD0/ $\overline{\text{SCK10}}$ /P10 pins as general-purpose port pins, see **CHAPTER 4 PORT FUNCTIONS**.

12.4.2 Asynchronous serial interface (UART) mode

In this mode, 1-byte data is transmitted/received following a start bit, and a full-duplex operation can be performed.

A dedicated UART baud rate generator is incorporated, so that communication can be executed at a wide range of baud rates.

(1) Registers used

- Asynchronous serial interface operation mode register 0 (ASIM0)
- Asynchronous serial interface reception error status register 0 (ASIS0)
- Baud rate generator control register 0 (BRGC0)
- Port mode register 1 (PM1)
- Port register 1 (P1)

The basic procedure of setting an operation in the UART mode is as follows.

- <1> Set the BRGC0 register (see **Figure 12-4**).
- <2> Set bits 1 to 4 (SL0, CL0, PS00, and PS01) of the ASIM0 register (see **Figure 12-2**).
- <3> Set bit 7 (POWER0) of the ASIM0 register to 1.
- <4> Set bit 6 (TXE0) of the ASIM0 register to 1. → Transmission is enabled.
Set bit 5 (RXE0) of the ASIM0 register to 1. → Reception is enabled.
- <5> Write data to the TXS0 register. → Data transmission is started.

Caution Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 12-2. Relationship Between Register Settings and Pins

| POWER0 | TXE0 | RXE0 | PM10 | P10 | PM11 | P11 | UART0 Operation | Pin Function | |
|--------|------|------|------|-----|------|-----|----------------------------|----------------|---------------|
| | | | | | | | | TxD0/SCK10/P10 | RxD0/SI10/P11 |
| 0 | 0 | 0 | × | × | × | × | Stop | SCK10/P10 | SI10/P11 |
| 1 | 0 | 1 | × | × | 1 | × | Reception | SCK10/P10 | RxD0 |
| | 1 | 0 | 0 | 1 | × | × | Transmission | TxD0 | SI10/P11 |
| | 1 | 1 | 0 | 1 | 1 | × | Transmission/ reception | TxD0 | RxD0 |

Note Can be set as port function or serial interface CSI10.

Remark ×: don't care

POWER0: Bit 7 of asynchronous serial interface operation mode register 0 (ASIM0)

TXE0: Bit 6 of ASIM0

RXE0: Bit 5 of ASIM0

PM1×: Port mode register

P1×: Port output latch

(2) Communication operation

(a) Format and waveform example of normal transmit/receive data

Figures 12-6 and 12-7 show the format and waveform example of the normal transmit/receive data.

Figure 12-6. Format of Normal UART Transmit/Receive Data



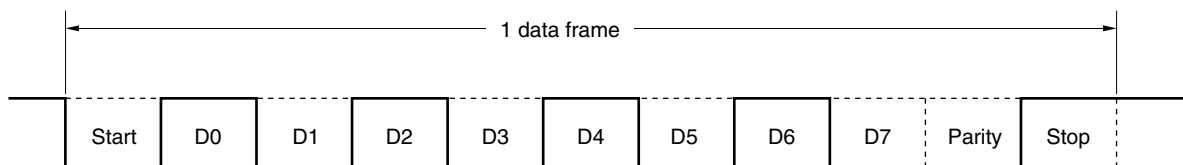
One data frame consists of the following bits.

- Start bit ... 1 bit
- Character bits ... 7 or 8 bits (LSB first)
- Parity bit ... Even parity, odd parity, 0 parity, or no parity
- Stop bit ... 1 or 2 bits

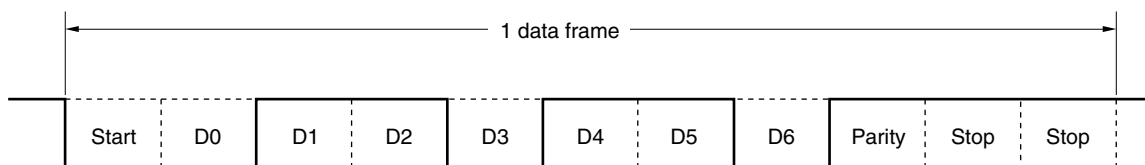
The character bit length, parity, and stop bit length in one data frame are specified by asynchronous serial interface operation mode register 0 (ASIM0).

Figure 12-7. Example of Normal UART Transmit/Receive Data Waveform

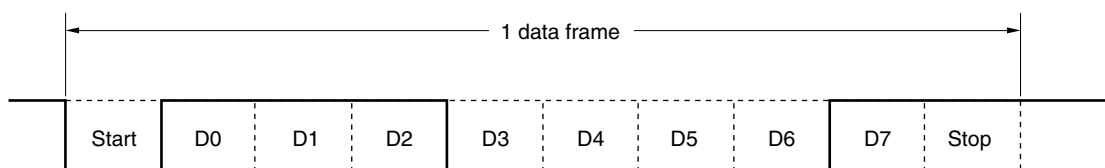
1. Data length: 8 bits, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



2. Data length: 7 bits, Parity: Odd parity, Stop bit: 2 bits, Communication data: 36H



3. Data length: 8 bits, Parity: None, Stop bit: 1 bit, Communication data: 87H



(b) Parity types and operation

The parity bit is used to detect a bit error in communication data. Usually, the same type of parity bit is used on both the transmission and reception sides. With even parity and odd parity, a 1-bit (odd number) error can be detected. With zero parity and no parity, an error cannot be detected.

(i) Even parity

- Transmission

Transmit data, including the parity bit, is controlled so that the number of bits that are “1” is even.

The value of the parity bit is as follows.

If transmit data has an odd number of bits that are “1”: 1

If transmit data has an even number of bits that are “1”: 0

- Reception

The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is odd, a parity error occurs.

(ii) Odd parity

- Transmission

Unlike even parity, transmit data, including the parity bit, is controlled so that the number of bits that are “1” is odd.

If transmit data has an odd number of bits that are “1”: 0

If transmit data has an even number of bits that are “1”: 1

- Reception

The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is even, a parity error occurs.

(iii) 0 parity

The parity bit is cleared to 0 when data is transmitted, regardless of the transmit data.

The parity bit is not detected when the data is received. Therefore, a parity error does not occur regardless of whether the parity bit is “0” or “1”.

(iv) No parity

No parity bit is appended to the transmit data.

Reception is performed assuming that there is no parity bit when data is received. Because there is no parity bit, a parity error does not occur.

(c) Transmission

If bit 7 (POWER0) of asynchronous serial interface operation mode register 0 (ASIM0) is set to 1 and bit 6 (TXE0) of ASIM0 is then set to 1, transmission is enabled. Transmission can be started by writing transmit data to transmit shift register 0 (TXS0). The start bit, parity bit, and stop bit are automatically appended to the data.

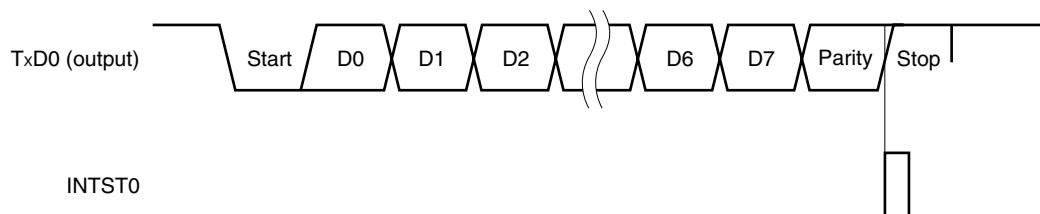
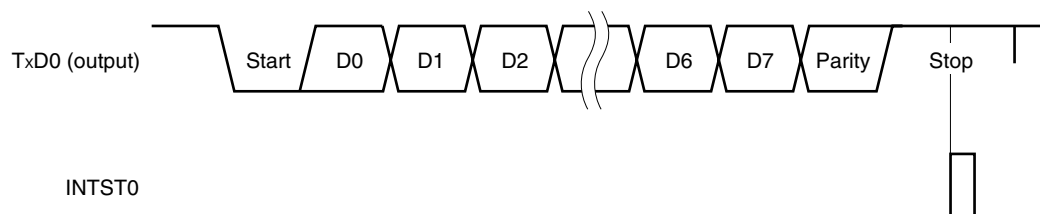
When transmission is started, the start bit is output from the TxD0 pin, and the transmit data is output followed by the rest of the data in order starting from the LSB. When transmission is completed, the parity and stop bits set by ASIM0 are appended and a transmission completion interrupt request (INTST0) is generated.

Transmission is stopped until the data to be transmitted next is written to TXS0.

Figure 12-8 shows the timing of the transmission completion interrupt request (INTST0). This interrupt occurs as soon as the last stop bit has been output.

Caution After transmit data is written to TXS0, do not write the next transmit data before the transmission completion interrupt signal (INTST0) is generated.

Figure 12-8. Transmission Completion Interrupt Request Timing

1. Stop bit length: 1**2. Stop bit length: 2**

(d) Reception

Reception is enabled and the RxD0 pin input is sampled when bit 7 (POWER0) of asynchronous serial interface operation mode register 0 (ASIM0) is set to 1 and then bit 5 (RXE0) of ASIM0 is set to 1.

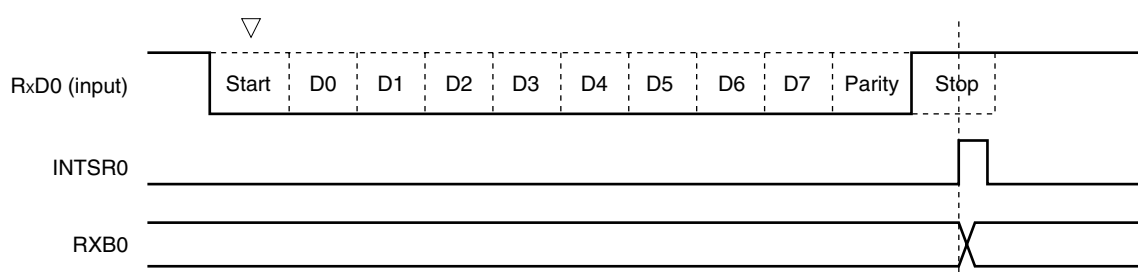
The 5-bit counter of the baud rate generator starts counting when the falling edge of the RxD0 pin input is detected. When the set value of baud rate generator control register 0 (BRGC0) has been counted, the RxD0 pin input is sampled again (▽ in Figure 12-9). If the RxD0 pin is low level at this time, it is recognized as a start bit.

When the start bit is detected, reception is started, and serial data is sequentially stored in receive shift register 0 (RXS0) at the set baud rate. When the stop bit has been received, the reception completion interrupt (INTSR0) is generated and the data of RXS0 is written to receive buffer register 0 (RXB0). If an overrun error (OVE0) occurs, however, the receive data is not written to RXB0.

Even if a parity error (PE0) occurs while reception is in progress, reception continues to the reception position of the stop bit, and an reception error interrupt (INTSR0) is generated after completion of reception.

INTSR0 occurs upon completion of reception and in case of a reception error.

Figure 12-9. Reception Completion Interrupt Request Timing



- Cautions**
1. If a reception error occurs, read asynchronous serial interface reception error status register 0 (ASIS0) and then read receive buffer register 0 (RXB0) to clear the error flag. Otherwise, an overrun error will occur when the next data is received, and the reception error status will persist.
 2. Reception is always performed with the “number of stop bits = 1”. The second stop bit is ignored.

(e) Reception error

Three types of errors may occur during reception: a parity error, framing error, or overrun error. If the error flag of asynchronous serial interface reception error status register 0 (ASIS0) is set as a result of data reception, a reception error interrupt (INTSR0) is generated.

Which error has occurred during reception can be identified by reading the contents of ASIS0 in the reception error interrupt (INTSR0) servicing (see **Figure 12-3**).

The contents of ASIS0 are cleared to 0 when ASIS0 is read.

Table 12-3. Cause of Reception Error

| Reception Error | Cause |
|-----------------|--|
| Parity error | The parity specified for transmission does not match the parity of the receive data. |
| Framing error | Stop bit is not detected. |
| Overrun error | Reception of the next data is completed before data is read from receive buffer register 0 (RXB0). |

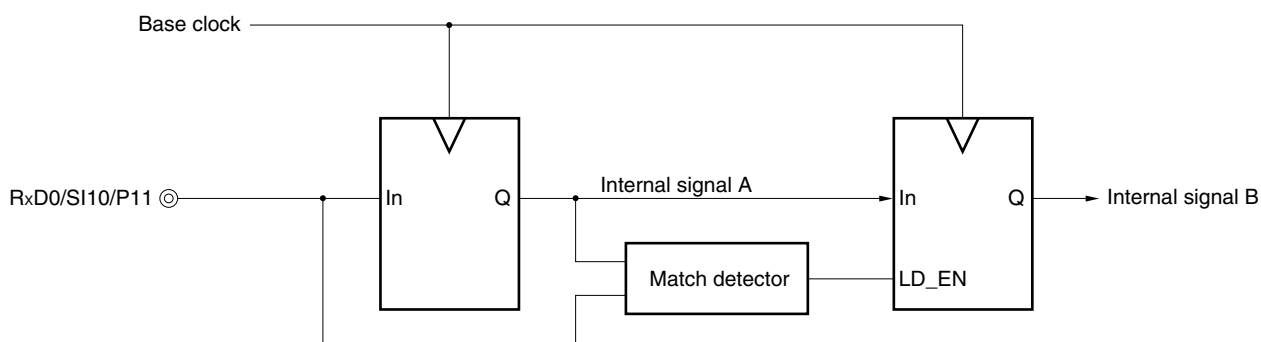
(f) Noise filter of receive data

The Rx/D0 signal is sampled using the base clock output by the prescaler block.

If two sampled values are the same, the output of the match detector changes, and the data is sampled as input data.

Because the circuit is configured as shown in Figure 12-10, the internal processing of the reception operation is delayed by two clocks from the external signal status.

Figure 12-10. Noise Filter Circuit



12.4.3 Dedicated baud rate generator

The dedicated baud rate generator consists of a source clock selector and a 5-bit programmable counter, and generates a serial clock for transmission/reception of UART0.

Separate 5-bit counters are provided for transmission and reception.

(1) Configuration of baud rate generator

- Base clock

The clock selected by bits 7 and 6 (TPS01 and TPS00) of baud rate generator control register 0 (BRGC0) is supplied to each module when bit 7 (POWER0) of asynchronous serial interface operation mode register 0 (ASIM0) is 1. This clock is called the base clock and its frequency is called f_{CLK0} . The base clock is fixed to low level when POWER0 = 0.

- Transmission counter

This counter stops operation, cleared to 0, when bit 7 (POWER0) or bit 6 (TXE0) of asynchronous serial interface operation mode register 0 (ASIM0) is 0.

It starts counting when POWER0 = 1 and TXE0 = 1.

The counter is cleared to 0 when the first data transmitted is written to transmit shift register 0 (TXS0).

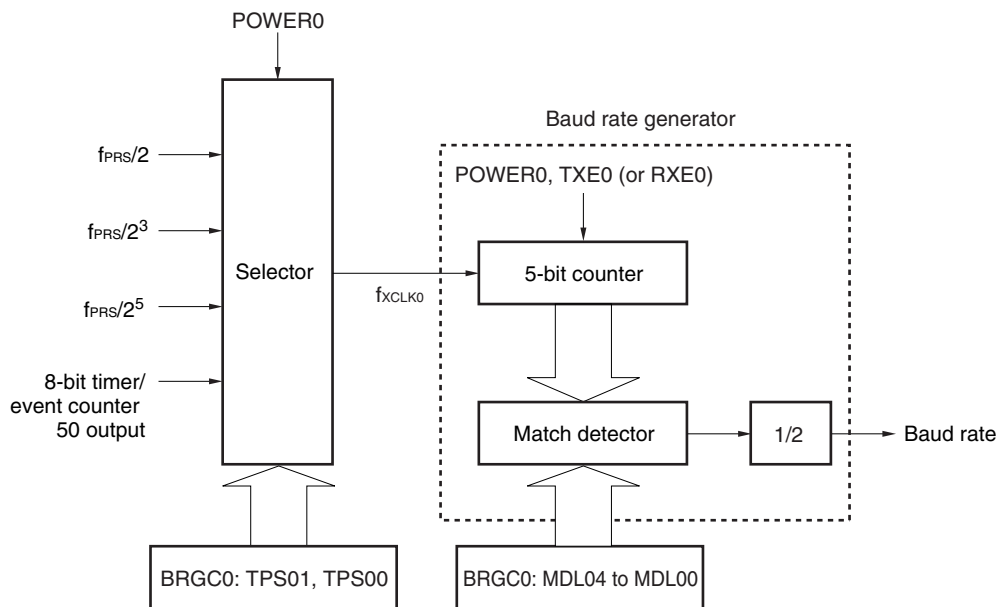
- Reception counter

This counter stops operation, cleared to 0, when bit 7 (POWER0) or bit 5 (RXE0) of asynchronous serial interface operation mode register 0 (ASIM0) is 0.

It starts counting when the start bit has been detected.

The counter stops operation after one frame has been received, until the next start bit is detected.

Figure 12-11. Configuration of Baud Rate Generator



Remark $POWER0$: Bit 7 of asynchronous serial interface operation mode register 0 (ASIM0)
 $TXE0$: Bit 6 of ASIM0
 $RXE0$: Bit 5 of ASIM0
 $BRGC0$: Baud rate generator control register 0

(2) Generation of serial clock

A serial clock to be generated can be specified by using baud rate generator control register 0 (BRGC0).

Select the clock to be input to the 5-bit counter by using bits 7 and 6 (TPS01 and TPS00) of BRGC0.

Bits 4 to 0 (MDL04 to MDL00) of BRGC0 can be used to select the division value ($f_{XCLK0}/8$ to $f_{XCLK0}/31$) of the 5-bit counter.

12.4.4 Calculation of baud rate

(1) Baud rate calculation expression

The baud rate can be calculated by the following expression.

- Baud rate = $\frac{f_{XCLK0}}{2 \times k}$ [bps]

f_{XCLK0} : Frequency of base clock selected by the TPS01 and TPS00 bits of the BRGC0 register

k: Value set by the MDL04 to MDL00 bits of the BRGC0 register (k = 8, 9, 10, ..., 31)

Table 12-4. Set Value of TPS01 and TPS00

| TPS01 | TPS00 | Base clock (f_{XCLK0}) selection | | | |
|-------|-------|--------------------------------------|-------------------|-------------------|--------------------|
| | | | $f_{PRS} = 2$ MHz | $f_{PRS} = 5$ MHz | $f_{PRS} = 10$ MHz |
| 0 | 0 | TM50 output ^{Note} | | | |
| 0 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 1 | 0 | $f_{PRS}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz |
| 1 | 1 | $f_{PRS}/2^5$ | 62.5 kHz | 156.25 kHz | 312.5 kHz |

Note Note the following points when selecting the TM50 output as the base clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.
It is not necessary to enable (TOE50 = 1) TO50 output in any mode.

(2) Error of baud rate

The baud rate error can be calculated by the following expression.

$$\bullet \text{ Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (correct baud rate)}} - 1 \right) \times 100 [\%]$$

- Cautions**
1. Keep the baud rate error during transmission to within the permissible error range at the reception destination.
 2. Make sure that the baud rate error during reception satisfies the range shown in (4) Permissible baud rate range during reception.

Example: Frequency of base clock = 2.5 MHz = 2,500,000 Hz
 Set value of MDL04 to MDL00 bits of BRGC0 register = 10000B (k = 16)
 Target baud rate = 76,800 bps

$$\begin{aligned} \text{Baud rate} &= 2.5 \text{ M}/(2 \times 16) \\ &= 2,500,000/(2 \times 16) = 78,125 [\text{bps}] \end{aligned}$$

$$\begin{aligned} \text{Error} &= (78,125/76,800 - 1) \times 100 \\ &= 1.725 [\%] \end{aligned}$$

(3) Example of setting baud rate**Table 12-5. Set Data of Baud Rate Generator**

| Baud Rate [bps] | f _{PRS} = 2.0 MHz | | | | f _{PRS} = 5.0 MHz | | | | f _{PRS} = 10.0 MHz | | | |
|-----------------|----------------------------|----|------------------|---------|----------------------------|----|------------------|---------|-----------------------------|----|------------------|---------|
| | TPS01, TPS00 | k | Calculated Value | ERR [%] | TPS01, TPS00 | k | Calculated Value | ERR [%] | TPS01, TPS00 | k | Calculated Value | ERR [%] |
| 4800 | 2H | 26 | 4808 | 0.16 | 3H | 16 | 4883 | 1.73 | – | – | – | – |
| 9600 | 2H | 13 | 9615 | 0.16 | 3H | 8 | 9766 | 1.73 | 3H | 16 | 9766 | 1.73 |
| 10400 | 2H | 12 | 10417 | 0.16 | 2H | 30 | 10417 | 0.16 | 3H | 15 | 10417 | 0.16 |
| 19200 | 1H | 26 | 19231 | 0.16 | 2H | 16 | 19531 | 1.73 | 3H | 8 | 19531 | 1.73 |
| 24000 | 1H | 21 | 23810 | –0.79 | 2H | 13 | 24038 | 0.16 | 2H | 26 | 24038 | 0.16 |
| 31250 | 1H | 16 | 31250 | 0 | 2H | 10 | 31250 | 0 | 2H | 20 | 31250 | 0 |
| 33600 | 1H | 15 | 33333 | –0.79 | 2H | 9 | 34722 | 3.34 | 2H | 19 | 32895 | –2.1 |
| 38400 | 1H | 13 | 38462 | 0.16 | 2H | 8 | 39063 | 1.73 | 2H | 16 | 39063 | 1.73 |
| 56000 | 1H | 9 | 55556 | –0.79 | 1H | 22 | 56818 | 1.46 | 2H | 11 | 56818 | 1.46 |
| 62500 | 1H | 8 | 62500 | 0 | 1H | 20 | 62500 | 0 | 2H | 10 | 62500 | 0 |
| 76800 | – | – | – | – | 1H | 16 | 78125 | 1.73 | 2H | 8 | 78125 | 1.73 |
| 115200 | – | – | – | – | 1H | 11 | 113636 | –1.36 | 1H | 22 | 113636 | –1.36 |
| 153600 | – | – | – | – | 1H | 8 | 156250 | 1.73 | 1H | 16 | 156250 | 1.73 |
| 312500 | – | – | – | – | – | – | – | – | 1H | 8 | 312500 | 0 |
| 625000 | – | – | – | – | – | – | – | – | – | – | – | – |

Remark TPS01, TPS00: Bits 7 and 6 of baud rate generator control register 0 (BRGC0) (setting of base clock (f_{CLK0}))

k: Value set by the MDL04 to MDL00 bits of BRGC0 (k = 8, 9, 10, ..., 31)

f_{PRS}: Peripheral hardware clock frequency

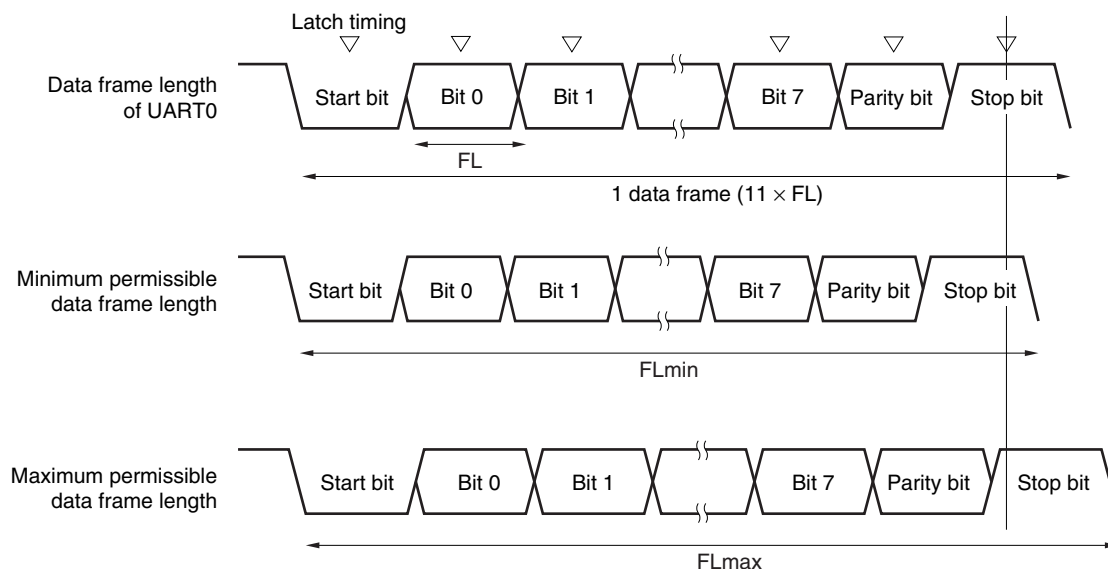
ERR: Baud rate error

(4) Permissible baud rate range during reception

The permissible error from the baud rate at the transmission destination during reception is shown below.

Caution Make sure that the baud rate error during reception is within the permissible error range, by using the calculation expression shown below.

Figure 12-12. Permissible Baud Rate Range During Reception



As shown in Figure 12-12, the latch timing of the receive data is determined by the counter set by baud rate generator control register 0 (BRGC0) after the start bit has been detected. If the last data (stop bit) meets this latch timing, the data can be correctly received.

Assuming that 11-bit data is received, the theoretical values can be calculated as follows.

$$FL = (\text{Brate})^{-1}$$

Brate: Baud rate of UART0

k: Set value of BRGC0

FL: 1-bit data length

Margin of latch timing: 2 clocks

$$\text{Minimum permissible data frame length: } FL_{\min} = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$$

Therefore, the maximum receivable baud rate at the transmission destination is as follows.

$$BR_{\max} = (FL_{\min}/11)^{-1} = \frac{22k}{21k+2} \text{ Brate}$$

Similarly, the maximum permissible data frame length can be calculated as follows.

$$\frac{10}{11} \times FL_{\max} = 11 \times FL - \frac{k+2}{2 \times k} \times FL = \frac{21k-2}{2 \times k} FL$$

$$FL_{\max} = \frac{21k-2}{20k} FL \times 11$$

Therefore, the minimum receivable baud rate at the transmission destination is as follows.

$$BR_{\min} = (FL_{\max}/11)^{-1} = \frac{20k}{21k-2} \text{ Brate}$$

The permissible baud rate error between UART0 and the transmission destination can be calculated from the above minimum and maximum baud rate expressions, as follows.

Table 12-6. Maximum/Minimum Permissible Baud Rate Error

| Division Ratio (k) | Maximum Permissible Baud Rate Error | Minimum Permissible Baud Rate Error |
|--------------------|-------------------------------------|-------------------------------------|
| 8 | +3.53% | -3.61% |
| 16 | +4.14% | -4.19% |
| 24 | +4.34% | -4.38% |
| 31 | +4.44% | -4.47% |

- Remarks**
1. The permissible error of reception depends on the number of bits in one frame, input clock frequency, and division ratio (k). The higher the input clock frequency and the higher the division ratio (k), the higher the permissible error.
 2. k: Set value of BRGCO

CHAPTER 13 SERIAL INTERFACE UART6

13.1 Functions of Serial Interface UART6

Serial interface UART6 has the following two modes.

(1) Operation stop mode

This mode is used when serial communication is not executed and can enable a reduction in the power consumption. For details, see **13.4.1 Operation stop mode**.

(2) Asynchronous serial interface (UART) mode

This mode supports the LIN (Local Interconnect Network)-bus. The functions of this mode are outlined below. For details, see **13.4.2 Asynchronous serial interface (UART) mode** and **13.4.3 Dedicated baud rate generator**.

- Maximum transfer rate: 625 kbps
- Two-pin configuration
 - TxD6: Transmit data output pin
 - RxD6: Receive data input pin
- Data length of communication data can be selected from 7 or 8 bits.
- Dedicated internal 8-bit baud rate generator allowing any baud rate to be set
- Transmission and reception can be performed independently (full duplex operation).
- MSB- or LSB-first communication selectable
- Inverted transmission operation
- Sync break field transmission from 13 to 20 bits
- More than 11 bits can be identified for sync break field reception (SBF reception flag provided).

- Cautions**
1. The TxD6 output inversion function inverts only the transmission side and not the reception side. To use this function, the reception side must be ready for reception of inverted data.
 2. If clock supply to serial interface UART6 is not stopped (e.g., in the HALT mode), normal operation continues. If clock supply to serial interface UART6 is stopped (e.g., in the STOP mode), each register stops operating, and holds the value immediately before clock supply was stopped. The TxD6 pin also holds the value immediately before clock supply was stopped and outputs it. However, the operation is not guaranteed after clock supply is resumed. Therefore, reset the circuit so that POWER6 = 0, RXE6 = 0, and TXE6 = 0.
 3. Set POWER6 = 1 and then set TXE6 = 1 (transmission) or RXE6 = 1 (reception) to start communication.
 4. TXE6 and RXE6 are synchronized by the base clock (f_{XCLK6}) set by CKSR6. To enable transmission or reception again, set TXE6 or RXE6 to 1 at least two clocks of the base clock after TXE6 or RXE6 has been cleared to 0. If TXE6 or RXE6 is set within two clocks of the base clock, the transmission circuit or reception circuit may not be initialized.
 5. Set transmit data to TXB6 at least one base clock (f_{XCLK6}) after setting TXE6 = 1.
 6. If data is continuously transmitted, the communication timing from the stop bit to the next start bit is extended two operating clocks of the macro. However, this does not affect the result of communication because the reception side initializes the timing when it has detected a start bit. Do not use the continuous transmission function if the interface is used in LIN communication operation.

Remark LIN stands for Local Interconnect Network and is a low-speed (1 to 20 kbps) serial communication protocol intended to aid the cost reduction of an automotive network.

LIN communication is single-master communication, and up to 15 slaves can be connected to one master.

The LIN slaves are used to control the switches, actuators, and sensors, and these are connected to the LIN master via the LIN network.

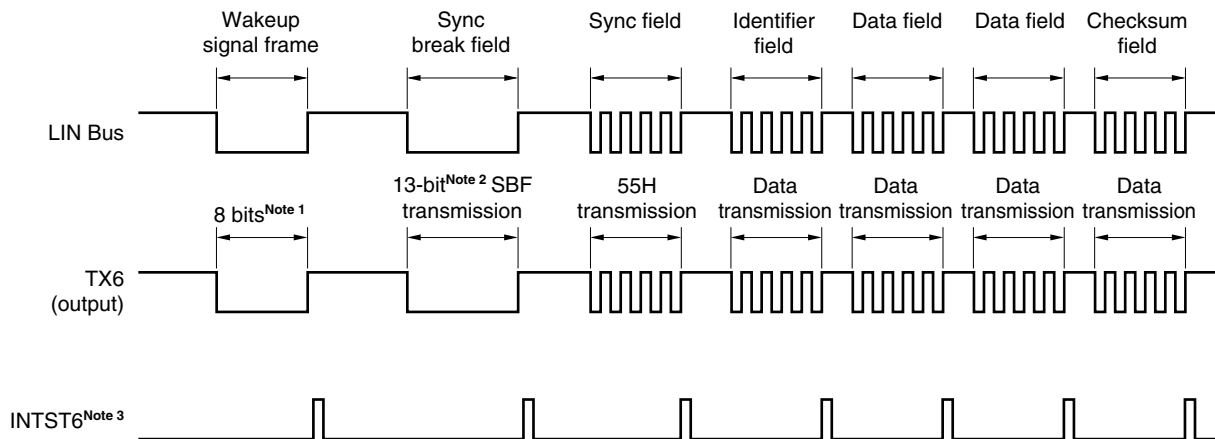
Normally, the LIN master is connected to a network such as CAN (Controller Area Network).

In addition, the LIN bus uses a single-wire method and is connected to the nodes via a transceiver that complies with ISO9141.

In the LIN protocol, the master transmits a frame with baud rate information and the slave receives it and corrects the baud rate error. Therefore, communication is possible when the baud rate error in the slave is $\pm 15\%$ or less.

Figures 13-1 and 13-2 outline the transmission and reception operations of LIN.

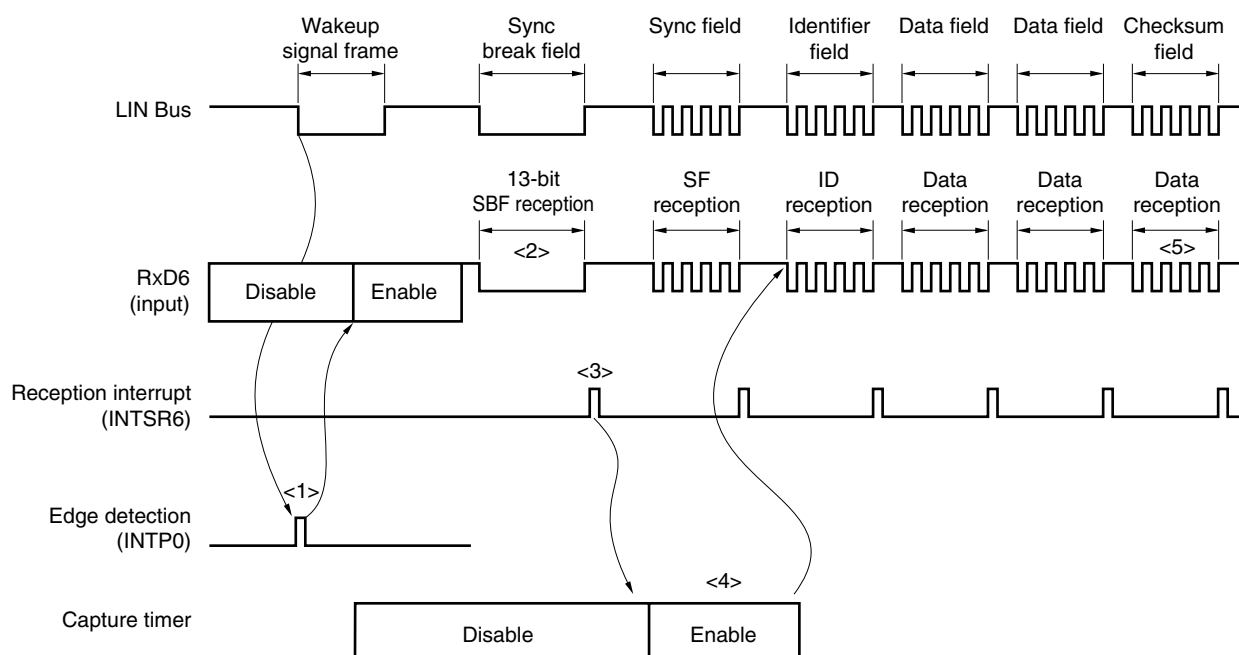
Figure 13-1. LIN Transmission Operation



- Notes**
1. The wakeup signal frame is substituted by 80H transmission in the 8-bit mode.
 2. The sync break field is output by hardware. The output width is the bit length set by bits 4 to 2 (SBL62 to SBL60) of asynchronous serial interface control register 6 (ASICL6) (see **13.4.2 (2) (h) SBF transmission**).
 3. INTST6 is output on completion of each transmission. It is also output when SBF is transmitted.

Remark The interval between each field is controlled by software.

Figure 13-2. LIN Reception Operation



Reception processing is as follows.

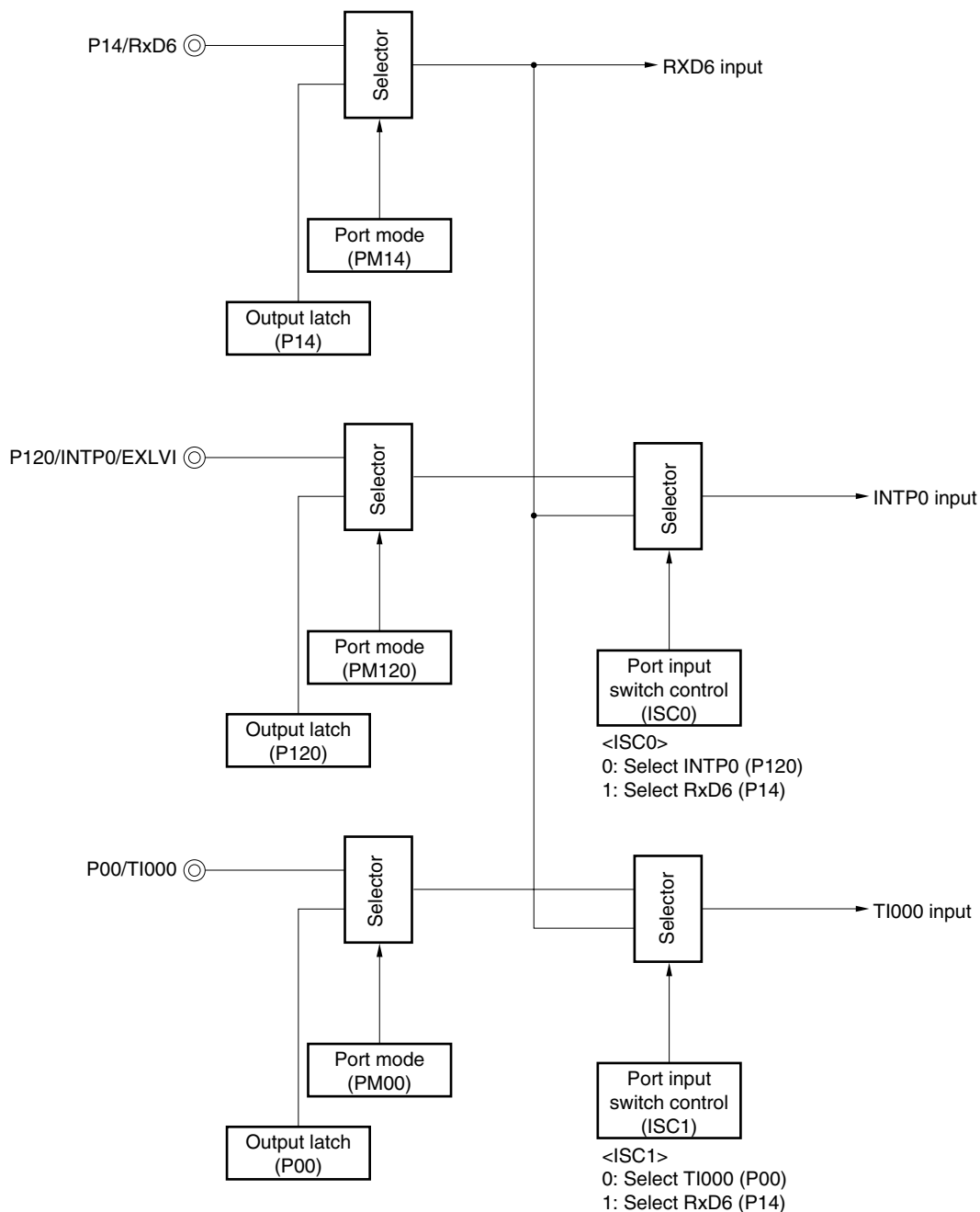
- <1> The wakeup signal is detected at the edge of the pin, and enables UART6 and sets the SBF reception mode.
- <2> Reception continues until the STOP bit is detected. When an SBF with low-level data of 11 bits or more has been detected, it is assumed that SBF reception has been completed correctly, and an interrupt signal is output. If an SBF with low-level data of less than 11 bits has been detected, it is assumed that an SBF reception error has occurred. The interrupt signal is not output and the SBF reception mode is restored.
- <3> If SBF reception has been completed correctly, an interrupt signal is output. Start 16-bit timer/event counter 00 by the SBF reception end interrupt servicing and measure the bit interval (pulse width) of the sync field (see **6.4.8 Pulse width measurement operation**). Detection of errors OVE6, PE6, and FE6 is suppressed, and error detection processing of UART communication and data transfer of the shift register and RXB6 is not performed. The shift register holds the reset value FFH.
- <4> Calculate the baud rate error from the bit interval of the sync field, disable UART6 after SF reception, and then re-set baud rate generator control register 6 (BRGC6).
- <5> Distinguish the checksum field by software. Also perform processing by software to initialize UART6 after reception of the checksum field and to set the SBF reception mode again.

Figure 13-3 shows the port configuration for LIN reception operation.

The wakeup signal transmitted from the LIN master is received by detecting the edge of the external interrupt (INTP0). The length of the sync field transmitted from the LIN master can be measured using the external event capture operation of 16-bit timer/event counter 00, and the baud rate error can be calculated.

The input source of the reception port input (RxD6) can be input to the external interrupt (INTP0) and 16-bit timer/event counter 00 by port input switch control (ISC0/ISC1), without connecting RxD6 and INTP0/TI000 externally.

Figure 13-3. Port Configuration for LIN Reception Operation



Remark ISC0, ISC1: Bits 0 and 1 of the input switch control register (ISC) (see **Figure 13-11**)

The peripheral functions used in the LIN communication operation are shown below.

<Peripheral functions used>

- External interrupt (INTP0); wakeup signal detection
Use: Detects the wakeup signal edges and detects start of communication.
- 16-bit timer/event counter 00 (TI000); baud rate error detection
Use: Detects the baud rate error (measures the TI000 input edge interval in the capture mode) by detecting the sync field (SF) length and divides it by the number of bits.
- Serial interface UART6

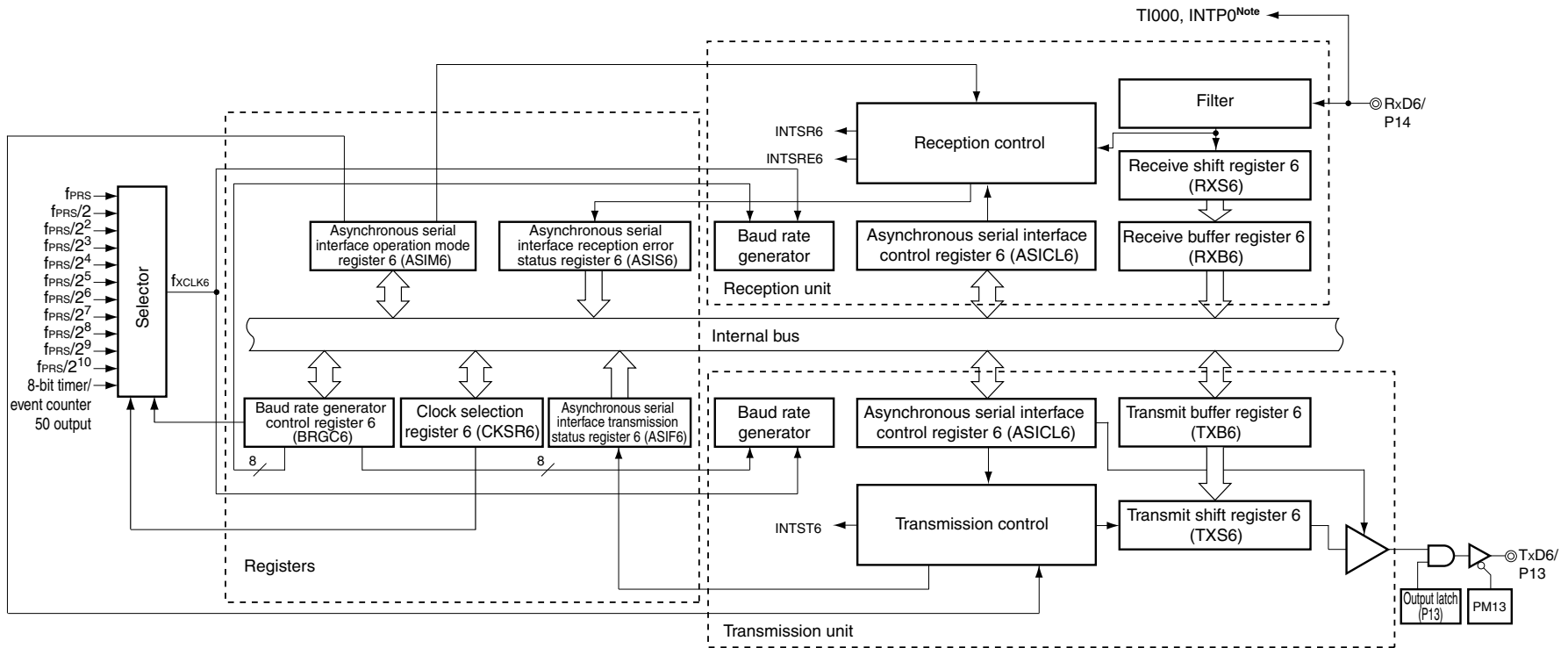
13.2 Configuration of Serial Interface UART6

Serial interface UART6 includes the following hardware.

Table 13-1. Configuration of Serial Interface UART6

| Item | Configuration |
|-------------------|--|
| Registers | Receive buffer register 6 (RXB6) Receive shift register 6 (RXS6) Transmit buffer register 6 (TXB6) Transmit shift register 6 (TXS6) |
| Control registers | Asynchronous serial interface operation mode register 6 (ASIM6) Asynchronous serial interface reception error status register 6 (ASIS6) Asynchronous serial interface transmission status register 6 (ASIF6) Clock selection register 6 (CKSR6) Baud rate generator control register 6 (BRGC6) Asynchronous serial interface control register 6 (ASICL6) Input switch control register (ISC) Port mode register 1 (PM1) Port register 1 (P1) |

Figure 13-4. Block Diagram of Serial Interface UART6



Note Selectable with input switch control register (ISC).

(1) Receive buffer register 6 (RXB6)

This 8-bit register stores parallel data converted by receive shift register 6 (RXS6).

Each time 1 byte of data has been received, new receive data is transferred to this register from RXS6. If the data length is set to 7 bits, data is transferred as follows.

- In LSB-first reception, the receive data is transferred to bits 0 to 6 of RXB6 and the MSB of RXB6 is always 0.
- In MSB-first reception, the receive data is transferred to bits 1 to 7 of RXB6 and the LSB of RXB6 is always 0.

If an overrun error (OVE6) occurs, the receive data is not transferred to RXB6.

RXB6 can be read by an 8-bit memory manipulation instruction. No data can be written to this register.

Reset signal generation sets this register to FFH.

(2) Receive shift register 6 (RXS6)

This register converts the serial data input to the RxD6 pin into parallel data.

RXS6 cannot be directly manipulated by a program.

(3) Transmit buffer register 6 (TXB6)

This buffer register is used to set transmit data. Transmission is started when data is written to TXB6.

This register can be read or written by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

- Cautions**
1. Do not write data to TXB6 when bit 1 (TXBF6) of asynchronous serial interface transmission status register 6 (ASIF6) is 1.
 2. Do not refresh (write the same value to) TXB6 by software during a communication operation (when bits 7 and 6 (POWER6, TXE6) of asynchronous serial interface operation mode register 6 (ASIM6) are 1 or when bits 7 and 5 (POWER6, RXE6) of ASIM6 are 1).
 3. Set transmit data to TXB6 at least one base clock (f_{CLK6}) after setting TXE6 = 1.

(4) Transmit shift register 6 (TXS6)

This register transmits the data transferred from TXB6 from the TxD6 pin as serial data. Data is transferred from TXB6 immediately after TXB6 is written for the first transmission, or immediately before INTST6 occurs after one frame was transmitted for continuous transmission. Data is transferred from TXB6 and transmitted from the TxD6 pin at the falling edge of the base clock.

TXS6 cannot be directly manipulated by a program.

13.3 Registers Controlling Serial Interface UART6

Serial interface UART6 is controlled by the following nine registers.

- Asynchronous serial interface operation mode register 6 (ASIM6)
- Asynchronous serial interface reception error status register 6 (ASIS6)
- Asynchronous serial interface transmission status register 6 (ASIF6)
- Clock selection register 6 (CKSR6)
- Baud rate generator control register 6 (BRGC6)
- Asynchronous serial interface control register 6 (ASICL6)
- Input switch control register (ISC)
- Port mode register 1 (PM1)
- Port register 1 (P1)

(1) Asynchronous serial interface operation mode register 6 (ASIM6)

This 8-bit register controls the serial communication operations of serial interface UART6.

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

Reset signal generation sets this register to 01H.

Remark ASIM6 can be refreshed (the same value is written) by software during a communication operation (when bits 7 and 6 (POWER6, TXE6) of ASIM6 = 1 or bits 7 and 5 (POWER6, RXE6) of ASIM6 = 1).

Figure 13-5. Format of Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) (1/2)

Address: FF50H After reset: 01H R/W

| | | | | | | | | |
|--------|--------|------|------|------|------|-----|-----|-------|
| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
| ASIM6 | POWER6 | TXE6 | RXE6 | PS61 | PS60 | CL6 | SL6 | ISRM6 |

| | |
|---------------------|--|
| POWER6 | Enables/disables operation of internal operation clock |
| 0 ^{Note 1} | Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit ^{Note 2} . |
| 1 | Enables operation of the internal operation clock |

| | |
|------|--|
| TXE6 | Enables/disables transmission |
| 0 | Disables transmission (synchronously resets the transmission circuit). |
| 1 | Enables transmission |

| | |
|------|--|
| RXE6 | Enables/disables reception |
| 0 | Disables reception (synchronously resets the reception circuit). |
| 1 | Enables reception |

- Notes**
1. If POWER6 = 0 is set while transmitting data, the output of the TxD6 pin will be fixed to high level (if TXDLV6 = 0). Furthermore, the input from the RxD6 pin will be fixed to high level.
 2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), bit 7 (SBRF6) and bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6), and receive buffer register 6 (RXB6) are reset.

Figure 13-5. Format of Asynchronous Serial Interface Operation Mode Register 6 (ASIM6) (2/2)

| PS61 | PS60 | Transmission operation | Reception operation |
|------|------|-----------------------------|---------------------------------------|
| 0 | 0 | Does not output parity bit. | Reception without parity |
| 0 | 1 | Outputs 0 parity. | Reception as 0 parity ^{Note} |
| 1 | 0 | Outputs odd parity. | Judges as odd parity. |
| 1 | 1 | Outputs even parity. | Judges as even parity. |

| CL6 | Specifies character length of transmit/receive data |
|-----|---|
| 0 | Character length of data = 7 bits |
| 1 | Character length of data = 8 bits |

| SL6 | Specifies number of stop bits of transmit data |
|-----|--|
| 0 | Number of stop bits = 1 |
| 1 | Number of stop bits = 2 |

| ISRM6 | Enables/disables occurrence of reception completion interrupt in case of error |
|-------|--|
| 0 | “INTSRE6” occurs in case of error (at this time, INTSR6 does not occur). |
| 1 | “INTSR6” occurs in case of error (at this time, INTSRE6 does not occur). |

Note If “reception as 0 parity” is selected, the parity is not judged. Therefore, bit 2 (PE6) of asynchronous serial interface reception error status register 6 (ASIS6) is not set and the error interrupt does not occur.

- Cautions**
1. To start the transmission, set POWER6 to 1 and then set TXE6 to 1. To stop the transmission, clear TXE6 to 0, and then clear POWER6 to 0.
 2. To start the reception, set POWER6 to 1 and then set RXE6 to 1. To stop the reception, clear RXE6 to 0, and then clear POWER6 to 0.
 3. Set POWER6 to 1 and then set RXE6 to 1 while a high level is input to the RxD6 pin. If POWER6 is set to 1 and RXE6 is set to 1 while a low level is input, reception is started.
 4. TXE6 and RXE6 are synchronized by the base clock (f_{XCLK6}) set by CKSR6. To enable transmission or reception again, set TXE6 or RXE6 to 1 at least two clocks of the base clock after TXE6 or RXE6 has been cleared to 0. If TXE6 or RXE6 is set within two clocks of the base clock, the transmission circuit or reception circuit may not be initialized.
 5. Set transmit data to TXB6 at least one base clock (f_{XCLK6}) after setting TXE6 = 1.
 6. Clear the TXE6 and RXE6 bits to 0 before rewriting the PS61, PS60, and CL6 bits.
 7. Fix the PS61 and PS60 bits to 0 when used in LIN communication operation.
 8. Clear TXE6 to 0 before rewriting the SL6 bit. Reception is always performed with “the number of stop bits = 1”, and therefore, is not affected by the set value of the SL6 bit.
 9. Make sure that RXE6 = 0 when rewriting the ISRM6 bit.

(2) Asynchronous serial interface reception error status register 6 (ASIS6)

This register indicates an error status on completion of reception by serial interface UART6. It includes three error flag bits (PE6, FE6, OVE6).

This register is read-only by an 8-bit memory manipulation instruction.

Reset signal generation, or clearing bit 7 (POWER6) or bit 5 (RXE6) of ASIM6 to 0 clears this register to 00H. 00H is read when this register is read. If a reception error occurs, read ASIS6 and then read receive buffer register 6 (RXB6) to clear the error flag.

Figure 13-6. Format of Asynchronous Serial Interface Reception Error Status Register 6 (ASIS6)

Address: FF53H After reset: 00H R

| | | | | | | | | |
|--------|---|---|---|---|---|-----|-----|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ASIS6 | 0 | 0 | 0 | 0 | 0 | PE6 | FE6 | OVE6 |

| | |
|-----|---|
| PE6 | Status flag indicating parity error |
| 0 | If POWER6 = 0 or RXE6 = 0, or if ASIS6 register is read |
| 1 | If the parity of transmit data does not match the parity bit on completion of reception |

| | |
|-----|--|
| FE6 | Status flag indicating framing error |
| 0 | If POWER6 = 0 or RXE6 = 0, or if ASIS6 register is read |
| 1 | If the stop bit is not detected on completion of reception |

| | |
|------|--|
| OVE6 | Status flag indicating overrun error |
| 0 | If POWER6 = 0 or RXE6 = 0, or if ASIS6 register is read |
| 1 | If receive data is set to the RXB6 register and the next reception operation is completed before the data is read. |

- Cautions**
1. The operation of the PE6 bit differs depending on the set values of the PS61 and PS60 bits of asynchronous serial interface operation mode register 6 (ASIM6).
 2. For the stop bit of the receive data, only the first stop bit is checked regardless of the number of stop bits.
 3. If an overrun error occurs, the next receive data is not written to receive buffer register 6 (RXB6) but discarded.
 4. If data is read from ASIS6, a wait cycle is generated. Do not read data from ASIS6 when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

(3) Asynchronous serial interface transmission status register 6 (ASIF6)

This register indicates the status of transmission by serial interface UART6. It includes two status flag bits (TXBF6 and TXSF6).

Transmission can be continued without disruption even during an interrupt period, by writing the next data to the TXB6 register after data has been transferred from the TXB6 register to the TXS6 register.

This register is read-only by an 8-bit memory manipulation instruction.

Reset signal generation, or clearing bit 7 (POWER6) or bit 6 (TXE6) of ASIM6 to 0 clears this register to 00H.

Figure 13-7. Format of Asynchronous Serial Interface Transmission Status Register 6 (ASIF6)

Address: FF55H After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|---|---|---|-------|-------|
| ASIF6 | 0 | 0 | 0 | 0 | 0 | 0 | TXBF6 | TXSF6 |

| TXBF6 | Transmit buffer data flag |
|-------|--|
| 0 | If POWER6 = 0 or TXE6 = 0, or if data is transferred to transmit shift register 6 (TXS6) |
| 1 | If data is written to transmit buffer register 6 (TXB6) (if data exists in TXB6) |

| TXSF6 | Transmit shift register data flag |
|-------|---|
| 0 | If POWER6 = 0 or TXE6 = 0, or if the next data is not transferred from transmit buffer register 6 (TXB6) after completion of transfer |
| 1 | If data is transferred from transmit buffer register 6 (TXB6) (if data transmission is in progress) |

- Cautions**
1. To transmit data continuously, write the first transmit data (first byte) to the TXB6 register. Be sure to check that the TXBF6 flag is "0". If so, write the next transmit data (second byte) to the TXB6 register. If data is written to the TXB6 register while the TXBF6 flag is "1", the transmit data cannot be guaranteed.
 2. To initialize the transmission unit upon completion of continuous transmission, be sure to check that the TXSF6 flag is "0" after generation of the transmission completion interrupt, and then execute initialization. If initialization is executed while the TXSF6 flag is "1", the transmit data cannot be guaranteed.

(4) Clock selection register 6 (CKSR6)

This register selects the base clock of serial interface UART6.

CKSR6 can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Remark CKSR6 can be refreshed (the same value is written) by software during a communication operation (when bits 7 and 6 (POWER6, TXE6) of ASIM6 = 1 or bits 7 and 5 (POWER6, RXE6) of ASIM6 = 1).

Figure 13-8. Format of Clock Selection Register 6 (CKSR6)

Address: FF56H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|-------|-------|-------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CKSR6 | 0 | 0 | 0 | 0 | TPS63 | TPS62 | TPS61 | TPS60 |

| TPS63 | TPS62 | TPS61 | TPS60 | Base clock (f_{CLK6}) selection | | | |
|------------------|-------|-------|-------|-------------------------------------|---------------------------|----------------------------|------------|
| | | | | $f_{PRS} = 2 \text{ MHz}$ | $f_{PRS} = 5 \text{ MHz}$ | $f_{PRS} = 10 \text{ MHz}$ | |
| 0 | 0 | 0 | 0 | f_{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 0 | 0 | 1 | $f_{PRS}/2$ | 1 MHz | 2.5 MHz | 5 MHz |
| 0 | 0 | 1 | 0 | $f_{PRS}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz |
| 0 | 0 | 1 | 1 | $f_{PRS}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz |
| 0 | 1 | 0 | 0 | $f_{PRS}/2^4$ | 125 kHz | 312.5 kHz | 625 kHz |
| 0 | 1 | 0 | 1 | $f_{PRS}/2^5$ | 62.5 kHz | 156.25 kHz | 312.5 kHz |
| 0 | 1 | 1 | 0 | $f_{PRS}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 0 | 1 | 1 | 1 | $f_{PRS}/2^7$ | 15.625 kHz | 39.06 kHz | 78.13 kHz |
| 1 | 0 | 0 | 0 | $f_{PRS}/2^8$ | 7.813 kHz | 19.53 kHz | 39.06 kHz |
| 1 | 0 | 0 | 1 | $f_{PRS}/2^9$ | 3.906 kHz | 9.77 kHz | 19.53 kHz |
| 1 | 0 | 1 | 0 | $f_{PRS}/2^{10}$ | 1.953 kHz | 4.88 kHz | 9.77 kHz |
| 1 | 0 | 1 | 1 | TM50 output ^{Note} | | | |
| Other than above | | | | Setting prohibited | | | |

Note Note the following points when selecting the TM50 output as the base clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.
It is not necessary to enable (TOE50 = 1) TO50 output in any mode.

Caution Make sure POWER6 = 0 when rewriting TPS63 to TPS60.

- Remarks**
1. f_{PRS} : Peripheral hardware clock frequency
 2. TMC506: Bit 6 of 8-bit timer mode control register 50 (TMC50)
TMC501: Bit 1 of TMC50

(5) Baud rate generator control register 6 (BRGC6)

This register sets the division value of the 8-bit counter of serial interface UART6.

BRGC6 can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Remark BRGC6 can be refreshed (the same value is written) by software during a communication operation (when bits 7 and 6 (POWER6, TXE6) of ASIM6 = 1 or bits 7 and 5 (POWER6, RXE6) of ASIM6 = 1).

Figure 13-9. Format of Baud Rate Generator Control Register 6 (BRGC6)

Address: FF57H After reset: FFH R/W

| | | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|-------|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BRGC6 | MDL67 | MDL66 | MDL65 | MDL64 | MDL63 | MDL62 | MDL61 | MDL60 |

| MDL67 | MDL66 | MDL65 | MDL64 | MDL63 | MDL62 | MDL61 | MDL60 | k | Output clock selection of 8-bit counter |
|-------|-------|-------|-------|-------|-------|-------|-------|-----|---|
| 0 | 0 | 0 | 0 | 0 | 0 | × | × | × | Setting prohibited |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 4 | $f_{XCLK6}/4$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 5 | $f_{XCLK6}/5$ |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 6 | $f_{XCLK6}/6$ |
| • | • | • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • | • | • |
| • | • | • | • | • | • | • | • | • | • |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 252 | $f_{XCLK6}/252$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 253 | $f_{XCLK6}/253$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 254 | $f_{XCLK6}/254$ |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 255 | $f_{XCLK6}/255$ |

- Cautions**
1. Make sure that bit 6 (TXE6) and bit 5 (RXE6) of the ASIM6 register = 0 when rewriting the MDL67 to MDL60 bits.
 2. The baud rate is the output clock of the 8-bit counter divided by 2.

- Remarks**
1. f_{XCLK6} : Frequency of base clock selected by the TPS63 to TPS60 bits of CKSR6 register
 2. k: Value set by MDL67 to MDL60 bits (k = 4, 5, 6, ..., 255)
 3. ×: Don't care

(6) Asynchronous serial interface control register 6 (ASICL6)

This register controls the serial communication operations of serial interface UART6.

ASICL6 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 16H.

Caution ASICL6 can be refreshed (the same value is written) by software during a communication operation (when bits 7 and 6 (POWER6, TXE6) of ASIM6 = 1 or bits 7 and 5 (POWER6, RXE6) of ASIM6 = 1). However, do not set both SBRT6 and SBTT6 to 1 by a refresh operation during SBF reception (SBRT6 = 1) or SBF transmission (until INTST6 occurs since SBTT6 has been set (1)), because it may re-trigger SBF reception or SBF transmission.

Figure 13-10. Format of Asynchronous Serial Interface Control Register 6 (ASICL6) (1/2)

Address: FF58H After reset: 16H R/W^{Note}

| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|-------|-------|-------|-------|-------|-------|------|--------|
| ASICL6 | SBRF6 | SBRT6 | SBTT6 | SBL62 | SBL61 | SBL60 | DIR6 | TXDLV6 |

| SBRF6 | SBF reception status flag |
|-------|---|
| 0 | If POWER6 = 0 and RXE6 = 0 or if SBF reception has been completed correctly |
| 1 | SBF reception in progress |

| SBRT6 | SBF reception trigger |
|-------|-----------------------|
| 0 | – |
| 1 | SBF reception trigger |

| SBTT6 | SBF transmission trigger |
|-------|--------------------------|
| 0 | – |
| 1 | SBF transmission trigger |

Note Bit 7 is read-only.

Figure 13-10. Format of Asynchronous Serial Interface Control Register 6 (ASICL6) (2/2)

| SBL62 | SBL61 | SBL60 | SBF transmission output width control |
|-------|-------|-------|---------------------------------------|
| 1 | 0 | 1 | SBF is output with 13-bit length. |
| 1 | 1 | 0 | SBF is output with 14-bit length. |
| 1 | 1 | 1 | SBF is output with 15-bit length. |
| 0 | 0 | 0 | SBF is output with 16-bit length. |
| 0 | 0 | 1 | SBF is output with 17-bit length. |
| 0 | 1 | 0 | SBF is output with 18-bit length. |
| 0 | 1 | 1 | SBF is output with 19-bit length. |
| 1 | 0 | 0 | SBF is output with 20-bit length. |

| DIR6 | First-bit specification |
|------|-------------------------|
| 0 | MSB |
| 1 | LSB |

| TXDLV6 | Enables/disables inverting TxD6 output |
|--------|--|
| 0 | Normal output of TxD6 |
| 1 | Inverted output of TxD6 |

- Cautions**
1. In the case of an SBF reception error, the mode returns to the SBF reception mode. The status of the SBRF6 flag is held (1).
 2. Before setting the SBRT6 bit, make sure that bit 7 (POWER6) and bit 5 (RXE6) of ASIM6 = 1. After setting the SBRT6 bit to 1, do not clear it to 0 before SBF reception is completed (before an interrupt request signal is generated).
 3. The read value of the SBRT6 bit is always 0. SBRT6 is automatically cleared to 0 after SBF reception has been correctly completed.
 4. Before setting the SBTT6 bit to 1, make sure that bit 7 (POWER6) and bit 6 (TXE6) of ASIM6 = 1. After setting the SBTT6 bit to 1, do not clear it to 0 before SBF transmission is completed (before an interrupt request signal is generated).
 5. The read value of the SBTT6 bit is always 0. SBTT6 is automatically cleared to 0 at the end of SBF transmission.
 6. Do not set the SBRT6 bit to 1 during reception, and do not set the SBTT6 bit to 1 during transmission.
 7. Before rewriting the DIR6 and TXDLV6 bits, clear the TXE6 and RXE6 bits to 0.
 8. When the TXDLV6 bit is set to 1 (inverted TxD6 output), the TxD6/SCLA0/P60 pin cannot be used as a general-purpose port, regardless of the settings of POWER6 and TXE6. When using the TxD6/SCLA0/P60 pin as a general-purpose port, clear the TXDLV6 bit to 0 (normal TxD6 output).

(7) Input switch control register (ISC)

The input switch control register (ISC) is used to receive a status signal transmitted from the master during LIN (Local Interconnect Network) reception.

The signal input from the P14/RxD6 pin is selected as the input source of INTPO and TI000 when ISC0 and ISC1 are set to 1.

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

Reset signal generation sets this register to 00H.

Figure 13-11. Format of Input Switch Control Register (ISC)

Address: FF4FH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|---|---|---|------|------|
| ISC | 0 | 0 | 0 | 0 | 0 | 0 | ISC1 | ISC0 |

| ISC1 | TI000 input source selection |
|------|------------------------------|
| 0 | TI000 (P00) |
| 1 | RxD6 (P14) |

| ISC0 | INTPO input source selection |
|------|------------------------------|
| 0 | INTPO (P120) |
| 1 | RxD6 (P14) |

(8) Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.

When using the P13/TxD6 pin for serial interface data output, clear PM13 to 0 and set the output latch of P13 to 1.

When using the P14/RxD6 pin for serial interface data input, set PM14 to 1. The output latch of P14 at this time may be 0 or 1.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to FFH.

Figure 13-12. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|------|------|------|------|------|------|------|------|
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |

| PM1n | P1n pin I/O mode selection (n = 0 to 7) |
|------|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

13.4 Operation of Serial Interface UART6

Serial interface UART6 has the following two modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode

13.4.1 Operation stop mode

In this mode, serial communication cannot be executed; therefore, the power consumption can be reduced. In addition, the pins can be used as ordinary port pins in this mode. To set the operation stop mode, clear bits 7, 6, and 5 (POWER6, TXE6, and RXE6) of ASIM6 to 0.

(1) Register used

The operation stop mode is set by asynchronous serial interface operation mode register 6 (ASIM6).

ASIM6 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 01H.

Address: FF50H After reset: 01H R/W

| Symbol | <7> | <6> | <5> | 4 | 3 | 2 | 1 | 0 |
|--------|--------|------|------|------|------|-----|-----|-------|
| ASIM6 | POWER6 | TXE6 | RXE6 | PS61 | PS60 | CL6 | SL6 | ISRM6 |

| | |
|---------------------|--|
| POWER6 | Enables/disables operation of internal operation clock |
| 0 ^{Note 1} | Disables operation of the internal operation clock (fixes the clock to low level) and asynchronously resets the internal circuit ^{Note 2} . |

| | |
|------|--|
| TXE6 | Enables/disables transmission |
| 0 | Disables transmission operation (synchronously resets the transmission circuit). |

| | |
|------|--|
| RXE6 | Enables/disables reception |
| 0 | Disables reception (synchronously resets the reception circuit). |

- Notes**
1. If POWER6 = 0 is set while transmitting data, the output of the TxD6 pin will be fixed to high level (if TXDLV6 = 0). Furthermore, the input from the RxD6 pin will be fixed to high level.
 2. Asynchronous serial interface reception error status register 6 (ASIS6), asynchronous serial interface transmission status register 6 (ASIF6), bit 7 (SBRF6) and bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6), and receive buffer register 6 (RXB6) are reset.

Caution Clear POWER6 to 0 after clearing TXE6 and RXE6 to 0 to stop the operation.

To start the communication, set POWER6 to 1, and then set TXE6 or RXE6 to 1.

Remark To use the RxD6/P14 and TxD6/P13 pins as general-purpose port pins, see **CHAPTER 4 PORT FUNCTIONS**.

13.4.2 Asynchronous serial interface (UART) mode

In this mode, data of 1 byte is transmitted/received following a start bit, and a full-duplex operation can be performed.

A dedicated UART baud rate generator is incorporated, so that communication can be executed at a wide range of baud rates.

(1) Registers used

- Asynchronous serial interface operation mode register 6 (ASIM6)
- Asynchronous serial interface reception error status register 6 (ASIS6)
- Asynchronous serial interface transmission status register 6 (ASIF6)
- Clock selection register 6 (CKSR6)
- Baud rate generator control register 6 (BRGC6)
- Asynchronous serial interface control register 6 (ASICL6)
- Input switch control register (ISC)
- Port mode register 1 (PM1)
- Port register 1 (P1)

The basic procedure of setting an operation in the UART mode is as follows.

- <1> Set the CKSR6 register (see **Figure 13-8**).
- <2> Set the BRGC6 register (see **Figure 13-9**).
- <3> Set bits 0 to 4 (ISRM6, SL6, CL6, PS60, PS61) of the ASIM6 register (see **Figure 13-5**).
- <4> Set bits 0 and 1 (TXDLV6, DIR6) of the ASICL6 register (see **Figure 13-10**).
- <5> Set bit 7 (POWER6) of the ASIM6 register to 1.
- <6> Set bit 6 (TXE6) of the ASIM6 register to 1. → Transmission is enabled.
Set bit 5 (RXE6) of the ASIM6 register to 1. → Reception is enabled.
- <7> Write data to transmit buffer register 6 (TXB6). → Data transmission is started.

Caution Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 13-2. Relationship Between Register Settings and Pins

| POWER6 | TXE6 | RXE6 | PM13 | P13 | PM14 | P14 | UART6 Operation | Pin Function | |
|--------|------|------|------|-----|------|-----|------------------------|--------------|----------|
| | | | | | | | | TxD6/P13 | RxD6/P14 |
| 0 | 0 | 0 | × | × | × | × | Stop | P13 | P14 |
| 1 | 0 | 1 | × | × | 1 | × | Reception | P13 | RxD6 |
| | 1 | 0 | 0 | 1 | × | × | Transmission | TxD6 | P14 |
| | 1 | 1 | 0 | 1 | 1 | × | Transmission/reception | TxD6 | RxD6 |

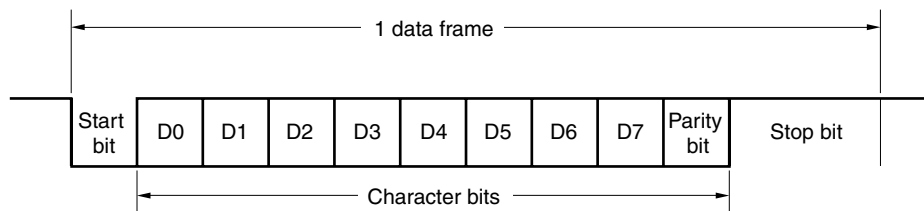
Note Can be set as port function.

Remark ×: don't care
 POWER6: Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)
 TXE6: Bit 6 of ASIM6
 RXE6: Bit 5 of ASIM6
 PM1×: Port mode register
 P1×: Port output latch

(2) Communication operation**(a) Format and waveform example of normal transmit/receive data**

Figures 13-13 and 13-14 show the format and waveform example of the normal transmit/receive data.

Figure 13-13. Format of Normal UART Transmit/Receive Data

1. LSB-first transmission/reception**2. MSB-first transmission/reception**

One data frame consists of the following bits.

- Start bit ... 1 bit
- Character bits ... 7 or 8 bits
- Parity bit ... Even parity, odd parity, 0 parity, or no parity
- Stop bit ... 1 or 2 bits

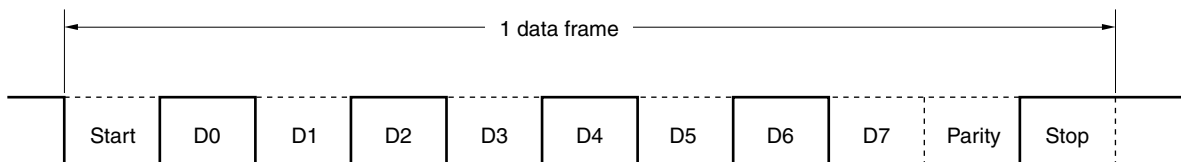
The character bit length, parity, and stop bit length in one data frame are specified by asynchronous serial interface operation mode register 6 (ASIM6).

Whether data is communicated with the LSB or MSB first is specified by bit 1 (DIR6) of asynchronous serial interface control register 6 (ASICL6).

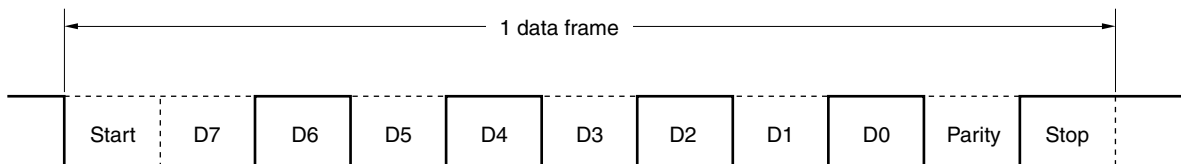
Whether the Tx/D6 pin outputs normal or inverted data is specified by bit 0 (TXDLV6) of ASICL6.

Figure 13-14. Example of Normal UART Transmit/Receive Data Waveform (1/2)

1. Data length: 8 bits, LSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



2. Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H



3. Data length: 8 bits, MSB first, Parity: Even parity, Stop bit: 1 bit, Communication data: 55H, Tx/D6 pin inverted output

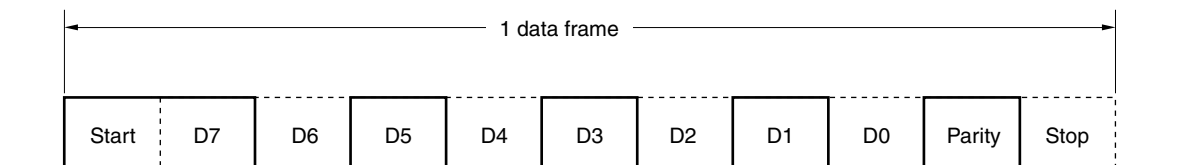
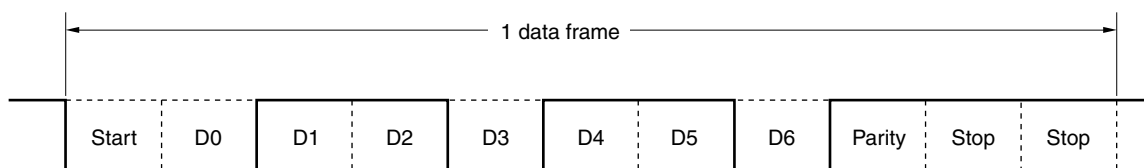
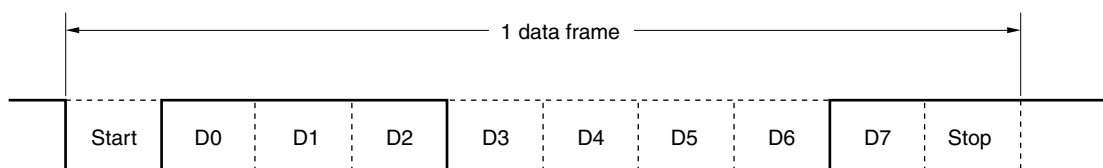


Figure 13-14. Example of Normal UART Transmit/Receive Data Waveform (2/2)

4. Data length: 7 bits, LSB first, Parity: Odd parity, Stop bit: 2 bits, Communication data: 36H



5. Data length: 8 bits, LSB first, Parity: None, Stop bit: 1 bit, Communication data: 87H



(b) Parity types and operation

The parity bit is used to detect a bit error in communication data. Usually, the same type of parity bit is used on both the transmission and reception sides. With even parity and odd parity, a 1-bit (odd number) error can be detected. With zero parity and no parity, an error cannot be detected.

Caution Fix the PS61 and PS60 bits to 0 when the device is used in LIN communication operation.

(i) Even parity

• Transmission

Transmit data, including the parity bit, is controlled so that the number of bits that are "1" is even. The value of the parity bit is as follows.

If transmit data has an odd number of bits that are "1": 1

If transmit data has an even number of bits that are "1": 0

• Reception

The number of bits that are "1" in the receive data, including the parity bit, is counted. If it is odd, a parity error occurs.

(ii) Odd parity

- Transmission

Unlike even parity, transmit data, including the parity bit, is controlled so that the number of bits that are “1” is odd.

If transmit data has an odd number of bits that are “1”: 0

If transmit data has an even number of bits that are “1”: 1

- Reception

The number of bits that are “1” in the receive data, including the parity bit, is counted. If it is even, a parity error occurs.

(iii) 0 parity

The parity bit is cleared to 0 when data is transmitted, regardless of the transmit data.

The parity bit is not detected when the data is received. Therefore, a parity error does not occur regardless of whether the parity bit is “0” or “1”.

(iv) No parity

No parity bit is appended to the transmit data.

Reception is performed assuming that there is no parity bit when data is received. Because there is no parity bit, a parity error does not occur.

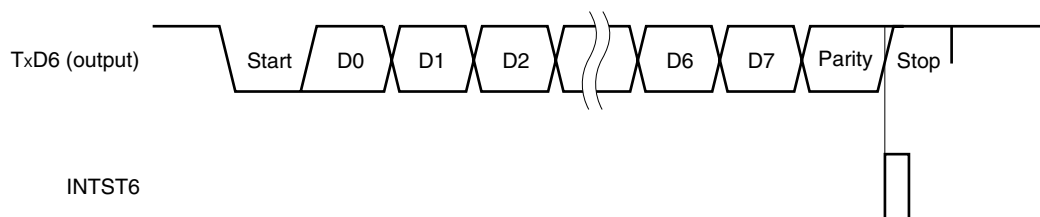
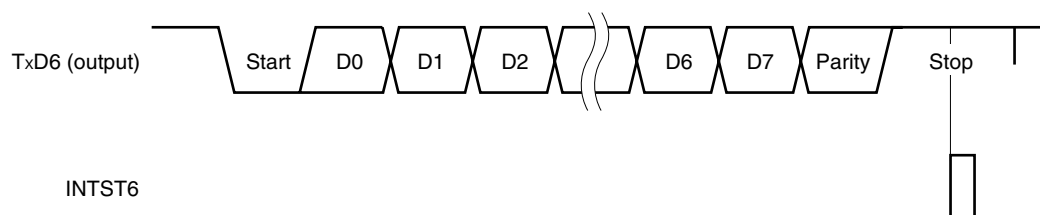
(c) Normal transmission

When bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and bit 6 (TXE6) of ASIM6 is then set to 1, transmission is enabled. Transmission can be started by writing transmit data to transmit buffer register 6 (TXB6). The start bit, parity bit, and stop bit are automatically appended to the data.

When transmission is started, the data in TXB6 is transferred to transmit shift register 6 (TXS6). After that, the transmit data is sequentially output from TXS6 to the TxD6 pin. When transmission is completed, the parity and stop bits set by ASIM6 are appended and a transmission completion interrupt request (INTST6) is generated.

Transmission is stopped until the data to be transmitted next is written to TXB6.

Figure 13-15 shows the timing of the transmission completion interrupt request (INTST6). This interrupt occurs as soon as the last stop bit has been output.

Figure 13-15. Normal Transmission Completion Interrupt Request Timing**1. Stop bit length: 1****2. Stop bit length: 2****(d) Continuous transmission**

The next transmit data can be written to transmit buffer register 6 (TXB6) as soon as transmit shift register 6 (TXS6) has started its shift operation. Consequently, even while the INTST6 interrupt is being serviced after transmission of one data frame, data can be continuously transmitted and an efficient communication rate can be realized. In addition, the TXB6 register can be efficiently written twice (2 bytes) without having to wait for the transmission time of one data frame, by reading bit 0 (TXSF6) of asynchronous serial interface transmission status register 6 (ASIF6) when the transmission completion interrupt has occurred.

To transmit data continuously, be sure to reference the ASIF6 register to check the transmission status and whether the TXB6 register can be written, and then write the data.

Cautions 1. The TXBF6 and TXSF6 flags of the ASIF6 register change from “10” to “11”, and to “01” during continuous transmission. To check the status, therefore, do not use a combination of the TXBF6 and TXSF6 flags for judgment. Read only the TXBF6 flag when executing continuous transmission.

2. When the device is use in LIN communication operation, the continuous transmission function cannot be used. Make sure that asynchronous serial interface transmission status register 6 (ASIF6) is 00H before writing transmit data to transmit buffer register 6 (TXB6).

| TXBF6 | Writing to TXB6 Register |
|-------|--------------------------|
| 0 | Writing enabled |
| 1 | Writing disabled |

Caution To transmit data continuously, write the first transmit data (first byte) to the TXB6 register. Be sure to check that the TXBF6 flag is “0”. If so, write the next transmit data (second byte) to the TXB6 register. If data is written to the TXB6 register while the TXBF6 flag is “1”, the transmit data cannot be guaranteed.

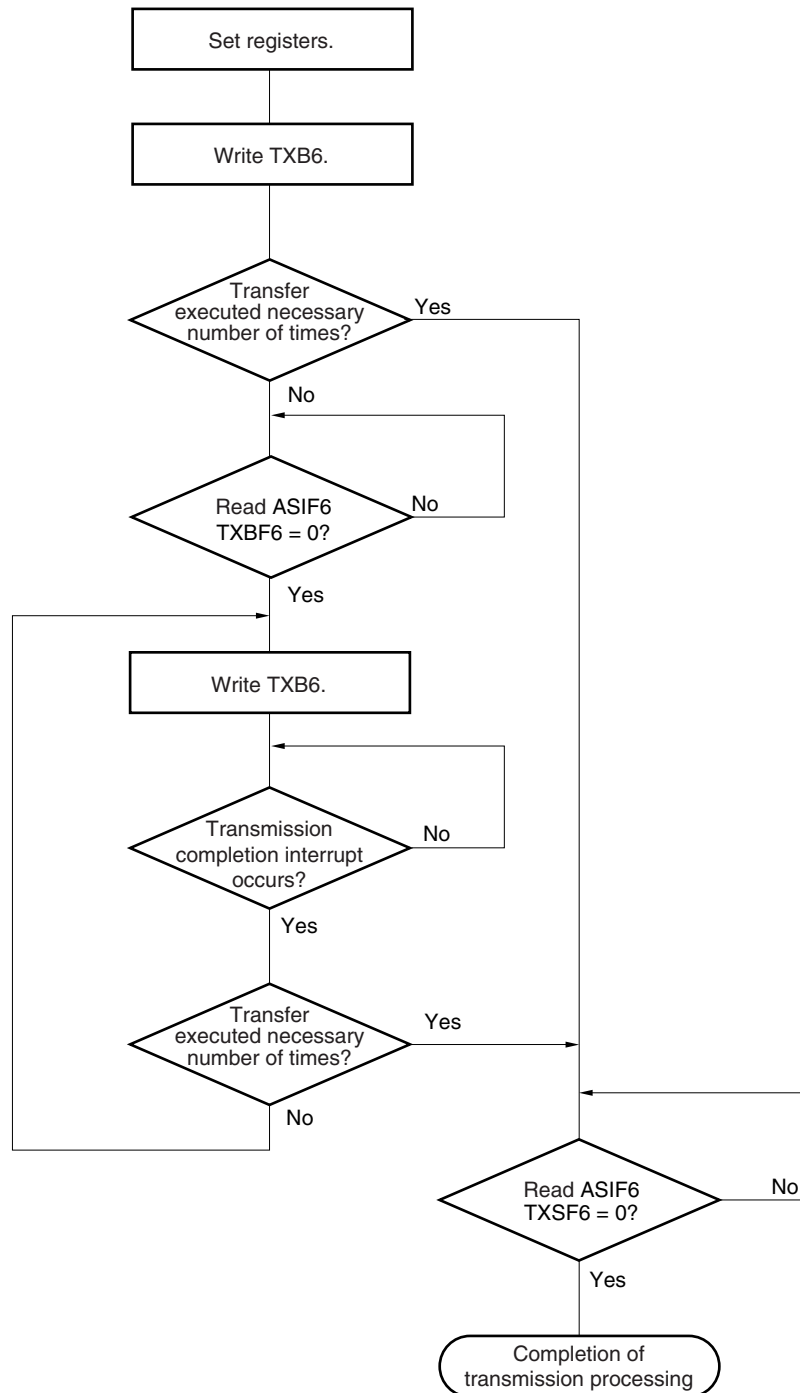
The communication status can be checked using the TXSF6 flag.

| TXSF6 | Transmission Status |
|-------|------------------------------|
| 0 | Transmission is completed. |
| 1 | Transmission is in progress. |

- Cautions**
1. To initialize the transmission unit upon completion of continuous transmission, be sure to check that the TXSF6 flag is “0” after generation of the transmission completion interrupt, and then execute initialization. If initialization is executed while the TXSF6 flag is “1”, the transmit data cannot be guaranteed.
 2. During continuous transmission, the next transmission may complete before execution of INTST6 interrupt servicing after transmission of one data frame. As a countermeasure, detection can be performed by developing a program that can count the number of transmit data and by referencing the TXSF6 flag.

Figure 13-16 shows an example of the continuous transmission processing flow.

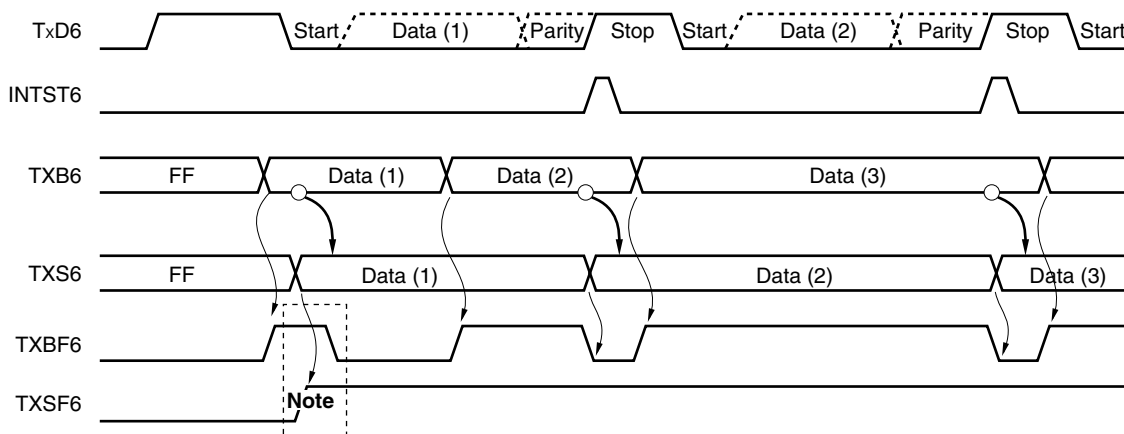
Figure 13-16. Example of Continuous Transmission Processing Flow



Remark TXB6: Transmit buffer register 6
 ASIF6: Asynchronous serial interface transmission status register 6
 TXBF6: Bit 1 of ASIF6 (transmit buffer data flag)
 TXSF6: Bit 0 of ASIF6 (transmit shift register data flag)

Figure 13-17 shows the timing of starting continuous transmission, and Figure 13-18 shows the timing of ending continuous transmission.

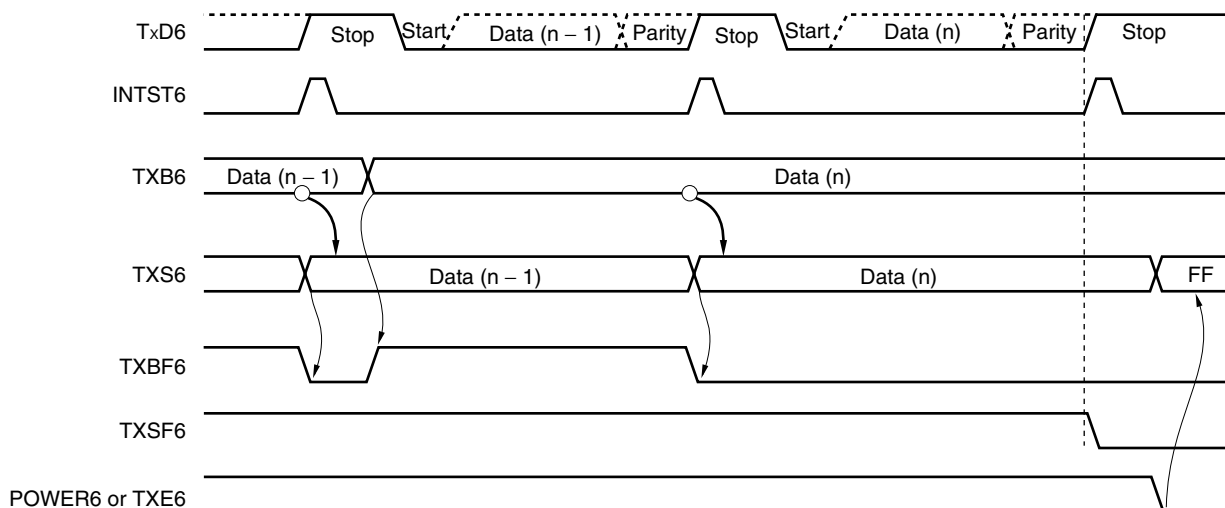
Figure 13-17. Timing of Starting Continuous Transmission



Note When ASIF6 is read, there is a period in which TXBF6 and TXSF6 = 1, 1. Therefore, judge whether writing is enabled using only the TXBF6 bit.

- Remark**
- TxD6: TxD6 pin (output)
 - INTST6: Interrupt request signal
 - TXB6: Transmit buffer register 6
 - TXS6: Transmit shift register 6
 - ASIF6: Asynchronous serial interface transmission status register 6
 - TXBF6: Bit 1 of ASIF6
 - TXSF6: Bit 0 of ASIF6

Figure 13-18. Timing of Ending Continuous Transmission



| | | |
|---------------|---------|--|
| Remark | TxD6: | TxD6 pin (output) |
| | INTST6: | Interrupt request signal |
| | TXB6: | Transmit buffer register 6 |
| | TXS6: | Transmit shift register 6 |
| | ASIF6: | Asynchronous serial interface transmission status register 6 |
| | TXBF6: | Bit 1 of ASIF6 |
| | TXSF6: | Bit 0 of ASIF6 |
| | POWER6: | Bit 7 of asynchronous serial interface operation mode register (ASIM6) |
| | TXE6: | Bit 6 of asynchronous serial interface operation mode register (ASIM6) |

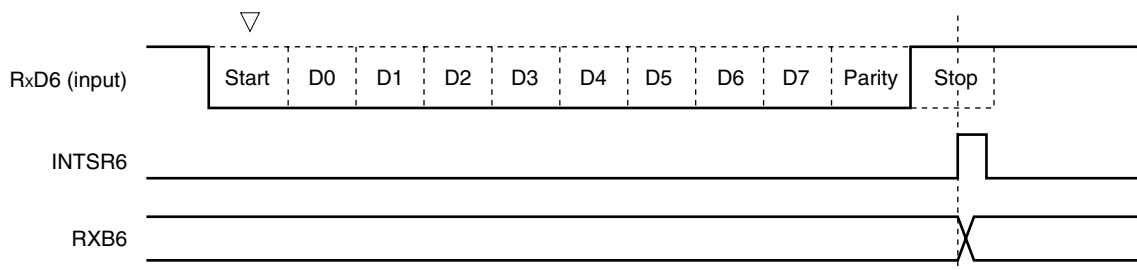
(e) Normal reception

Reception is enabled and the RxD6 pin input is sampled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1.

The 8-bit counter of the baud rate generator starts counting when the falling edge of the RxD6 pin input is detected. When the set value of baud rate generator control register 6 (BRGC6) has been counted, the RxD6 pin input is sampled again (in Figure 13-19). If the RxD6 pin is low level at this time, it is recognized as a start bit. When the start bit is detected, reception is started, and serial data is sequentially stored in the receive shift register (RXS6) at the set baud rate. When the stop bit has been received, the reception completion interrupt (INTSR6) is generated and the data of RXS6 is written to receive buffer register 6 (RXB6). If an overrun error (OVE6) occurs, however, the receive data is not written to RXB6.

Even if a parity error (PE6) occurs while reception is in progress, reception continues to the reception position of the stop bit, and a reception error interrupt (INTSR6/INTSRE6) is generated on completion of reception.

Figure 13-19. Reception Completion Interrupt Request Timing



- Cautions**
1. If a reception error occurs, read ASIS6 and then RXB6 to clear the error flag. Otherwise, an overrun error will occur when the next data is received, and the reception error status will persist.
 2. Reception is always performed with the “number of stop bits = 1”. The second stop bit is ignored.
 3. Be sure to read asynchronous serial interface reception error status register 6 (ASIS6) before reading RXB6.

(f) Reception error

Three types of errors may occur during reception: a parity error, framing error, or overrun error. If the error flag of asynchronous serial interface reception error status register 6 (ASIS6) is set as a result of data reception, a reception error interrupt request (INTSR6/INTSRE6) is generated.

Which error has occurred during reception can be identified by reading the contents of ASIS6 in the reception error interrupt (INTSR6/INTSRE6) servicing (see **Figure 13-6**).

The contents of ASIS6 are cleared to 0 when ASIS6 is read.

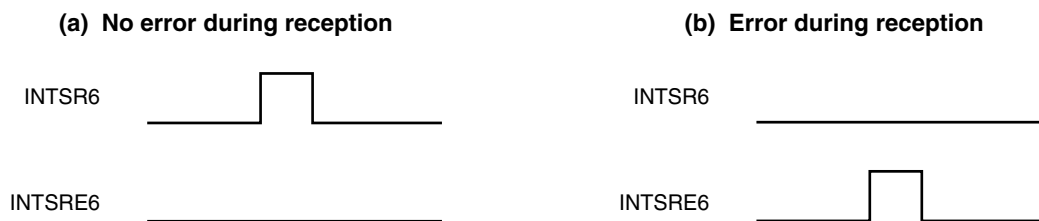
Table 13-3. Cause of Reception Error

| Reception Error | Cause |
|-----------------|--|
| Parity error | The parity specified for transmission does not match the parity of the receive data. |
| Framing error | Stop bit is not detected. |
| Overrun error | Reception of the next data is completed before data is read from receive buffer register 6 (RXB6). |

The reception error interrupt can be separated into reception completion interrupt (INTSR6) and error interrupt (INTSRE6) by clearing bit 0 (ISRM6) of asynchronous serial interface operation mode register 6 (ASIM6) to 0.

Figure 13-20. Reception Error Interrupt

1. If ISRM6 is cleared to 0 (reception completion interrupt (INTSR6) and error interrupt (INTSRE6) are separated)



2. If ISRM6 is set to 1 (error interrupt is included in INTSR6)



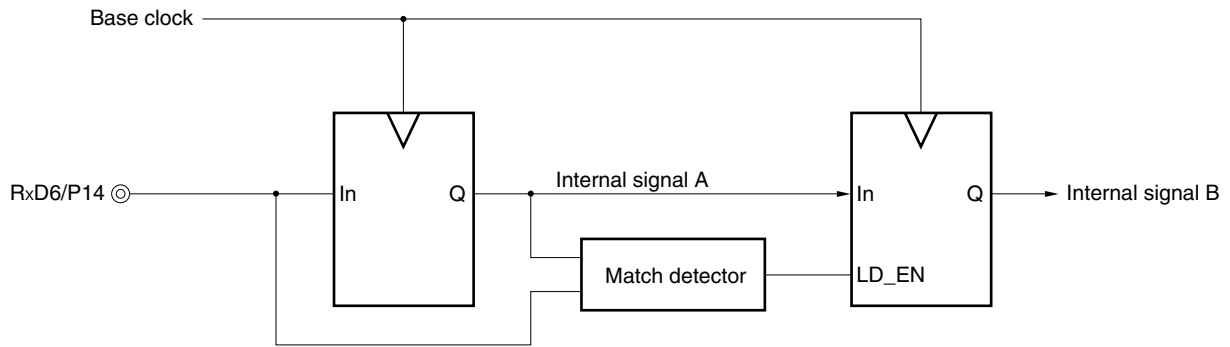
(g) Noise filter of receive data

The RxD6 signal is sampled with the base clock output by the prescaler block.

If two sampled values are the same, the output of the match detector changes, and the data is sampled as input data.

Because the circuit is configured as shown in Figure 13-21, the internal processing of the reception operation is delayed by two clocks from the external signal status.

Figure 13-21. Noise Filter Circuit

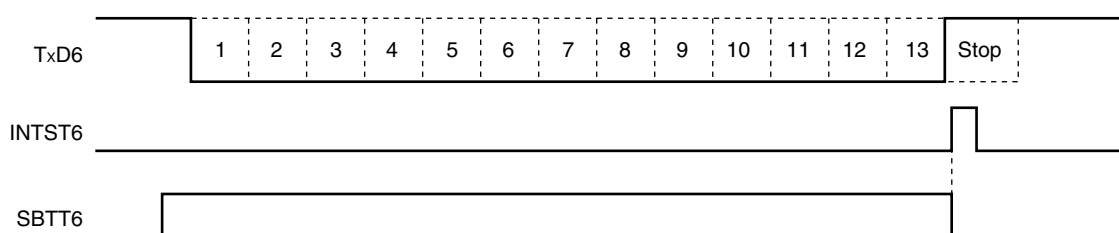


(h) SBF transmission

When the device is use in LIN communication operation, the SBF (Synchronous Break Field) transmission control function is used for transmission. For the transmission operation of LIN, see **Figure 13-1 LIN Transmission Operation**.

When bit 7 (POWER6) of asynchronous serial interface mode register 6 (ASIM6) is set to 1, the TxD6 pin outputs high level. Next, when bit 6 (TXE6) of ASIM6 is set to 1, the transmission enabled status is entered, and SBF transmission is started by setting bit 5 (SBTT6) of asynchronous serial interface control register 6 (ASICL6) to 1. Thereafter, a low level of bits 13 to 20 (set by bits 4 to 2 (SBL62 to SBL60) of ASICL6) is output. Following the end of SBF transmission, the transmission completion interrupt request (INTST6) is generated and SBTT6 is automatically cleared. Thereafter, the normal transmission mode is restored.

Transmission is suspended until the data to be transmitted next is written to transmit buffer register 6 (TXB6), or until SBTT6 is set to 1.

Figure 13-22. SBF Transmission

Remark Tx D6: TxD6 pin (output)

INTST6: Transmission completion interrupt request

SBTT6: Bit 5 of asynchronous serial interface control register 6 (ASICL6)

(i) SBF reception

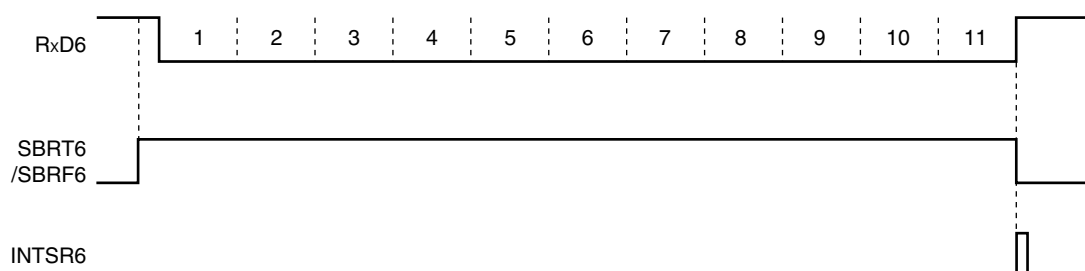
When the device is used in LIN communication operation, the SBF (Synchronous Break Field) reception control function is used for reception. For the reception operation of LIN, see **Figure 13-2 LIN Reception Operation**.

Reception is enabled when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is set to 1 and then bit 5 (RXE6) of ASIM6 is set to 1. SBF reception is enabled when bit 6 (SBRT6) of asynchronous serial interface control register 6 (ASICL6) is set to 1. In the SBF reception enabled status, the RxD6 pin is sampled and the start bit is detected in the same manner as the normal reception enable status.

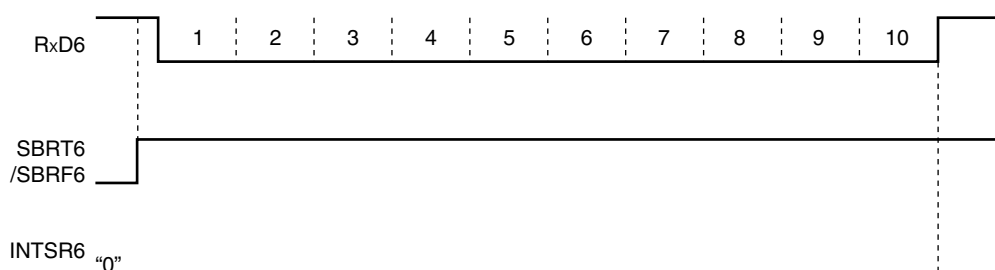
When the start bit has been detected, reception is started, and serial data is sequentially stored in the receive shift register 6 (RXS6) at the set baud rate. When the stop bit is received and if the width of SBF is 11 bits or more, a reception completion interrupt request (INTSR6) is generated as normal processing. At this time, the SBRF6 and SBRT6 bits are automatically cleared, and SBF reception ends. Detection of errors, such as OVE6, PE6, and FE6 (bits 0 to 2 of asynchronous serial interface reception error status register 6 (ASIS6)) is suppressed, and error detection processing of UART communication is not performed. In addition, data transfer between receive shift register 6 (RXS6) and receive buffer register 6 (RXB6) is not performed, and the reset value of FFH is retained. If the width of SBF is 10 bits or less, an interrupt does not occur as error processing after the stop bit has been received, and the SBF reception mode is restored. In this case, the SBRF6 and SBRT6 bits are not cleared.

Figure 13-23. SBF Reception

1. Normal SBF reception (stop bit is detected with a width of more than 10.5 bits)



2. SBF reception error (stop bit is detected with a width of 10.5 bits or less)



Remark RxD6: RxD6 pin (input)
 SBRT6: Bit 6 of asynchronous serial interface control register 6 (ASICL6)
 SBRF6: Bit 7 of ASICL6
 INTSR6: Reception completion interrupt request

13.4.3 Dedicated baud rate generator

The dedicated baud rate generator consists of a source clock selector and an 8-bit programmable counter, and generates a serial clock for transmission/reception of UART6.

Separate 8-bit counters are provided for transmission and reception.

(1) Configuration of baud rate generator

- Base clock

The clock selected by bits 3 to 0 (TPS63 to TPS60) of clock selection register 6 (CKSR6) is supplied to each module when bit 7 (POWER6) of asynchronous serial interface operation mode register 6 (ASIM6) is 1. This clock is called the base clock and its frequency is called f_{CLK6} . The base clock is fixed to low level when POWER6 = 0.

- Transmission counter

This counter stops operation, cleared to 0, when bit 7 (POWER6) or bit 6 (TXE6) of asynchronous serial interface operation mode register 6 (ASIM6) is 0.

It starts counting when POWER6 = 1 and TXE6 = 1.

The counter is cleared to 0 when the first data transmitted is written to transmit buffer register 6 (TXB6).

If data are continuously transmitted, the counter is cleared to 0 again when one frame of data has been completely transmitted. If there is no data to be transmitted next, the counter is not cleared to 0 and continues counting until POWER6 or TXE6 is cleared to 0.

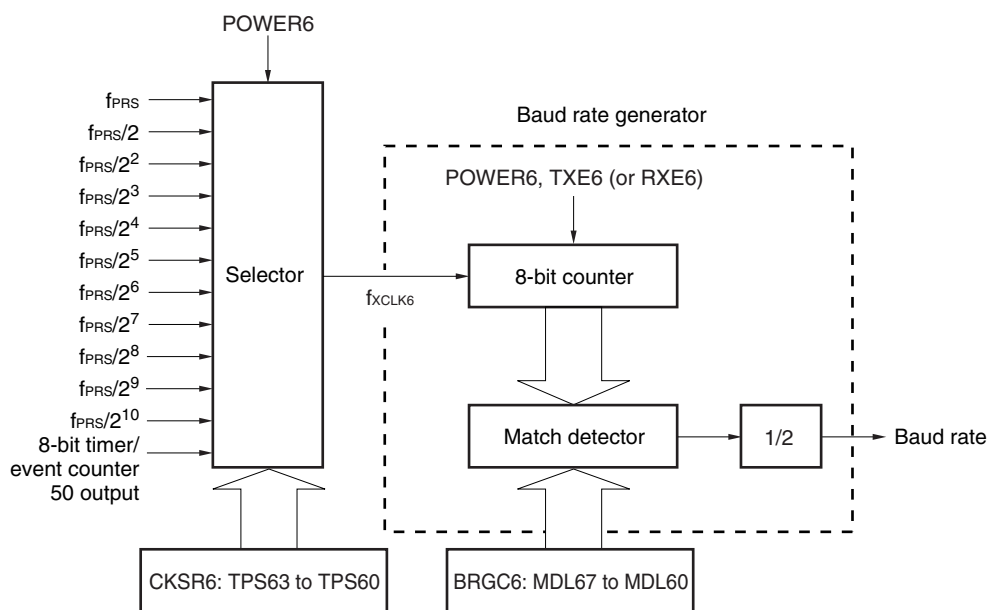
- Reception counter

This counter stops operation, cleared to 0, when bit 7 (POWER6) or bit 5 (RXE6) of asynchronous serial interface operation mode register 6 (ASIM6) is 0.

It starts counting when the start bit has been detected.

The counter stops operation after one frame has been received, until the next start bit is detected.

Figure 13-24. Configuration of Baud Rate Generator



- Remark**
- POWER6: Bit 7 of asynchronous serial interface operation mode register 6 (ASIM6)
 - TXE6: Bit 6 of ASIM6
 - RXE6: Bit 5 of ASIM6
 - CKSR6: Clock selection register 6
 - BRGC6: Baud rate generator control register 6

(2) Generation of serial clock

A serial clock to be generated can be specified by using clock selection register 6 (CKSR6) and baud rate generator control register 6 (BRGC6).

The clock to be input to the 8-bit counter can be set by bits 3 to 0 (TPS63 to TPS60) of CKSR6 and the division value ($f_{XCLK6}/4$ to $f_{XCLK6}/255$) of the 8-bit counter can be set by bits 7 to 0 (MDL67 to MDL60) of BRGC6.

13.4.4 Calculation of baud rate

(1) Baud rate calculation expression

The baud rate can be calculated by the following expression.

- Baud rate = $\frac{f_{XCLK6}}{2 \times k}$ [bps]

f_{XCLK6} : Frequency of base clock selected by TPS63 to TPS60 bits of CKSR6 register

k: Value set by MDL67 to MDL60 bits of BRGC6 register (k = 4, 5, 6, ..., 255)

Table 13-4. Set Value of TPS63 to TPS60

| TPS63 | TPS62 | TPS61 | TPS60 | | Base Clock (f _{CLK6}) Selection | | |
|------------------|-------|-------|-------|-----------------------------------|---|-----------------------------|------------------------------|
| | | | | | f _{PRS} = 2 MHz | f _{PRS} = 5 MHz | f _{PRS} = 10 MHz |
| 0 | 0 | 0 | 0 | f _{PRS} | 2 MHz | 5 MHz | 10 MHz |
| 0 | 0 | 0 | 1 | f _{PRS} /2 | 1 MHz | 2.5 MHz | 5 MHz |
| 0 | 0 | 1 | 0 | f _{PRS} /2 ² | 500 kHz | 1.25 MHz | 2.5 MHz |
| 0 | 0 | 1 | 1 | f _{PRS} /2 ³ | 250 kHz | 625 kHz | 1.25 MHz |
| 0 | 1 | 0 | 0 | f _{PRS} /2 ⁴ | 125 kHz | 312.5 kHz | 625 kHz |
| 0 | 1 | 0 | 1 | f _{PRS} /2 ⁵ | 62.5 kHz | 156.25 kHz | 312.5 kHz |
| 0 | 1 | 1 | 0 | f _{PRS} /2 ⁶ | 31.25 kHz | 78.13 kHz | 156.25 kHz |
| 0 | 1 | 1 | 1 | f _{PRS} /2 ⁷ | 15.625 kHz | 39.06 kHz | 78.13 kHz |
| 1 | 0 | 0 | 0 | f _{PRS} /2 ⁸ | 7.813 kHz | 19.53 kHz | 39.06 kHz |
| 1 | 0 | 0 | 1 | f _{PRS} /2 ⁹ | 3.906 kHz | 9.77 kHz | 19.53 kHz |
| 1 | 0 | 1 | 0 | f _{PRS} /2 ¹⁰ | 1.953 kHz | 4.88 kHz | 9.77 kHz |
| 1 | 0 | 1 | 1 | TM50 output ^{Note} | | | |
| Other than above | | | | Setting prohibited | | | |

Note Note the following points when selecting the TM50 output as the base clock.

- Mode in which the count clock is cleared and started upon a match of TM50 and CR50 (TMC506 = 0)
Start the operation of 8-bit timer/event counter 50 first and then enable the timer F/F inversion operation (TMC501 = 1).
- PWM mode (TMC506 = 1)
Start the operation of 8-bit timer/event counter 50 first and then set the count clock to make the duty = 50%.
It is not necessary to enable (TOE50 = 1) TO50 output in any mode.

(2) Error of baud rate

The baud rate error can be calculated by the following expression.

$$\bullet \text{ Error (\%)} = \left(\frac{\text{Actual baud rate (baud rate with error)}}{\text{Desired baud rate (correct baud rate)}} - 1 \right) \times 100 \text{ [\%]}$$

- Cautions**
1. Keep the baud rate error during transmission to within the permissible error range at the reception destination.
 2. Make sure that the baud rate error during reception satisfies the range shown in (4) Permissible baud rate range during reception.

Example: Frequency of base clock = 10 MHz = 10,000,000 Hz
Set value of MDL67 to MDL60 bits of BRGC6 register = 00100001B (k = 33)
Target baud rate = 153600 bps

$$\begin{aligned} \text{Baud rate} &= 10 \text{ M} / (2 \times 33) \\ &= 10000000 / (2 \times 33) = 151,515 \text{ [bps]} \end{aligned}$$

$$\begin{aligned} \text{Error} &= (151515/153600 - 1) \times 100 \\ &= -1.357 \text{ [\%]} \end{aligned}$$

(3) Example of setting baud rate

Table 13-5. Set Data of Baud Rate Generator

| Baud Rate [bps] | f _{PRS} = 2.0 MHz | | | | f _{PRS} = 5.0 MHz | | | | f _{PRS} = 10.0 MHz | | | |
|--------------------|----------------------------|----|---------------------|------------|----------------------------|----|---------------------|------------|-----------------------------|----|---------------------|------------|
| | TPS63 to TPS60 | k | Calculated Value | ERR [%] | TPS63 to TPS60 | k | Calculated Value | ERR [%] | TPS63 to TPS60 | k | Calculated Value | ERR [%] |
| 300 | 8H | 13 | 301 | 0.16 | 7H | 65 | 301 | 0.16 | 8H | 65 | 301 | 0.16 |
| 600 | 7H | 13 | 601 | 0.16 | 6H | 65 | 601 | 0.16 | 7H | 65 | 601 | 0.16 |
| 1200 | 6H | 13 | 1202 | 0.16 | 5H | 65 | 1202 | 0.16 | 6H | 65 | 1202 | 0.16 |
| 2400 | 5H | 13 | 2404 | 0.16 | 4H | 65 | 2404 | 0.16 | 5H | 65 | 2404 | 0.16 |
| 4800 | 4H | 13 | 4808 | 0.16 | 3H | 65 | 4808 | 0.16 | 4H | 65 | 4808 | 0.16 |
| 9600 | 3H | 13 | 9615 | 0.16 | 2H | 65 | 9615 | 0.16 | 3H | 65 | 9615 | 0.16 |
| 19200 | 2H | 13 | 19231 | 0.16 | 1H | 65 | 19231 | 0.16 | 2H | 65 | 19231 | 0.16 |
| 24000 | 1H | 21 | 23810 | -0.79 | 3H | 13 | 24038 | 0.16 | 4H | 13 | 24038 | 0.16 |
| 31250 | 1H | 16 | 31250 | 0 | 4H | 5 | 31250 | 0 | 5H | 5 | 31250 | 0 |
| 38400 | 1H | 13 | 38462 | 0.16 | 0H | 65 | 38462 | 0.16 | 1H | 65 | 38462 | 0.16 |
| 48000 | 0H | 21 | 47619 | -0.79 | 2H | 13 | 48077 | 0.16 | 3H | 13 | 48077 | 0.16 |
| 76800 | 0H | 13 | 76923 | 0.16 | 0H | 33 | 75758 | -1.36 | 0H | 65 | 76923 | 0.16 |
| 115200 | 0H | 9 | 111111 | -3.55 | 1H | 11 | 113636 | -1.36 | 0H | 43 | 116279 | 0.94 |
| 153600 | - | - | - | - | 1H | 8 | 156250 | 1.73 | 0H | 33 | 151515 | -1.36 |
| 312500 | - | - | - | - | 0H | 8 | 312500 | 0 | 1H | 8 | 312500 | 0 |
| 625000 | - | - | - | - | 0H | 4 | 625000 | 0 | 1H | 4 | 625000 | 0 |

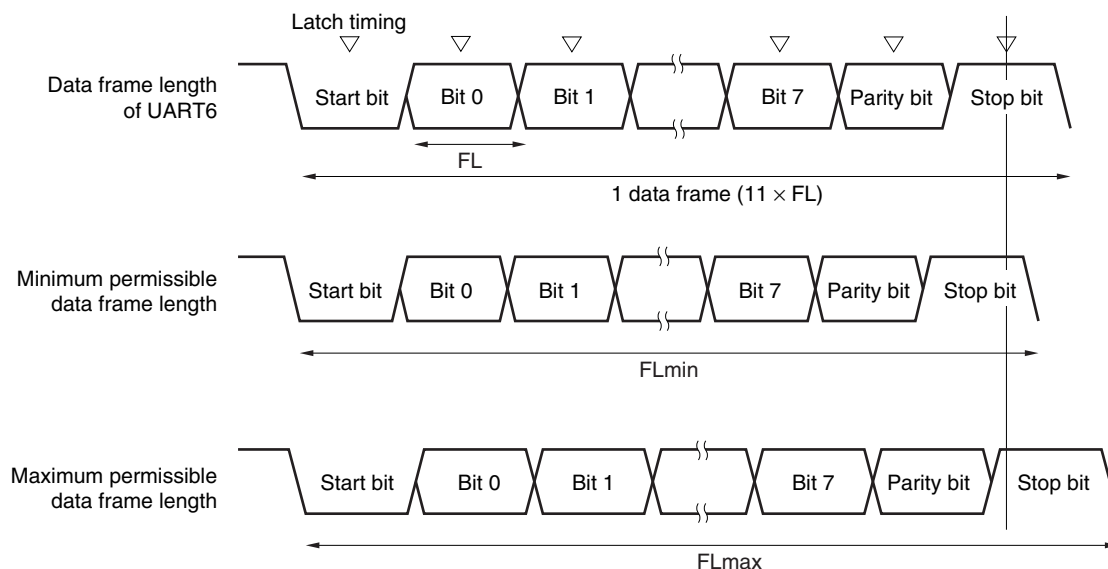
Remark TPS63 to TPS60: Bits 3 to 0 of clock selection register 6 (CKSR6) (setting of base clock (f_{CLK6}))
k: Value set by MDL67 to MDL60 bits of baud rate generator control register 6 (BRGC6) (k = 4, 5, 6, ..., 255)
f_{PRS}: Peripheral hardware clock frequency
ERR: Baud rate error

(4) Permissible baud rate range during reception

The permissible error from the baud rate at the transmission destination during reception is shown below.

Caution Make sure that the baud rate error during reception is within the permissible error range, by using the calculation expression shown below.

Figure 13-25. Permissible Baud Rate Range During Reception



As shown in Figure 13-25, the latch timing of the receive data is determined by the counter set by baud rate generator control register 6 (BRGC6) after the start bit has been detected. If the last data (stop bit) meets this latch timing, the data can be correctly received.

Assuming that 11-bit data is received, the theoretical values can be calculated as follows.

$$FL = (\text{Brate})^{-1}$$

Brate: Baud rate of UART6

k: Set value of BRGC6

FL: 1-bit data length

Margin of latch timing: 2 clocks

$$\text{Minimum permissible data frame length: } FL_{\min} = 11 \times FL - \frac{k-2}{2k} \times FL = \frac{21k+2}{2k} FL$$

Therefore, the maximum receivable baud rate at the transmission destination is as follows.

$$BR_{\max} = (FL_{\min}/11)^{-1} = \frac{22k}{21k+2} \text{ Brate}$$

Similarly, the maximum permissible data frame length can be calculated as follows.

$$\frac{10}{11} \times FL_{\max} = 11 \times FL - \frac{k+2}{2 \times k} \times FL = \frac{21k-2}{2 \times k} FL$$

$$FL_{\max} = \frac{21k-2}{20k} FL \times 11$$

Therefore, the minimum receivable baud rate at the transmission destination is as follows.

$$BR_{\min} = (FL_{\max}/11)^{-1} = \frac{20k}{21k-2} \text{ Brate}$$

The permissible baud rate error between UART6 and the transmission destination can be calculated from the above minimum and maximum baud rate expressions, as follows.

Table 13-6. Maximum/Minimum Permissible Baud Rate Error

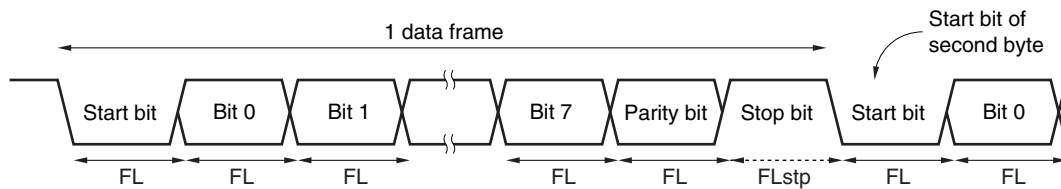
| Division Ratio (k) | Maximum Permissible Baud Rate Error | Minimum Permissible Baud Rate Error |
|--------------------|-------------------------------------|-------------------------------------|
| 4 | +2.33% | -2.44% |
| 8 | +3.53% | -3.61% |
| 20 | +4.26% | -4.31% |
| 50 | +4.56% | -4.58% |
| 100 | +4.66% | -4.67% |
| 255 | +4.72% | -4.73% |

- Remarks**
1. The permissible error of reception depends on the number of bits in one frame, input clock frequency, and division ratio (k). The higher the input clock frequency and the higher the division ratio (k), the higher the permissible error.
 2. k: Set value of BRGC6

(5) Data frame length during continuous transmission

When data is continuously transmitted, the data frame length from a stop bit to the next start bit is extended by two clocks of base clock from the normal value. However, the result of communication is not affected because the timing is initialized on the reception side when the start bit is detected.

Figure 13-26. Data Frame Length During Continuous Transmission



Where the 1-bit data length is FL, the stop bit length is FLstp, and base clock frequency is f_{CLK6} , the following expression is satisfied.

$$FL_{stp} = FL + 2/f_{CLK6}$$

Therefore, the data frame length during continuous transmission is:

$$\text{Data frame length} = 11 \times FL + 2/f_{CLK6}$$

CHAPTER 14 SERIAL INTERFACE CSI10

14.1 Functions of Serial Interface CSI10

Serial interface CSI10 has the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

(1) Operation stop mode

This mode is used when serial communication is not performed and can enable a reduction in the power consumption. For details, see **14.4.1 Operation stop mode**.

(2) 3-wire serial I/O mode (MSB/LSB-first selectable)

This mode is used to communicate 8-bit data using three lines: a serial clock line ($\overline{\text{SCK10}}$) and two serial data lines (SI10 and SO10).

The processing time of data communication can be shortened in the 3-wire serial I/O mode because transmission and reception can be simultaneously executed.

In addition, whether 8-bit data is communicated with the MSB or LSB first can be specified, so this interface can be connected to any device.

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface. For details, see **14.4.2 3-wire serial I/O mode**.

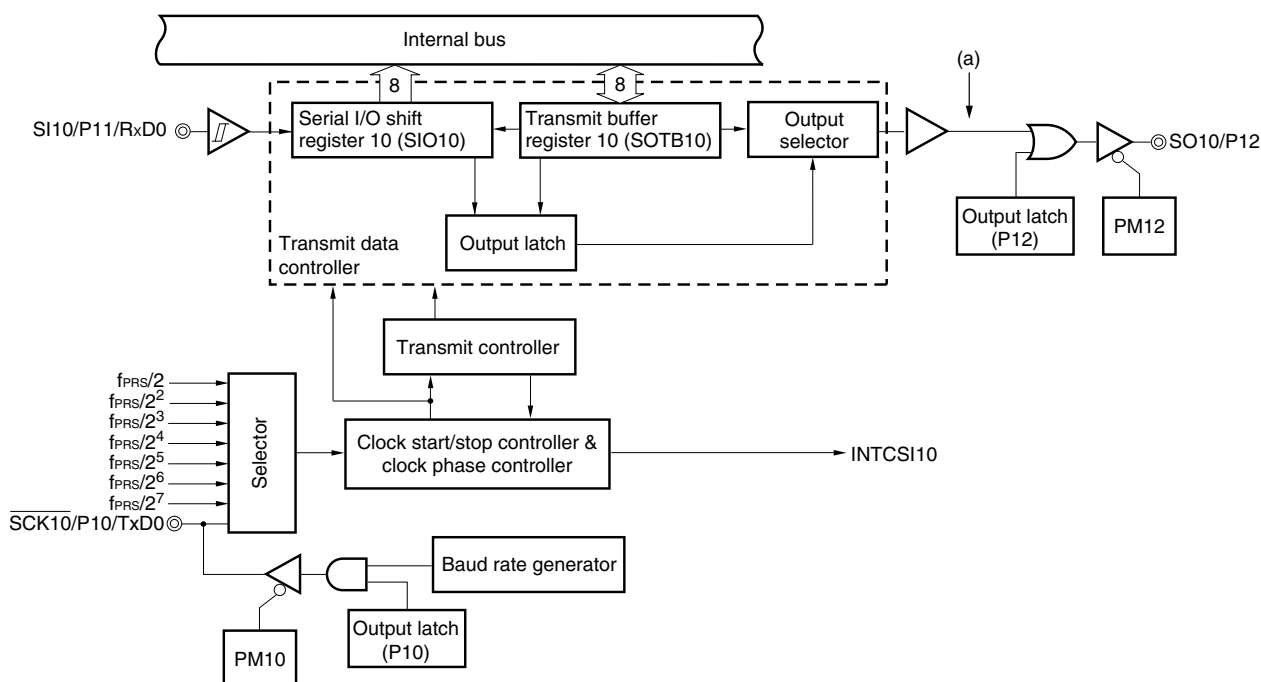
14.2 Configuration of Serial Interface CSI10

Serial interface CSI10 includes the following hardware.

Table 14-1. Configuration of Serial Interface CSI10

| Item | Configuration |
|-------------------|---|
| Controller | Transmit controller Clock start/stop controller & clock phase controller |
| Registers | Transmit buffer register 10 (SOTB10) Serial I/O shift register 10 (SIO10) |
| Control registers | Serial operation mode register 10 (CSIM10) Serial clock selection register 10 (CSIC10) Port mode register 1 (PM1) Port register 1 (P1) |

Figure 14-1. Block Diagram of Serial Interface CSI10



Remark (a): SO10 output

(1) Transmit buffer register 10 (SOTB10)

This register sets the transmit data.

Transmission/reception is started by writing data to SOTB10 when bit 7 (CSIE10) and bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 1.

The data written to SOTB10 is converted from parallel data into serial data by serial I/O shift register 10, and output to the serial output pin (SO10).

SOTB10 can be written or read by an 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Caution Do not access SOTB10 when CSOT10 = 1 (during serial communication).

(2) Serial I/O shift register 10 (SIO10)

This is an 8-bit register that converts data from parallel data into serial data and vice versa.

This register can be read by an 8-bit memory manipulation instruction.

Reception is started by reading data from SIO10 if bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 0.

During reception, the data is read from the serial input pin (SI10) to SIO10.

Reset signal generation sets this register to 00H.

Caution Do not access SIO10 when CSOT10 = 1 (during serial communication).

14.3 Registers Controlling Serial Interface CSI10

Serial interface CSI10 is controlled by the following four registers.

- Serial operation mode register 10 (CSIM10)
- Serial clock selection register 10 (CSIC10)
- Port mode register 1 (PM1)
- Port register 1 (P1)

(1) Serial operation mode register 10 (CSIM10)

CSIM10 is used to select the operation mode and enable or disable operation.

CSIM10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Figure 14-2. Format of Serial Operation Mode Register 10 (CSIM10)

Address: FF80H After reset: 00H R/W^{Note 1}

| | | | | | | | | |
|--------|--------|--------|---|-------|---|---|---|--------|
| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CSIM10 | CSIE10 | TRMD10 | 0 | DIR10 | 0 | 0 | 0 | CSOT10 |

| | |
|--------|---|
| CSIE10 | Operation control in 3-wire serial I/O mode |
| 0 | Disables operation ^{Note 2} and asynchronously resets the internal circuit ^{Note 3} . |
| 1 | Enables operation |

| | |
|--------------------------|---------------------------------------|
| TRMD10 ^{Note 4} | Transmit/receive mode control |
| 0 ^{Note 5} | Receive mode (transmission disabled). |
| 1 | Transmit/receive mode |

| | |
|-------------------------|-------------------------|
| DIR10 ^{Note 6} | First bit specification |
| 0 | MSB |
| 1 | LSB |

| | |
|--------|-------------------------------|
| CSOT10 | Communication status flag |
| 0 | Communication is stopped. |
| 1 | Communication is in progress. |

- Notes**
1. Bit 0 is a read-only bit.
 2. To use P10/ $\overline{\text{SCK10}}$ /TxD0 and P12/SO10 as general-purpose ports, set CSIM10 in the default status (00H).
 3. Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.
 4. Do not rewrite TRMD10 when CSOT10 = 1 (during serial communication).
 5. The SO10 output (see (a) in **Figure 14-1**) is fixed to the low level when TRMD10 is 0. Reception is started when data is read from SIO10.
 6. Do not rewrite DIR10 when CSOT10 = 1 (during serial communication).

Caution Be sure to clear bit 5 to 0.

(2) Serial clock selection register 10 (CSIC10)

This register specifies the timing of the data transmission/reception and sets the serial clock.

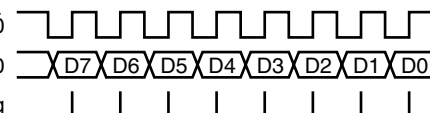
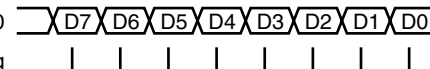

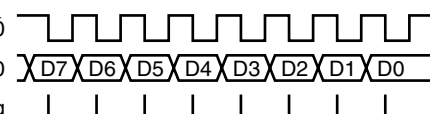
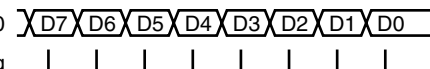

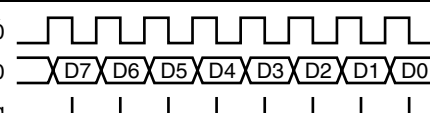
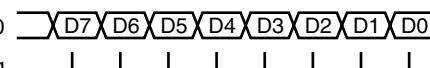

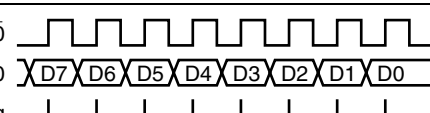
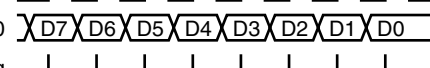

CSIC10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets this register to 00H.

Figure 14-3. Format of Serial Clock Selection Register 10 (CSIC10)

Address: FF81H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|-------|-------|--------|--------|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CSIC10 | 0 | 0 | 0 | CKP10 | DAP10 | CKS102 | CKS101 | CKS100 |

| CKP10 | DAP10 | Specification of data transmission/reception timing | Type |
|-------|-------|--|------|
| 0 | 0 | $\overline{\text{SCK10}}$  SO10  SI10 input timing  | 1 |
| 0 | 1 | SCK10  SO10  SI10 input timing  | 2 |
| 1 | 0 | $\overline{\text{SCK10}}$  SO10  SI10 input timing  | 3 |
| 1 | 1 | SCK10  SO10  SI10 input timing  | 4 |

| CKS102 | CKS101 | CKS100 | CSI10 serial clock selection ^{Note 1} | | | Mode | |
|--------|--------|--------|---|----------------------------------|-----------------------------------|------------|-------------|
| | | | $f_{\text{PRS}} = 2 \text{ MHz}$ | $f_{\text{PRS}} = 5 \text{ MHz}$ | $f_{\text{PRS}} = 10 \text{ MHz}$ | | |
| 0 | 0 | 0 | $f_{\text{PRS}}/2$ | 1 MHz | 2.5 MHz | 5 MHz | Master mode |
| 0 | 0 | 1 | $f_{\text{PRS}}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz | |
| 0 | 1 | 0 | $f_{\text{PRS}}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz | |
| 0 | 1 | 1 | $f_{\text{PRS}}/2^4$ | 125 kHz | 312.5 kHz | 625 kHz | |
| 1 | 0 | 0 | $f_{\text{PRS}}/2^5$ | 62.5 kHz | 156.25 kHz | 312.5 kHz | |
| 1 | 0 | 1 | $f_{\text{PRS}}/2^6$ | 31.25 kHz | 78.13 kHz | 156.25 kHz | |
| 1 | 1 | 0 | $f_{\text{PRS}}/2^7$ | 15.63 kHz | 39.06 kHz | 78.13 kHz | |
| 1 | 1 | 1 | External clock input to $\overline{\text{SCK10}}$ ^{Note 2} | | | Slave mode | |

Notes 1. Set the serial clock to satisfy the following conditions.

Serial clock $\leq 5 \text{ MHz}$

2. Do not start communication with the external clock from the $\overline{\text{SCK10}}$ pin when in the STOP mode.

Cautions 1. Do not write to CSIC10 while CSIE10 = 1 (operation enabled).

2. To use P10/ $\overline{\text{SCK10}}$ /Tx/D0 and P12/SO10 as general-purpose ports, set CSIC10 in the default status (00H).

3. The phase type of the data clock is type 1 after reset.

Remark f_{PRS} : Peripheral hardware clock oscillation frequency

(3) Port mode register 1 (PM1)

This register sets port 1 input/output in 1-bit units.

When using P10/ $\overline{\text{SCK10}}$ as the clock output pin of the serial interface, clear PM10 to 0, and set the output latches of P10 to 1.

When using P12/SO10 as the data output pin of the serial interface, clear PM12 and the output latches of P12 to 0.

When using P10/ $\overline{\text{SCK10}}$ as the clock input pin of the serial interface and P11/SI10/RxD0 as the data input pin, set PM10 and PM11 to 1. At this time, the output latches of P10 and P11 may be 0 or 1.

PM1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 14-4. Format of Port Mode Register 1 (PM1)

Address: FF21H After reset: FFH R/W

| | | | | | | | | |
|--------|------|------|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PM1 | PM17 | PM16 | PM15 | PM14 | PM13 | PM12 | PM11 | PM10 |

| PM1n | P1n pin I/O mode selection (n = 0 to 7) |
|------|---|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

14.4 Operation of Serial Interface CSI10

Serial interface CSI10 can be used in the following two modes.

- Operation stop mode
- 3-wire serial I/O mode

14.4.1 Operation stop mode

Serial communication is not executed in this mode. Therefore, the power consumption can be reduced. In addition, the P10/ $\overline{\text{SCK10}}$ /TxD0, P11/SI10/RxD0, and P12/SO10 pins can be used as ordinary I/O port pins in this mode.

(1) Register used

The operation stop mode is set by serial operation mode register 10 (CSIM10).

To set the operation stop mode, clear bit 7 (CSIE10) of CSIM10 to 0.

(a) Serial operation mode register 10 (CSIM10)

CSIM10 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets CSIM10n to 00H.

Address: FF80H After reset: 00H R/W

| Symbol | <7> | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|--------|---|-------|---|---|---|--------|
| CSIM10 | CSIE10 | TRMD10 | 0 | DIR10 | 0 | 0 | 0 | CSOT10 |
| CSIE10 | Operation control in 3-wire serial I/O mode | | | | | | | |
| 0 | Disables operation ^{Note 1} and asynchronously resets the internal circuit ^{Note 2} . | | | | | | | |

Notes 1. To use P10/ $\overline{\text{SCK10}}$ /TxD0 and P12/SO10 as general-purpose ports, set CSIM10 in the default status (00H).

2. Bit 0 (CSOT10) of CSIM10 and serial I/O shift register 10 (SIO10) are reset.

14.4.2 3-wire serial I/O mode

The 3-wire serial I/O mode is used for connecting peripheral ICs and display controllers with a clocked serial interface.

In this mode, communication is executed by using three lines: the serial clock ($\overline{\text{SCK10}}$), serial output (SO10), and serial input (SI10) lines.

(1) Registers used

- Serial operation mode register 10 (CSIM10)
- Serial clock selection register 10 (CSIC10)
- Port mode register 1 (PM1)
- Port register 1 (P1)

The basic procedure of setting an operation in the 3-wire serial I/O mode is as follows.

- <1> Set the CSIC10 register (see **Figures 14-3**).
- <2> Set bits 0, 4, and 6 (CSOT10, DIR10, and TRMD10) of the CSIM10 register (see **Figures 14-2**).
- <3> Set bit 7 (CSIE10) of the CSIM10 register to 1. → Transmission/reception is enabled.
- <4> Write data to transmit buffer register 10 (SOTB10). → Data transmission/reception is started.
Read data from serial I/O shift register 10 (SIO10). → Data reception is started.

Caution Take relationship with the other party of communication when setting the port mode register and port register.

The relationship between the register settings and pins is shown below.

Table 14-2. Relationship Between Register Settings and Pins

| CSIE10 | TRMD10 | PM11 | P11 | PM12 | P12 | PM10 | P10 | CSI10 Operation | Pin Function | | |
|--------|--------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---|-------------------|----------|--|
| | | | | | | | | | SI10/RxD0/ P11 | SO10/P12 | $\overline{\text{SCK10}}$ / TxD0/P10 |
| 0 | × | × ^{Note 1} | × ^{Note 1} | × ^{Note 1} | × ^{Note 1} | × ^{Note 1} | × ^{Note 1} | Stop | RxD0/P11 | P12 | TxD0/ P10 ^{Note 2} |
| 1 | 0 | 1 | × | × ^{Note 1} | × ^{Note 1} | 1 | × | Slave reception ^{Note 3} | SI10 | P12 | $\overline{\text{SCK10}}$ (input) ^{Note 3} |
| 1 | 1 | × ^{Note 1} | × ^{Note 1} | 0 | 0 | 1 | × | Slave transmission ^{Note 3} | RxD0/P11 | SO10 | $\overline{\text{SCK10}}$ (input) ^{Note 3} |
| 1 | 1 | 1 | × | 0 | 0 | 1 | × | Slave transmission/ reception ^{Note 3} | SI10 | SO10 | $\overline{\text{SCK10}}$ (input) ^{Note 3} |
| 1 | 0 | 1 | × | × ^{Note 1} | × ^{Note 1} | 0 | 1 | Master reception | SI10 | P12 | $\overline{\text{SCK10}}$ (output) |
| 1 | 1 | × ^{Note 1} | × ^{Note 1} | 0 | 0 | 0 | 1 | Master transmission | RxD0/P11 | SO10 | $\overline{\text{SCK10}}$ (output) |
| 1 | 1 | 1 | × | 0 | 0 | 0 | 1 | Master transmission/ reception | SI10 | SO10 | $\overline{\text{SCK10}}$ (output) |

Notes 1. Can be set as port function.

2. To use P10/ $\overline{\text{SCK10}}$ /TxD0 as port pins, clear CKP10 to 0.

3. To use the slave mode, set CKS102, CKS101, and CKS100 to 1, 1, 1.

Remark

- ×: don't care
- CSIE10: Bit 7 of serial operation mode register 10 (CSIM10)
- TRMD10: Bit 6 of CSIM10
- CKP10: Bit 4 of serial clock selection register 10 (CSIC10)
- CKS102, CKS101, CKS100: Bits 2 to 0 of CSIC10
- PM1×: Port mode register
- P1×: Port output latch

(2) Communication operation

In the 3-wire serial I/O mode, data is transmitted or received in 8-bit units. Each bit of the data is transmitted or received in synchronization with the serial clock.

Data can be transmitted or received if bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 1. Transmission/reception is started when a value is written to transmit buffer register 10 (SOTB10). In addition, data can be received when bit 6 (TRMD10) of serial operation mode register 10 (CSIM10) is 0.

Reception is started when data is read from serial I/O shift register 10 (SIO10).

After communication has been started, bit 0 (CSOT10) of CSIM10 is set to 1. When communication of 8-bit data has been completed, a communication completion interrupt request flag (CSIIF10) is set, and CSOT10 is cleared to 0. Then the next communication is enabled.

Caution Do not access the control register and data register when CSOT10 = 1 (during serial communication).

Figure 14-5. Timing in 3-Wire Serial I/O Mode (1/2)

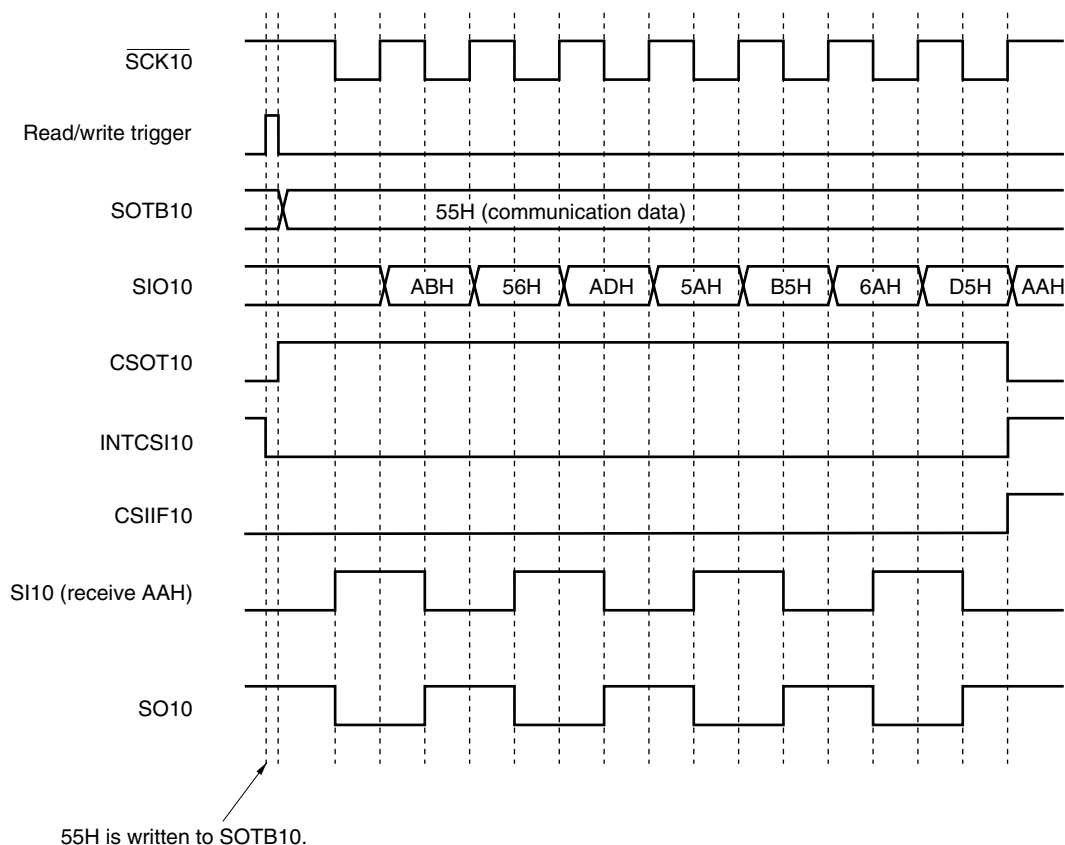
(1) Transmission/reception timing (Type 1: TRMD10 = 1, DIR10 = 0, CKP10 = 0, DAP10 = 0)

Figure 14-5. Timing in 3-Wire Serial I/O Mode (2/2)

(2) Transmission/reception timing (Type 2: TRMD10 = 1, DIR10 = 0, CKP10 = 0, DAP10 = 1)

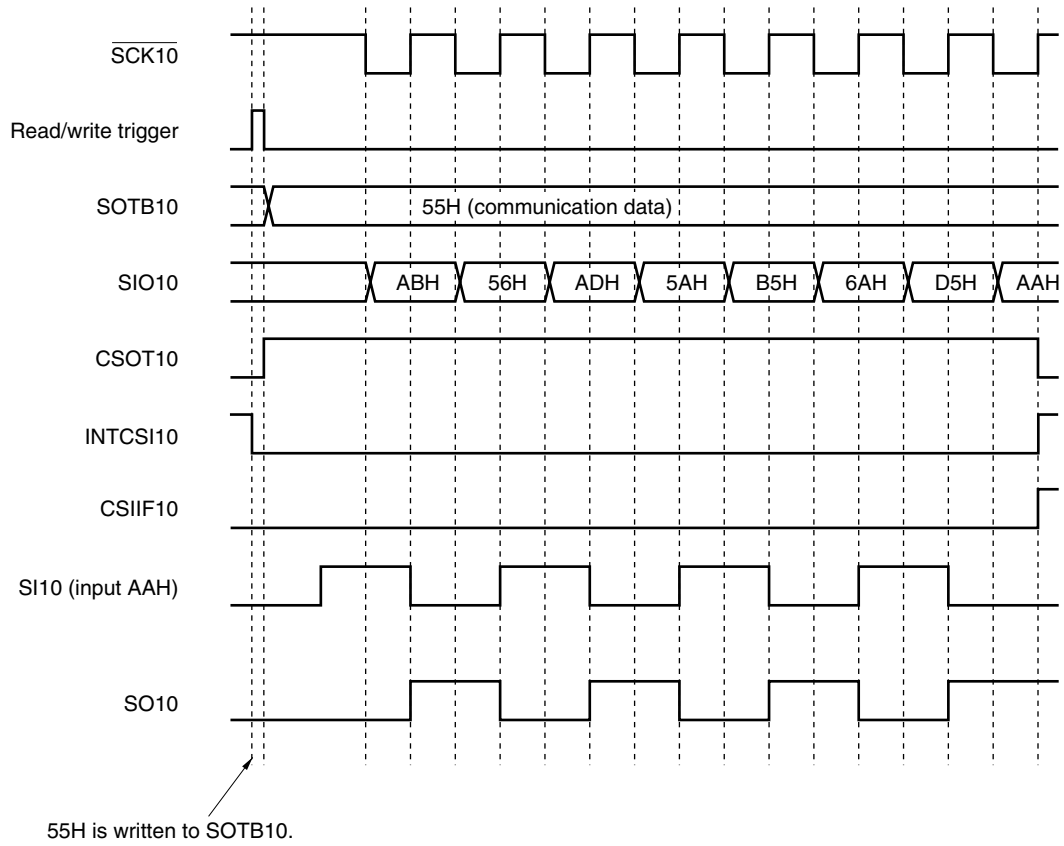
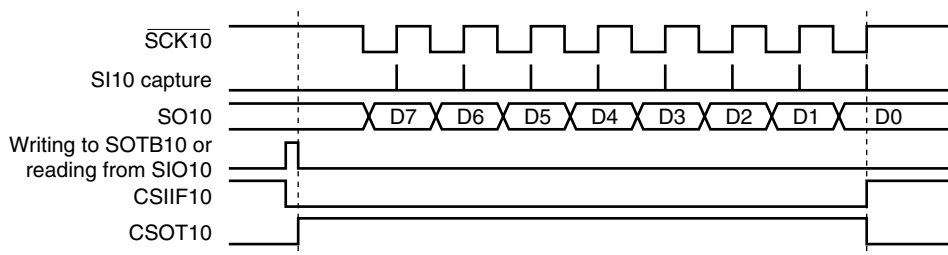
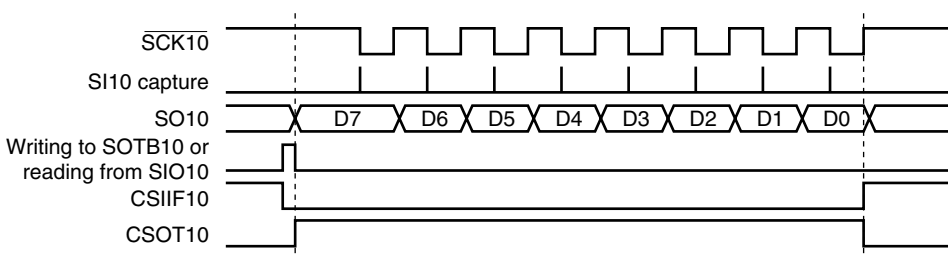


Figure 14-6. Timing of Clock/Data Phase

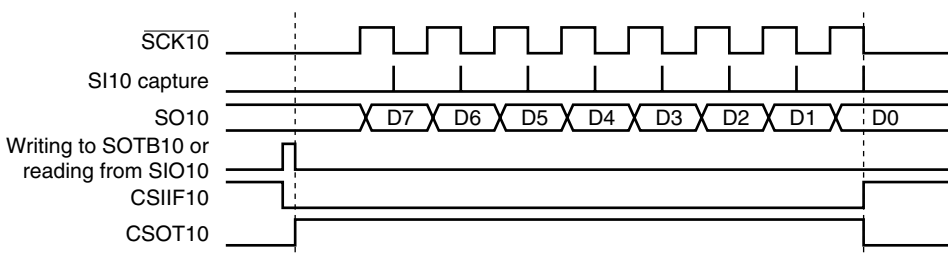
(a) Type 1: CKP10 = 0, DAP10 = 0, DIR10 = 0



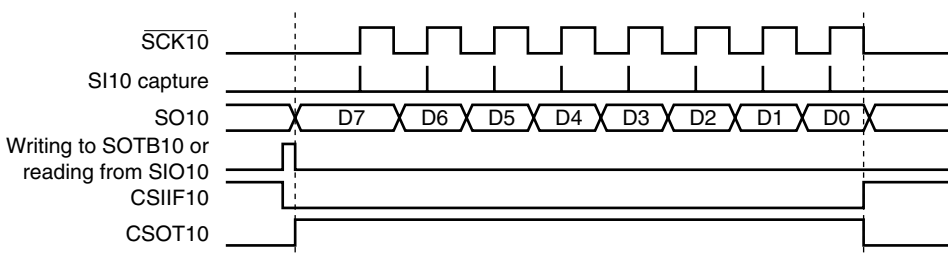
(b) Type 2: CKP10 = 0, DAP10 = 1, DIR10 = 0



(c) Type 3: CKP10 = 1, DAP10 = 0, DIR10 = 0



(d) Type 4: CKP10 = 1, DAP10 = 1, DIR10 = 0

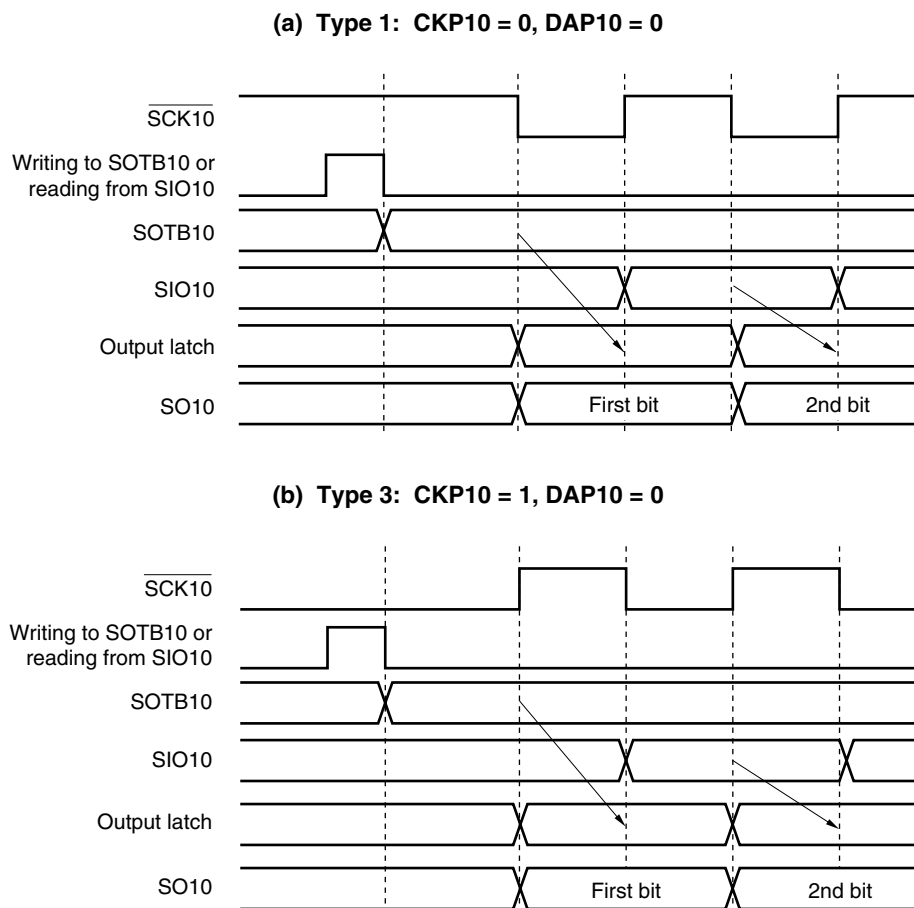


Remark The above figure illustrates a communication operation where data is transmitted with the MSB first.

(3) Timing of output to SO10 pin (first bit)

When communication is started, the value of transmit buffer register 10 (SOTB10) is output from the SO10 pin. The output operation of the first bit at this time is described below.

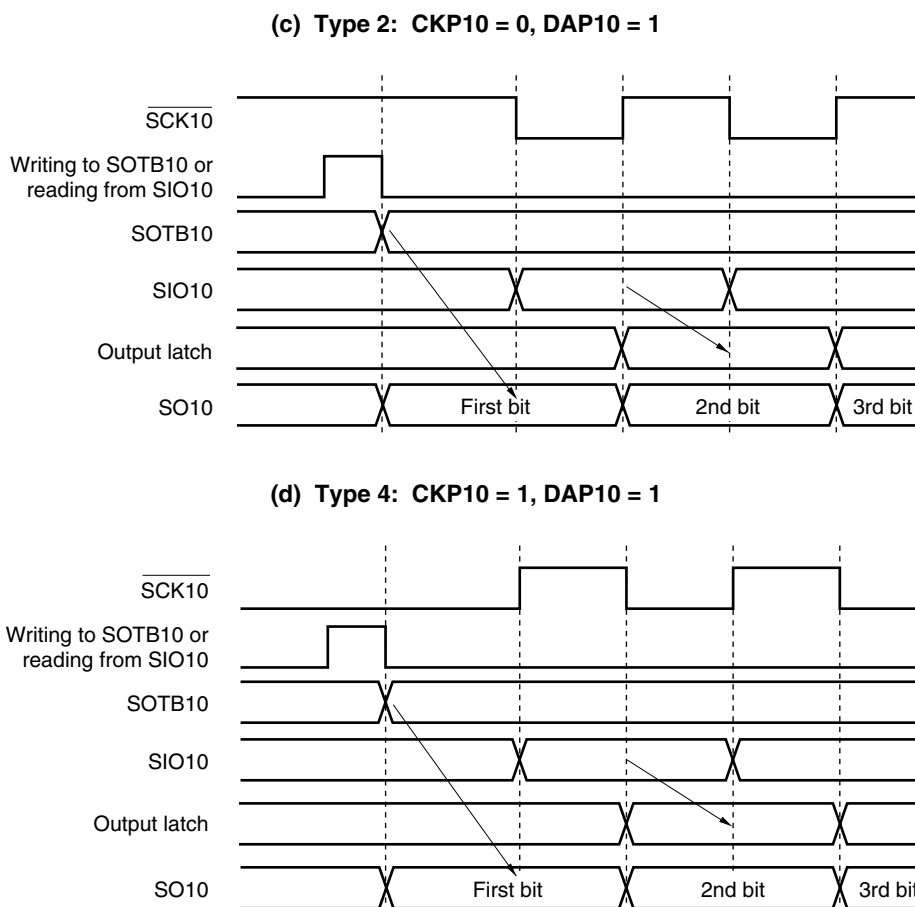
Figure 14-7. Output Operation of First Bit (1/2)



The first bit is directly latched by the SOTB10 register to the output latch at the falling (or rising) edge of $\overline{SCK10}$, and output from the SO10 pin via an output selector. Then, the value of the SOTB10 register is transferred to the SIO10 register at the next rising (or falling) edge of $\overline{SCK10}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO10 register via the S110 pin.

The second and subsequent bits are latched by the SIO10 register to the output latch at the next falling (or rising) edge of $\overline{SCK10}$, and the data is output from the SO10 pin.

Figure 14-7. Output Operation of First Bit (2/2)



The first bit is directly latched by the SOTB10 register at the falling edge of the write signal of the SOTB10 register or the read signal of the SIO10 register, and output from the SO10 pin via an output selector. Then, the value of the SOTB10 register is transferred to the SIO10 register at the next falling (or rising) edge of $\overline{SCK10}$, and shifted one bit. At the same time, the first bit of the receive data is stored in the SIO10 register via the S110 pin.

The second and subsequent bits are latched by the SIO10 register to the output latch at the next rising (or falling) edge of $\overline{SCK10}$, and the data is output from the SO10 pin.

(4) Output value of SO10 pin (last bit)

After communication has been completed, the SO10 pin holds the output value of the last bit.

Figure 14-8. Output Value of SO10 Pin (Last Bit) (1/2)

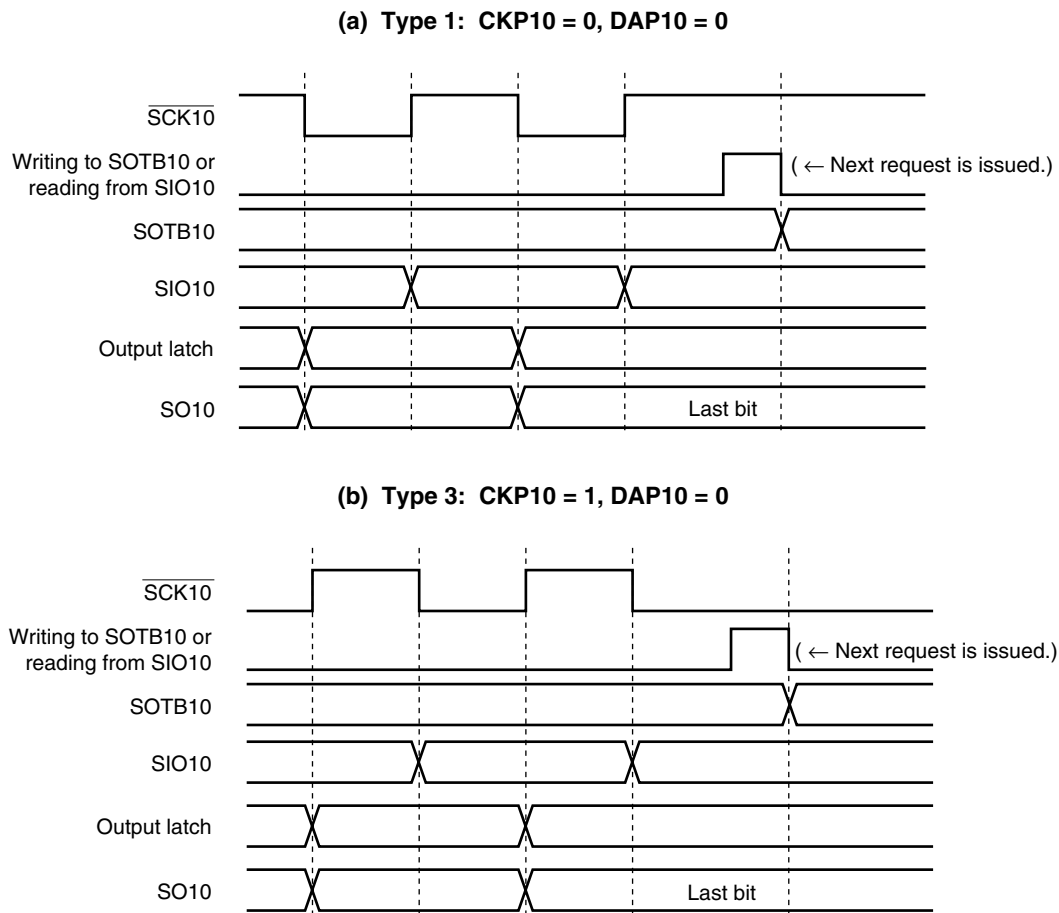
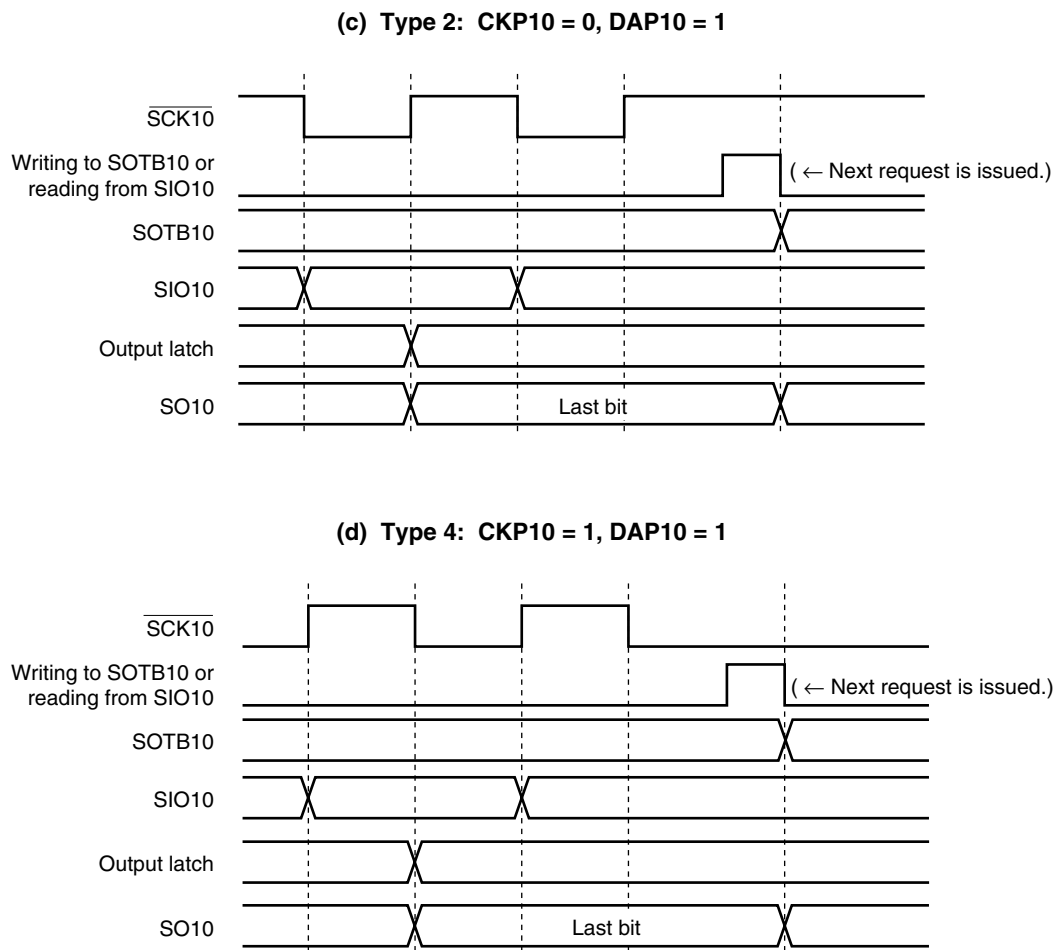


Figure 14-8. Output Value of SO10 Pin (Last Bit) (2/2)



(5) SO10 output (see (a) in Figure 14-1)

The status of the SO10 output is as follows if bit 7 (CSIE10) of serial operation mode register 10 (CSIM10) is cleared to 0.

Table 14-3. SO10 Output Status

| TRMD10 | DAP10 | DIR10 | SO10 Output ^{Note 1} |
|------------------------------|-----------|-----------|---|
| TRMD10 = 0 ^{Note 2} | – | – | Outputs low level ^{Note 2} |
| TRMD10 = 1 | DAP10 = 0 | – | Value of SO10 latch (low-level output) |
| | DAP10 = 1 | DIR10 = 0 | Value of bit 7 of SOTB10 |
| | | DIR10 = 1 | Value of bit 0 of SOTB10 |

Notes 1. The actual output of the SO10/P12 pin is determined according to PM12 and P12, as well as the SO10 output.

2. Status after reset

Caution If a value is written to TRMD10, DAP10, and DIR10, the output value of SO10 changes.

CHAPTER 15 SERIAL INTERFACE IIC0

15.1 Functions of Serial Interface IIC0

Serial interface IIC0 has the following two 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) I²C bus mode (multimaster supported)

This mode is used for 8-bit data transfers with several devices via two lines: a serial clock (SCL0) line and a serial data bus (SDA0) line.

This mode complies with the I²C 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 I²C bus.

Since the SCL0 and SDA0 pins are used for open drain outputs, IIC0 requires pull-up resistors for the serial clock line and the serial data bus line.

Figure 15-1 shows a block diagram of serial interface IIC0.

Figure 15-1. Block Diagram of Serial Interface IIC0

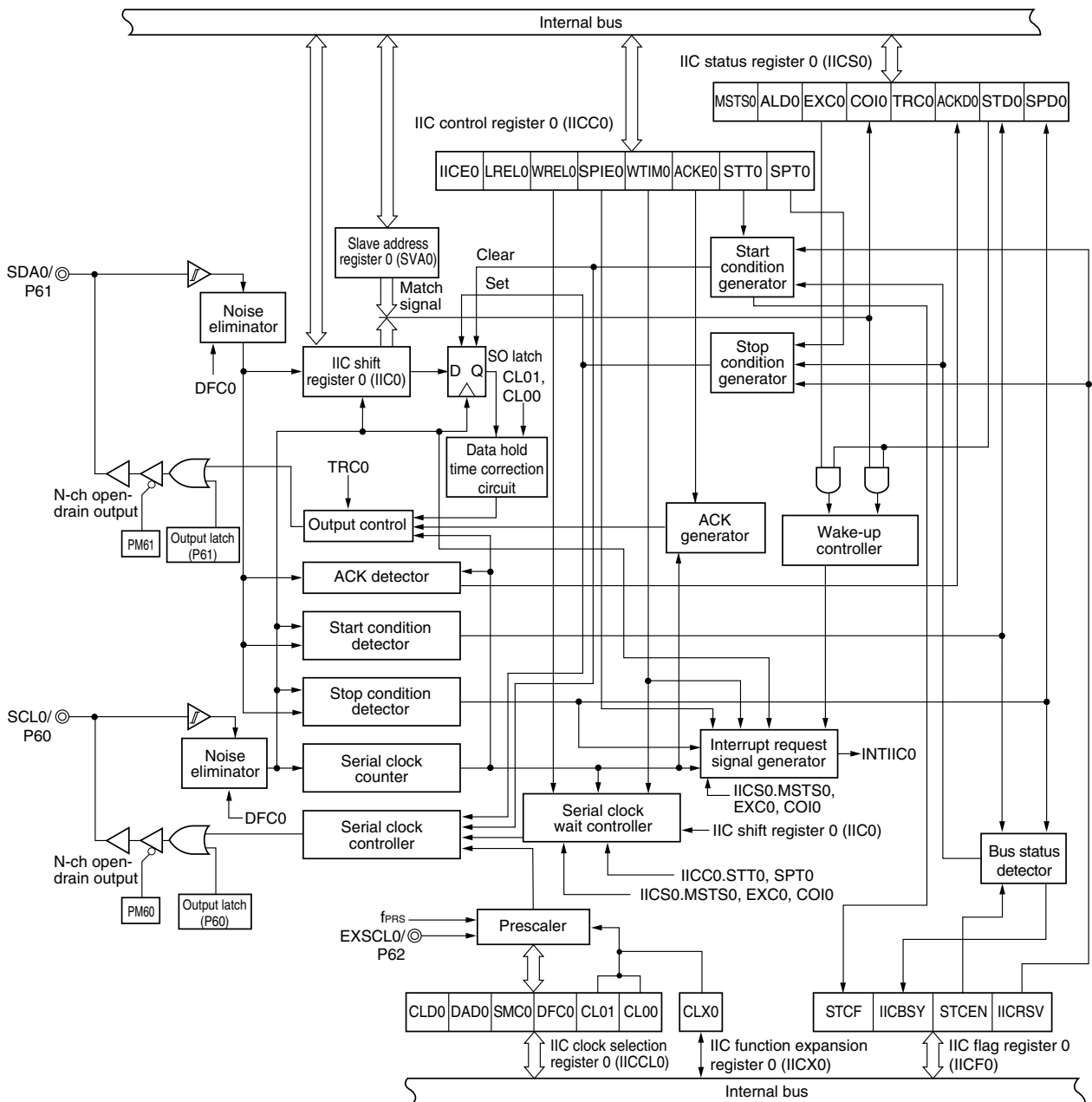
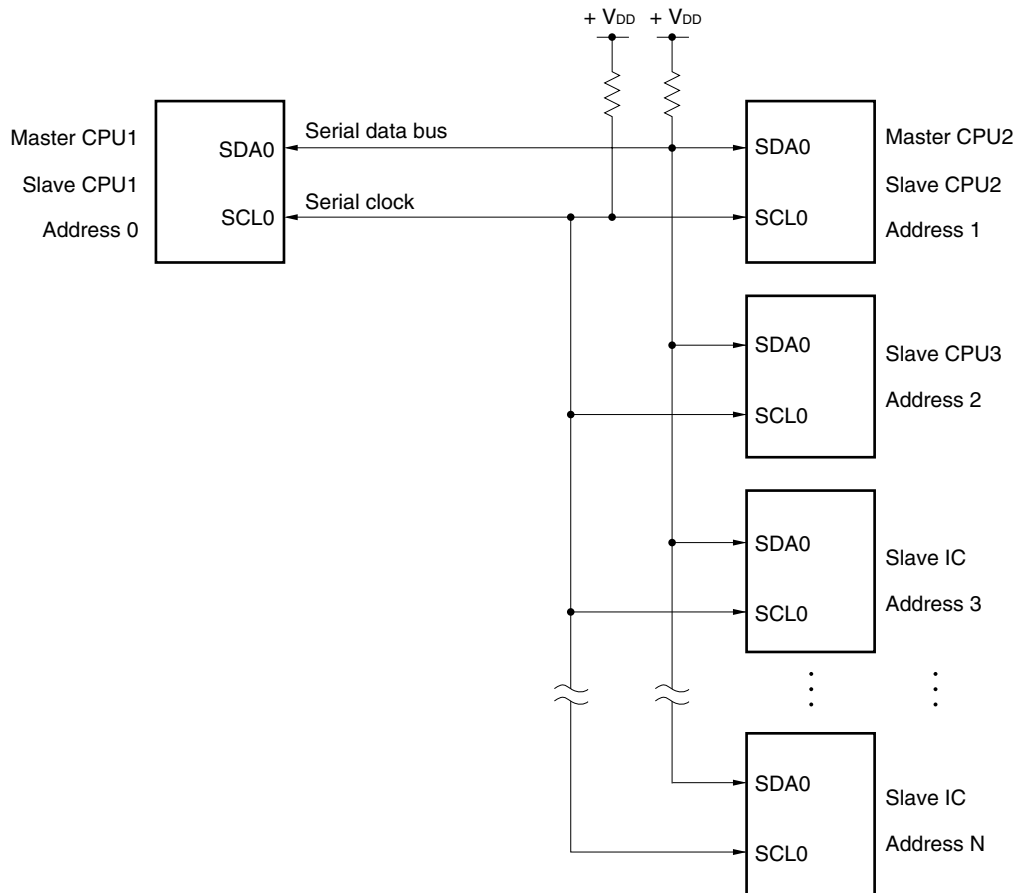


Figure 15-2 shows a serial bus configuration example.

Figure 15-2. Serial Bus Configuration Example Using I²C Bus



15.2 Configuration of Serial Interface IIC0

Serial interface IIC0 includes the following hardware.

Table 15-1. Configuration of Serial Interface IIC0

| Item | Configuration |
|-------------------|--|
| Registers | IIC shift register 0 (IIC0) Slave address register 0 (SVA0) |
| Control registers | IIC control register 0 (IICC0) IIC status register 0 (IICS0) IIC flag register 0 (IICF0) IIC clock selection register 0 (IICCL0) IIC function expansion register 0 (IICX0) Port mode register 6 (PM6) Port register 6 (P6) |

(1) IIC shift register 0 (IIC0)

IIC0 is used to convert 8-bit serial data to 8-bit parallel data and vice versa in synchronization with the serial clock.

IIC0 can be used for both transmission and reception.

The actual transmit and receive operations can be controlled by writing and reading operations to IIC0.

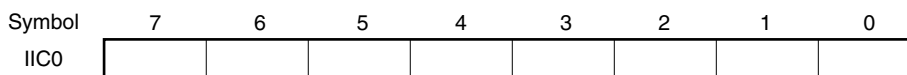
Cancel the wait state and start data transfer by writing data to IIC0 during the wait period.

IIC0 is set by an 8-bit memory manipulation instruction.

Reset signal generation clears IIC0 to 00H.

Figure 15-3. Format of IIC Shift Register 0 (IIC0)

Address: FFA5H After reset: 00H R/W



- Cautions**
1. Do not write data to IIC0 during data transfer.
 2. Write or read IIC0 only during the wait period. Accessing IIC0 in a communication state other than during the wait period is prohibited. When the device serves as the master, however, IIC0 can be written only once after the communication trigger bit (STT0) is set to 1.
 3. When communication is reserved, write data to the IIC0 register after the interrupt triggered by a stop condition is detected.

(2) Slave address register 0 (SVA0)

This register stores seven bits of local addresses {A6, A5, A4, A3, A2, A1, A0} when in slave mode.

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

However, rewriting to this register is prohibited while $STD0 = 1$ (while the start condition is detected).

Reset signal generation clears SVA0 to 00H.

Figure 15-4. Format of Slave Address Register 0 (SVA0)

Address: FFA7H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|-------------------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SVA0 | | | | | | | | 0 ^{Note} |

Note Bit 0 is fixed to 0.

(3) SO latch

The SO latch is used to retain the SDA0 pin's output level.

(4) Wake-up controller

This circuit generates an interrupt request (INTIIC0) when the address received by this register matches the address value set to slave address register 0 (SVA0) or when an extension code is received.

(5) Prescaler

This selects the sampling clock to be used.

(6) 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.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).

An I²C interrupt request is generated by the following two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by WTIM0 bit)
- Interrupt request generated when a stop condition is detected (set by SPIE0 bit)

Remark WTIM0 bit: Bit 3 of IIC control register 0 (IICC0)

SPIE0 bit: Bit 4 of IIC control register 0 (IICC0)

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) ACK generator, stop condition detector, start condition detector, and ACK detector

These circuits generate and detect each status.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

(12) Start condition generator

This circuit generates a start condition when the STT0 bit is set to 1.

However, in the communication reservation disabled status (IICRSV bit = 1), when the bus is not released (IICBSY bit = 1), start condition requests are ignored and the STCF bit is set to 1.

(13) Stop condition generator

This circuit generates a stop condition when the SPT0 bit is set to 1.

(14) 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 STCEN bit.

| | | |
|---------------|-------------|---|
| Remark | STT0 bit: | Bit 1 of IIC control register 0 (IICC0) |
| | SPT0 bit: | Bit 0 of IIC control register 0 (IICC0) |
| | IICRSV bit: | Bit 0 of IIC flag register 0 (IICF0) |
| | IICBSY bit: | Bit 6 of IIC flag register 0 (IICF0) |
| | STCF bit: | Bit 7 of IIC flag register 0 (IICF0) |
| | STCEN bit: | Bit 1 of IIC flag register 0 (IICF0) |

15.3 Registers to Control Serial Interface IIC0

Serial interface IIC0 is controlled by the following seven registers.

- IIC control register 0 (IICC0)
- IIC flag register 0 (IICF0)
- IIC status register 0 (IICS0)
- IIC clock selection register 0 (IICCL0)
- IIC function expansion register 0 (IICX0)
- Port mode register 6 (PM6)
- Port register 6 (P6)

(1) IIC control register 0 (IICC0)

This register is used to enable/stop I²C operations, set wait timing, and set other I²C operations.

IICC0 register is set by a 1-bit or 8-bit memory manipulation instruction. However, set the SPIE0, WTIM0, and ACKE0 bits while IICE0 bit = 0 or during the wait period. These bits can be set at the same time when the IICE0 bit is set from "0" to "1".

Reset signal generation clears IICC0 to 00H.

Figure 15-5. Format of IIC Control Register 0 (IICC0) (1/4)

Address: FFA6H After reset: 00H R/W

| | | | | | | | | |
|--------|-------|-------|-------|-------|-------|-------|------|------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| IICC0 | IICE0 | LRELO | WRELO | SPIE0 | WTIM0 | ACKE0 | STT0 | SPT0 |

| | |
|---|--|
| IICE0 | I ² C operation enable |
| 0 | Stop operation. Reset IIC status register 0 (IICS0) ^{Note 1} . Stop internal operation. |
| 1 | Enable operation. |
| Be sure to set this bit (1) while the SCL0 and SDA0 lines are at high level. | |
| Condition for clearing (IICE0 = 0) | |
| <ul style="list-style-type: none"> • Cleared by instruction • Reset | |
| Condition for setting (IICE0 = 1) | |
| <ul style="list-style-type: none"> • Set by instruction | |

| | |
|--|--|
| LRELO ^{Notes 2,3} | 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 SCL0 and SDA0 lines are set to high impedance. The following flags of IIC control register 0 (IICC0) and IIC status register 0 (IICS0) are cleared to 0. • STT0 • SPT0 • MSTS0 • EXC0 • COI0 • TRC0 • ACKD0 • STD0 |
| The standby mode following exit from communications remains in effect until the following communications entry conditions are met. | |
| <ul style="list-style-type: none"> • After a stop condition is detected, restart is in master mode. • An address match or extension code reception occurs after the start condition. | |
| Condition for clearing (LRELO = 0) | |
| <ul style="list-style-type: none"> • Automatically cleared after execution • Reset | |
| Condition for setting (LRELO = 1) | |
| <ul style="list-style-type: none"> • Set by instruction | |

| | |
|---|--|
| WRELO ^{Notes 2,3} | Wait cancellation |
| 0 | Do not cancel wait |
| 1 | Cancel wait. This setting is automatically cleared after wait is canceled. |
| When WRELO is set (wait canceled) during the wait period at the ninth clock pulse in the transmission status (TRC0 = 1), the SDA0 line goes into the high impedance state (TRC0 = 0). | |
| Condition for clearing (WRELO = 0) | |
| <ul style="list-style-type: none"> • Automatically cleared after execution • Reset | |
| Condition for setting (WRELO = 1) | |
| <ul style="list-style-type: none"> • Set by instruction | |

- Notes 1.** The IICS0 register, the STCF0 and IICBSY bits of the IICF0 register, and the CLD0 and DAD0 bits of the IICCL0 register are reset.
- 2.** The signals of these bits are invalid while the IICE0 bit is 0.
- 3.** When the LRELO and WRELO bits are read, 0 is always read.

Caution If the operation of I²C is enabled (IICE0 = 1) when the SCL0 line is high level, the SDA0 line is low level, and the digital filter is turned on (DFC0 of the IICCL0 register = 1), a start condition will be inadvertently detected immediately. In this case, set (1) the LRELO bit by using a 1-bit memory manipulation instruction immediately after enabling operation of I²C (IICE0 = 1).

Figure 15-5. Format of IIC Control Register 0 (IICC0) (2/4)

| | | |
|---|--|--|
| SPIE0 ^{Note 1} | Enable/disable generation of interrupt request when stop condition is detected | |
| 0 | Disable | |
| 1 | Enable | |
| Condition for clearing (SPIE0 = 0) | | Condition for setting (SPIE0 = 1) |
| <ul style="list-style-type: none"> • Cleared by instruction • Reset | | <ul style="list-style-type: none"> • Set by instruction |

| | | |
|---|---|--|
| WTIM0 ^{Note 1} | 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 (WTIM0 = 0) | | Condition for setting (WTIM0 = 1) |
| <ul style="list-style-type: none"> • Cleared by instruction • Reset | | <ul style="list-style-type: none"> • Set by instruction |

| | | |
|---|--|--|
| ACKE0 ^{Notes 1, 2} | Acknowledgment control | |
| 0 | Disable acknowledgment. | |
| 1 | Enable acknowledgment. During the ninth clock period, the SDA0 line is set to low level. | |
| Condition for clearing (ACKE0 = 0) | | Condition for setting (ACKE0 = 1) |
| <ul style="list-style-type: none"> • Cleared by instruction • Reset | | <ul style="list-style-type: none"> • Set by instruction |

Notes 1. This flag's signal is invalid when IICE0 = 0.

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 acknowledge is generated regardless of the set value.

Figure 15-5. Format of IIC Control Register 0 (IICC0) (3/4)

| STT0 ^{Note} | Start condition trigger | | | | |
|--|---|-----------------------------------|----------------------------------|---|--|
| 0 | Do not generate a start condition. | | | | |
| 1 | <p>When bus is released (in standby state, when IICBSY = 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"> When communication reservation function is enabled (IICRSV = 0) Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released. When communication reservation function is disabled (IICRSV = 1) Even if this bit is set (1), the STT0 is cleared and the STT0 clear flag (STCF) is set (1). No start condition is generated. <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"> For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when ACKE0 has been cleared to 0 and slave has been notified of final reception. 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. Cannot be set to 1 at the same time as stop condition trigger (SPT0). Setting the STT0 bit to 1 and then setting it again before it is cleared to 0 is prohibited. | | | | | |
| <table border="1"> <thead> <tr> <th>Condition for clearing (STT0 = 0)</th> <th>Condition for setting (STT0 = 1)</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> Cleared by setting SST0 bit to 1 while communication reservation is prohibited. Cleared by loss in arbitration Cleared after start condition is generated by master device Cleared by LRELO = 1 (exit from communications) When IICE0 = 0 (operation stop) Reset </td> <td> <ul style="list-style-type: none"> Set by instruction </td> </tr> </tbody> </table> | | Condition for clearing (STT0 = 0) | Condition for setting (STT0 = 1) | <ul style="list-style-type: none"> Cleared by setting SST0 bit to 1 while communication reservation is prohibited. Cleared by loss in arbitration Cleared after start condition is generated by master device Cleared by LRELO = 1 (exit from communications) When IICE0 = 0 (operation stop) Reset | <ul style="list-style-type: none"> Set by instruction |
| Condition for clearing (STT0 = 0) | Condition for setting (STT0 = 1) | | | | |
| <ul style="list-style-type: none"> Cleared by setting SST0 bit to 1 while communication reservation is prohibited. Cleared by loss in arbitration Cleared after start condition is generated by master device Cleared by LRELO = 1 (exit from communications) When IICE0 = 0 (operation stop) Reset | <ul style="list-style-type: none"> Set by instruction | | | | |

Note The signal of this bit is invalid while IICE0 is 0.

Remarks 1. Bit 1 (STT0) becomes 0 when it is read after data setting.

2. IICRSV: Bit 0 of IIC flag register (IICF0)

STCF: Bit 7 of IIC flag register (IICF0)

Figure 15-5. Format of IIC Control Register 0 (IICC0) (4/4)

| SPT0 | 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"> • For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when ACKE0 has been cleared to 0 and slave has been notified of final reception. • 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. • Cannot be set to 1 at the same time as start condition trigger (STT0). • SPT0 bit can be set to 1 only when in master mode. • When WTIM0 has been cleared to 0, if SPT0 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. WTIM0 should be changed from 0 to 1 during the wait period following the output of eight clocks, and SPT0 bit should be set to 1 during the wait period that follows the output of the ninth clock. • Setting SPT0 bit to 1 and then setting it again before it is cleared to 0 is prohibited. | | | | | |
| <table border="1"> <thead> <tr> <th>Condition for clearing (SPT0 = 0)</th> <th>Condition for setting (SPT0 = 1)</th> </tr> </thead> <tbody> <tr> <td> <ul style="list-style-type: none"> • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • Cleared by LRELO = 1 (exit from communications) • When IICE0 = 0 (operation stop) • Reset </td> <td> <ul style="list-style-type: none"> • Set by instruction </td> </tr> </tbody> </table> | | Condition for clearing (SPT0 = 0) | Condition for setting (SPT0 = 1) | <ul style="list-style-type: none"> • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • Cleared by LRELO = 1 (exit from communications) • When IICE0 = 0 (operation stop) • Reset | <ul style="list-style-type: none"> • Set by instruction |
| Condition for clearing (SPT0 = 0) | Condition for setting (SPT0 = 1) | | | | |
| <ul style="list-style-type: none"> • Cleared by loss in arbitration • Automatically cleared after stop condition is detected • Cleared by LRELO = 1 (exit from communications) • When IICE0 = 0 (operation stop) • Reset | <ul style="list-style-type: none"> • Set by instruction | | | | |

Caution When bit 3 (TRC0) of the IIC status register 0 (IICS0) is set to 1 (transmission status), bit 5 (WRELO) of the IICC0 register is set to 1 during the ninth clock and wait is canceled, after which the TRC0 bit is cleared (reception status) and the SDA0 line is set to high impedance. Release the wait performed while the TRC bit is 1 (transmission status) by writing to the IIC shift register.

Remark Bit 0 (SPT0) becomes 0 when it is read after data setting.

(2) IIC status register 0 (IICS0)

This register indicates the status of I²C.

IICS0 is read by a 1-bit or 8-bit memory manipulation instruction only when STT0 = 1 and during the wait period.

Reset signal generation clears IICS0 to 00H.

Caution If data is read from IICS0 register, a wait cycle is generated. Do not read data from IICS0 register when the peripheral hardware clock (f_{PRS}) is stopped. For details, see CHAPTER 27 CAUTIONS FOR WAIT.

Figure 15-6. Format of IIC Status Register 0 (IICS0) (1/3)

Address: FFAAH After reset: 00H R

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

| | | | | | | | | |
|-------|-------|------|------|------|------|-------|------|------|
| IICS0 | MSTS0 | ALD0 | EXC0 | COI0 | TRC0 | ACKD0 | STD0 | SPD0 |
|-------|-------|------|------|------|------|-------|------|------|

| | | |
|--|---|---|
| MSTS0 | Master device status | |
| 0 | Slave device status or communication standby status | |
| 1 | Master device communication status | |
| Condition for clearing (MSTS0 = 0) | | Condition for setting (MSTS0 = 1) |
| <ul style="list-style-type: none"> • When a stop condition is detected • When ALD0 = 1 (arbitration loss) • Cleared by LREL0 = 1 (exit from communications) • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When a start condition is generated |

| | | |
|---|--|--|
| ALD0 | 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". MSTS0 bit is cleared. | |
| Condition for clearing (ALD0 = 0) | | Condition for setting (ALD0 = 1) |
| <ul style="list-style-type: none"> • Automatically cleared after IICS0 register is read^{Note} • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When the arbitration result is a "loss". |

| | | |
|--|---------------------------------------|---|
| EXC0 | Detection of extension code reception | |
| 0 | Extension code was not received. | |
| 1 | Extension code was received. | |
| Condition for clearing (EXC0 = 0) | | Condition for setting (EXC0 = 1) |
| <ul style="list-style-type: none"> • When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 (exit from communications) • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When the higher four bits of the received address data is either "0000" or "1111" (set at the rising edge of the eighth clock). |

Note This register is also cleared when a 1-bit memory manipulation instruction is executed for bits other than IICS0 register. Therefore, when using the ALD0 bit, read the data of this bit before the data of the other bits.

Remark LREL0: Bit 6 of IIC control register 0 (IICC0)

IICE0: Bit 7 of IIC control register 0 (IICC0)

Figure 15-6. Format of IIC Status Register 0 (IICS0) (2/3)

| COI0 | Detection of matching addresses | |
|--|---------------------------------|---|
| 0 | Addresses do not match. | |
| 1 | Addresses match. | |
| Condition for clearing (COI0 = 0) | | Condition for setting (COI0 = 1) |
| <ul style="list-style-type: none"> • When a start condition is detected • When a stop condition is detected • Cleared by LREL0 = 1 (exit from communications) • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When the received address matches the local address (slave address register 0 (SVA0)) (set at the rising edge of the eighth clock). |

| TRC0 | Detection of transmit/receive status | |
|--|--|---|
| 0 | Receive status (other than transmit status). The SDA0 line is set for high impedance. | |
| 1 | Transmit status. The value in the SO0 latch is enabled for output to the SDA0 line (valid starting at the falling edge of the first byte's ninth clock). | |
| Condition for clearing (TRC0 = 0) | | Condition for setting (TRC0 = 1) |
| <Both master and slave> <ul style="list-style-type: none"> • When a stop condition is detected • Cleared by LREL0 = 1 (exit from communications) • When the IICE0 bit changes from 1 to 0 (operation stop) • Cleared by WREL0 = 1^{Note} (wait cancel) • When the ALD0 bit changes from 0 to 1 (arbitration loss) • Reset • When not used for communication (MSTS0, EXC0, COI0 = 0) <Master> <ul style="list-style-type: none"> • When "1" is output to the first byte's LSB (transfer direction specification bit) <Slave> <ul style="list-style-type: none"> • When a start condition is detected • When "0" is input to the first byte's LSB (transfer direction specification bit) | | <Master> <ul style="list-style-type: none"> • When a start condition is generated • When 0 (master transmission) is output to the LSB (transfer direction specification bit) of the first byte (during address transfer) <Slave> <ul style="list-style-type: none"> • When 1 (slave transmission) is input to the LSB (transfer direction specification bit) of the first byte from the master (during address transfer) |

Note When bit 3 (TRC0) of the IIC status register 0 (IICS0) is set to 1 (transmission status), bit 5 (WREL0) of the IIC control register 0 (IICC0) is set to 1 during the ninth clock and wait is canceled, after which the TRC0 bit is cleared (reception status) and the SDA0 line is set to high impedance. Release the wait performed while TRC0 bit is 1 (transmission status) by writing to the IIC shift register.

Remark LREL0: Bit 6 of IIC control register 0 (IICC0)
IICE0: Bit 7 of IIC control register 0 (IICC0)

Figure 15-6. Format of IIC Status Register 0 (IICS0) (3/3)

| ACKD0 | Detection of acknowledge ($\overline{\text{ACK}}$) | |
|---|--|--|
| 0 | Acknowledge was not detected. | |
| 1 | Acknowledge was detected. | |
| Condition for clearing (ACKD0 = 0) | | Condition for setting (ACKD0 = 1) |
| <ul style="list-style-type: none"> • When a stop condition is detected • At the rising edge of the next byte's first clock • Cleared by LREL0 = 1 (exit from communications) • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • After the SDA0 line is set to low level at the rising edge of ninth clock of SCL0 line |

| STD0 | 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 (STD0 = 0) | | Condition for setting (STD0 = 1) |
| <ul style="list-style-type: none"> • When a stop condition is detected • At the rising edge of the next byte's first clock following address transfer • Cleared by LREL0 = 1 (exit from communications) • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When a start condition is detected |

| SPD0 | 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 (SPD0 = 0) | | Condition for setting (SPD0 = 1) |
| <ul style="list-style-type: none"> • At the rising edge of the address transfer byte's first clock following setting of this bit and detection of a start condition • When IICE0 changes from 1 to 0 (operation stop) • Reset | | <ul style="list-style-type: none"> • When a stop condition is detected |

Remark LREL0: Bit 6 of IIC control register 0 (IICC0)
IICE0: Bit 7 of IIC control register 0 (IICC0)

(3) IIC flag register 0 (IICF0)

This register sets the operation mode of I²C and indicates the status of the I²C bus.

This register can be set by a 1-bit or 8-bit memory manipulation instruction. However, the STT0 clear flag (STCF) and I²C bus status flag (IICBSY) are read-only.

The IICRSV bit can be used to enable/disable the communication reservation function.

The STCEN bit can be used to set the initial value of the IICBSY bit.

The IICRSV and STCEN bits can be written only when the operation of I²C is disabled (bit 7 (IICE0) of the IIC control register 0 (IICC0) = 0). When operation is enabled, the IICF0 register can be read.

Reset signal generation clears this register to 00H.

Figure 15-7. Format of IIC Flag Register 0 (IICF0)

Address: FFABH After reset: 00H R/W^{Note}

| | | | | | | | | |
|--------|------|--------|---|---|---|---|-------|--------|
| Symbol | <7> | <6> | 5 | 4 | 3 | 2 | <1> | <0> |
| IICF0 | STCF | IICBSY | 0 | 0 | 0 | 0 | STCEN | IICRSV |

| | | |
|---|--|--|
| STCF | STT0 clear flag | |
| 0 | Generate start condition | |
| 1 | Start condition generation unsuccessful: clear STT0 flag | |
| Condition for clearing (STCF = 0) | | Condition for setting (STCF = 1) |
| <ul style="list-style-type: none"> Cleared by STT0 = 1 When IICE0 = 0 (operation stop) Reset | | <ul style="list-style-type: none"> Generating start condition unsuccessful and STT0 bit cleared to 0 when communication reservation is disabled (IICRSV = 1). |

| | | |
|---|--|---|
| IICBSY | I ² C bus status flag | |
| 0 | Bus release status (communication initial status when STCEN = 1) | |
| 1 | Bus communication status (communication initial status when STCEN = 0) | |
| Condition for clearing (IICBSY = 0) | | Condition for setting (IICBSY = 1) |
| <ul style="list-style-type: none"> Detection of stop condition When IICE0 = 0 (operation stop) Reset | | <ul style="list-style-type: none"> Detection of start condition Setting of IICE0 bit when STCEN = 0 |

| | | |
|---|--|--|
| STCEN | Initial start enable trigger | |
| 0 | After operation is enabled (IICE0 = 1), enable generation of a start condition upon detection of a stop condition. | |
| 1 | After operation is enabled (IICE0 = 1), enable generation of a start condition without detecting a stop condition. | |
| Condition for clearing (STCEN = 0) | | Condition for setting (STCEN = 1) |
| <ul style="list-style-type: none"> Detection of start condition Reset | | <ul style="list-style-type: none"> Set by instruction |

| | | |
|---|--|--|
| IICRSV | Communication reservation function disable bit | |
| 0 | Enable communication reservation | |
| 1 | Disable communication reservation | |
| Condition for clearing (IICRSV = 0) | | Condition for setting (IICRSV = 1) |
| <ul style="list-style-type: none"> Cleared by instruction Reset | | <ul style="list-style-type: none"> Set by instruction |

Note Bits 6 and 7 are read-only.

- Cautions**
1. Write to STCEN bit only when the operation is stopped (IICE0 = 0).
 2. As the bus release status (IICBSY = 0) is recognized regardless of the actual bus status when STCEN = 1, when generating the first start condition (STT0 = 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 IICRSV bit only when the operation is stopped (IICE0 = 0).

Remark STT0: Bit 1 of IIC control register 0 (IICC0)
IICE0: Bit 7 of IIC control register 0 (IICC0)

(4) IIC clock selection register 0 (IICCL0)

This register is used to set the transfer clock for the I²C bus.

IICCL0 is set by a 1-bit or 8-bit memory manipulation instruction. However, the CLD0 and DAD0 bits are read-only.

The SMC0, CL01, and CL00 bits are set in combination with bit 0 (CLX0) of IIC function expansion register 0 (IICX0) (see **15.3 (6) I²C transfer clock setting method**).

Set IICCL0 while bit 7 (IICE0) of IIC control register 0 (IICC0) is 0.

Reset signal generation clears IICCL0 to 00H.

Figure 15-8. Format of IIC Clock Selection Register 0 (IICCL0)

Address: FFA8H After reset: 00H R/W^{Note}

| Symbol | 7 | 6 | <5> | <4> | <3> | <2> | 1 | 0 |
|--------|---|---|------|------|------|------|------|------|
| IICCL0 | 0 | 0 | CLD0 | DAD0 | SMC0 | DFC0 | CL01 | CL00 |

| CLD0 | Detection of SCL0 pin level (valid only when IICE0 = 1) |
|---|---|
| 0 | The SCL0 pin was detected at low level. |
| 1 | The SCL0 pin was detected at high level. |
| Condition for clearing (CLD0 = 0) | |
| <ul style="list-style-type: none"> • When the SCL0 pin is at low level • When IICE0 = 0 (operation stop) • Reset | |
| Condition for setting (CLD0 = 1) | |
| <ul style="list-style-type: none"> • When the SCL0 pin is at high level | |

| DAD0 | Detection of SDA0 pin level (valid only when IICE0 = 1) |
|---|---|
| 0 | The SDA0 pin was detected at low level. |
| 1 | The SDA0 pin was detected at high level. |
| Condition for clearing (DAD0 = 0) | |
| <ul style="list-style-type: none"> • When the SDA0 pin is at low level • When IICE0 = 0 (operation stop) • Reset | |
| Condition for setting (DAD0 = 1) | |
| <ul style="list-style-type: none"> • When the SDA0 pin is at high level | |

| SMC0 | Operation mode switching |
|------|------------------------------|
| 0 | Operates in standard mode. |
| 1 | Operates in high-speed mode. |

| DFC0 | Digital filter operation control |
|--|----------------------------------|
| 0 | Digital filter off. |
| 1 | Digital filter on. |
| Digital filter can be used only in high-speed mode. | |
| In high-speed mode, the transfer clock does not vary regardless of DFC0 bit set (1)/clear (0). | |
| The digital filter is used for noise elimination in high-speed mode. | |

Note Bits 4 and 5 are read-only.

Remark IICE0: Bit 7 of IIC control register 0 (IICC0)

(5) IIC function expansion register 0 (IICX0)

This register sets the function expansion of I²C.

IICX0 is set by a 1-bit or 8-bit memory manipulation instruction. The CLX0 bit is set in combination with bits 3, 1, and 0 (SMC0, CL01, and CL00) of IIC clock selection register 0 (IICCL0) (see **15.3 (6) I²C transfer clock setting method**).

Set IICX0 while bit 7 (IICE0) of IIC control register 0 (IICC0) is 0.

Reset signal generation clears IICX0 to 00H.

Figure 15-9. Format of IIC Function Expansion Register 0 (IICX0)

| Address: FFA9H | After reset: 00H | R/W | | | | | | | |
|----------------|------------------|-----|---|---|---|---|---|------|--|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> | |
| IICX0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | CLX0 | |

(6) I²C transfer clock setting method

The I²C transfer clock frequency (f_{SCL}) is calculated using the following expression.

$$f_{SCL} = 1/(m \times T + t_R + t_F)$$

$m = 12, 18, 24, 44, 66, 86$ (see **Table 15-2 Selection Clock Setting**)

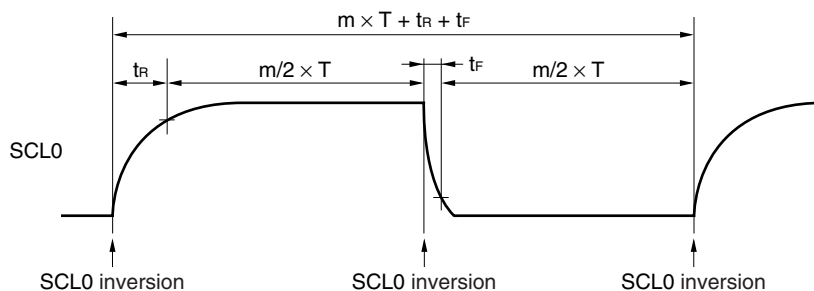
T : $1/f_w$

t_R : SCL0 rise time

t_F : SCL0 fall time

For example, the I²C transfer clock frequency (f_{SCL}) when $f_w = f_{PRS}/2 = 4.19$ MHz, $m = 86$, $t_R = 200$ ns, and $t_F = 50$ ns is calculated using following expression.

$$f_{SCL} = 1/(86 \times 238.7 \text{ ns} + 200 \text{ ns} + 50 \text{ ns}) \cong 48.1 \text{ kHz}$$



The selection clock is set using a combination of bits 3, 1, and 0 (SMC0, CL01, and CL00) of IIC clock selection register 0 (IICCL0) and bit 0 (CLX0) of IIC function expansion register 0 (IICX0).

Table 15-2. Selection Clock Setting

| IICX0 | IICCL0 | | | Selection Clock (fw) ^{Note} | Transfer Clock (fw/m) | Settable Selection Clock (fw) Range | Operation Mode |
|-------|--------|-------|-------|---|--------------------------|--|-----------------------------------|
| | Bit 0 | Bit 3 | Bit 1 | | | | |
| CLX0 | SMC0 | CL01 | CL00 | | | | |
| 0 | 0 | 0 | 0 | $f_{PRS}/2$ | fw/44 | 2.00 to 4.19 MHz | Normal mode (SMC0 bit = 0) |
| 0 | 0 | 0 | 1 | $f_{PRS}/2$ | fw/86 | 4.19 to 8.38 MHz | |
| 0 | 0 | 1 | 0 | $f_{PRS}/4$ | fw/86 | | |
| 0 | 0 | 1 | 1 | Setting prohibited | | | |
| 0 | 1 | 0 | × | $f_{PRS}/2$ | fw/24 | 4.00 to 8.38 MHz | High-speed mode (SMC0 bit = 1) |
| 0 | 1 | 1 | 0 | $f_{PRS}/4$ | fw/24 | | |
| 0 | 1 | 1 | 1 | Setting prohibited | | | |
| 1 | 0 | × | × | | | | |
| 1 | 1 | 0 | × | $f_{PRS}/2$ | fw/12 | 4.00 to 4.19 MHz | High-speed mode (SMC0 bit = 1) |
| 1 | 1 | 1 | 0 | $f_{PRS}/4$ | fw/12 | | |
| 1 | 1 | 1 | 1 | Setting prohibited | | | |

Note If the peripheral hardware clock (f_{PRS}) operates on the internal high-speed oscillation clock (f_{XH}) ($XSEL = 0$), set CLX0, SMC0, CL01 and CL00 as follows.

| IICX0 | IICCL0 | | | Selection Clock (fw) | Transfer Clock (fw/m) | Settable Selection Clock (fw) Range | Operation Mode |
|-------|--------|-------|-------|-------------------------|--------------------------|-------------------------------------|-----------------------------------|
| | Bit 0 | Bit 3 | Bit 1 | | | | |
| CLX0 | SMC0 | CL01 | CL00 | | | | |
| 0 | 0 | 0 | 0 | $f_{PRS}/2$ | fw/44 | 3.8 MHz to 4.2 MHz | Normal mode (SMC0 bit = 0) |
| 0 | 1 | 0 | × | $f_{PRS}/2$ | fw/24 | | High-speed mode (SMC0 bit = 1) |

Caution Determine the transfer clock frequency of I²C by using CLX0, SMC0, CL01, and CL00 before enabling the operation (by setting bit 7 (IICE0) of IIC control register 0 (IICC0) to 1). To change the transfer clock frequency, clear IICE0 once to 0.

- Remarks**
1. ×: don't care
 2. f_{PRS} : Peripheral hardware clock frequency

(7) Port mode register 6 (PM6)

This register sets the input/output of port 6 in 1-bit units.

When using the P60/SCL0 pin as clock I/O and the P61/SDA0 pin as serial data I/O, clear PM60 and PM61, and the output latches of P60 and P61 to 0.

Set IICE0 (bit 7 of IIC control register 0 (IICC0)) to 1 before setting the output mode because the P60/SCL0 and P61/SDA0 pins output a low level (fixed) when IICE0 is 0.

PM6 is set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets PM6 to FFH.

Figure 15-10. Format of Port Mode Register 6 (PM6)

Address: FF26H 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) |

15.4 I²C Bus Mode Functions

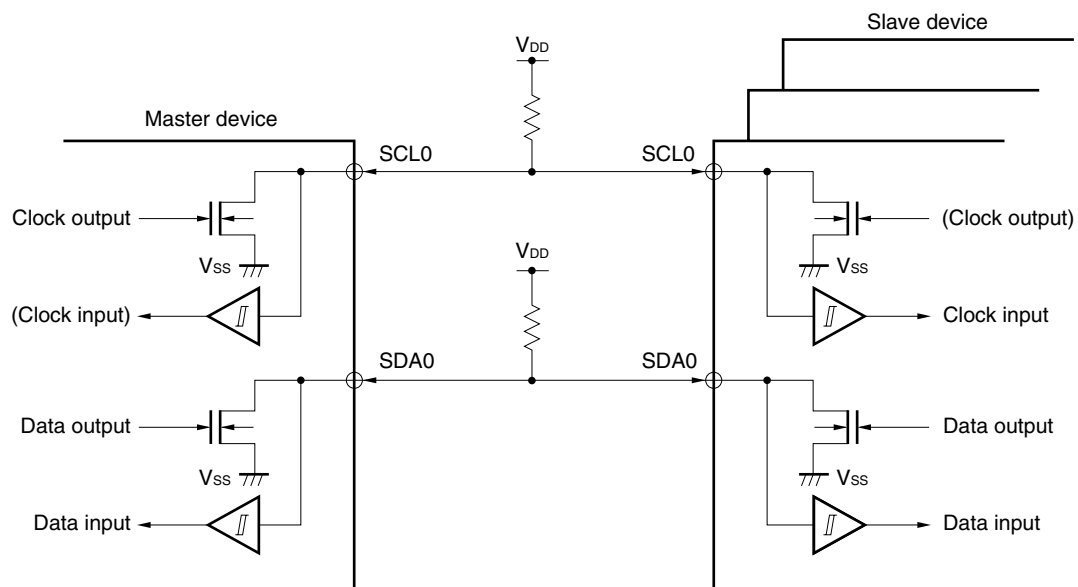
15.4.1 Pin configuration

The serial clock pin (SCL0) and serial data bus pin (SDA0) are configured as follows.

- (1) SCL0..... 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) SDA0 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.

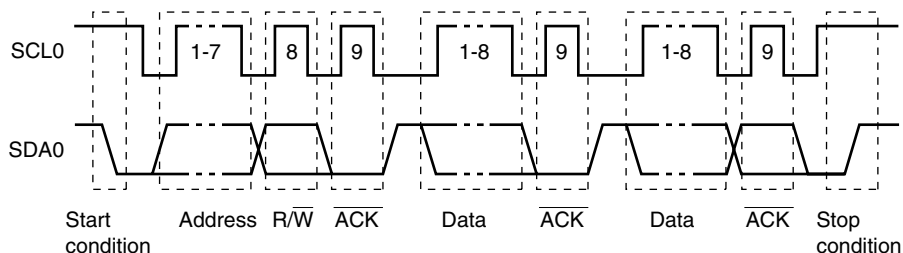
Figure 15-11. Pin Configuration Diagram



15.5 I²C Bus Definitions and Control Methods

The following section describes the I²C bus's serial data communication format and the signals used by the I²C bus. Figure 15-12 shows the transfer timing for the "start condition", "address", "data", and "stop condition" output via the I²C bus's serial data bus.

Figure 15-12. I²C Bus Serial Data Transfer Timing



The master device generates the start condition, slave address, and stop condition.

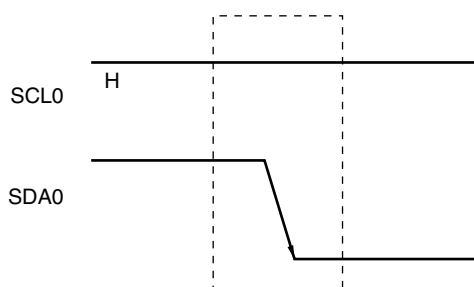
The acknowledge ($\overline{\text{ACK}}$) can be generated by either the master or slave device (normally, it is output by the device that receives 8-bit data).

The serial clock (SCL0) is continuously output by the master device. However, in the slave device, the SCL0's low level period can be extended and a wait can be inserted.

15.5.1 Start conditions

A start condition is met when the SCL0 pin is at high level and the SDA0 pin changes from high level to low level. The start conditions for the SCL0 pin and SDA0 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 15-13. Start Conditions



A start condition is output when bit 1 (STT0) of IIC control register 0 (IICC0) is set (to 1) after a stop condition has been detected (SPD0: Bit 0 = 1 in IIC status register 0 (IICS0)). When a start condition is detected, bit 1 (STD0) of IICS0 is set (to 1).

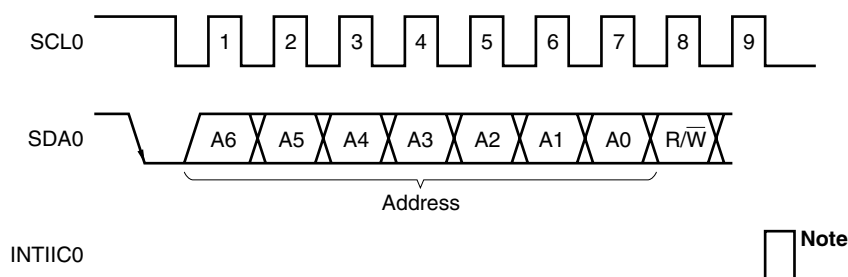
15.5.2 Addresses

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 slave address register 0 (SVA0). If the address data matches the SVA0 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 15-14. Address



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

The slave address and the eighth bit, which specifies the transfer direction as described in **15.5.3 Transfer direction specification** below, are together written to IIC shift register 0 (IIC0) and are then output. Received addresses are written to IIC0.

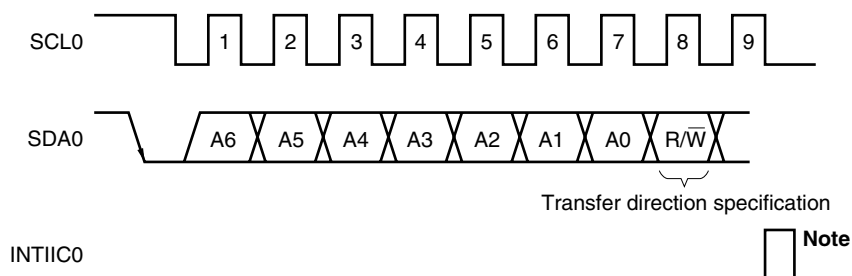
The slave address is assigned to the higher 7 bits of IIC0 register.

15.5.3 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 15-15. Transfer Direction Specification



Note INTIIC0 is not issued if data other than a local address or extension code is received during slave device operation.

15.5.4 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 (ACKD0) of IIC status register 0 (IICS0).

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.

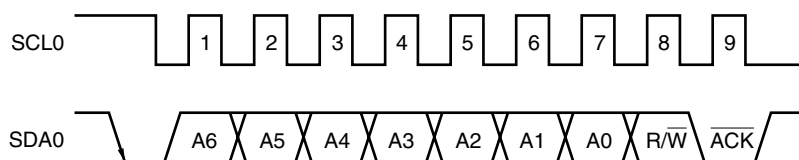
To generate $\overline{\text{ACK}}$, the reception side makes the SDA0 line low at the ninth clock (indicating normal reception).

Automatic generation of $\overline{\text{ACK}}$ is enabled by setting bit 2 (ACKE0) of IIC control register 0 (IICC0) to 1. Bit 3 (TRC0) of the IICS0 register is set by the data of the eighth bit that follows 7-bit address information. Usually, set ACEK0 bit to 1 for reception (TRC0 = 0).

If a slave can receive no more data during reception (TRC0 = 0) or does not require the next data item, then the slave must inform the master, by clearing ACEK0 bit to 0, that it will not receive any more data.

When the master does not require the next data item during reception (TRC0 = 0), it must clear ACEK0 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 15-16. $\overline{\text{ACK}}$



When the local address is received, $\overline{\text{ACK}}$ is automatically generated, regardless of the value of ACEK0 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 ACEK0 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.

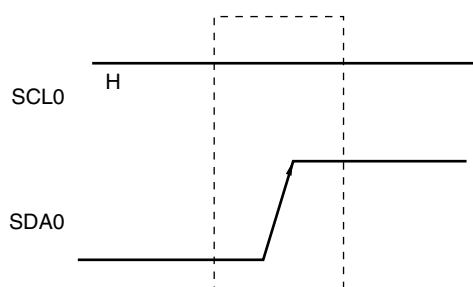
- When 8-clock wait state is selected (bit 3 (WTIM0) of IICC0 register = 0):
By setting ACEK0 bit to 1 before releasing the wait state, $\overline{\text{ACK}}$ is generated at the falling edge of the eighth clock of the SCL0 pin.
- When 9-clock wait state is selected (bit 3 (WTIM0) of IICC0 register = 1):
 $\overline{\text{ACK}}$ is generated by setting ACEK0 bit to 1 in advance.

15.5.5 Stop condition

When the SCL0 pin is at high level, changing the SDA0 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 15-17. Stop Condition



A stop condition is generated when bit 0 (SPT0) of IIC control register 0 (IICC0) is set to 1. When the stop condition is detected, bit 0 (SPD0) of IIC status register 0 (IICS0) is set to 1 and INTIIC0 is generated when bit 4 (SPIE0) of IICC0 register is set to 1.

15.5.6 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 SCL0 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 15-18. 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 ACKE0 = 1)

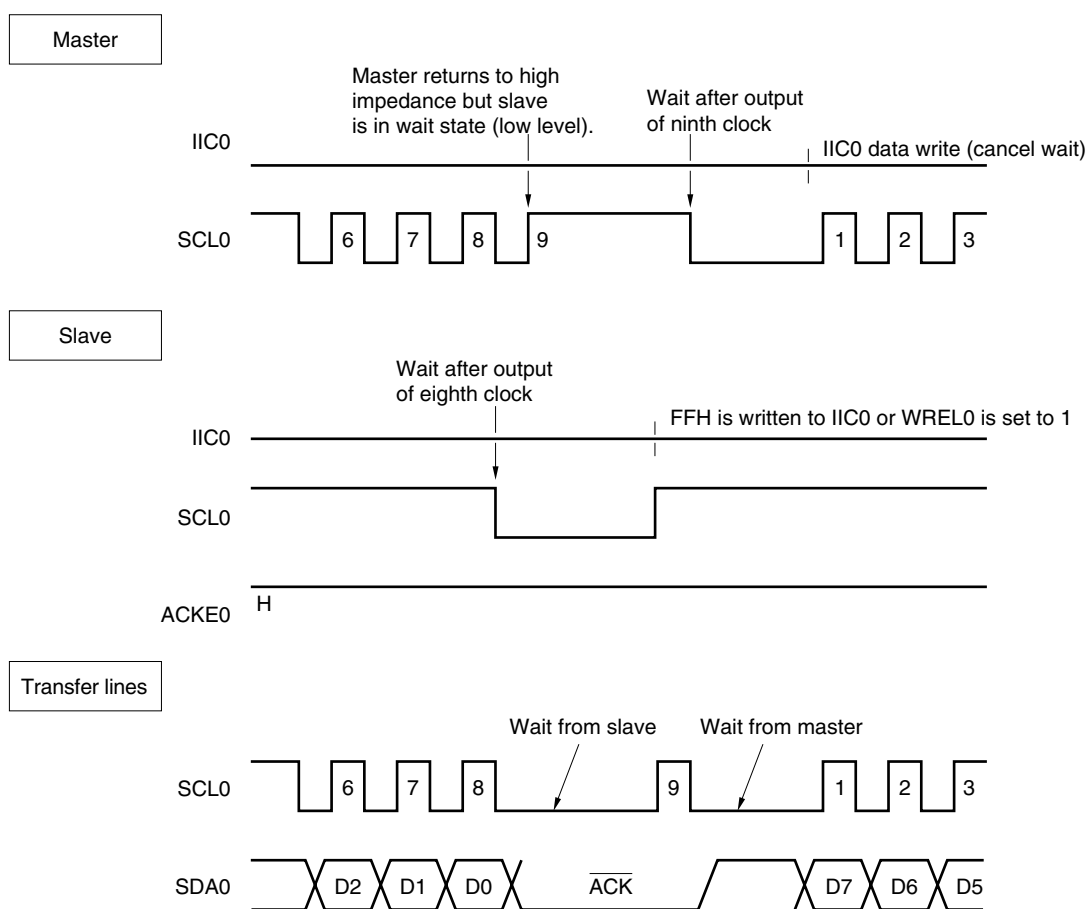
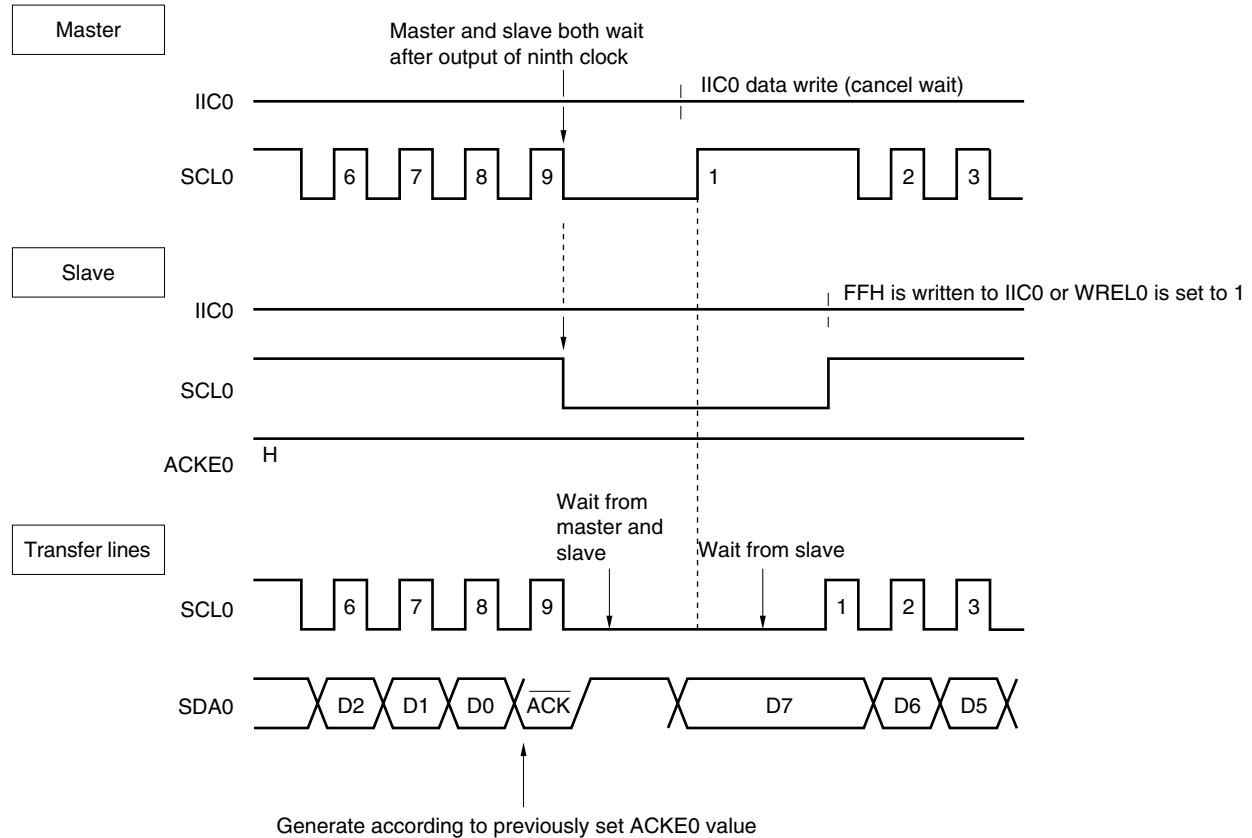


Figure 15-18. Wait (2/2)

(2) When master and slave devices both have a nine-clock wait
(master transmits, slave receives, and ACKE0 = 1)



Remark ACKE0: Bit 2 of IIC control register 0 (IICC0)

WRELO: Bit 5 of IIC control register 0 (IICC0)

A wait may be automatically generated depending on the setting of bit 3 (WTIM0) of IIC control register 0 (IICC0).

Normally, the receiving side cancels the wait state when bit 5 (WRELO) of IICC0 register is set to 1 or when FFH is written to IIC shift register 0 (IIC0), and the transmitting side cancels the wait state when data is written to IIC0 register.

The master device can also cancel the wait state via either of the following methods.

- By setting bit 1 (STT0) of IICC0 register to 1
- By setting bit 0 (SPT0) of IICC0 register to 1

15.5.7 Canceling wait

The I²C usually cancels a wait state by the following processing.

- Writing data to IIC shift register 0 (IIC0)
- Setting bit 5 (WRELO) of IIC control register 0 (IICC0) (canceling wait)
- Setting bit 1 (STT0) of IIC0 register (generating start condition)^{Note}
- Setting bit 0 (SPT0) of IIC0 register (generating stop condition)^{Note}

Note Master only

When the above wait canceling processing is executed, the I²C cancels the wait state and communication is resumed.

To cancel a wait state and transmit data (including addresses), write the data to IIC0 register.

To receive data after canceling a wait state, or to complete data transmission, set bit 5 (WRELO) of the IIC0 control register 0 (IICC0) to 1.

To generate a restart condition after canceling a wait state, set bit 1 (STT0) of IICC0 register to 1.

To generate a stop condition after canceling a wait state, set bit 0 (SPT0) of IICC0 register to 1.

Execute the canceling processing only once for one wait state.

If, for example, data is written to IIC0 register after canceling a wait state by setting WRELO bit to 1, an incorrect value may be output to SDA0 line because the timing for changing the SDA0 line conflicts with the timing for writing IIC0 register.

In addition to the above, communication is stopped if IICE0 bit is cleared to 0 when communication has been aborted, so that the wait state can be canceled.

If the I²C bus has deadlocked due to noise, processing is saved from communication by setting bit 6 (LRELO) of IICC0 register, so that the wait state can be canceled.

15.5.8 Interrupt request (INTIIC0) generation timing and wait control

The setting of bit 3 (WTIM0) of IIC control register 0 (IICC0) determines the timing by which INTIIC0 is generated and the corresponding wait control, as shown in Table 15-3.

Table 15-3. INTIIC0 Generation Timing and Wait Control

| WTIM0 | During Slave Device Operation | | | During Master Device Operation | | |
|-------|-------------------------------|---------------------|---------------------|--------------------------------|----------------|-------------------|
| | Address | Data Reception | Data Transmission | Address | Data Reception | Data Transmission |
| 0 | 9 ^{Notes 1, 2} | 8 ^{Note 2} | 8 ^{Note 2} | 9 | 8 | 8 |
| 1 | 9 ^{Notes 1, 2} | 9 ^{Note 2} | 9 ^{Note 2} | 9 | 9 | 9 |

Notes 1. The slave device's INTIIC0 signal and wait period occurs at the falling edge of the ninth clock only when there is a match with the address set to slave address register 0 (SVA0).

At this point, \overline{ACK} is generated regardless of the value set to bit 2 (ACKE0) of the IICC0 register. For a slave device that has received an extension code, INTIIC0 occurs at the falling edge of the eighth clock.

However, if the address does not match after restart, INTIIC0 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 slave address register 0 (SVA0) and extension code is not received, neither INTIIC0 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 WTIM0 bit.
- Master device operation: Interrupt and wait timing occur at the falling edge of the ninth clock regardless of the WTIM0 bit.

(2) During data reception

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(3) During data transmission

- Master/slave device operation: Interrupt and wait timing are determined according to the WTIM0 bit.

(4) Wait cancellation method

The four wait cancellation methods are as follows.

- Writing data to IIC shift register 0 (IIC0)
- Setting bit 5 (WRELO) of IIC control register 0 (IICC0) (canceling wait)
- Setting bit 1 (STT0) of IIC0 register (generating start condition)^{Note}
- Setting bit 0 (SPT0) of IIC0 register (generating stop condition)^{Note}

Note Master only.

When an 8-clock wait has been selected (WTIM0 = 0), the presence/absence of $\overline{\text{ACK}}$ generation must be determined prior to wait cancellation.

(5) Stop condition detection

INTIIC0 is generated when a stop condition is detected (only when SPIE0 = 1).

15.5.9 Address match detection method

In I²C 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 (INTIIC0) occurs when a local address has been set to slave address register 0 (SVA0) and when the address set to SVA0 matches the slave address sent by the master device, or when an extension code has been received.

15.5.10 Error detection

In I²C bus mode, the status of the serial data bus (SDA0) during data transmission is captured by IIC shift register 0 (IIC0) of the transmitting device, so the IIC0 data prior to transmission can be compared with the transmitted IIC0 data to enable detection of transmission errors. A transmission error is judged as having occurred when the compared data values do not match.

15.5.11 Extension code

- (1) When the higher 4 bits of the receive address are either “0000” or “1111”, the extension code reception flag (EXC0) is set to 1 for extension code reception and an interrupt request (INTIIC0) is issued at the falling edge of the eighth clock. The local address stored in slave address register 0 (SVA0) is not affected.
- (2) If “11110xx0” is set to SVA0 register by a 10-bit address transfer and “11110xx0” is transferred from the master device, the results are as follows. Note that INTIIC0 occurs at the falling edge of the eighth clock.
 - Higher four bits of data match: EXC0 = 1
 - Seven bits of data match: COI0 = 1

Remark EXC0: Bit 5 of IIC status register 0 (IICS0)
COI0: Bit 4 of IIC status register 0 (IICS0)

- (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 (LREL0) of the IIC control register 0 (IICC0) to 1 to set the standby mode for the next communication operation.

Table 15-4. Bit Definitions of Main Extension Code

| 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 (for address authentication) |
| 1 1 1 1 0 x x | 1 | 10-bit slave address specification (for read command issuance after address match) |

Remark For extension codes other than the above, refer to THE I²C-BUS SPECIFICATION published by NXP.

15.5.12 Arbitration

When several master devices simultaneously generate a start condition (when STT0 is set to 1 before STD0 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 (ALD0) in IIC status register 0 (IICS0) is set (1) via the timing by which the arbitration loss occurred, and the SCL0 and SDA0 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 ALD0 = 1 setting that has been made by software.

For details of interrupt request timing, see **15.5.17 Timing of I²C interrupt request (INTIIC0) occurrence.**

Remark STD0: Bit 1 of IIC status register 0 (IICS0)
STT0: Bit 1 of IIC control register 0 (IICC0)

Figure 15-19. Arbitration Timing Example

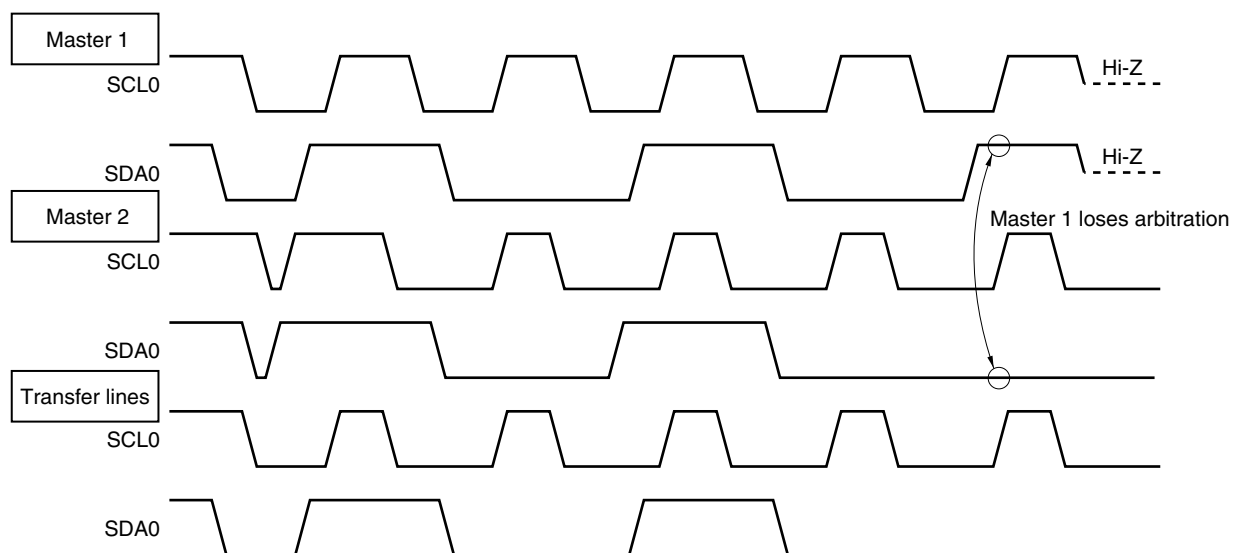


Table 15-5. Status During Arbitration and Interrupt Request Generation Timing

| Status During Arbitration | Interrupt Request Generation Timing |
|--|--|
| During address transmission | At falling edge of eighth or ninth clock following byte transfer ^{Note 1} |
| 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 SPIE0 = 1) ^{Note 2} |
| When data is at low level while attempting to generate a restart condition | At falling edge of eighth or ninth clock following byte transfer ^{Note 1} |
| When stop condition is detected while attempting to generate a restart condition | When stop condition is generated (when SPIE0 = 1) ^{Note 2} |
| When data is at low level while attempting to generate a stop condition | At falling edge of eighth or ninth clock following byte transfer ^{Note 1} |
| When SCL0 is at low level while attempting to generate a restart condition | |

- Notes 1.** When WTIM0 bit (bit 3 of IIC control register 0 (IICC0)) = 1, an interrupt request occurs at the falling edge of the ninth clock. When WTIM0 = 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 SPIE0 = 1 for master device operation.

Remark SPIE0: Bit 4 of IIC control register 0 (IICC0)

15.5.13 Wakeup function

The I²C bus slave function is a function that generates an interrupt request signal (INTIIC0) when a local address and extension code have been received.

This function makes processing more efficient by preventing unnecessary INTIIC0 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.

However, when a stop condition is detected, bit 4 (SPIE0) of IIC control register 0 (IICC0) is set regardless of the wakeup function, and this determines whether interrupt requests are enabled or disabled.

15.5.14 Communication reservation

(1) When communication reservation function is enabled (bit 0 (IICRSV) of IIC flag register 0 (IICF0) = 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 when bit 6 (LREL0) of IIC control register 0 (IICC0) was set to 1).

If bit 1 (STT0) of IICC0 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 IIC shift register 0 (IIC0) after bit 4 (SPIE0) of IICC0 was set to 1, and it was detected by generation of an interrupt request signal (INTIIC0) that the bus was released (detection of the stop condition), then the device automatically starts communication as the master. Data written to IIC0 before the stop condition is detected is invalid.

When STT0 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 MSTS0 bit (bit 7 of IIC status register 0 (IICS0)) after STT0 bit is set to 1 and the wait time elapses.

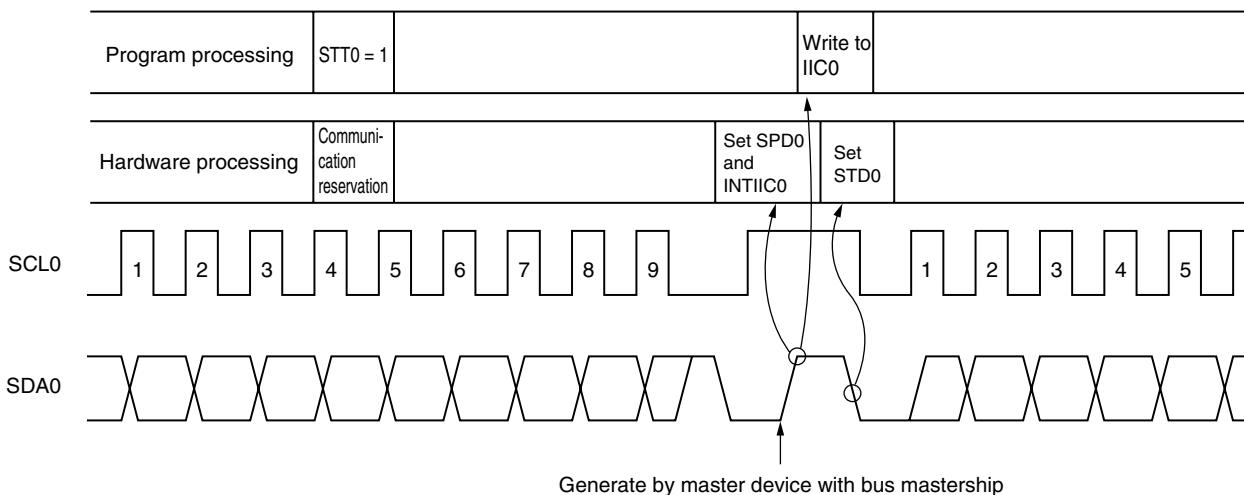
The wait periods, which should be set via software, are listed in Table 15-6.

Table 15-6. Wait Periods

| CLX0 | SMC0 | CL01 | CL00 | Wait Period |
|------|------|------|------|-------------|
| 0 | 0 | 0 | 0 | 46 clocks |
| 0 | 0 | 0 | 1 | 86 clocks |
| 0 | 0 | 1 | 0 | 172 clocks |
| 0 | 0 | 1 | 1 | 34 clocks |
| 0 | 1 | 0 | 0 | 30 clocks |
| 0 | 1 | 0 | 1 | |
| 0 | 1 | 1 | 0 | 60 clocks |
| 0 | 1 | 1 | 1 | 12 clocks |
| 1 | 1 | 0 | 0 | 18 clocks |
| 1 | 1 | 0 | 1 | |
| 1 | 1 | 1 | 0 | 36 clocks |

Figure 15-20 shows the communication reservation timing.

Figure 15-20. Communication Reservation Timing



- Remark** IIC0: IIC shift register 0
 STT0: Bit 1 of IIC control register 0 (IICC0)
 STD0: Bit 1 of IIC status register 0 (IICS0)
 SPD0: Bit 0 of IIC status register 0 (IICS0)

Communication reservations are accepted via the following timing. After bit 1 (STD0) of IIC status register 0 (IICS0) is set to 1, a communication reservation can be made by setting bit 1 (STT0) of IIC control register 0 (IICC0) to 1 before a stop condition is detected.

Figure 15-21. Timing for Accepting Communication Reservations

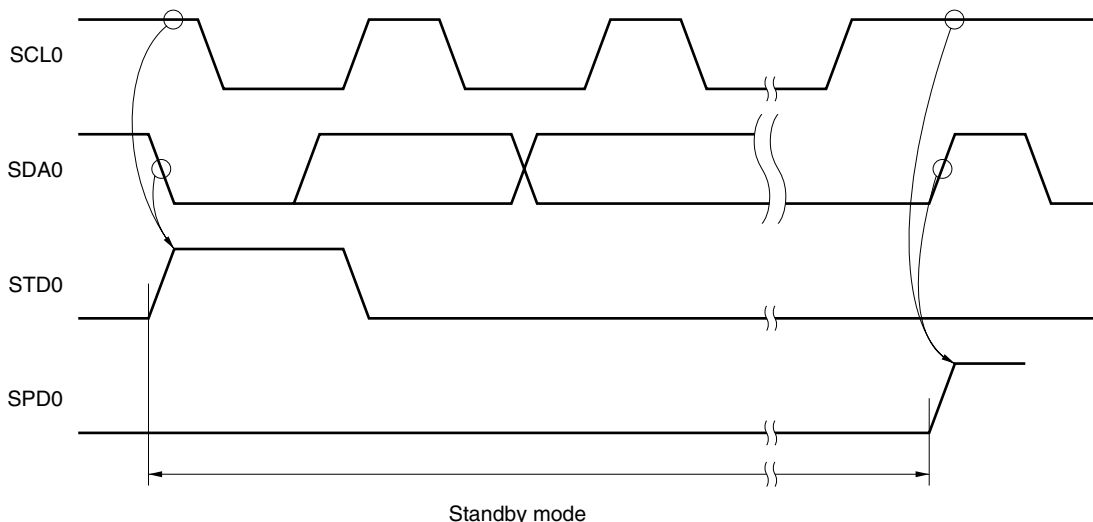
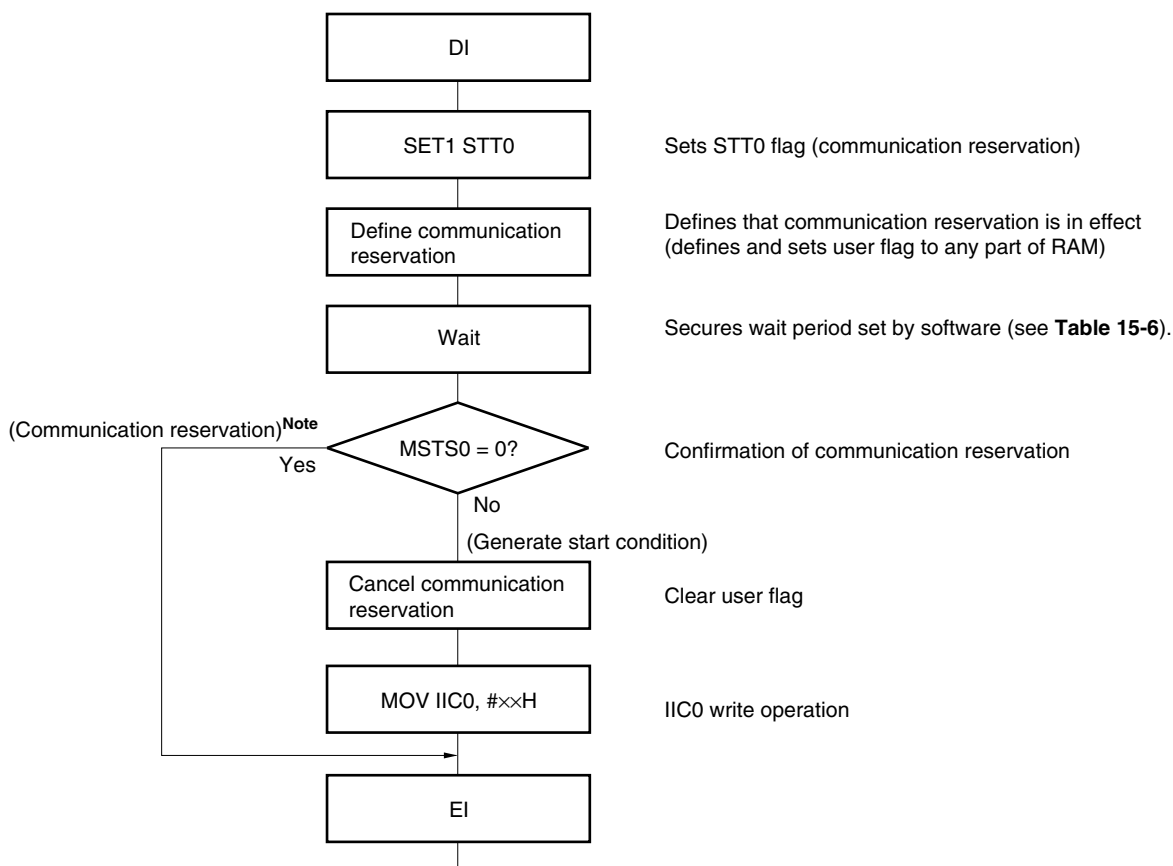


Figure 15-22 shows the communication reservation protocol.

Figure 15-22. Communication Reservation Protocol



Note The communication reservation operation executes a write to IIC shift register 0 (IIC0) when a stop condition interrupt request occurs.

Remark STT0: Bit 1 of IIC control register 0 (IICC0)
 MSTS0: Bit 7 of IIC status register 0 (IICS0)
 IIC0: IIC shift register 0

(2) When communication reservation function is disabled (bit 0 (IICRSV) of IIC flag register 0 (IICF0) = 1)

When bit 1 (STT0) of IIC control register 0 (IICC0) 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 when bit 6 (LREL0) of IICC0 register was set to 1)

To confirm whether the start condition was generated or request was rejected, check STCF flag (bit 7 of IICF0). The time shown in Table 15-7 is required until STCF flag is set to 1 after setting STT0 = 1. Therefore, secure the time by software.

Table 15-7. Wait Periods

| CL01 | CL00 | Wait Period |
|------|------|-------------|
| 0 | 0 | 6 clocks |
| 0 | 1 | 6 clocks |
| 1 | 0 | 12 clocks |
| 1 | 1 | 3 clocks |

15.5.15 Cautions

- (1) When STCEN (bit 1 of IIC flag register 0 (IICF0)) = 0

Immediately after I²C operation is enabled (IICE0 = 1), the bus communication status (IICBSY flag (bit 6 of IICF0) = 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 IIC clock selection register 0 (IICCL0).
- <2> Set bit 7 (IICE0) of IIC control register 0 (IICC0) to 1.
- <3> Set bit 0 (SPT0) of IICC0 to 1.

- (2) When STCEN = 1

Immediately after I²C operation is enabled (IICE0 = 1), the bus released status (IICBSY = 0) is recognized regardless of the actual bus status. To generate the first start condition (STT0 (bit 1 of IIC control register 0 (IICC0)) = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.

- (3) If other I²C communications are already in progress

If I²C operation is enabled and the device participates in communication already in progress when the SDA0 pin is low and the SCL0 pin is high, the macro of I²C recognizes that the SDA0 pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code, $\overline{\text{ACK}}$ is returned, but this interferes with other I²C communications. To avoid this, start I²C in the following sequence.

- <1> Clear bit 4 (SPIE0) of IICC0 register to 0 to disable generation of an interrupt request signal (INTIIC0) when the stop condition is detected.
- <2> Set bit 7 (IICE0) of IICC0 register to 1 to enable the operation of I²C.
- <3> Wait for detection of the start condition.
- <4> Set bit 6 (LREL0) of IICC0 register to 1 before $\overline{\text{ACK}}$ is returned (4 to 80 clocks after setting IICE0 bit to 1), to forcibly disable detection.

- (4) Determine the transfer clock frequency by using SMC0, CL01, CL00 bits (bits 3, 1, and 0 of IICL0 register), and CLX0 bit (bit 0 of IICX0 register) before enabling the operation (IICE0 = 1). To change the transfer clock frequency, clear IICE0 bit to 0 once.

- (5) Setting STT0 and SPT0 bits (bits 1 and 0 of IICC0 register) again after they are set and before they are cleared to 0 is prohibited.

- (6) When transmission is reserved, set SPIE0 bit (bit 4 of IICL0 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 IIC status register 0 (IICS0) 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 SPIE0 bit to 1 when MSTS0 bit (bit 7 of IIC status register 0 (IICS0)) is detected by software.

15.5.16 Communication operations

The following shows three operation procedures with the flowchart.

(1) Master operation in single master system

The flowchart when using the R7F0C011B, R7F0C012B, and R7F0C013B 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 I²C bus multimaster system, whether the bus is released or used cannot be judged by the I²C 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 R7F0C011B, R7F0C012B, and R7F0C013B takes 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 R7F0C011B, R7F0C012B, and R7F0C013B loses 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

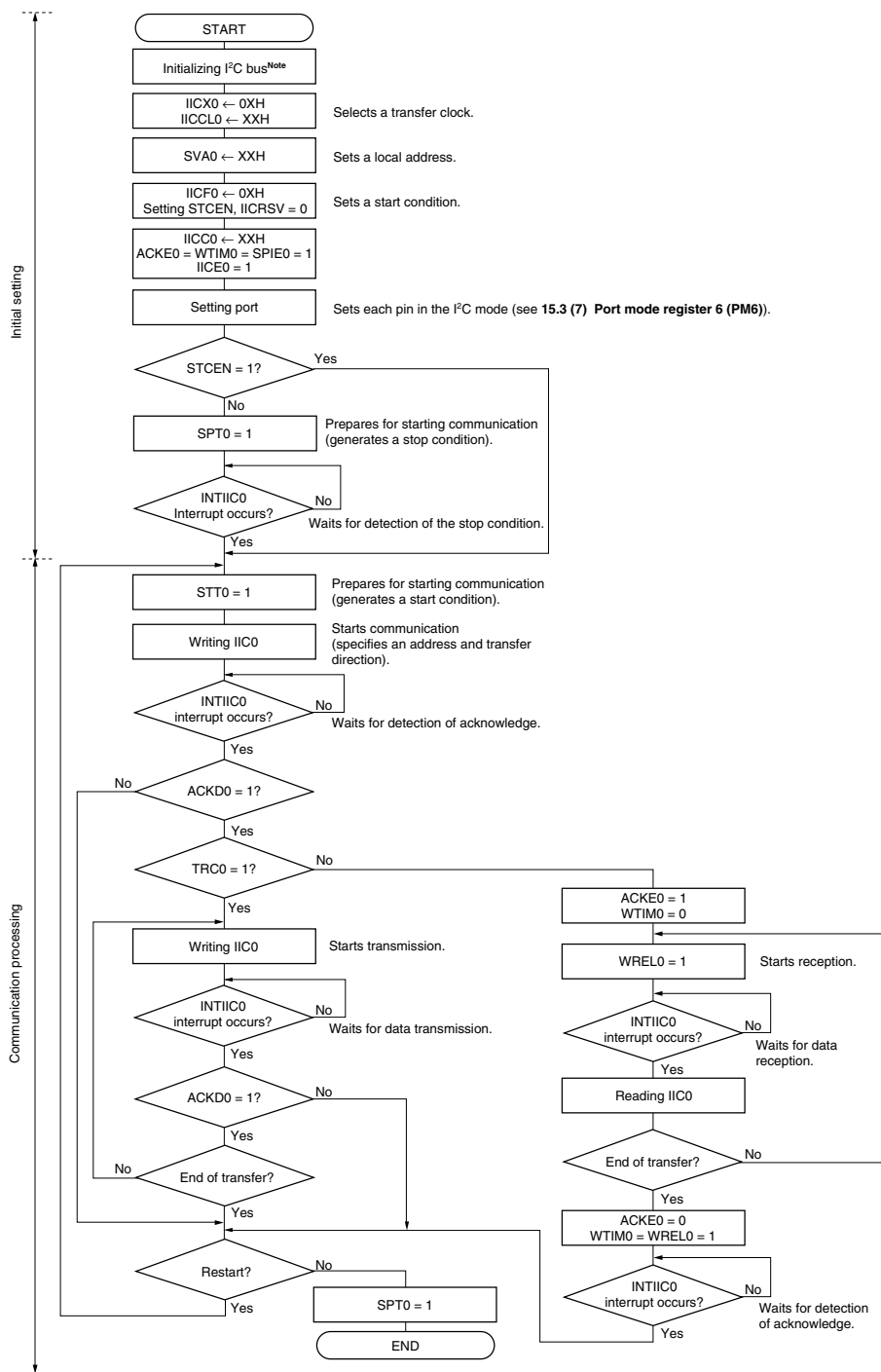
An example of when the R7F0C011B, R7F0C012B, and R7F0C013B is used as the I²C bus slave is shown below.

When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIIC0 interrupt occurrence (communication waiting). When an INTIIC0 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.

(1) Master operation in single-master system

Figure 15-23. Master Operation in Single-Master System

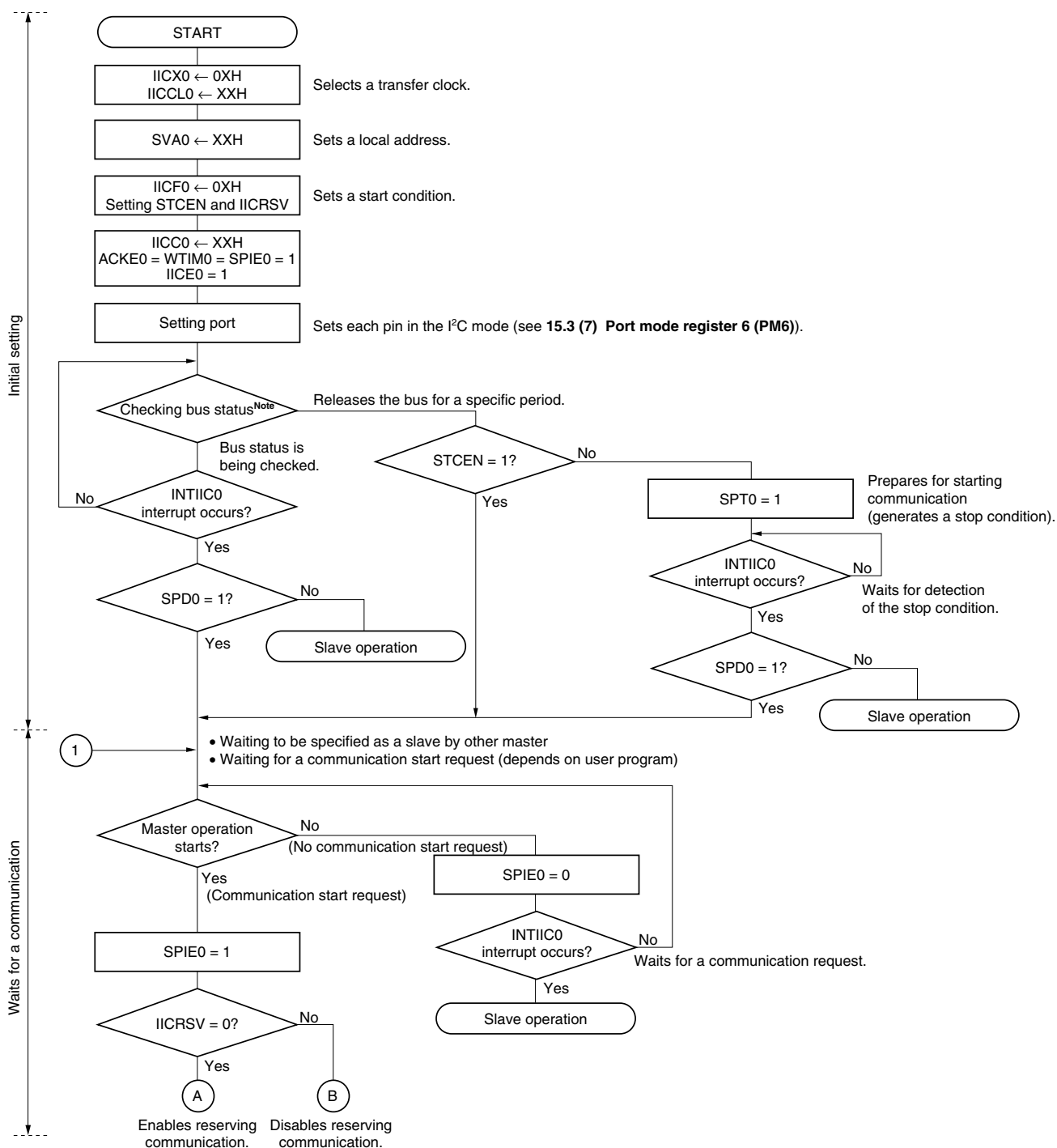


Note Release (SCL0 and SDA0 pins = high level) the I²C bus in conformance with the specifications of the product that is communicating. If EEPROM is outputting a low level to the SDA0 pin, for example, set the SCL0 pin in the output port mode, and output a clock pulse from the output port until the SDA0 pin is constantly at high level.

Remark Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.

(2) Master operation in multi-master system

Figure 15-24. Master Operation in Multi-Master System (1/3)



Note Confirm that the bus is released (CLD0 bit = 1, DAD0 bit = 1) for a specific period (for example, for a period of one frame). If the SDA0 pin is constantly at low level, decide whether to release the I²C bus (SCL0 and SDA0 pins = high level) in conformance with the specifications of the product that is communicating.

Figure 15-24. Master Operation in Multi-Master System (2/3)

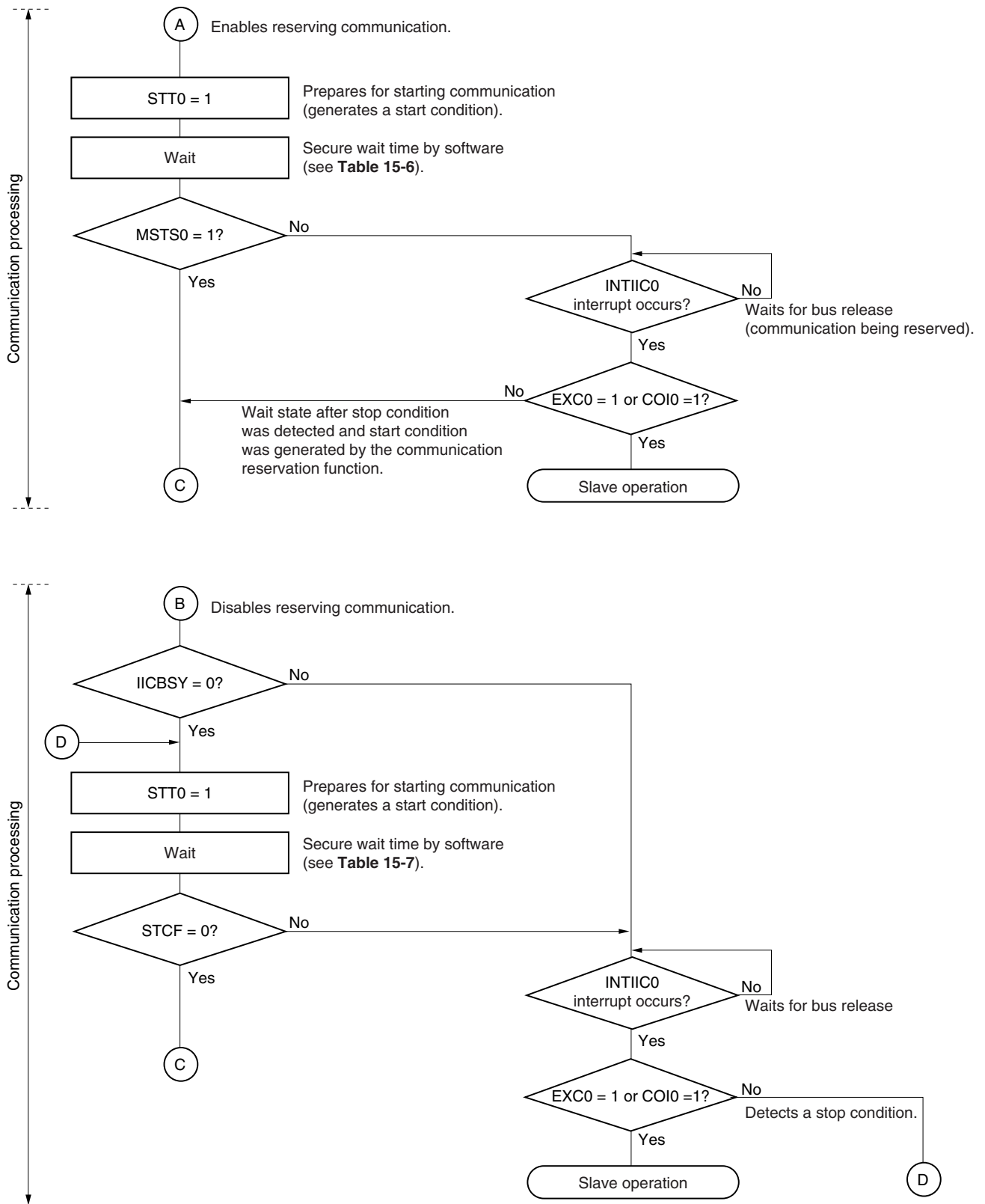
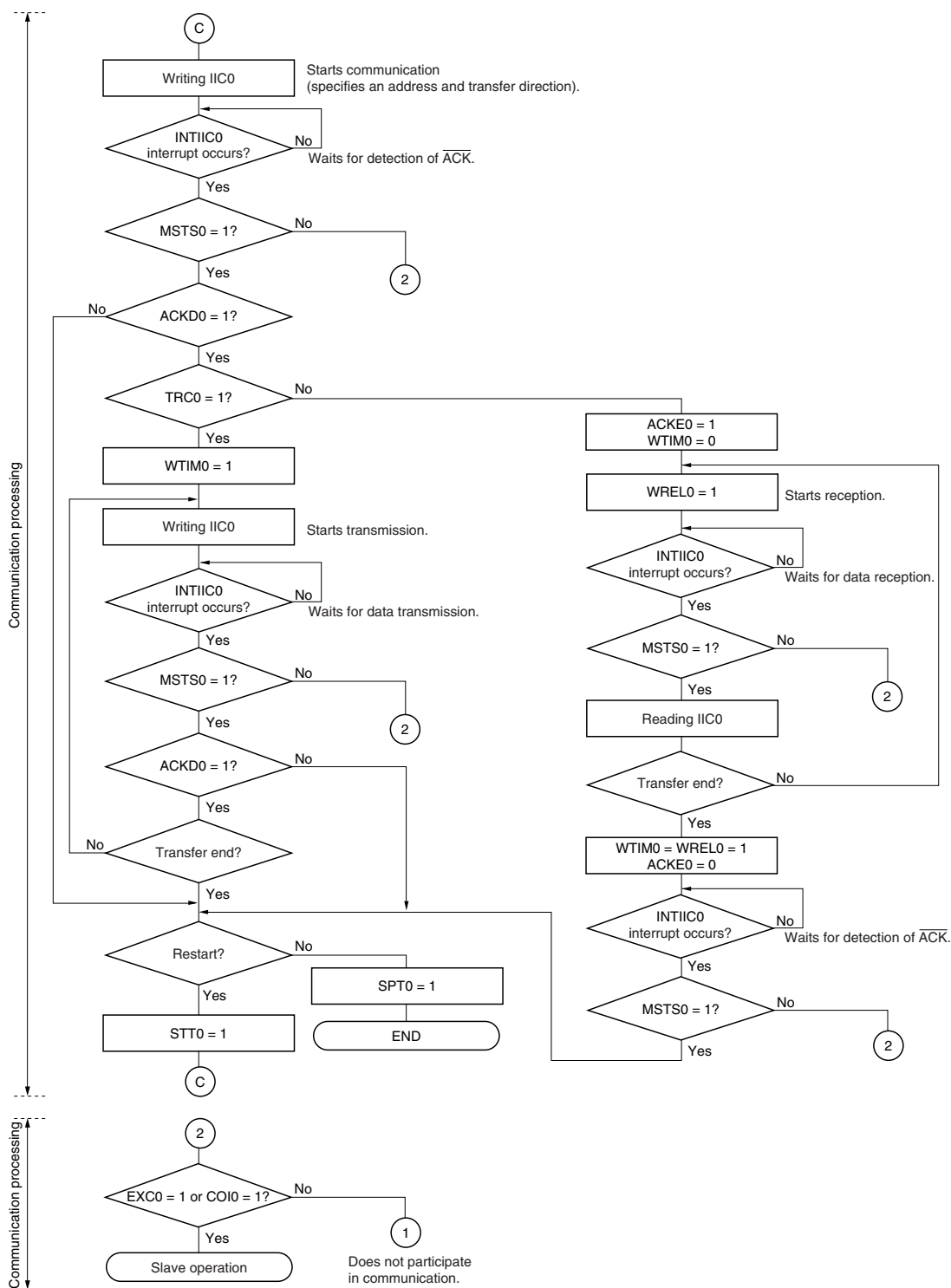


Figure 15-24. Master Operation in Multi-Master System (3/3)



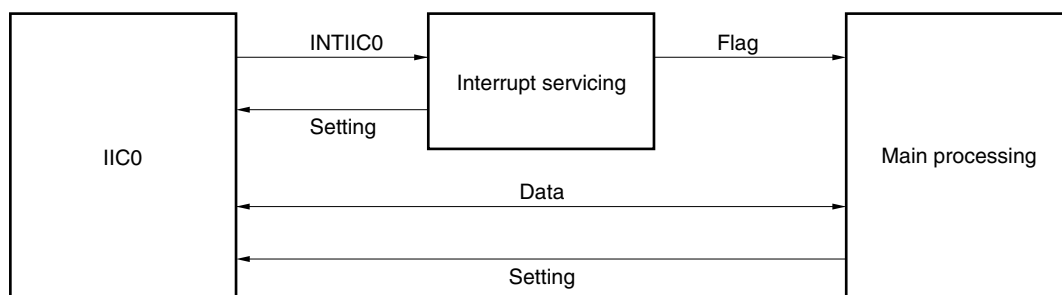
- Remarks 1.** Conform to the specifications of the product that is communicating, with respect to the transmission and reception formats.
- 2.** To use the device as a master in a multi-master system, read the MSTSO bit each time interrupt INTIIC0 has occurred to check the arbitration result.
- 3.** To use the device as a slave in a multi-master system, check the status by using the IICS0 and IICF0 registers each time interrupt INTIIC0 has occurred, and determine the processing to be performed next.

(3) Slave operation

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIIC0 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 INTIIC0 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 INTIIC0.

<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 INTIIC0 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 TRC0.

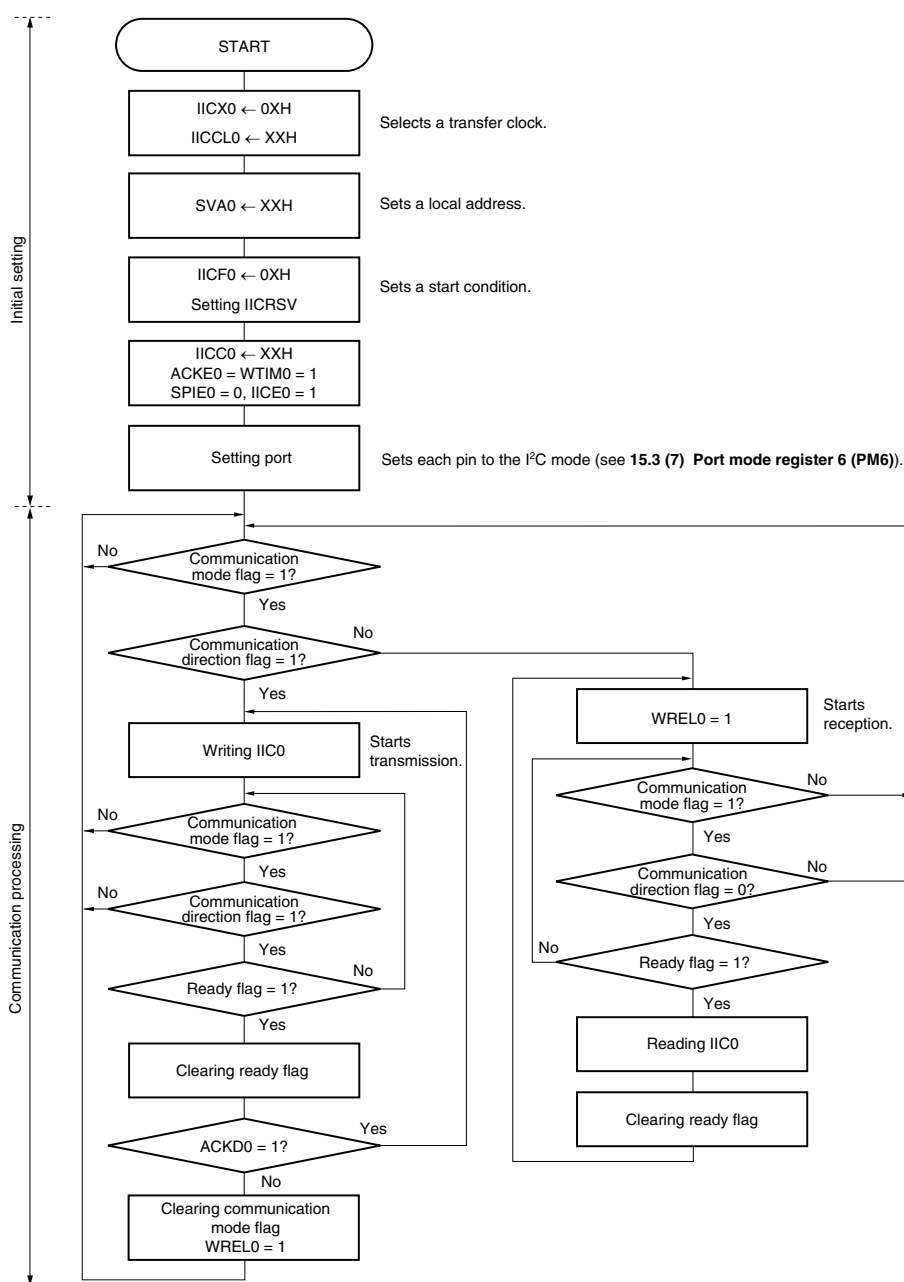
The main processing of the slave operation is explained next.

Start serial interface IIC0 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 15-25. Slave Operation Flowchart (1)



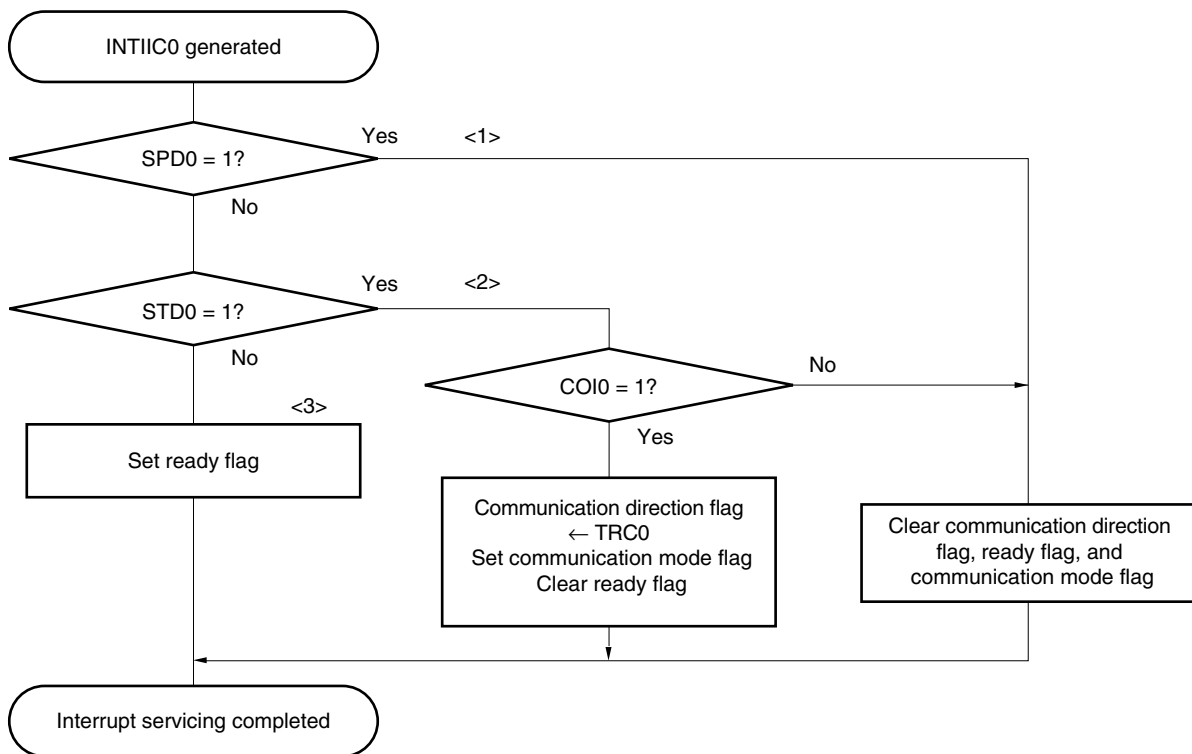
Remark Conform to the specifications of the product that is in communication, regarding the transmission and reception formats.

An example of the processing procedure of the slave with the INTIIC0 interrupt is explained below (processing is performed assuming that no extension code is used). The INTIIC0 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 I²C bus remaining in the wait state.

Remark <1> to <3> above correspond to <1> to <3> in **Figure 15-26 Slave Operation Flowchart (2)**.

Figure 15-26. Slave Operation Flowchart (2)



15.5.17 Timing of I²C interrupt request (INTIIC0) occurrence

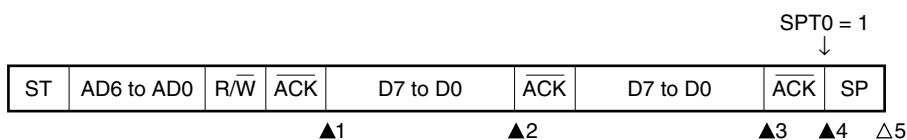
The timing of transmitting or receiving data and generation of interrupt request signal INTIIC0, and the value of the IICS0 register when the INTIIC0 signal is generated are shown below.

| | | |
|---------------|--------------------|----------------------------------|
| Remark | ST: | Start condition |
| | AD6 to AD0: | Address |
| | R/W: | Transfer direction specification |
| | \overline{ACK} : | Acknowledge |
| | D7 to D0: | Data |
| | SP: | Stop condition |

(1) Master device operation

(a) Start ~ Address ~ Data ~ Data ~ Stop (transmission/reception)

(i) When WTIM0 = 0



▲1: IICS0 = 1000×110B

▲2: IICS0 = 1000×000B

▲3: IICS0 = 1000×000B (Sets WTIM0 to 1^{Note})

▲4: IICS0 = 1000××00B (Sets SPT0 to 1)

Δ5: IICS0 = 00000001B

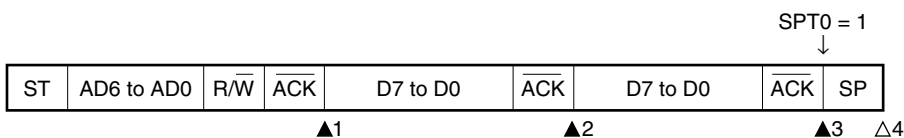
Note To generate a stop condition, set WTIM0 to 1 and change the timing for generating the INTIIC0 interrupt request signal.

Remark ▲: Always generated

Δ: Generated only when SPIE0 = 1

x: Don't care

(ii) When WTIM0 = 1



▲1: IICS0 = 1000×110B

▲2: IICS0 = 1000×100B

▲3: IICS0 = 1000××00B (Sets SPT0 to 1)

Δ4: IICS0 = 00000001B

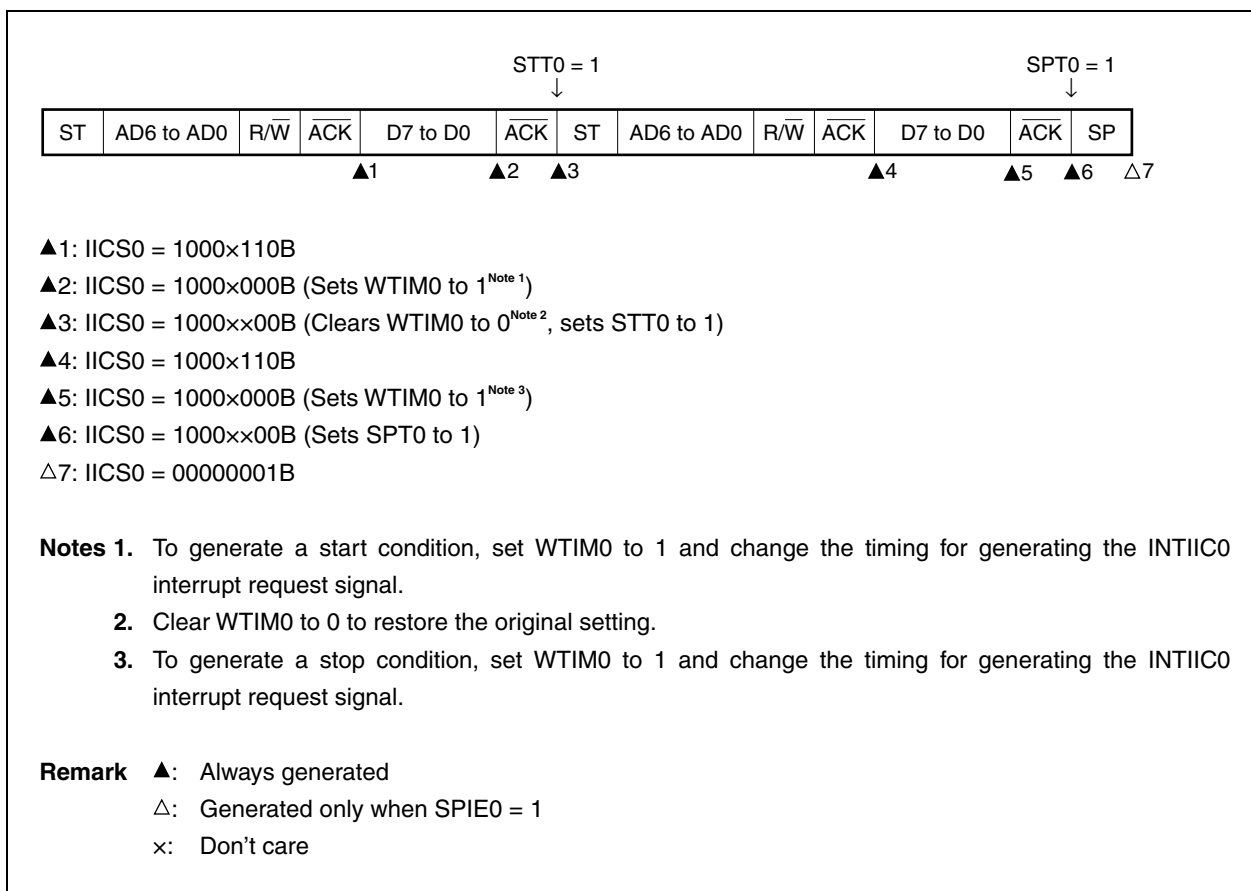
Remark ▲: Always generated

Δ: Generated only when SPIE0 = 1

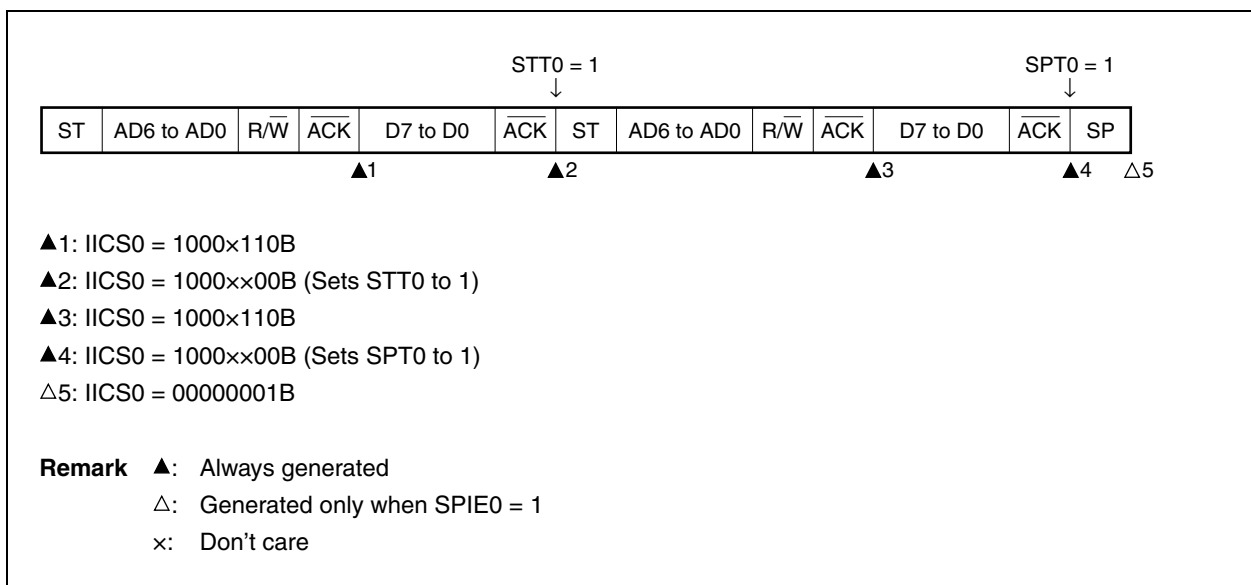
x: Don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop (restart)

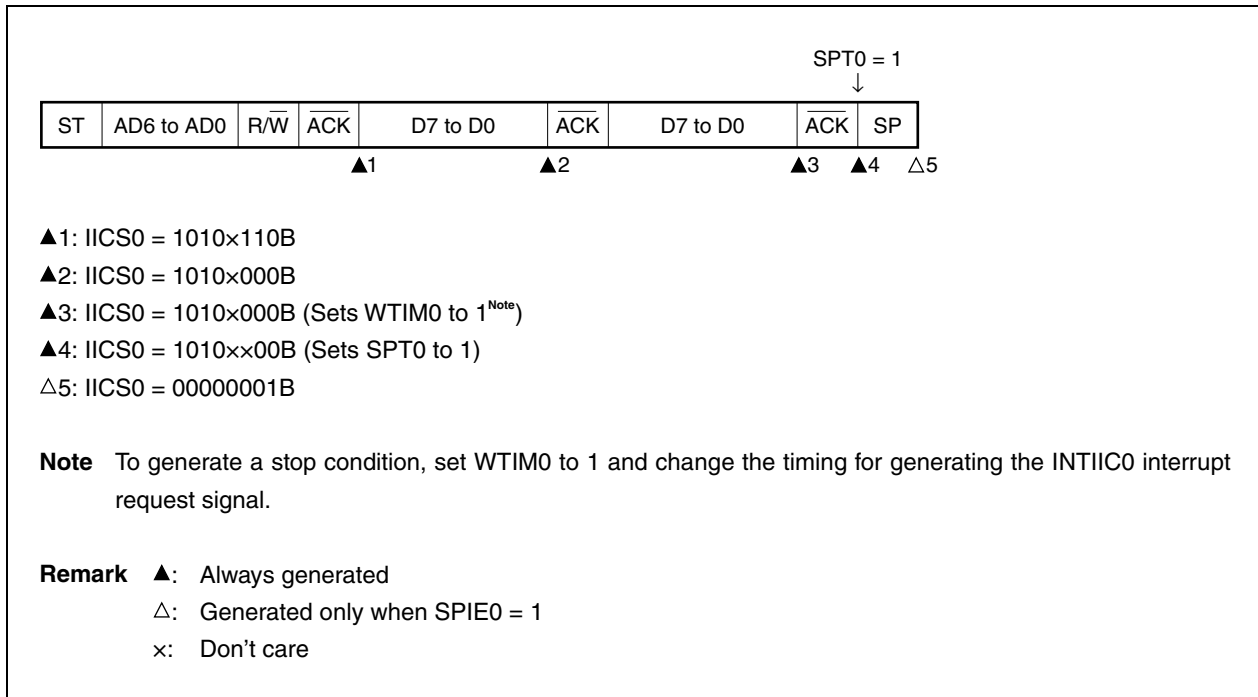
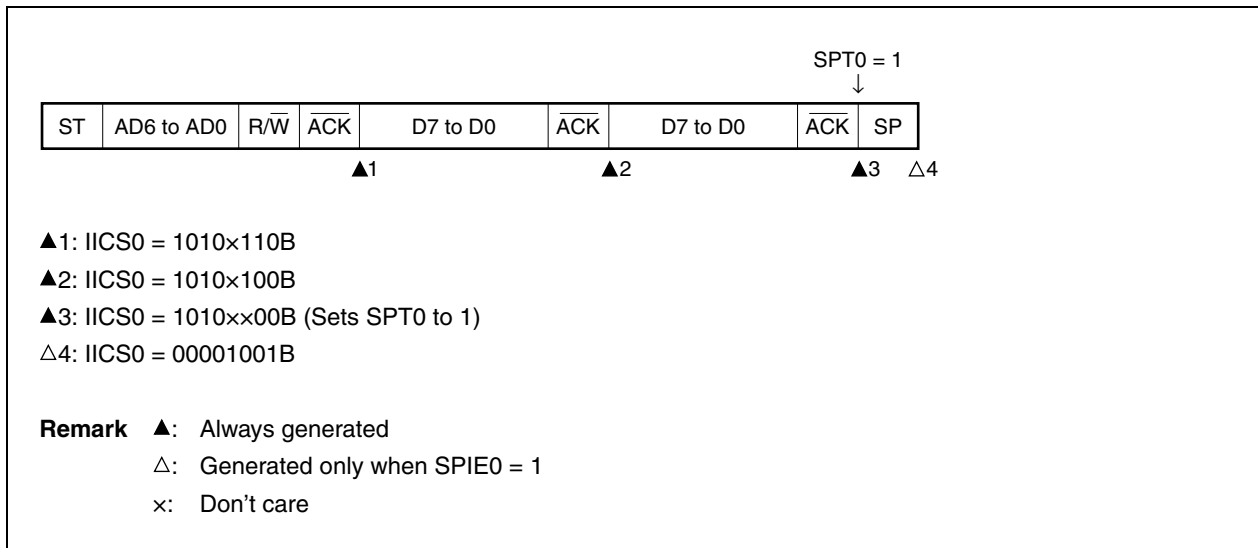
(i) When WTIM0 = 0

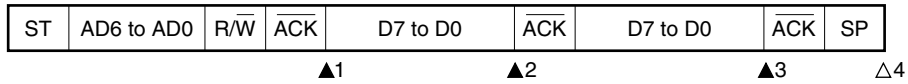


(ii) When WTIM0 = 1



(c) Start ~ Code ~ Data ~ Data ~ Stop (extension code transmission)

(i) When $WTIM0 = 0$ (ii) When $WTIM0 = 1$ 

(2) Slave device operation (slave address data reception)**(a) Start ~ Address ~ Data ~ Data ~ Stop****(i) When WTIM0 = 0**

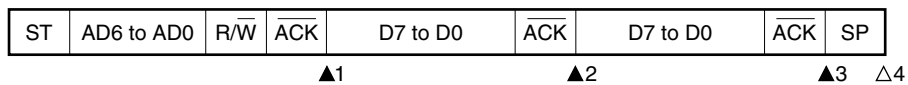
▲1: IICS0 = 0001x110B

▲2: IICS0 = 0001x000B

▲3: IICS0 = 0001x000B

△4: IICS0 = 00000001B

Remark ▲: Always generated
 △: Generated only when SPIE0 = 1
 x: Don't care

(ii) When WTIM0 = 1

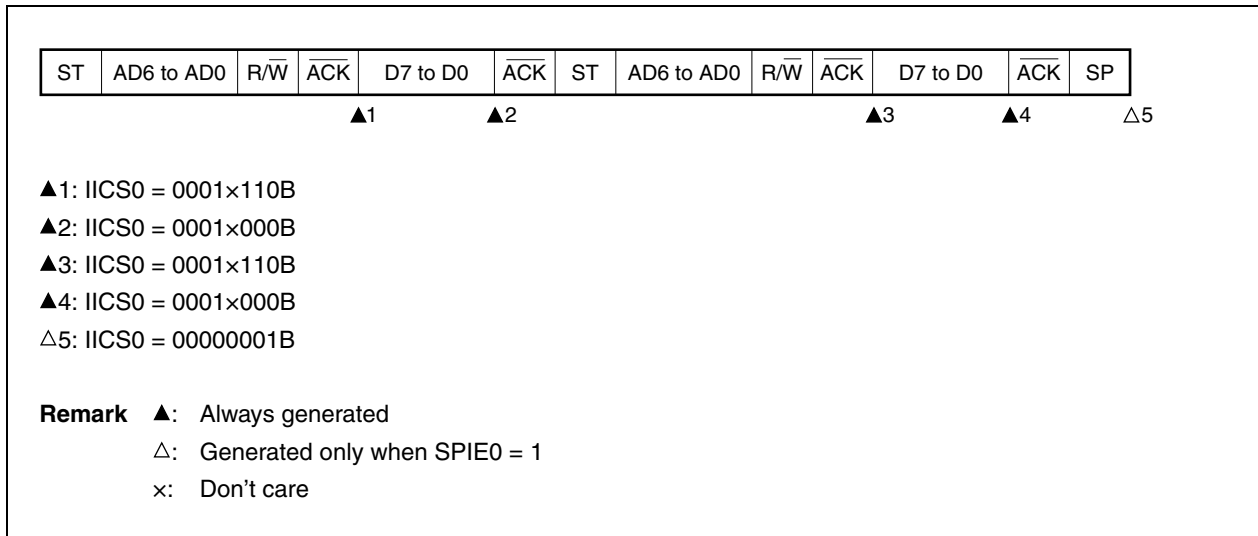
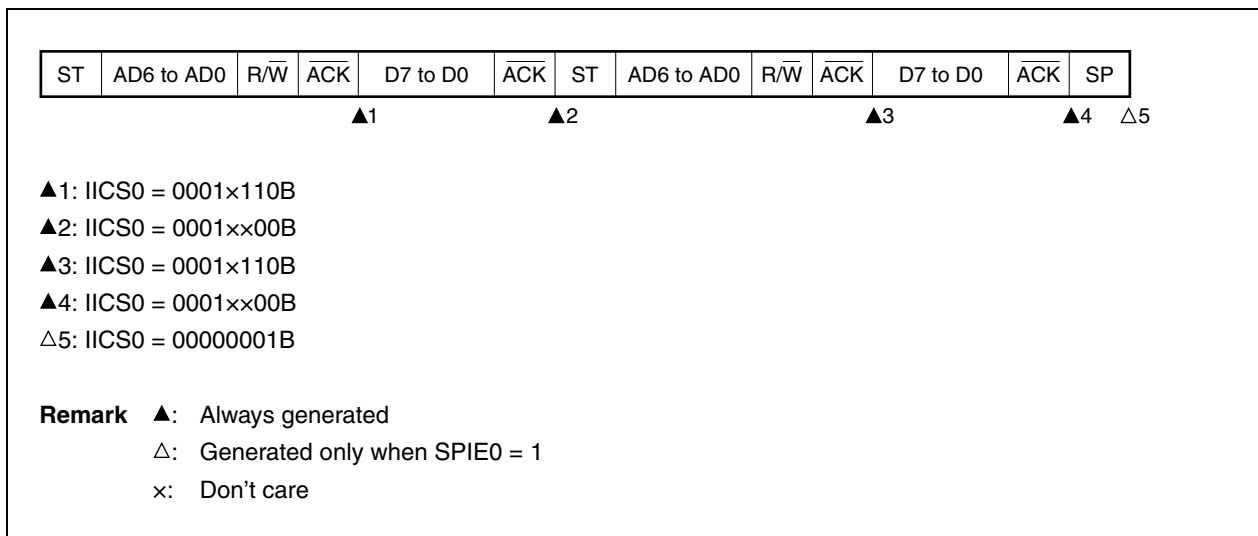
▲1: IICS0 = 0001x110B

▲2: IICS0 = 0001x100B

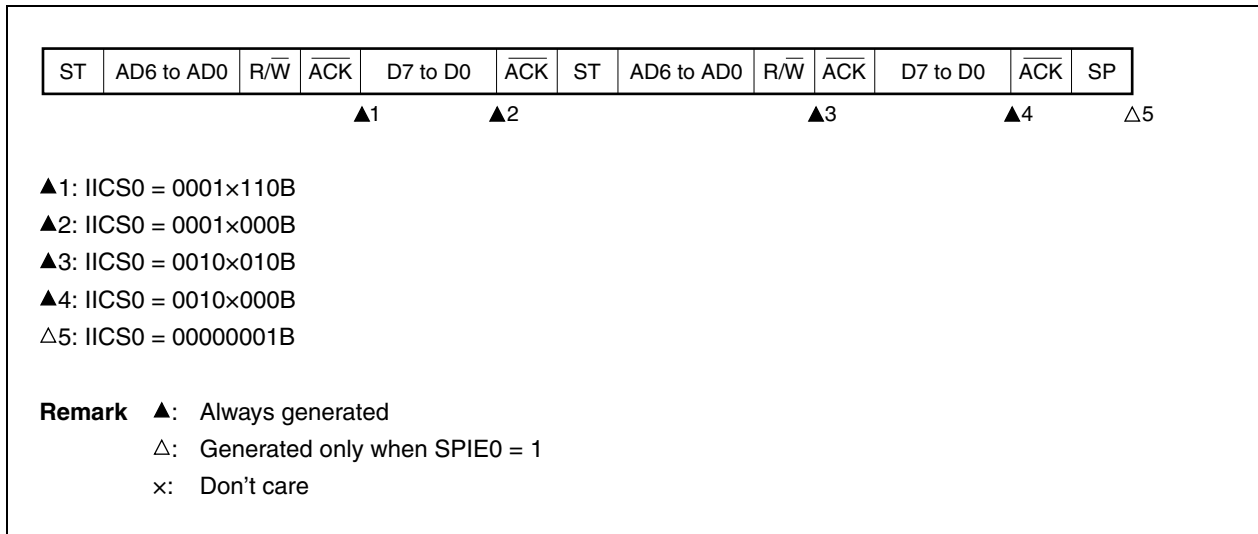
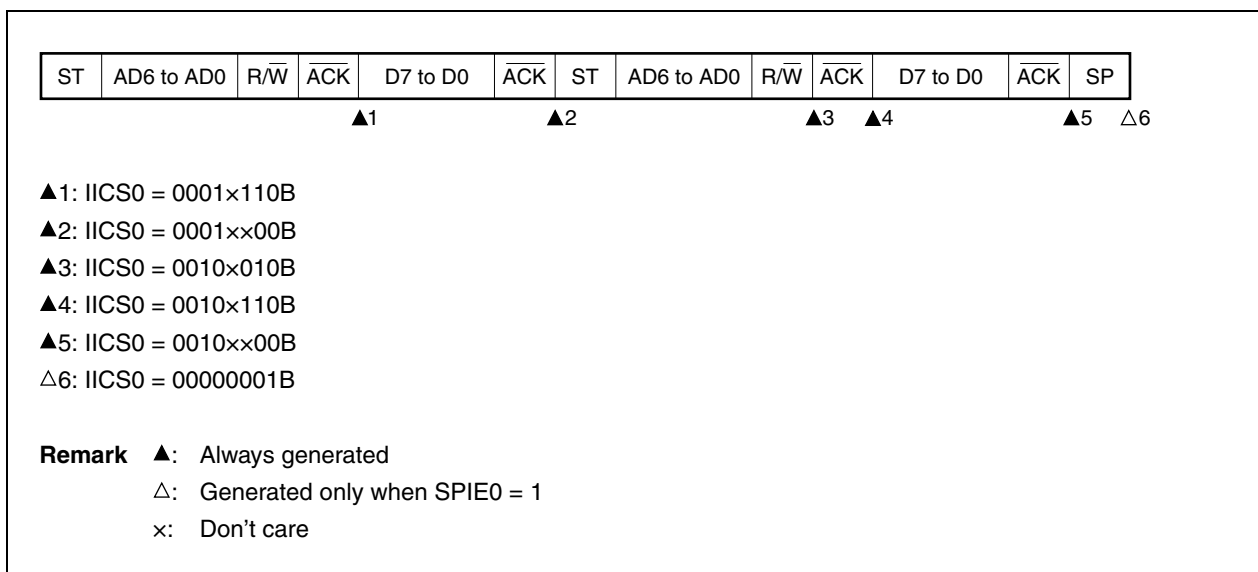
▲3: IICS0 = 0001xx00B

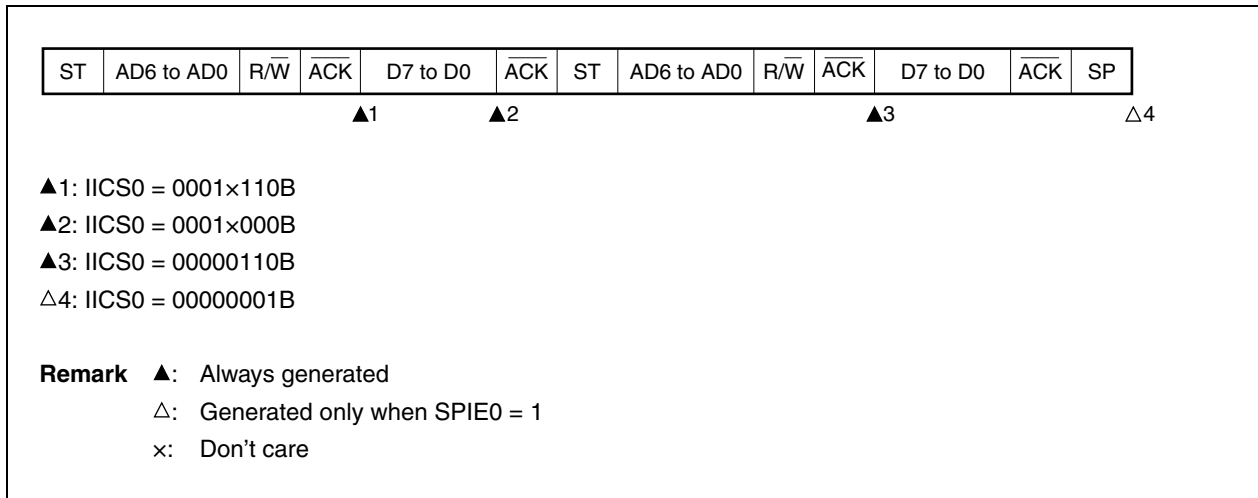
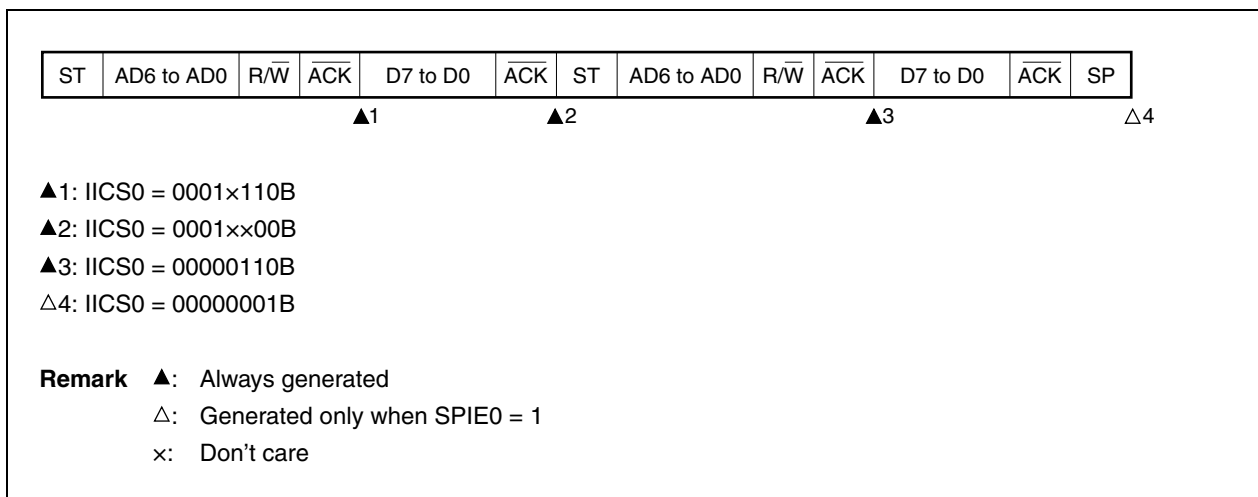
△4: IICS0 = 00000001B

Remark ▲: Always generated
 △: Generated only when SPIE0 = 1
 x: Don't care

(b) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**(i) When WTIM0 = 0 (after restart, matches with SVA0)****(ii) When WTIM0 = 1 (after restart, matches with SVA0)**

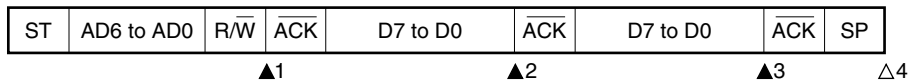
(c) Start ~ Address ~ Data ~ Start ~ Code ~ Data ~ Stop

(i) When $WTIM0 = 0$ (after restart, does not match address (= extension code))(ii) When $WTIM0 = 1$ (after restart, does not match address (= extension code))

(d) Start ~ Address ~ Data ~ Start ~ Address ~ Data ~ Stop**(i) When WTIM0 = 0 (after restart, does not match address (= not extension code))****(ii) When WTIM0 = 1 (after restart, does not match address (= not extension code))**

(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 WTIM0 = 0**

▲1: IICS0 = 0010×010B

▲2: IICS0 = 0010×000B

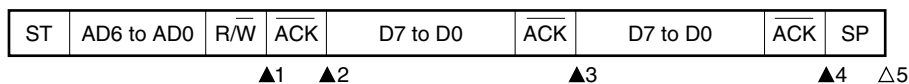
▲3: IICS0 = 0010×000B

△4: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(ii) When WTIM0 = 1

▲1: IICS0 = 0010×010B

▲2: IICS0 = 0010×110B

▲3: IICS0 = 0010×100B

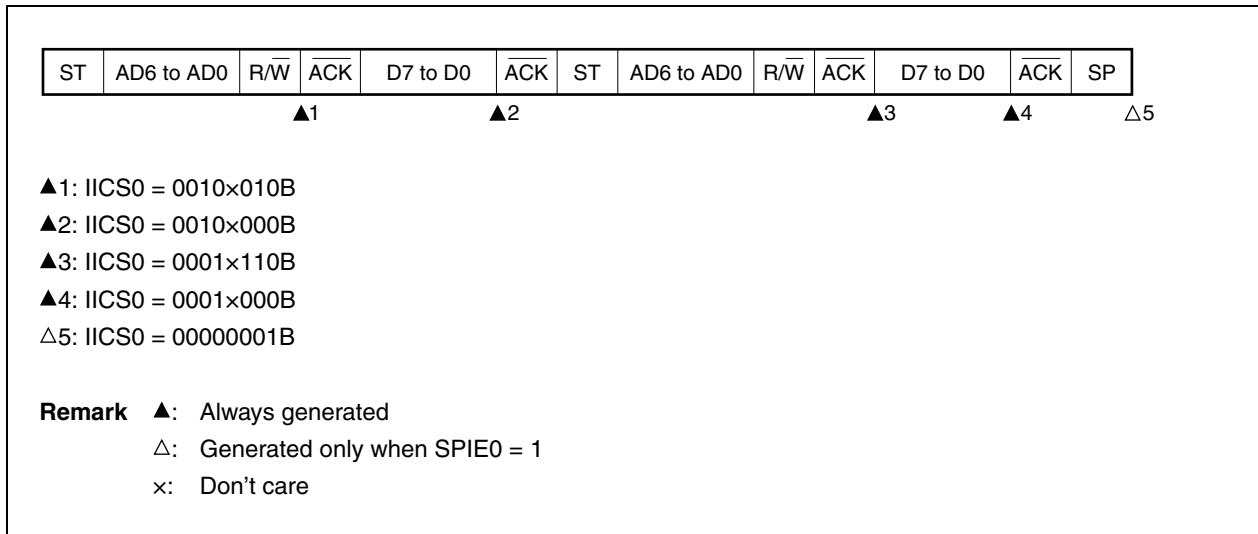
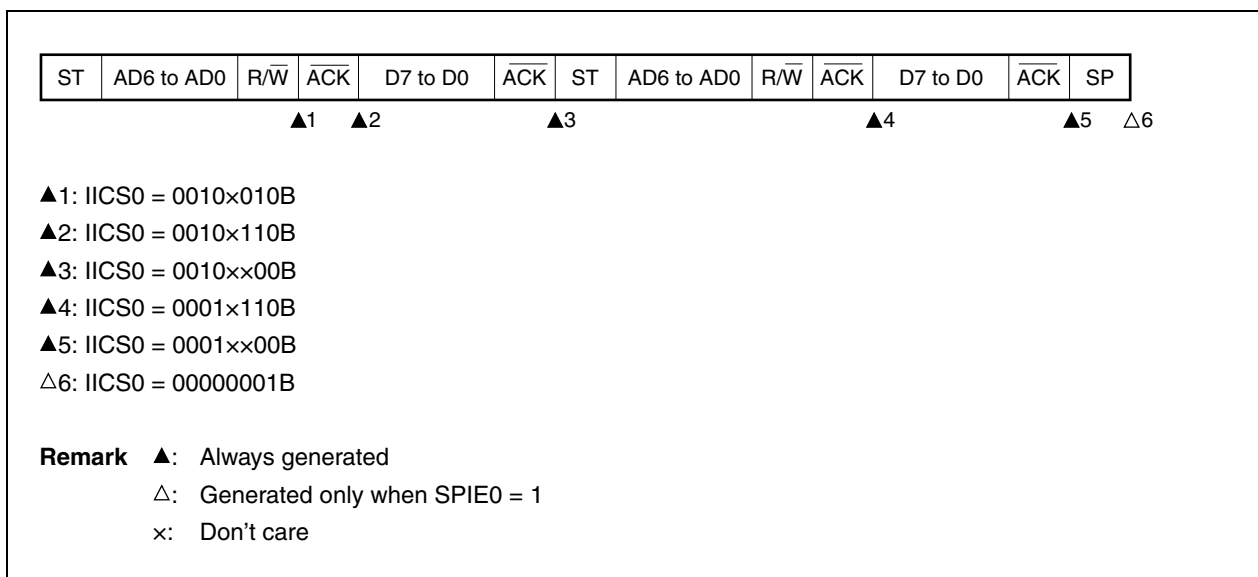
▲4: IICS0 = 0010××00B

△5: IICS0 = 00000001B

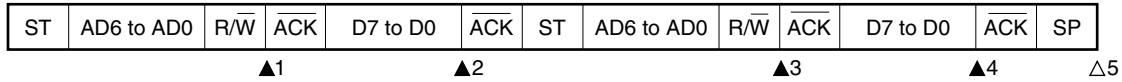
Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(b) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**(i) When WTIM0 = 0 (after restart, matches SVA0)****(ii) When WTIM0 = 1 (after restart, matches SVA0)**

(c) Start ~ Code ~ Data ~ Start ~ Code ~ Data ~ Stop

(i) When $WTIM0 = 0$ (after restart, extension code reception)

▲1: IICS0 = 0010x010B

▲2: IICS0 = 0010x000B

▲3: IICS0 = 0010x010B

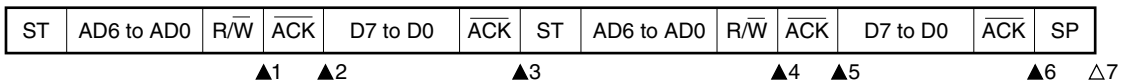
▲4: IICS0 = 0010x000B

△5: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(ii) When $WTIM0 = 1$ (after restart, extension code reception)

▲1: IICS0 = 0010x010B

▲2: IICS0 = 0010x110B

▲3: IICS0 = 0010xx00B

▲4: IICS0 = 0010x010B

▲5: IICS0 = 0010x110B

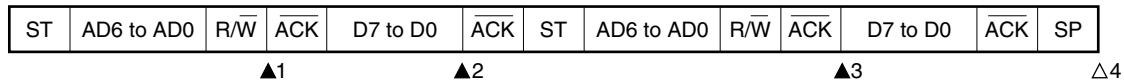
▲6: IICS0 = 0010xx00B

△7: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(d) Start ~ Code ~ Data ~ Start ~ Address ~ Data ~ Stop**(i) When WTIM0 = 0 (after restart, does not match address (= not extension code))**

▲1: IICS0 = 00100010B

▲2: IICS0 = 00100000B

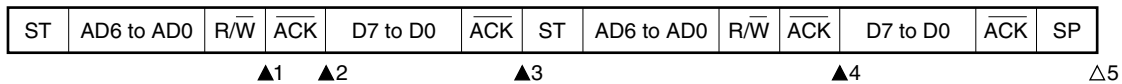
▲3: IICS0 = 00000110B

▲4: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(ii) When WTIM0 = 1 (after restart, does not match address (= not extension code))

▲1: IICS0 = 00100010B

▲2: IICS0 = 00100110B

▲3: IICS0 = 00100x00B

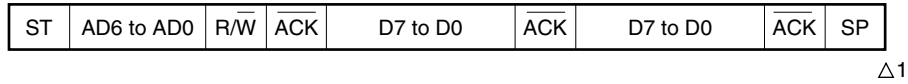
▲4: IICS0 = 00000110B

▲5: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

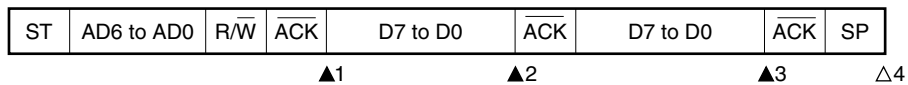
(4) Operation without communication**(a) Start ~ Code ~ Data ~ Data ~ Stop**

△1: IICS0 = 00000001B

Remark △: Generated only when SPIE0 = 1

(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 MSTS0 bit each time interrupt request signal INTIIC0 has occurred to check the arbitration result.

(a) When arbitration loss occurs during transmission of slave address data**(i) When WTIMO = 0**

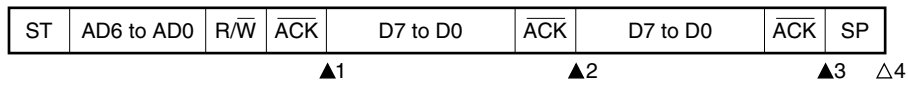
▲1: IICS0 = 0101x110B

▲2: IICS0 = 0001x000B

▲3: IICS0 = 0001x000B

△4: IICS0 = 00000001B

Remark ▲: Always generated
 △: Generated only when SPIE0 = 1
 x: Don't care

(ii) When $WTIM0 = 1$ 

▲1: IICS0 = 0101×110B

▲2: IICS0 = 0001×100B

▲3: IICS0 = 0001××00B

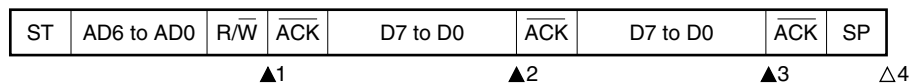
△4: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(b) When arbitration loss occurs during transmission of extension code

(i) When $WTIM0 = 0$ 

▲1: IICS0 = 0110×010B

▲2: IICS0 = 0010×000B

▲3: IICS0 = 0010×000B

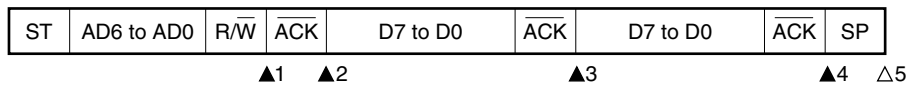
△4: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(ii) When WTIM0 = 1



▲1: IICS0 = 0110x010B

▲2: IICS0 = 0010x110B

▲3: IICS0 = 0010x100B

▲4: IICS0 = 0010xx00B

△5: IICS0 = 00000001B

Remark ▲: Always generated

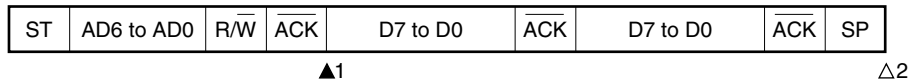
△: Generated only when SPIE0 = 1

x: Don't care

(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 MSTS0 bit each time interrupt request signal INTIIC0 has occurred to check the arbitration result.

(a) When arbitration loss occurs during transmission of slave address data (when WTIM0 = 1)

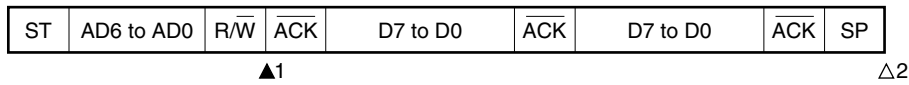


▲1: IICS0 = 01000110B

△2: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

(b) When arbitration loss occurs during transmission of extension code

▲1: IICS0 = 0110x010B

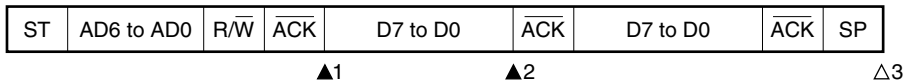
Sets LREL0 = 1 by software

△2: IICS0 = 00000001B

Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

(c) When arbitration loss occurs during transmission of data**(i) When WTIM0 = 0**

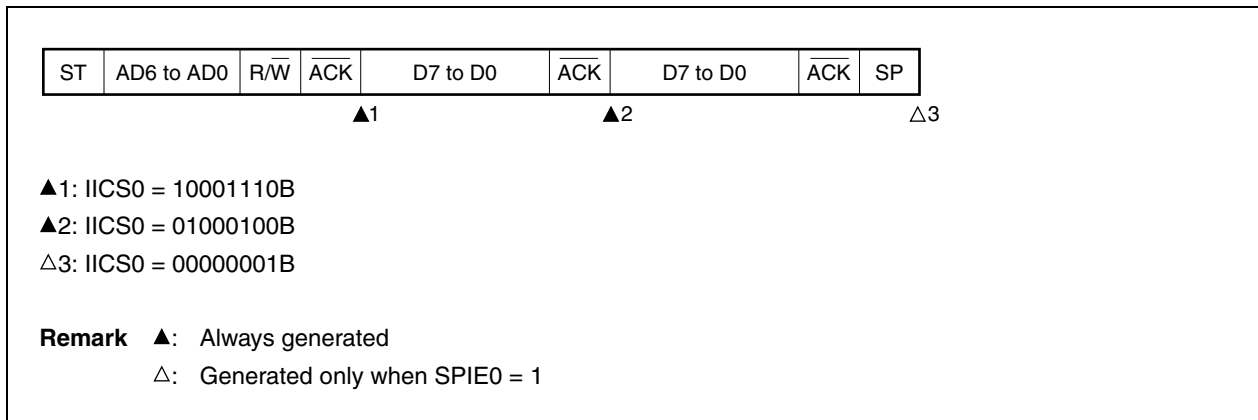
▲1: IICS0 = 10001110B

▲2: IICS0 = 01000000B

△3: IICS0 = 00000001B

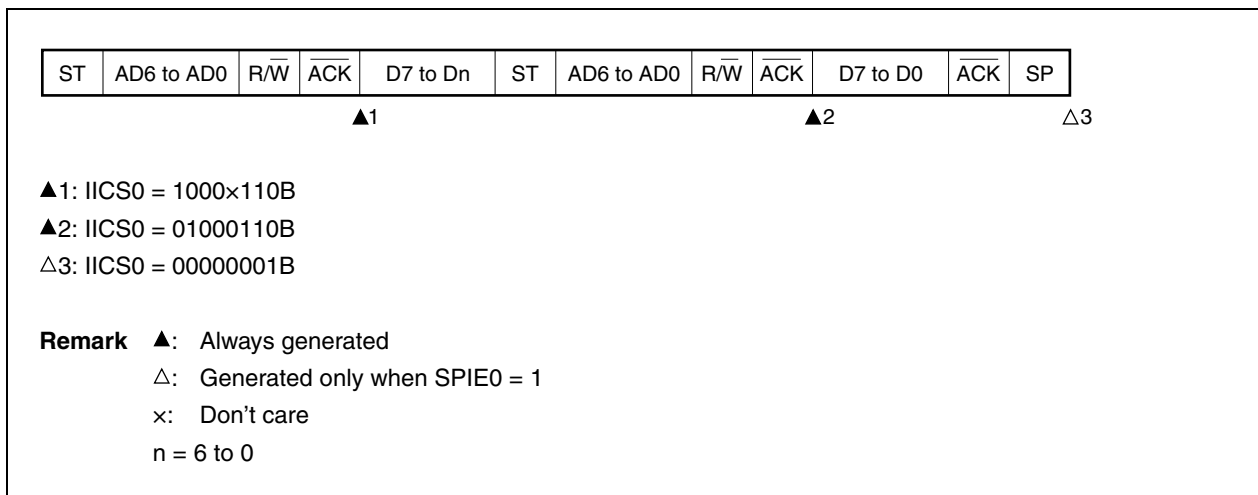
Remark ▲: Always generated

△: Generated only when SPIE0 = 1

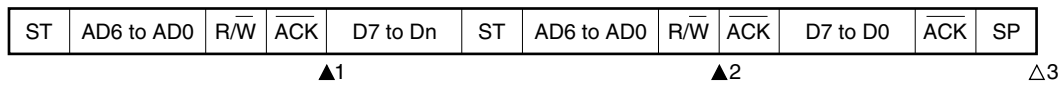
(ii) When $WTIM0 = 1$ 

(d) When loss occurs due to restart condition during data transfer

(i) Not extension code (Example: unmatched with SVA0)



(ii) Extension code



▲1: IICS0 = 1000×110B

▲2: IICS0 = 01100010B

Sets LREL0 = 1 by software

△3: IICS0 = 00000001B

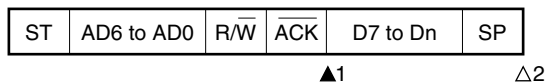
Remark ▲: Always generated

△: Generated only when SPIE0 = 1

x: Don't care

n = 6 to 0

(e) When loss occurs due to stop condition during data transfer



▲1: IICS0 = 10000110B

△2: IICS0 = 01000001B

Remark ▲: Always generated

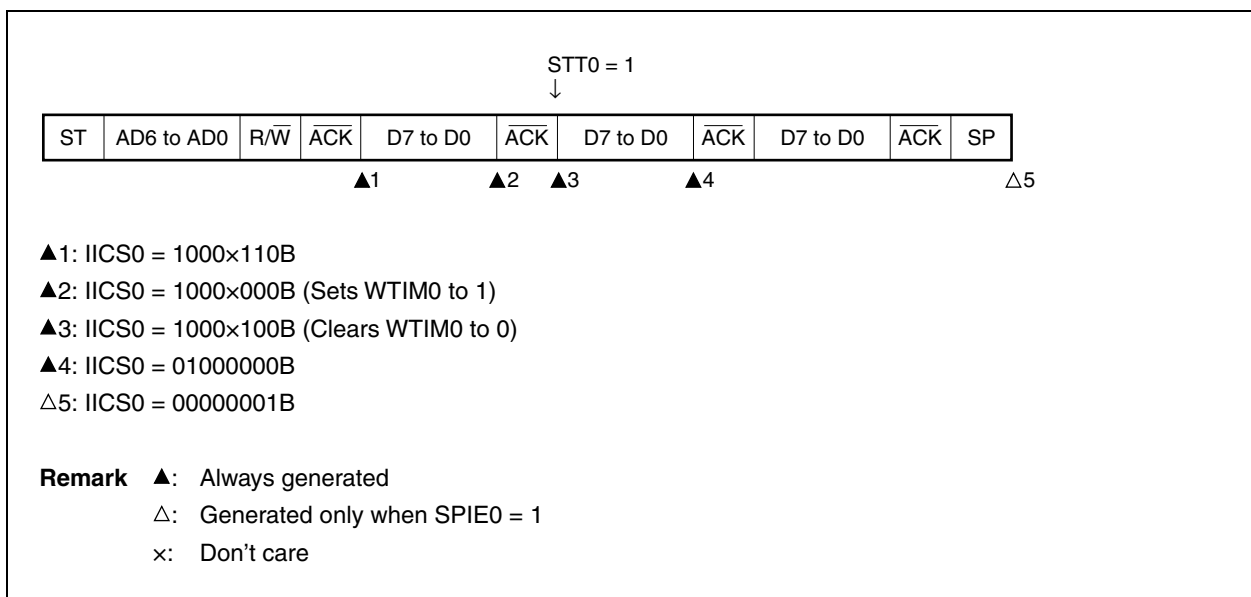
△: Generated only when SPIE0 = 1

x: Don't care

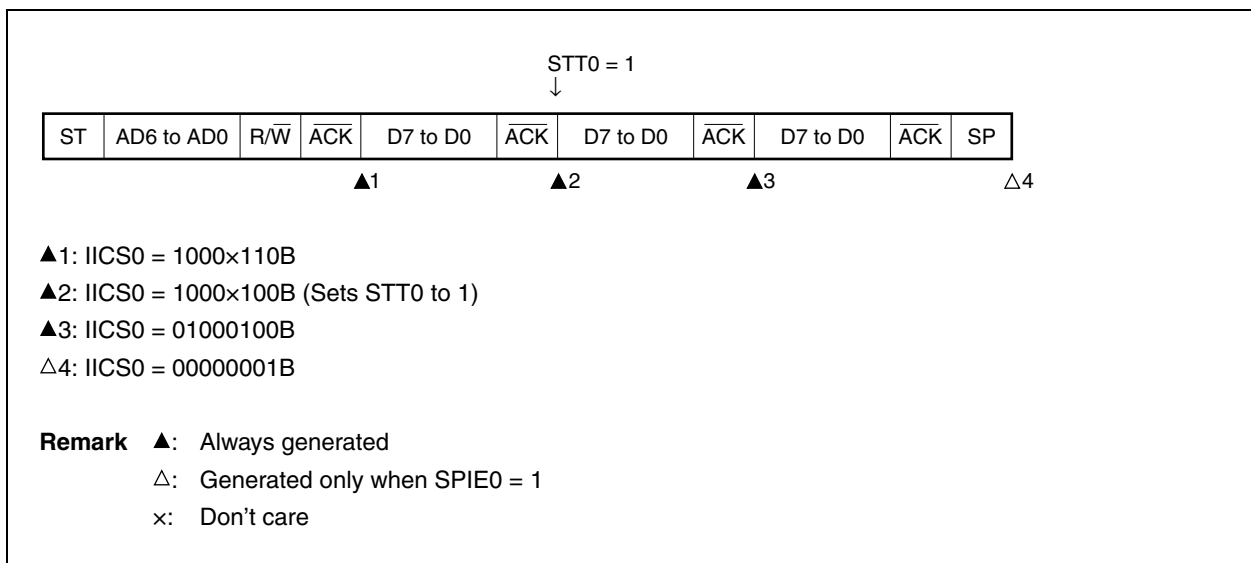
n = 6 to 0

(f) When arbitration loss occurs due to low-level data when attempting to generate a restart condition

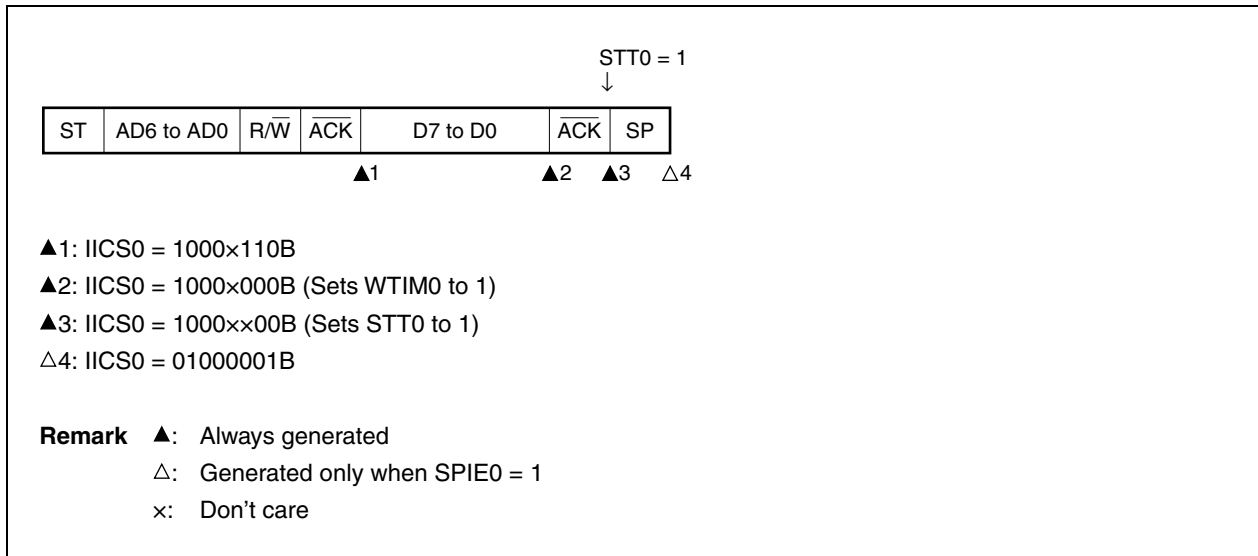
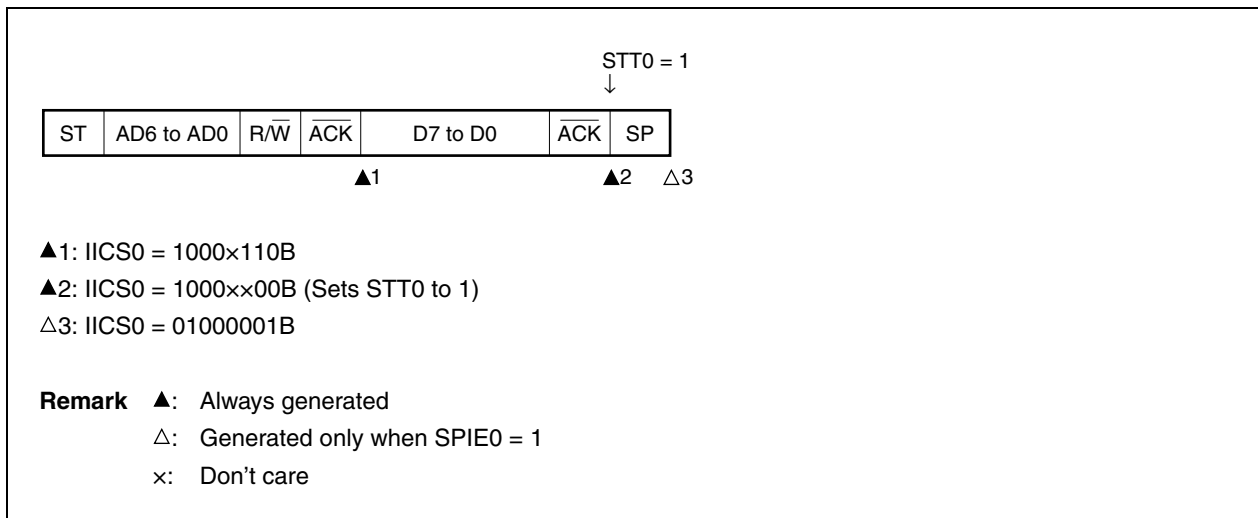
(i) When **WTIM0 = 0**



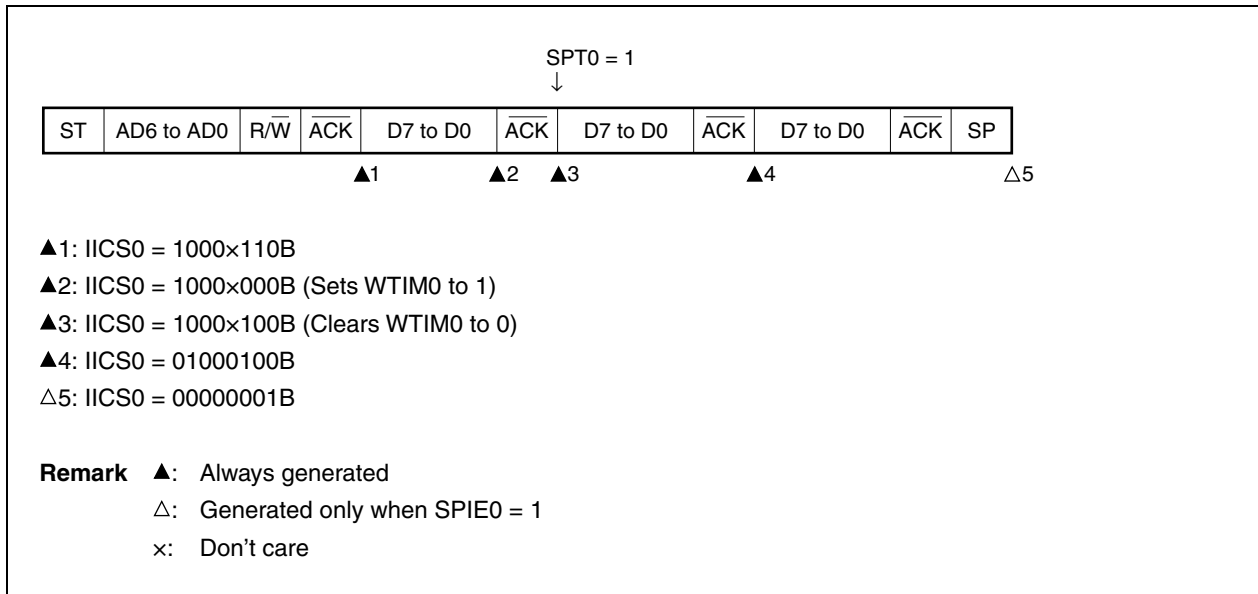
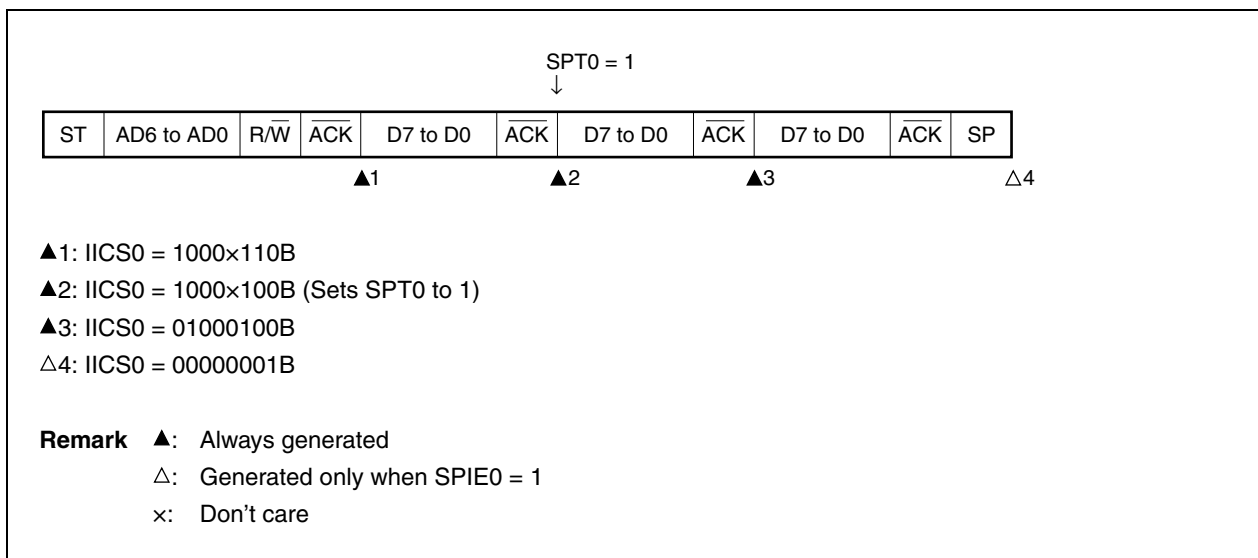
(ii) When **WTIM0 = 1**



(g) When arbitration loss occurs due to a stop condition when attempting to generate a restart condition

(i) When $WTIM0 = 0$ (ii) When $WTIM0 = 1$ 

(h) When arbitration loss occurs due to low-level data when attempting to generate a stop condition

(i) When $WTIM0 = 0$ (ii) When $WTIM0 = 1$ 

15.6 Timing Charts

When using the I²C 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 TRC0 bit (bit 3 of IIC status register 0 (IICS0)), which specifies the data transfer direction, and then starts serial communication with the slave device.

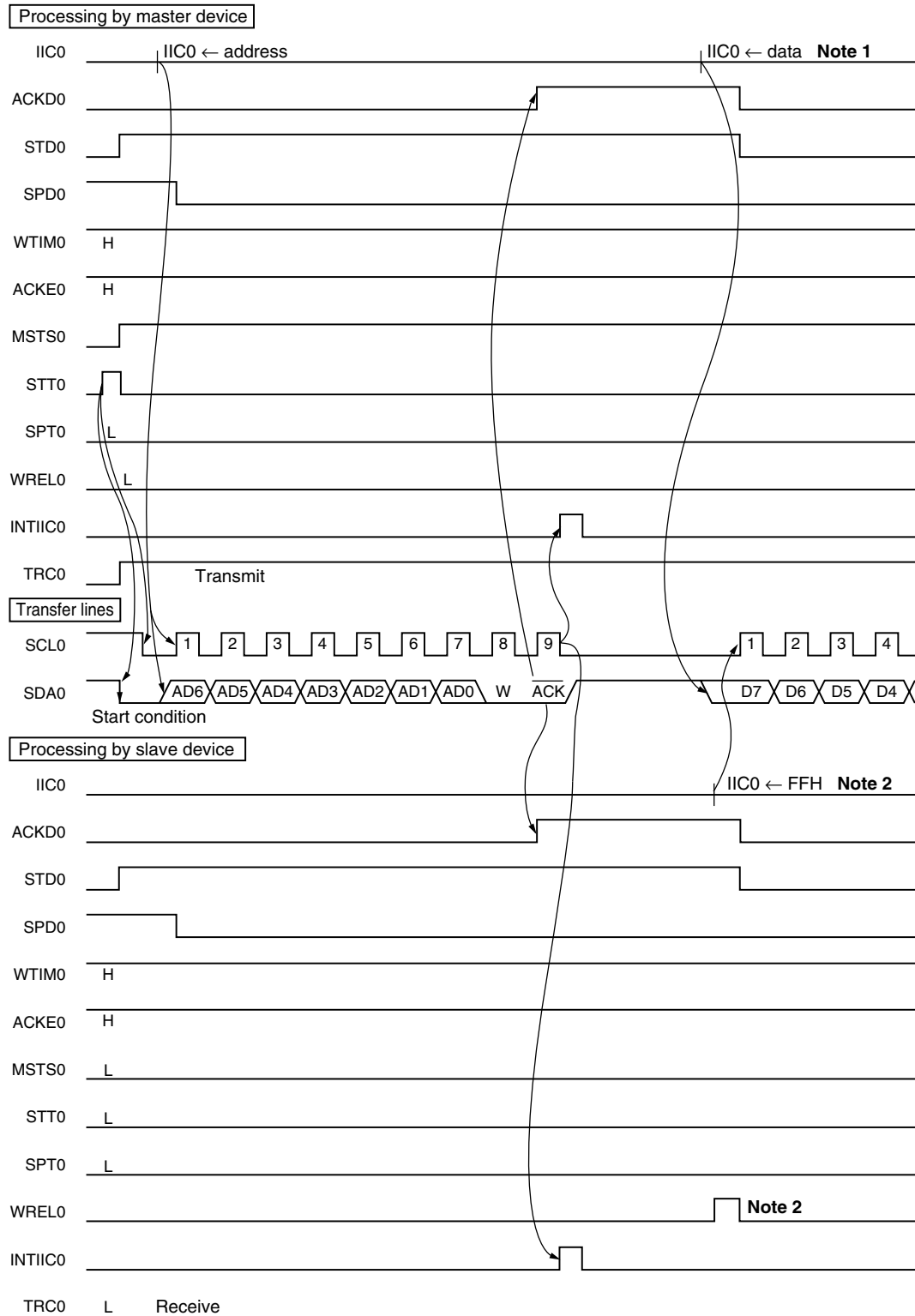
Figures 15-27 and 15-28 show timing charts of the data communication.

IIC shift register 0 (IIC0)'s shift operation is synchronized with the falling edge of the serial clock (SCL0). The transmit data is transferred to the SO0 latch and is output (MSB first) via the SDA0 pin.

Data input via the SDA0 pin is captured into IIC0 at the rising edge of SCL0.

**Figure 15-27. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (1/3)**

(1) Start condition ~ address

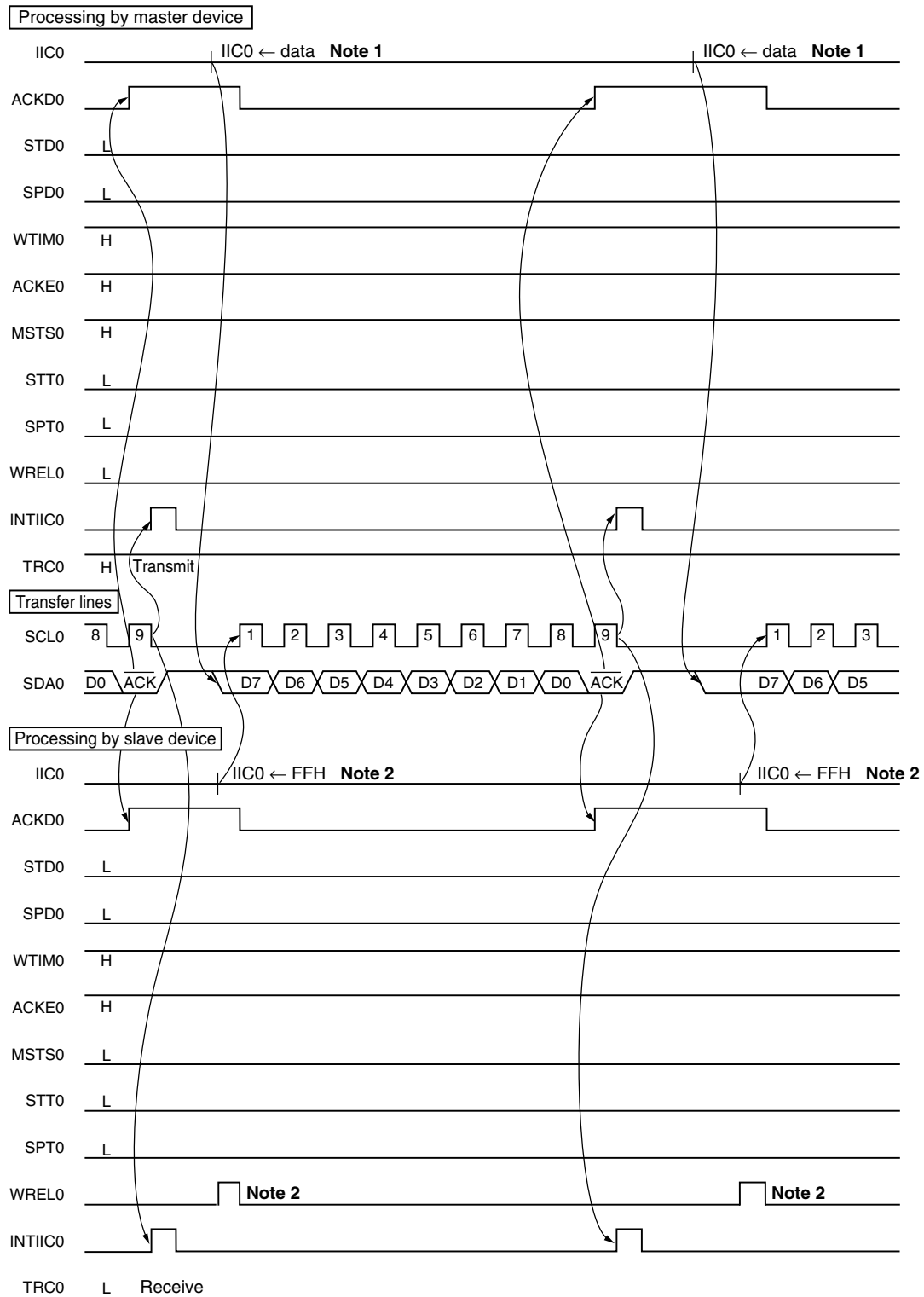


Notes 1. Write data to IIC0, not setting WRELO, in order to cancel a wait state during master transmission.

2. To cancel slave wait, write "FFH" to IIC0 or set WRELO.

**Figure 15-27. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (2/3)**

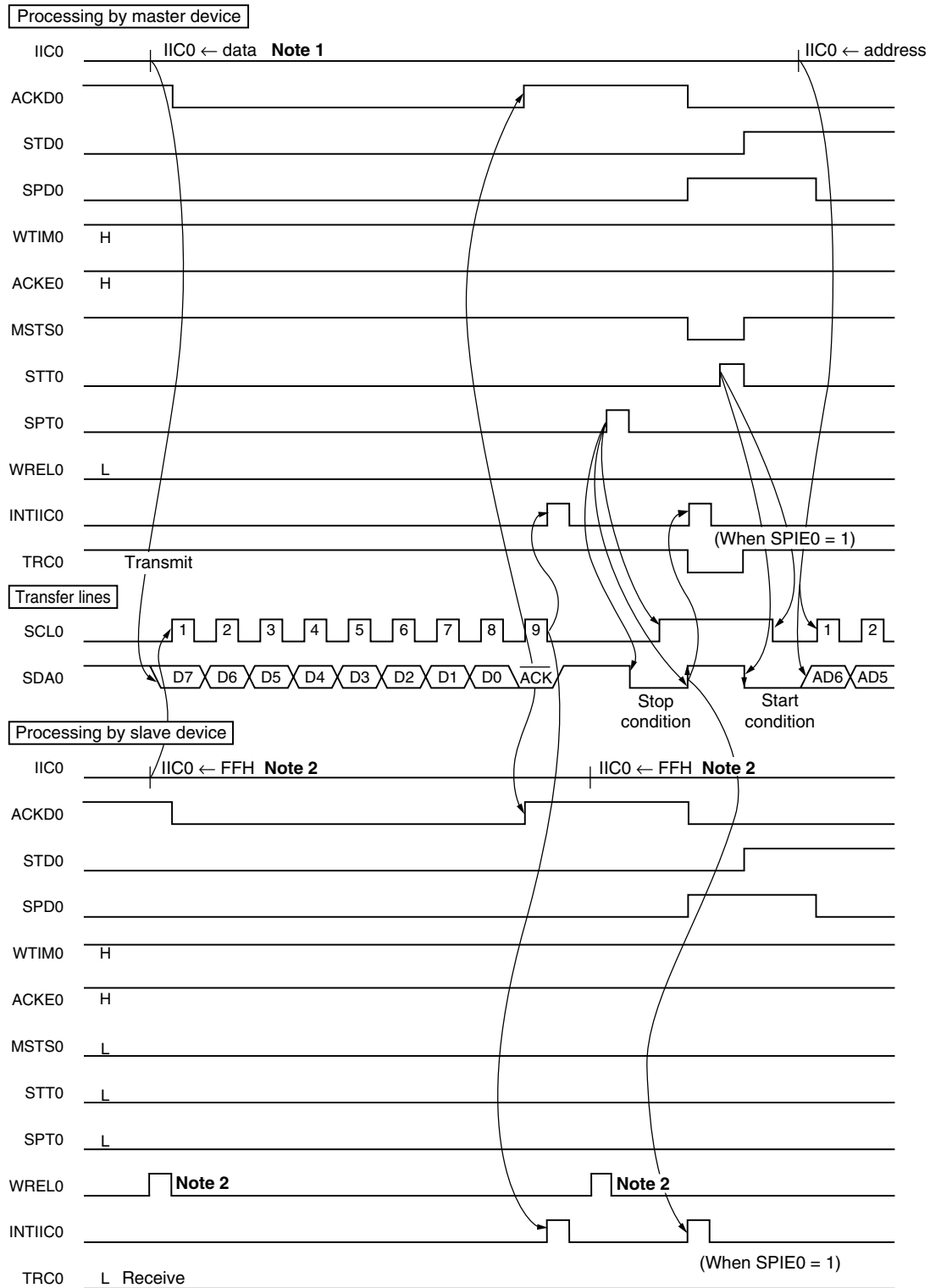
(2) Data



- Notes 1.** Write data to IIC0, not setting WRELO, in order to cancel a wait state during master transmission.
- 2.** To cancel slave wait, write "FFH" to IIC0 or set WRELO.

**Figure 15-27. Example of Master to Slave Communication
(When 9-Clock Wait Is Selected for Both Master and Slave) (3/3)**

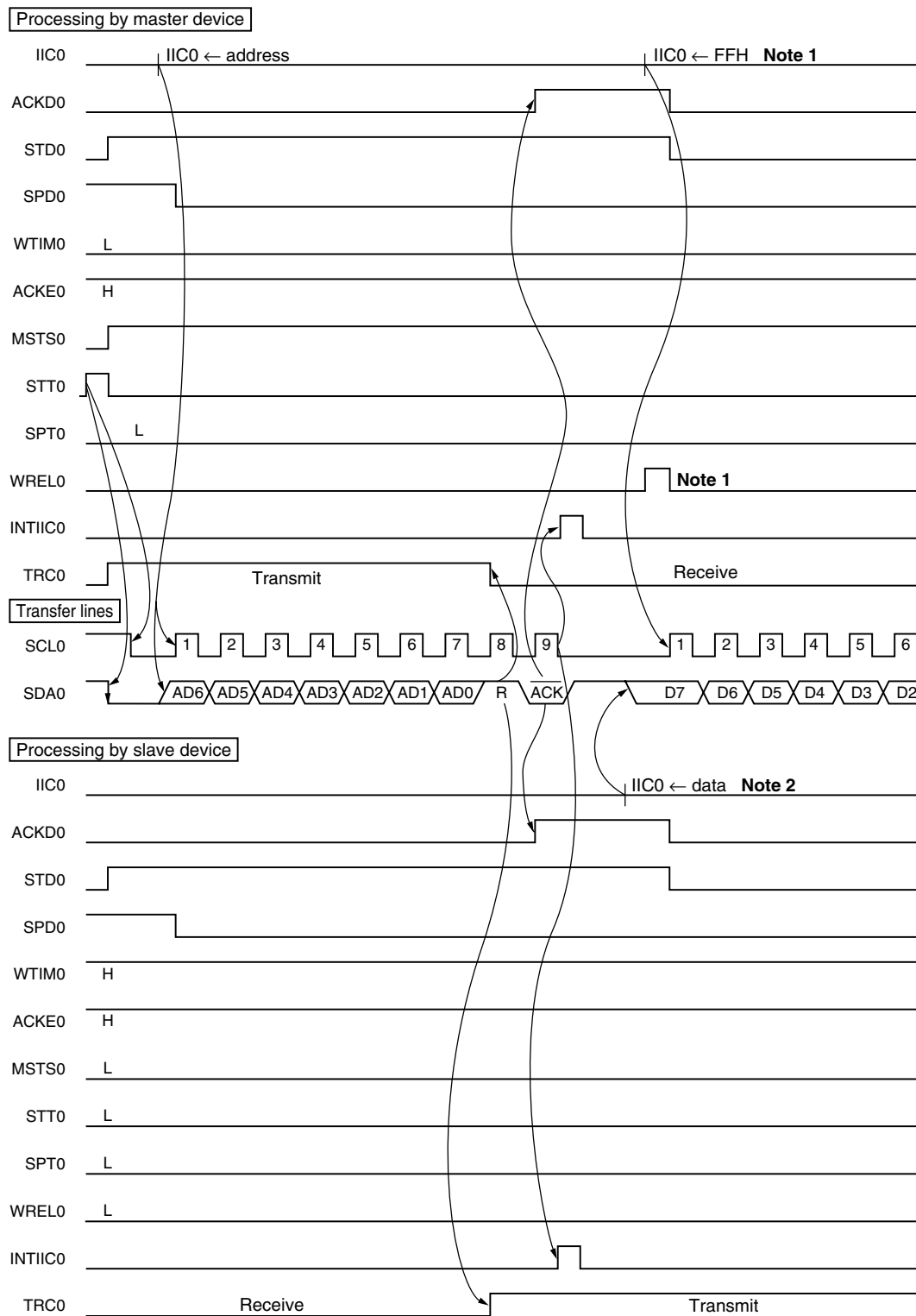
(3) Stop condition



- Notes 1.** Write data to IIC0, not setting WRELO, in order to cancel a wait state during master transmission.
2. To cancel slave wait, write "FFH" to IIC0 or set WRELO.

Figure 15-28. Example of Slave to Master Communication
 (When 8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (1/3)

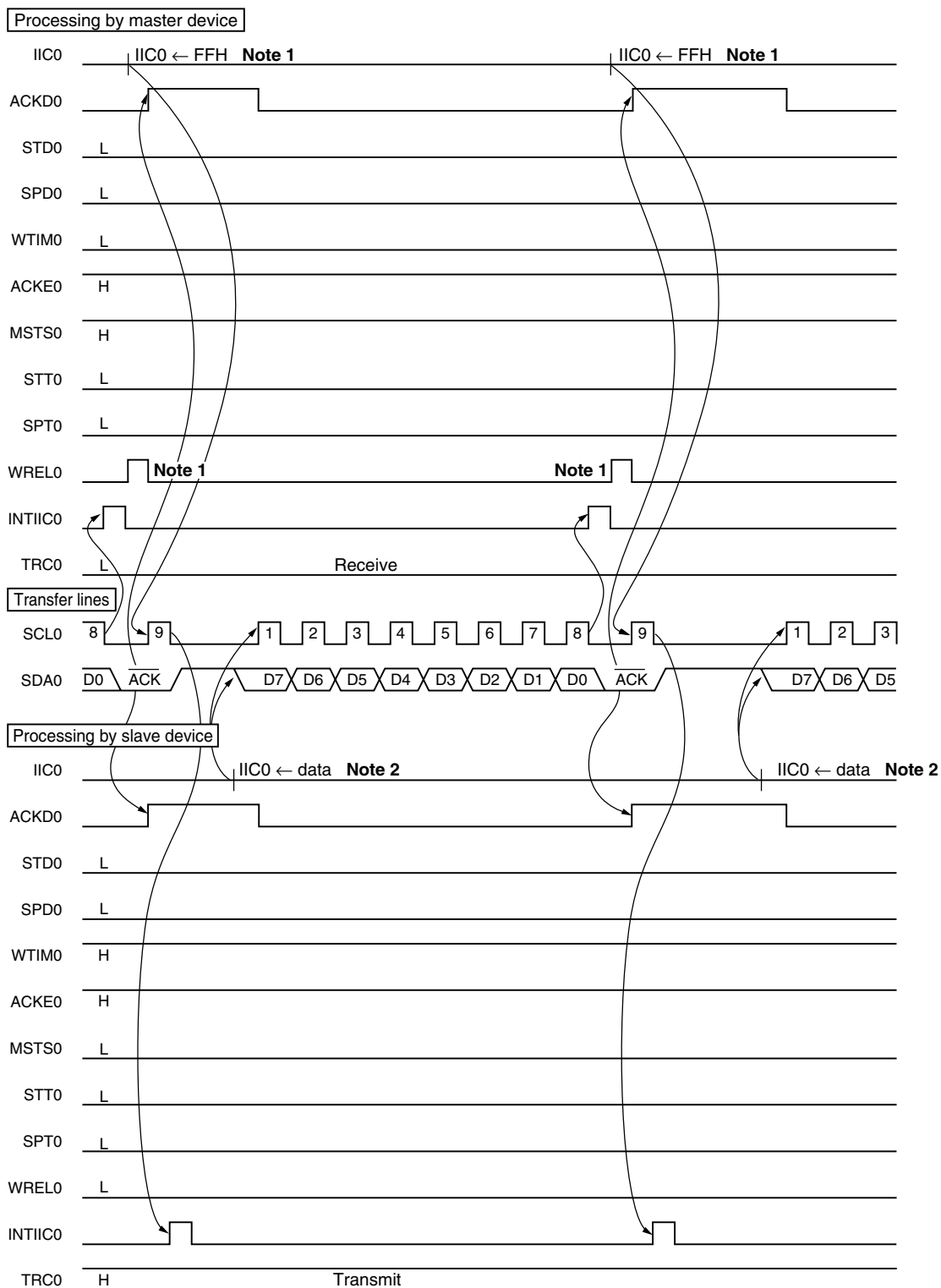
(1) Start condition ~ address



- Notes 1.** To cancel master wait, write "FFH" to IIC0 or set WRELO.
2. Write data to IIC0, not setting WRELO, in order to cancel a wait state during slave transmission.

Figure 15-28. Example of Slave to Master Communication
 (When 8-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (2/3)

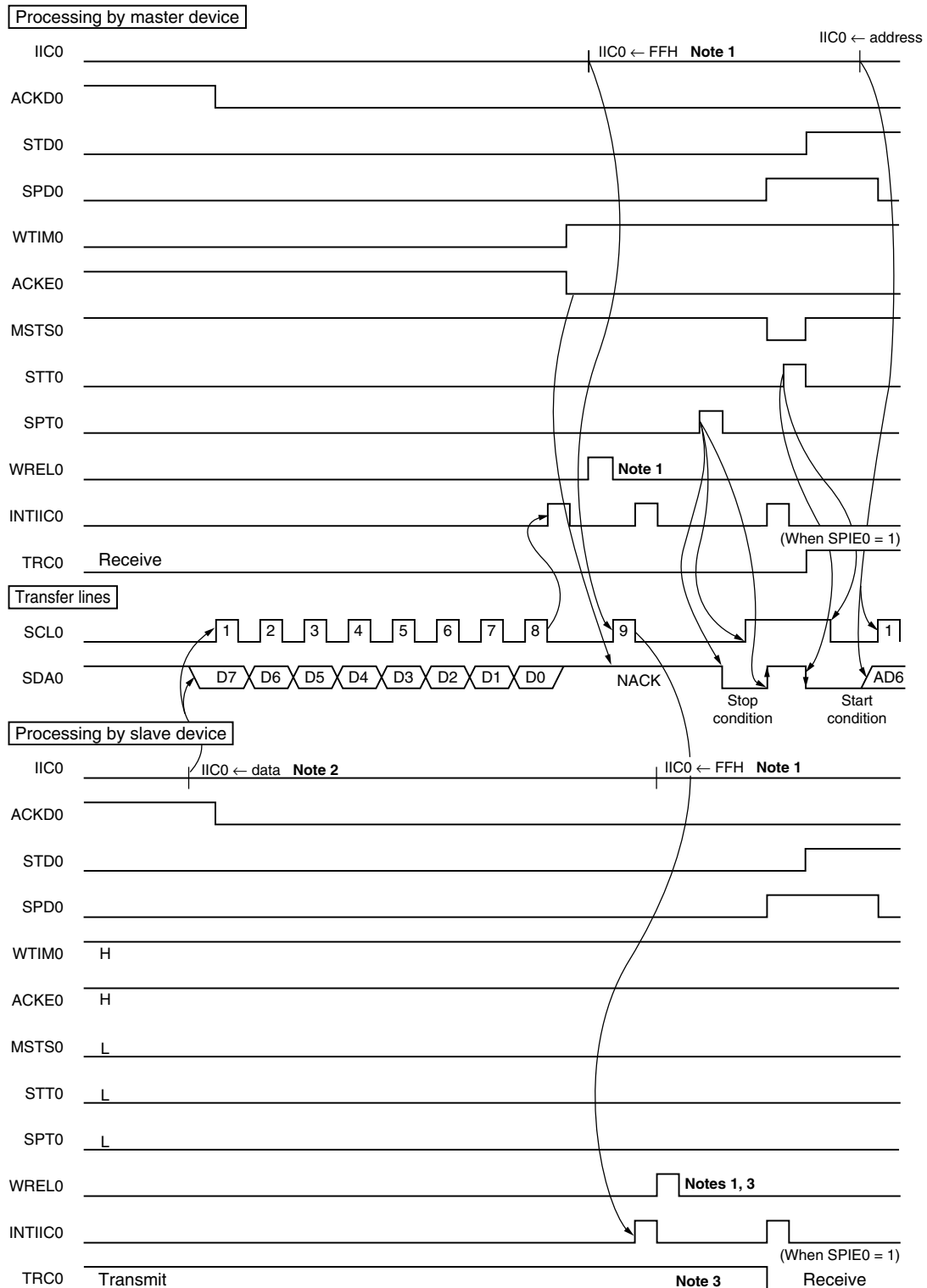
(2) Data



- Notes 1.** To cancel master wait, write "FFH" to IIC0 or set WRELO.
- 2.** Write data to IIC0, not setting WRELO, in order to cancel a wait state during slave transmission.

Figure 15-28. Example of Slave to Master Communication
 (When 8-Clock and 9-Clock Wait Is Selected for Master, 9-Clock Wait Is Selected for Slave) (3/3)

(3) Stop condition



- Notes 1.** To cancel wait, write "FFH" to IIC0 or set WRELO.
- 2.** Write data to IIC0, not setting WRELO, in order to cancel a wait state during slave transmission.
- 3.** If a wait state during slave transmission is canceled by setting WRELO, TRC0 will be cleared.

CHAPTER 16 INTERRUPT FUNCTIONS

16.1 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 a high interrupt priority group and a low interrupt priority group by setting the priority specification flag registers (PROL, PROH, PR1L, PR1H).

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 priority of vectored interrupt servicing. For the priority order, see **Table 16-1**.

A standby release signal is generated and STOP and HALT modes are released.

External interrupt requests and internal interrupt requests are provided as maskable interrupts.

(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.

16.2 Interrupt Sources and Configuration

The interrupt sources consist of maskable interrupts and software interrupts. In addition, they also have up to four reset sources (see **Table 16-1**).

Table 16-1. Interrupt Source List (1/2)

| Interrupt Type | Internal/ External | Basic Configuration Type ^{Note 1} | Default Priority ^{Note 2} | Interrupt Source | | Vector Table Address |
|----------------|--------------------|--|------------------------------------|--|--|----------------------|
| | | | | Name | Trigger | |
| Maskable | Internal | (A) | 0 | INTLVI | Low-voltage detection ^{Note 3} | 0004H |
| | External | (B) | 1 | INTP0 | Pin input edge detection | 0006H |
| | | | 2 | INTP1 | | 0008H |
| | | | 3 | INTP2 | | 000AH |
| | | | 4 | INTP3 | | 000CH |
| | | | 5 | INTP4 | | 000EH |
| | | | 6 | INTP5 | | 0010H |
| | Internal | (A) | 7 | INTSRE6 | UART6 reception error generation | 0012H |
| | | | 8 | INTSR6 | End of UART6 reception | 0014H |
| | | | 9 | INTST6 | End of UART6 transmission | 0016H |
| | | | 10 | INTCSI10/ INTST0 | End of CSI10 communication/end of UART0 transmission | 0018H |
| | | | 11 | INTTMH1 | Match between TMH1 and CMP01 (when compare register is specified) | 001AH |
| | | | 12 | INTTMH0 | Match between TMH0 and CMP00 (when compare register is specified) | 001CH |
| | | | 13 | INTTM50 | Match between TM50 and CR50 (when compare register is specified) | 001EH |
| | | | 14 | INTTM000 | Match between TM00 and CR000 (when compare register is specified), TIO10 pin valid edge detection (when capture register is specified) | 0020H |
| | | | 15 | INTTM010 | Match between TM00 and CR010 (when compare register is specified), TIO00 pin valid edge detection (when capture register is specified) | 0022H |
| | | | 16 | INTAD | End of A/D conversion | 0024H |
| | | | 17 | INTSR0 | End of UART0 reception or reception error generation | 0026H |
| | | | 18 | INTWTI | Watch timer reference time interval signal | 0028H |
| 19 | | | INTTM51 ^{Note 4} | Match between TM51 and CR51 (when compare register is specified) | 002AH | |

- Notes**
- Basic configuration types (A) to (C) correspond to (A) to (C) in **Figure 16-1**.
 - The default priority determines the sequence of processing vectored interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 22 indicates the lowest priority.
 - When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is cleared to 0.
 - When 8-bit timer/event counter 51 is used in the carrier generator mode, an interrupt is generated upon the timing when the INTTM5H1 signal is generated (see **Figure 8-13 Transfer Timing**).

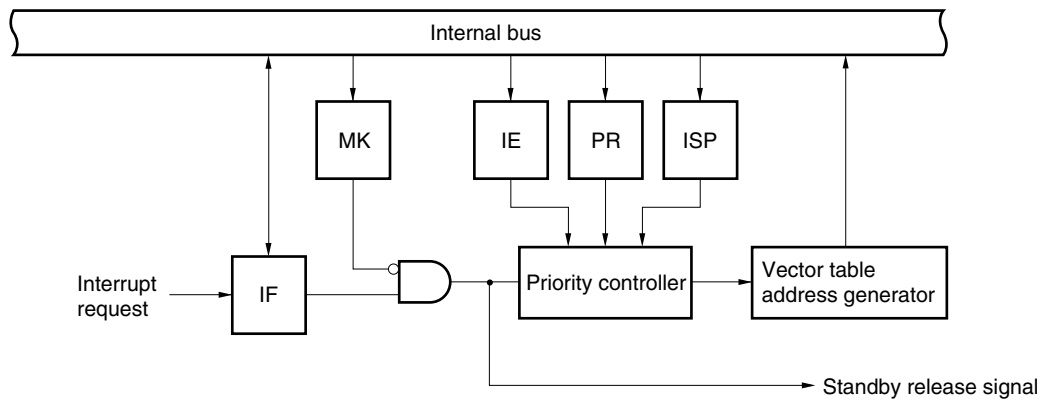
Table 16-1. Interrupt Source List (2/2)

| Interrupt Type | Internal/ External | Basic Configuration Type ^{Note 1} | Default Priority ^{Note 2} | Interrupt Source | | Vector Table Address |
|----------------|--------------------|--|------------------------------------|------------------|---|----------------------|
| | | | | Name | Trigger | |
| Maskable | Internal | (A) | 21 | INTWT | Watch timer overflow | 002EH |
| | Internal | (A) | 22 | INTIIC0 | End of IIC0 communication | 0034H |
| Software | – | (C) | – | BRK | BRK instruction execution | 003EH |
| Reset | – | – | – | RESET | Reset input | 0000H |
| | | | | POC | Power-on clear | |
| | | | | LVI | Low-voltage detection ^{Note 3} | |
| | | | | WDT | WDT overflow | |

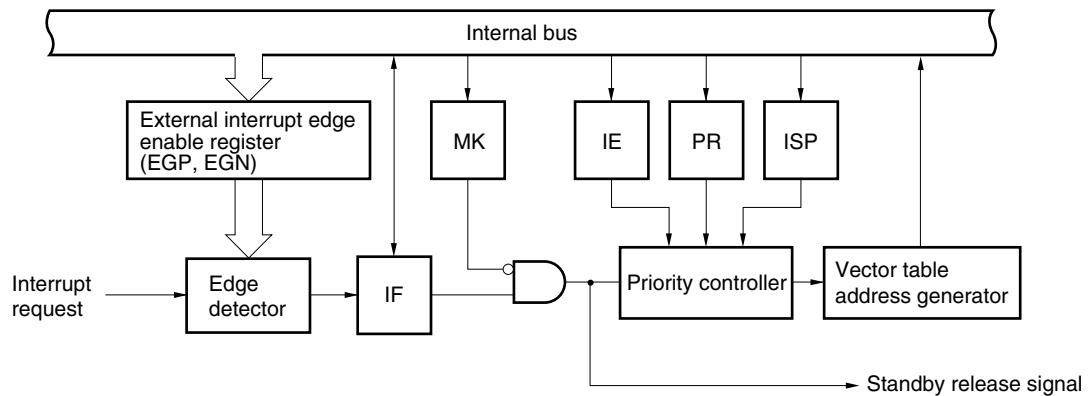
- Notes**
1. Basic configuration types (A) to (C) correspond to (A) to (C) in **Figure 16-1**.
 2. The default priority determines the sequence of processing vectored interrupts if two or more maskable interrupts occur simultaneously. Zero indicates the highest priority and 22 indicates the lowest priority.
 3. When bit 1 (LVIMD) of the low-voltage detection register (LVIM) is set to 1.

Figure 16-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal maskable interrupt



(B) External maskable interrupt (INTPn)

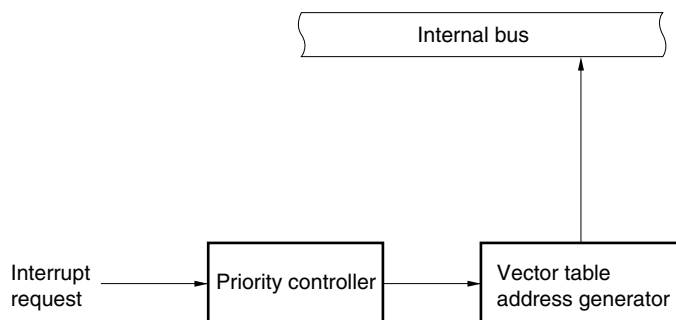


Remark n = 0 to 5

- IF: Interrupt request flag
- IE: Interrupt enable flag
- ISP: In-service priority flag
- MK: Interrupt mask flag
- PR: Priority specification flag

Figure 16-1. Basic Configuration of Interrupt Function (2/2)

(C) Software interrupt



- IF: Interrupt request flag
- IE: Interrupt enable flag
- ISP: In-service priority flag
- MK: Interrupt mask flag
- PR: Priority specification flag

16.3 Registers Controlling Interrupt Functions

The following 6 types of registers are used to control the interrupt functions.

- Interrupt request flag register (IF0L, IF0H, IF1L, IF1H)
- Interrupt mask flag register (MK0L, MK0H, MK1L, MK1H)
- Priority specification flag register (PR0L, PR0H, PR1L, PR1H)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 16-2 shows a list of interrupt request flags, interrupt mask flags, and priority specification flags corresponding to interrupt request sources.

Table 16-2. Flags Corresponding to Interrupt Request Sources (1/2)

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | | | | |
|------------------|---------------------------------|----------|----------------------------------|----------|----------------------------------|----------|----------------------------------|----------------------------------|----------------------------------|
| | | Register | | Register | | Register | | | |
| INTLVI | LVIF | IF0L | LVIMK | MK0L | LVIPR | PR0L | | | |
| INTP0 | PIF0 | | PMK0 | | PPR0 | | | | |
| INTP1 | PIF1 | | PMK1 | | PPR1 | | | | |
| INTP2 | PIF2 | | PMK2 | | PPR2 | | | | |
| INTP3 | PIF3 | | PMK3 | | PPR3 | | | | |
| INTP4 | PIF4 | | PMK4 | | PPR4 | | | | |
| INTP5 | PIF5 | | PMK5 | | PPR5 | | | | |
| INTSRE6 | SREIF6 | | SREMK6 | | SREPR6 | | | | |
| INTSR6 | SRIF6 | IF0H | SRMK6 | MK0H | SRPR6 | PR0H | | | |
| INTST6 | STIF6 | | STMK6 | | STPR6 | | | | |
| INTCSI10 | CSIF10 <small>Note 1</small> | | DUALIF0 <small>Note 1</small> | | CSIMK10 <small>Note 2</small> | | DUALMK0 <small>Note 2</small> | CSIPR10 <small>Note 3</small> | DUALPR0 <small>Note 3</small> |
| INTST0 | STIF0 <small>Note 1</small> | | | | STMK0 <small>Note 2</small> | | | STPR0 <small>Note 3</small> | |
| INTTMH1 | TMIFH1 | | | | TMMKH1 | | | TMPRH1 | |
| INTTMH0 | TMIFH0 | | | | TMMKH0 | | | TMPRH0 | |
| INTTM50 | TMIF50 | | | | TMMK50 | | | TMPR50 | |
| INTTM000 | TMIF000 | | | | TMMK000 | | | TMPR000 | |
| INTTM010 | TMIF010 | | | | TMMK010 | | | TMPR010 | |

- Notes**
1. If either interrupt source INTCSI10 or INTST0 is generated, bit 2 of IF0H is set (1).
 2. Bit 2 of MK0H supports both interrupt sources INTCSI10 and INTST0.
 3. Bit 2 of PR0H supports both interrupt sources INTCSI10 and INTST0.

Table 16-2. Flags Corresponding to Interrupt Request Sources (2/2)

| Interrupt Source | Interrupt Request Flag | | Interrupt Mask Flag | | Priority Specification Flag | |
|-------------------------|------------------------|----------|---------------------|----------|-----------------------------|----------|
| | | Register | | Register | | Register |
| INTAD | ADIF | IF1L | ADMK | MK1L | ADPR | PR1L |
| INTSR0 | SRIF0 | | SRMK0 | | SRPR0 | |
| INTWTI | WTIIF | | WTIMK | | WTIPR | |
| INTTM51 ^{Note} | TMIF51 | | TMMK51 | | TMPR51 | |
| INTWT | WTIF | | WTMK | | WTPR | |
| INTIIC0 | IICIF0 | IF1H | IICMK0 | MK1H | IICPR0 | PR1H |

Note When 8-bit timer/event counter 51 is used in the carrier generator mode, an interrupt is generated upon the timing when the INTTM5H1 signal is generated (see **Figure 8-13 Transfer Timing**).

(1) Interrupt request flag registers (IF0L, IF0H, IF1L, IF1H)

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.

IF0L, IF0H, IF1L, and IF1H are set by a 1-bit or 8-bit memory manipulation instruction. When IF0L and IF0H, and IF1L and IF1H are combined to form 16-bit registers IF0 and IF1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

- Cautions**
1. When operating a timer, serial interface, or A/D converter after standby release, operate it once after clearing the interrupt request flag. An interrupt request flag may be set by noise.
 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 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.

Figure 16-2. Format of Interrupt Request Flag Registers (IF0L, IF0H, IF1L, IF1H)

Address: FFE0H After reset: 00H R/W

| | | | | | | | | |
|--------|--------|------|------|------|------|------|------|------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| IF0L | SREIF6 | PIF5 | PIF4 | PIF3 | PIF2 | PIF1 | PIF0 | LVIF |

Address: FFE1H After reset: 00H R/W

| | | | | | | | | |
|--------|---------|---------|--------|--------|--------|----------------------------|-------|-------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| IF0H | TMIF010 | TMIF000 | TMIF50 | TMIFH0 | TMIFH1 | DUALIF0 CSIF10 STIF0 | STIF6 | SRIF6 |

Address: FFE2H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|------|---|--------|-------|-------|------|
| Symbol | 7 | 6 | <5> | 4 | <3> | <2> | <1> | <0> |
| IF1L | 0 | 0 | WTIF | 0 | TMIF51 | WTIIF | SRIF0 | ADIF |

Address: FFE3H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
| IF1H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | IICIF0 |

| | |
|-------|--|
| XXIFX | Interrupt request flag |
| 0 | No interrupt request signal is generated |
| 1 | Interrupt request is generated, interrupt request status |

Caution Be sure to clear bits 4, 6, and 7 of IF1L, and bits 1 to 7 of IF1H to 0.

(2) Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H)

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing.

MK0L, MK0H, MK1L, and MK1H are set by a 1-bit or 8-bit memory manipulation instruction. When MK0L and MK0H, and MK1L and MK1H are combined to form 16-bit registers MK0 and MK1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 16-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H)

Address: FFE4H After reset: FFH R/W

| | | | | | | | | |
|--------|--------|------|------|------|------|------|------|-------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| MK0L | SREMK6 | PMK5 | PMK4 | PMK3 | PMK2 | PMK1 | PMK0 | LVIMK |

Address: FFE5H After reset: FFH R/W

| | | | | | | | | |
|--------|---------|---------|--------|--------|--------|-----------------------------|-------|-------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| MK0H | TMMK010 | TMMK000 | TMMK50 | TMMKH0 | TMMKH1 | DUALMK0 CSIMK10 STMK0 | STMK6 | SRMK6 |

Address: FFE6H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|------|---|--------|-------|-------|------|
| Symbol | 7 | 6 | <5> | 4 | <3> | <2> | <1> | <0> |
| MK1L | 1 | 1 | WTMK | 1 | TMMK51 | WTIMK | SRMK0 | ADMK |

Address: FFE7H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
| MK1H | 1 | 1 | 1 | 1 | 1 | 1 | 1 | IICMK0 |

| | |
|-------|------------------------------|
| XXMKX | Interrupt servicing control |
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled |

Caution Be sure to set bits 4, 6, and 7 of MK1L and bits 1 to 7 of MK1H to 1.

(3) Priority specification flag registers (PR0L, PR0H, PR1L, PR1H)

The priority specification flag registers are used to set the corresponding maskable interrupt priority order.

PR0L, PR0H, PR1L, and PR1H are set by a 1-bit or 8-bit memory manipulation instruction. If PR0L and PR0H, and PR1L and PR1H are combined to form 16-bit registers PR0 and PR1, they are set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

Figure 16-4. Format of Priority Specification Flag Registers (PR0L, PR0H, PR1L, PR1H)

Address: FFE8H After reset: FFH R/W

| | | | | | | | | |
|--------|--------|------|------|------|------|------|------|-------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| PR0L | SREPR6 | PPR5 | PPR4 | PPR3 | PPR2 | PPR1 | PPR0 | LVIPR |

Address: FFE9H After reset: FFH R/W

| | | | | | | | | |
|--------|---------|---------|--------|--------|--------|-----------------------------|-------|-------|
| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
| PR0H | TMPR010 | TMPR000 | TMPR50 | TMPRH0 | TMPRH1 | DUALPR0 CSIPR10 STPR0 | STPR6 | SRPR6 |

Address: FFEAH After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|------|---|--------|-------|-------|------|
| Symbol | 7 | 6 | <5> | 4 | <3> | <2> | <1> | <0> |
| PR1L | 1 | 1 | WTPR | 1 | TMPR51 | WTIPR | SRPR0 | ADPR |

Address: FFE9H After reset: FFH R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|--------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
| PR1H | 1 | 1 | 1 | 1 | 1 | 1 | 1 | IICPR0 |

| | |
|-------|--------------------------|
| XXPRX | Priority level selection |
| 0 | High priority level |
| 1 | Low priority level |

Caution Be sure to set bits 4, 6, and 7 of PR1L and bits 1 to 7 of PR1H to 1.

(4) External interrupt rising edge enable register (EGP), external interrupt falling edge enable register (EGN)

These registers specify the valid edge for INTP_n.

EGP and EGN are set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears these registers to 00H.

Remark n = 0 to 5

Figure 16-5. Format of External Interrupt Rising Edge Enable Register (EGP) and External Interrupt Falling Edge Enable Register (EGN)

Address: FF48H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EGP | 0 | 0 | EGP5 | EGP4 | EGP3 | EGP2 | EGP1 | EGP0 |

Address: FF49H After reset: 00H R/W

| | | | | | | | | |
|--------|---|---|------|------|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EGN | 0 | 0 | EGN5 | EGN4 | EGN3 | EGN2 | EGN1 | EGN0 |

| EGP _n | EGN _n | INTP _n pin valid edge selection |
|------------------|------------------|--|
| 0 | 0 | Edge detection disabled |
| 0 | 1 | Falling edge |
| 1 | 0 | Rising edge |
| 1 | 1 | Both rising and falling edges |

Caution Be sure to clear bits 6 and 7 of EGP and EGN to 0.

Remark n = 0 to 5

Table 16-3 shows the ports corresponding to EGP_n and EGN_n.

Table 16-3. Ports Corresponding to EGP_n and EGN_n

| Detection Enable Register | | Edge Detection Port | Interrupt Request Signal |
|---------------------------|------|---------------------|--------------------------|
| EGP0 | EGN0 | P120 | INTP0 |
| EGP1 | EGN1 | P30 | INTP1 |
| EGP2 | EGN2 | P31 | INTP2 |
| EGP3 | EGN3 | P32 | INTP3 |
| EGP4 | EGN4 | P33 | INTP4 |
| EGP5 | EGN5 | P16 | INTP5 |

Caution Select the port mode by clearing EGP_n and EGN_n to 0 because an edge may be detected when the external interrupt function is switched to the port function.

Remark n = 0 to 5

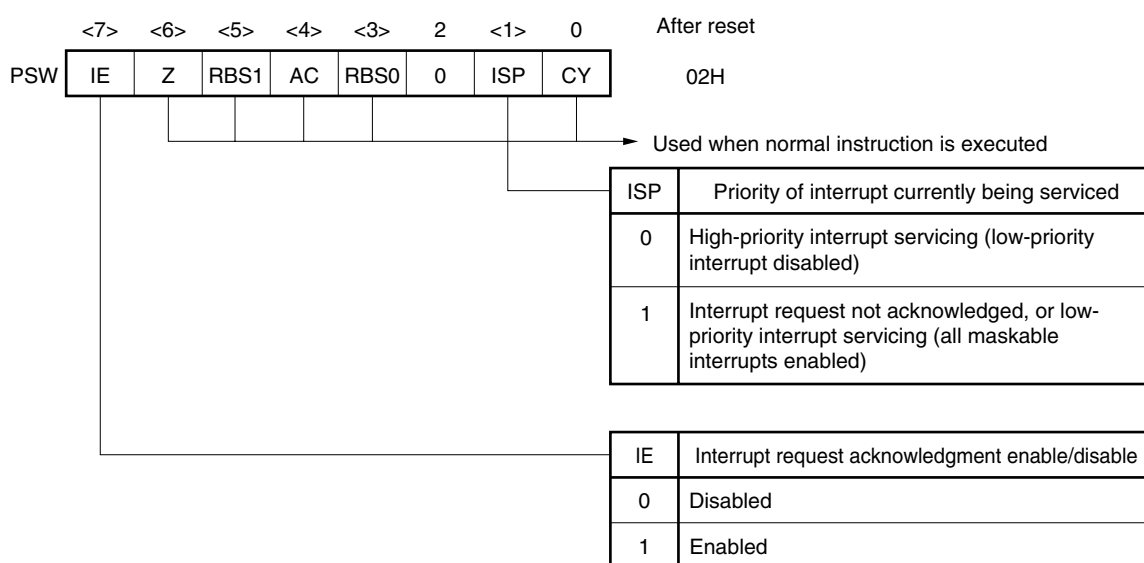
(5) Program status word (PSW)

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 ISP flag 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. If a maskable interrupt request is acknowledged, the contents of the priority specification flag of the acknowledged interrupt are transferred to the ISP flag. 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 02H.

Figure 16-6. Format of Program Status Word



16.4 Interrupt Servicing Operations

16.4.1 Maskable interrupt acknowledgment

A maskable interrupt 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 (when the ISP flag is reset to 0).

The times from generation of a maskable interrupt request until vectored interrupt servicing is performed are listed in Table 16-4 below.

For the interrupt request acknowledgment timing, see **Figures 16-8** and **16-9**.

Table 16-4. Time from Generation of Maskable Interrupt Until Servicing

| | Minimum Time | Maximum Time ^{Note} |
|----------------------------|--------------|------------------------------|
| When $\times\times PR = 0$ | 7 clocks | 32 clocks |
| When $\times\times PR = 1$ | 8 clocks | 33 clocks |

Note If an interrupt request is generated just before a divide instruction, the wait time becomes longer.

Remark 1 clock: $1/f_{CPU}$ (f_{CPU} : 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.

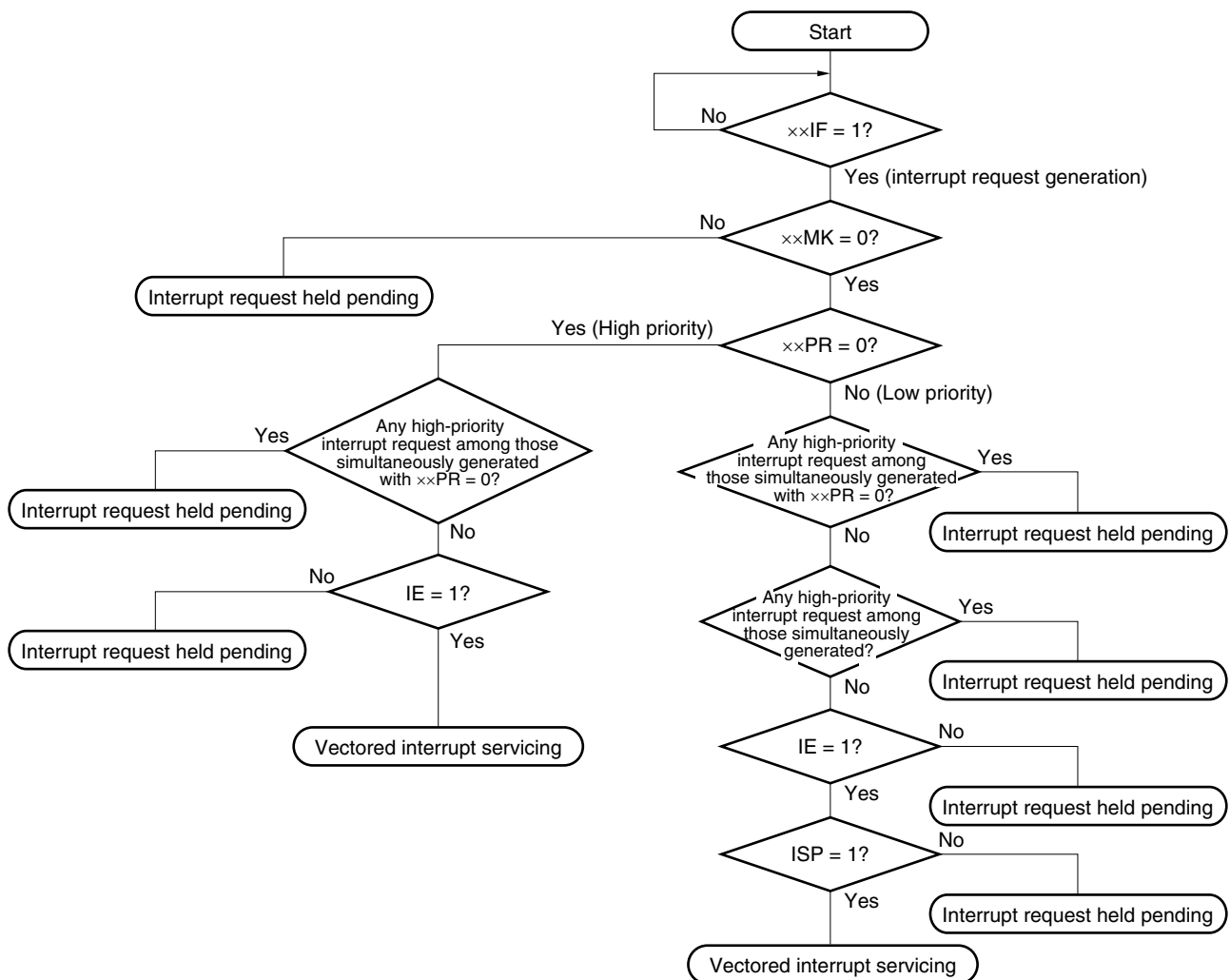
An interrupt request that is held pending is acknowledged when it becomes acknowledgeable.

Figure 16-7 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 ISP flag. 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 16-7. Interrupt Request Acknowledgment Processing Algorithm



xxIF: Interrupt request flag

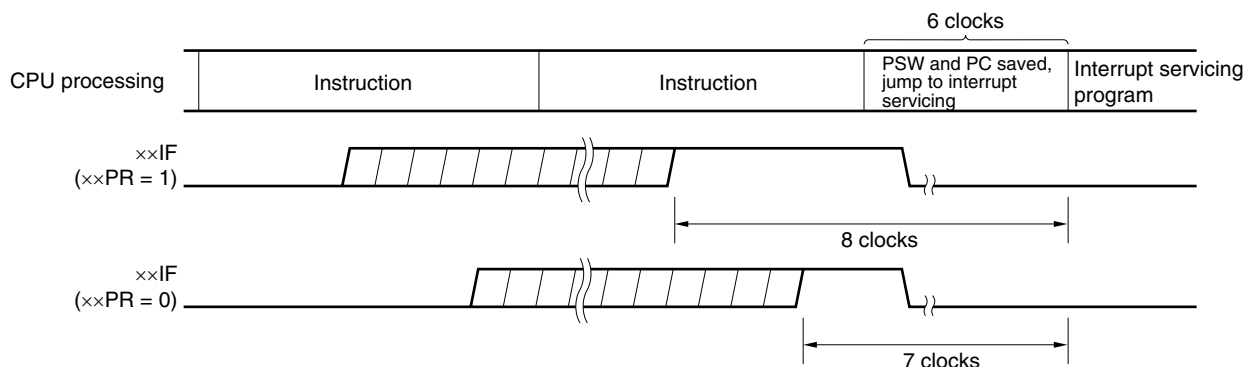
xxMK: Interrupt mask flag

xxPR: Priority specification flag

IE: Flag that controls acknowledgment of maskable interrupt request (1 = Enable, 0 = Disable)

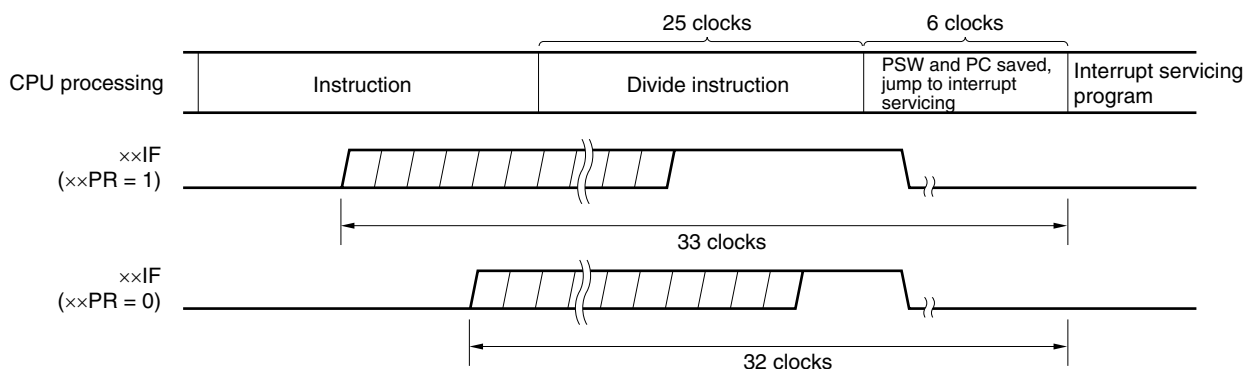
ISP: Flag that indicates the priority level of the interrupt currently being serviced (0 = high-priority interrupt servicing, 1 = No interrupt request acknowledged, or low-priority interrupt servicing)

Figure 16-8. Interrupt Request Acknowledgment Timing (Minimum Time)



Remark 1 clock: $1/f_{CPU}$ (f_{CPU} : CPU clock)

Figure 16-9. Interrupt Request Acknowledgment Timing (Maximum Time)



Remark 1 clock: $1/f_{CPU}$ (f_{CPU} : CPU clock)

16.4.2 Software interrupt request acknowledgment

A software interrupt acknowledge 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 (003EH, 003FH) are loaded into the PC and branched.

Restoring from a software interrupt is possible by using the RETB instruction.

Caution Do not use the RETI instruction for restoring from the software interrupt.

16.4.3 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 equal to or higher than that of the interrupt currently being serviced is generated, it is acknowledged for multiple interrupt servicing. If an interrupt with a priority 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.

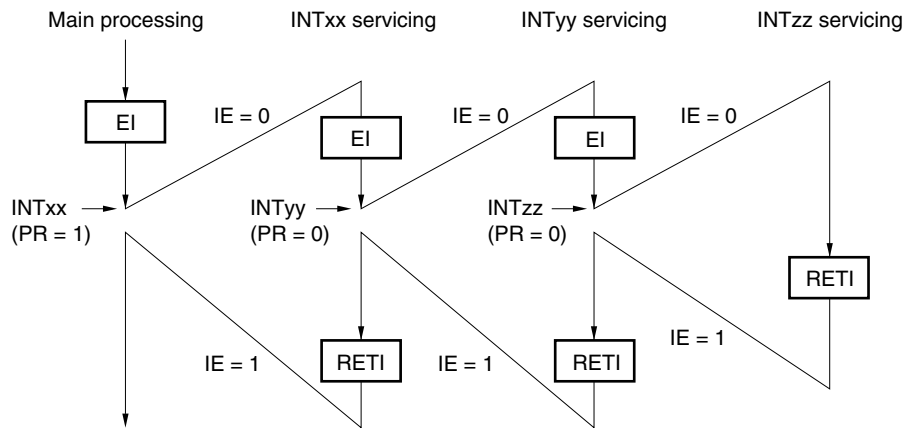
Table 16-5 shows relationship between interrupt requests enabled for multiple interrupt servicing and Figure 16-10 shows multiple interrupt servicing examples.

Table 16-5. Relationship Between Interrupt Requests Enabled for Multiple Interrupt Servicing During Interrupt Servicing

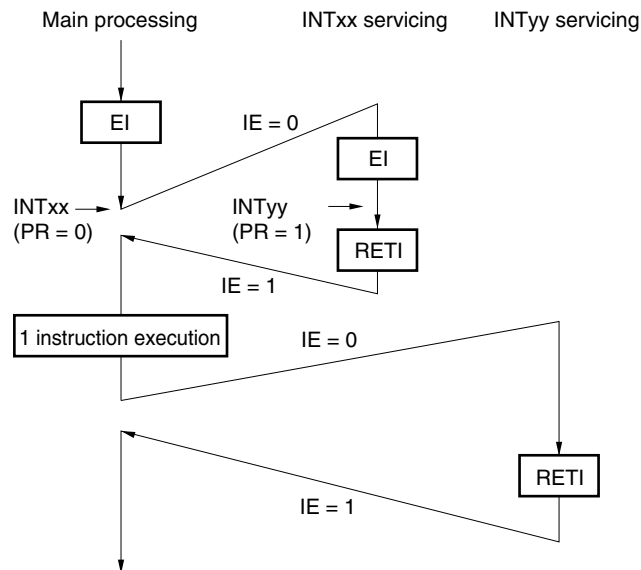
| Multiple Interrupt Request Interrupt Being Serviced | | Maskable Interrupt Request | | | | Software Interrupt Request |
|--|---------|----------------------------|--------|--------|--------|----------------------------------|
| | | PR = 0 | | PR = 1 | | |
| | | IE = 1 | IE = 0 | IE = 1 | IE = 0 | |
| Maskable interrupt | ISP = 0 | ○ | × | × | × | ○ |
| | ISP = 1 | ○ | × | ○ | × | ○ |
| Software interrupt | | ○ | × | ○ | × | ○ |

- Remarks**
- : Multiple interrupt servicing enabled
 - ×: Multiple interrupt servicing disabled
 - ISP and IE are flags contained in the PSW.
ISP = 0: An interrupt with higher priority is being serviced.
ISP = 1: No interrupt request has been acknowledged, or an interrupt with a lower priority is being serviced.
IE = 0: Interrupt request acknowledgment is disabled.
IE = 1: Interrupt request acknowledgment is enabled.
 - PR is a flag contained in PR0L, PR0H, PR1L, and PR1H.
PR = 0: Higher priority level
PR = 1: Lower priority level

Figure 16-10. 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

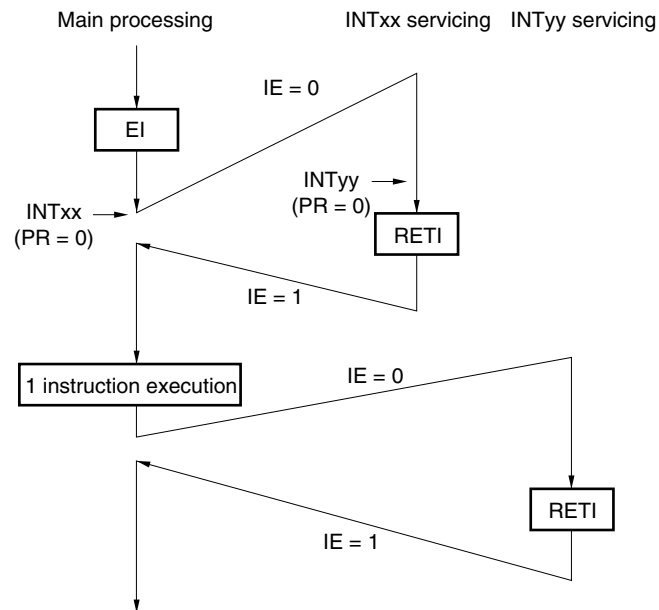
Interrupt request INTyy issued during servicing of interrupt INTxx is not acknowledged because its priority is lower than that of INTxx, and multiple interrupt servicing does not take place. The INTyy interrupt request is held pending, and is acknowledged following execution of one main processing instruction.

PR = 0: Higher priority level

PR = 1: Lower priority level

IE = 0: Interrupt request acknowledgment disabled

Figure 16-10. 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 = 0: Higher priority level

IE = 0: Interrupt request acknowledgment disabled

16.4.4 Interrupt request hold

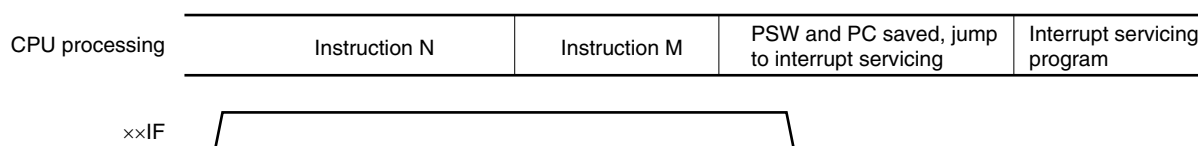
There are instructions where, even if an interrupt request is issued for them while another instruction is being executed, request acknowledgment is held pending until the end of execution of the next instruction. These instructions (interrupt request hold instructions) are listed below.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW. bit, CY
- MOV1 CY, PSW. bit
- AND1 CY, PSW. bit
- OR1 CY, PSW. bit
- XOR1 CY, PSW. bit
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW. bit, \$addr16
- BF PSW. bit, \$addr16
- BTCLR PSW. bit, \$addr16
- EI
- DI
- Manipulation instructions for the IF0L, IF0H, IF1L, IF1H, MK0L, MK0H, MK1L, MK1H, PR0L, PR0H, PR1L, and PR1H registers.

Caution The BRK instruction is not one of the above-listed interrupt request hold instructions. However, the software interrupt activated by executing the BRK instruction causes the IE flag to be cleared. Therefore, even if a maskable interrupt request is generated during execution of the BRK instruction, the interrupt request is not acknowledged.

Figure 16-11 shows the timing at which interrupt requests are held pending.

Figure 16-11. Interrupt Request Hold



- Remarks**
1. Instruction N: Interrupt request hold instruction
 2. Instruction M: Instruction other than interrupt request hold instruction
 3. The $\times\times$ PR (priority level) values do not affect the operation of $\times\times$ IF (interrupt request).

CHAPTER 17 STANDBY FUNCTION

17.1 Standby Function and Configuration

17.1.1 Standby function

The standby function is designed to reduce the operating current of the system. The following two 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, internal high-speed oscillator or internal low-speed 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 internal high-speed 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.

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.
 2. The following sequence is recommended for operating current reduction of the A/D converter when the standby function is used: First clear bit 7 (ADCS) and bit 0 (ADCE) of the A/D converter mode register (ADM) to 0 to stop the A/D conversion operation, and then execute the STOP instruction.

17.1.2 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)

Remark For the registers that start, stop, or select the clock, see **CHAPTER 5 CLOCK GENERATOR**.

(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. When X1 clock oscillation starts with the internal high-speed oscillation clock used as the CPU clock, the X1 clock oscillation stabilization time can be checked.

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

When reset is released (reset by $\overline{\text{RESET}}$ input, POC, LVI, and WDT), the STOP instruction and MSTOP (bit 7 of MOC register) = 1 clear OSTC to 00H.

Figure 17-1. Format of Oscillation Stabilization Time Counter Status Register (OSTC)

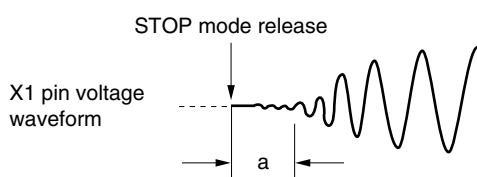
Address: FFA3H After reset: 00H R

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|--------|--------|--------|--------|--------|
| OSTC | 0 | 0 | 0 | MOST11 | MOST13 | MOST14 | MOST15 | MOST16 |

| MOST11 | MOST13 | MOST14 | MOST15 | MOST16 | Oscillation stabilization time status | |
|--------|--------|--------|--------|--------|---------------------------------------|---------------|
| | | | | | f _x = 10 MHz | |
| 1 | 0 | 0 | 0 | 0 | 2 ¹¹ /f _x min. | 204.8 μs min. |
| 1 | 1 | 0 | 0 | 0 | 2 ¹³ /f _x min. | 819.2 μs min. |
| 1 | 1 | 1 | 0 | 0 | 2 ¹⁴ /f _x min. | 1.64 ms min. |
| 1 | 1 | 1 | 1 | 0 | 2 ¹⁵ /f _x min. | 3.27 ms min. |
| 1 | 1 | 1 | 1 | 1 | 2 ¹⁶ /f _x min. | 6.55 ms min. |

- Cautions**
1. After the above time has elapsed, the bits are set to 1 in order from MOST11 and remain 1.
 2. The oscillation stabilization time counter counts up to the oscillation stabilization time set by OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.
 - Desired OSTC oscillation stabilization time ≤ Oscillation stabilization time set by OSTS

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC 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

(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 OSTS after the STOP mode is released.

When the internal high-speed oscillation clock is selected as the CPU clock, confirm with 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 OSTC.

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

Reset signal generation sets OSTS to 05H.

Figure 17-2. Format of Oscillation Stabilization Time Select Register (OSTS)

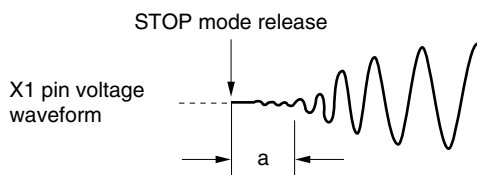
Address: FFA4H After reset: 05H R/W

| | | | | | | | | |
|--------|---|---|---|---|---|-------|-------|-------|
| 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 | |
|------------------|-------|-------|--|------------------------|
| | | | | $f_x = 10 \text{ MHz}$ |
| 0 | 0 | 1 | $2^{11}/f_x$ | 204.8 μs |
| 0 | 1 | 0 | $2^{13}/f_x$ | 819.2 μs |
| 0 | 1 | 1 | $2^{14}/f_x$ | 1.64 ms |
| 1 | 0 | 0 | $2^{15}/f_x$ | 3.27 ms |
| 1 | 0 | 1 | $2^{16}/f_x$ | 6.55 ms |
| Other than above | | | Setting prohibited | |

- Cautions**
- To set the STOP mode when the X1 clock is used as the CPU clock, set OSTS before executing the STOP instruction.
 - 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 OSTS. If the STOP mode is entered and then released while the internal high-speed oscillation clock is being used as the CPU clock, set the oscillation stabilization time as follows.
 - Desired OSTC oscillation stabilization time \leq Oscillation stabilization time set by OSTS

Note, therefore, that only the status up to the oscillation stabilization time set by OSTS is set to OSTC 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

17.2 Standby Function Operation

17.2.1 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 internal high-speed oscillation clock.

The operating statuses in the HALT mode are shown below.

Table 17-1. 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 Internal High-Speed Oscillation Clock (f_{RH}) | When CPU Is Operating on X1 Clock (f_x) | When CPU Is Operating on External Main System Clock (f_{EXCLK}) |
| Item | | | | |
| System clock | | Clock supply to the CPU is stopped | | |
| Main system clock | f_{RH} | Operation continues (cannot be stopped) | Status before HALT mode was set is retained | |
| | f_x | Status before HALT mode was set is retained | Operation continues (cannot be stopped) | Status before HALT mode was set is retained |
| | f_{EXCLK} | Operates or stops by external clock input | | Operation continues (cannot be stopped) |
| | f_{RL} | Status before HALT mode was set is retained | | |
| CPU | | Operation stopped | | |
| Flash memory | | | | |
| RAM | | Status before HALT mode was set is retained | | |
| Port (latch) | | | | |
| 16-bit timer/event counter | 00 | Operable | | |
| 8-bit timer/event counter | 50 | | | |
| | 51 | | | |
| 8-bit timer | H0 | | | |
| | H1 | | | |
| Watch timer | | | | |
| Watchdog timer | | Operable. Clock supply to watchdog timer stops when "internal low-speed oscillator can be stopped by software" is set by option byte. | | |
| A/D converter | | Operable | | |
| Serial interface | UART0 | | | |
| | UART6 | | | |
| | CSI10 | | | |
| | IIC0 | | | |
| Power-on-clear function | | | | |
| Low-voltage detection function | | | | |
| External interrupt | | | | |

Remarks 1. f_{RH} : Internal high-speed oscillation clock, f_x : X1 clock

f_{EXCLK} : External main system clock, f_{RL} : Internal low-speed oscillation clock

2. The functions mounted depend on the product. See **1.6 Block Diagram** and **1.7 Outline of Functions**.

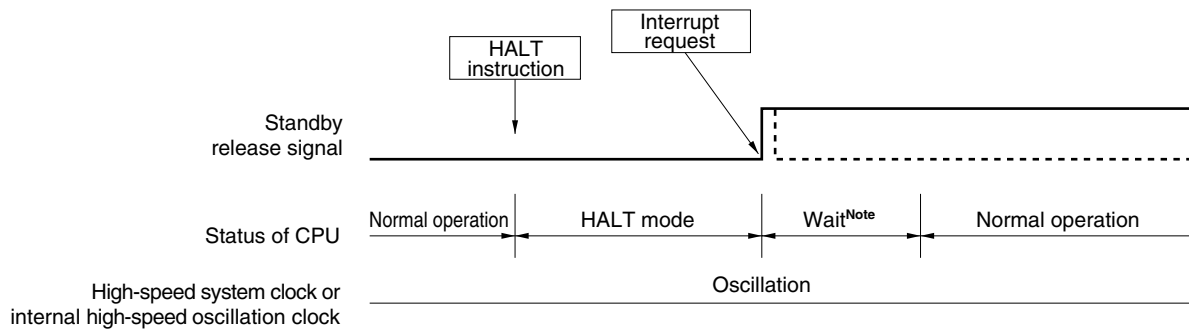
(2) HALT mode release

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 17-3. HALT Mode Release by Interrupt Request Generation



Note The wait time is as follows:

- When vectored interrupt servicing is carried out: 11 or 12 clocks
- When vectored interrupt servicing is not carried out: 4 or 5 clocks

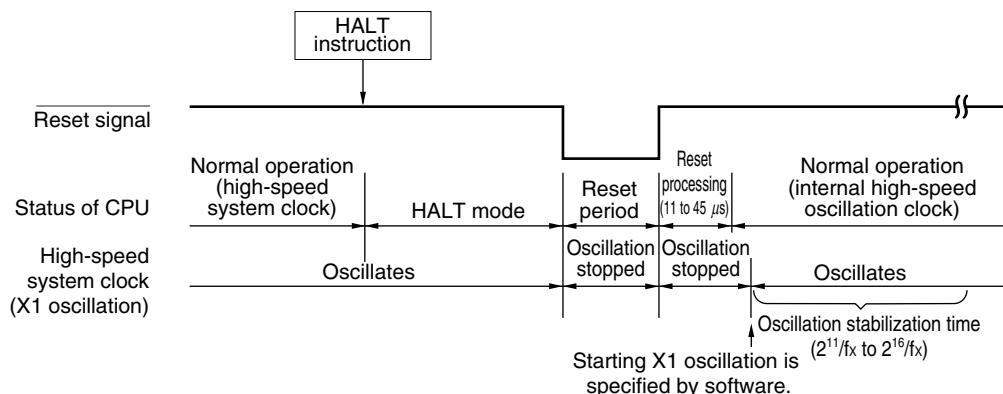
Remark The broken lines indicate the case when the interrupt request which has released the standby mode is acknowledged.

(b) Release by reset signal generation

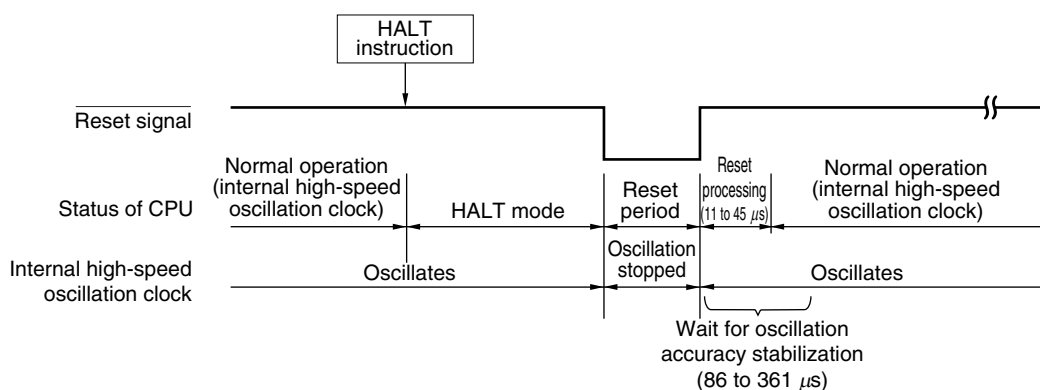
When the reset signal is generated, HALT mode is released, and then, as in the case with a normal reset operation, the program is executed after branching to the reset vector address.

Figure 17-4. HALT Mode Release by Reset

(1) When high-speed system clock is used as CPU clock



(2) When internal high-speed oscillation clock is used as CPU clock



Remark fx: X1 clock oscillation frequency

Table 17-2. Operation in Response to Interrupt Request in HALT Mode

| Release Source | MK _{xx} | PR _{xx} | IE | ISP | Operation |
|----------------------------|------------------|------------------|----|-----|------------------------------------|
| Maskable interrupt request | 0 | 0 | 0 | × | Next address instruction execution |
| | 0 | 0 | 1 | × | Interrupt servicing execution |
| | 0 | 1 | 0 | 1 | Next address instruction execution |
| | 0 | 1 | × | 0 | |
| | 0 | 1 | 1 | 1 | Interrupt servicing execution |
| | 1 | × | × | × | HALT mode held |
| Reset | – | – | × | × | Reset processing |

×: don't care

17.2.2 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 main system clock.

Caution Because the interrupt request signal is used to clear the standby mode, if there is an interrupt source with the interrupt request flag set and the interrupt mask flag reset, the standby mode is immediately cleared if set. Thus, the STOP mode is reset to the HALT mode immediately after execution of the STOP instruction and the system returns to the operating mode as soon as the wait time set using the oscillation stabilization time select register (OSTS) has elapsed.

The operating statuses in the STOP mode are shown below.

Table 17-3. 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 Internal High-Speed Oscillation Clock (f_{RH}) | When CPU Is Operating on X1 Clock (f_x) | When CPU Is Operating on External Main System Clock (f_{EXCLK}) |
| Item | | | | |
| System clock | | Clock supply to the CPU is stopped | | |
| Main system clock | f_{RH} | Stopped | | |
| | f_x | Stopped | | |
| | f_{EXCLK} | Input invalid | | |
| | f_{RL} | Status before STOP mode was set is retained | | |
| CPU | | Operation stopped | | |
| Flash memory | | Operation stopped | | |
| RAM | | Status before STOP mode was set is retained | | |
| Port (latch) | | Status before STOP mode was set is retained | | |
| 16-bit timer/event counter | 00 | Operation stopped | | |
| 8-bit timer/event counter | 50 ^{Note} | Operable only when TI50 is selected as the count clock | | |
| | 51 ^{Note} | Operable only when TI51 is selected as the count clock | | |
| 8-bit timer | H0 | Operable only when TM50 output is selected as the count clock during 8-bit timer/event counter 50 operation | | |
| | H1 | Operable only when f_{RL} , $f_{RL}/2^7$, $f_{RL}/2^9$ is selected as the count clock | | |
| Watch timer | | Operation stopped | | |
| Watchdog timer | | Operable. Clock supply to watchdog timer stops when "internal low-speed oscillator can be stopped by software" is set by option byte. | | |
| A/D converter | | Operation stopped | | |
| Serial interface | UART0 | Operable only when TM50 output is selected as the serial clock during 8-bit timer/event counter 50 operation | | |
| | UART6 | Operable only when TM50 output is selected as the serial clock during 8-bit timer/event counter 50 operation | | |
| | CSI10 ^{Note} | Operable only when external clock is selected as the serial clock | | |
| | IIC0 | Operable only when the external clock from EXSCL0/P62 pin is selected as the serial clock ^{Note 2} | | |
| Power-on-clear function | | Operable | | |
| Low-voltage detection function | | Operable | | |
| External interrupt | | Operable | | |

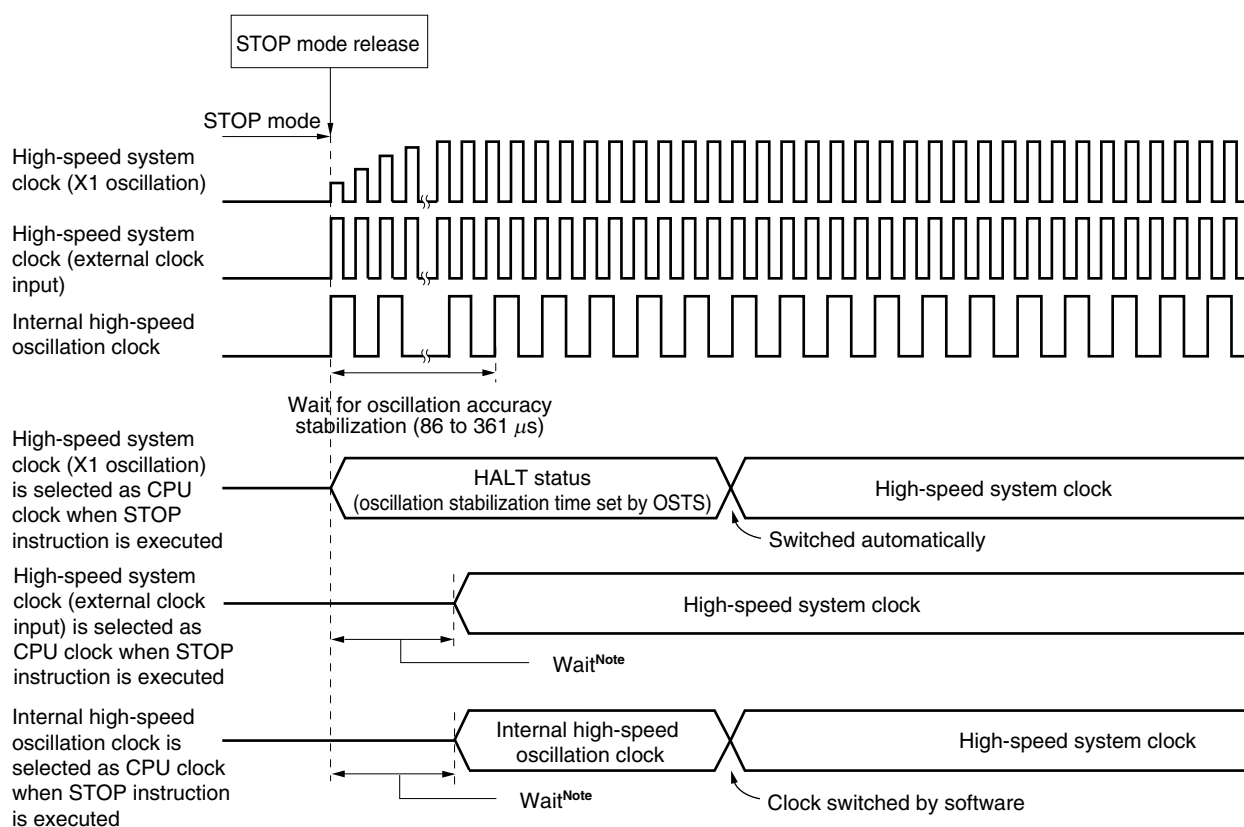
Note Do not start operation of these functions on the external clock input from peripheral hardware pins in the stop mode.

- Remarks 1.** f_{RH} : Internal high-speed oscillation clock, f_x : X1 clock
 f_{EXCLK} : External main system clock, f_{RL} : Internal low-speed oscillation clock
- 2.** The functions mounted depend on the product. See **1.6 Block Diagram** and **1.7 Outline of Functions**.

- Cautions**
1. 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.
 2. Even if “internal low-speed oscillator can be stopped by software” is selected by the option byte, the internal low-speed oscillation clock continues in the STOP mode in the status before the STOP mode is set. To stop the internal low-speed oscillator’s oscillation in the STOP mode, stop it by software and then execute the STOP instruction.
 3. To shorten oscillation stabilization time after the STOP mode is released when the CPU operates with the high-speed system clock (X1 oscillation), switch the CPU clock to the internal high-speed oscillation clock before the execution of the STOP instruction using the following procedure.
 - <1> Set RSTOP to 0 (starting oscillation of the internal high-speed oscillator) → <2> Set MCM0 to 0 (switching the CPU from X1 oscillation to internal high-speed oscillation) → <3> Check that MCS is 0 (checking the CPU clock) → <4> Check that RSTS is 1 (checking internal high-speed oscillation operation) → <5> Execute the STOP instruction
 Before changing the CPU clock from the internal high-speed oscillation 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).
 4. Execute the STOP instruction after having confirmed that the internal high-speed oscillator is operating stably (RSTS = 1).

(2) STOP mode release

Figure 17-5. Operation Timing When STOP Mode Is Released (When Unmasked Interrupt Request Is Generated)



Note The wait time is as follows:

- When vectored interrupt servicing is carried out: 17 or 18 clocks
- When vectored interrupt servicing is not carried out: 11 or 12 clocks

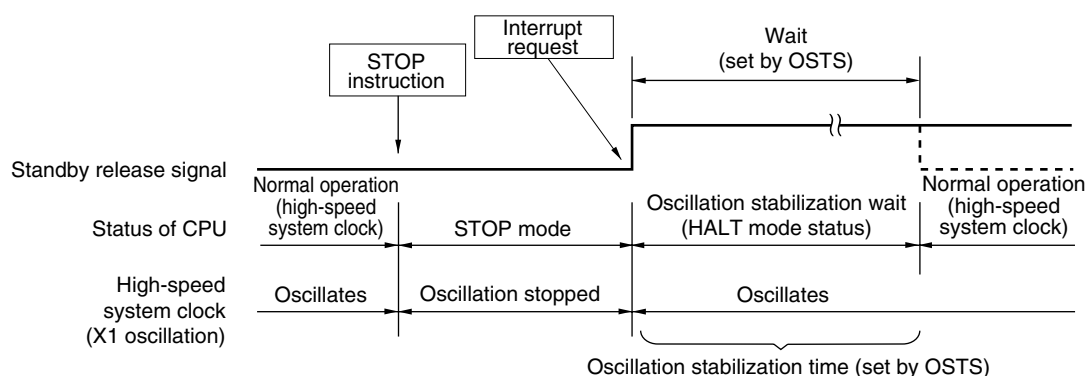
The STOP mode can be released by the following two sources.

(a) Release by unmasked interrupt request

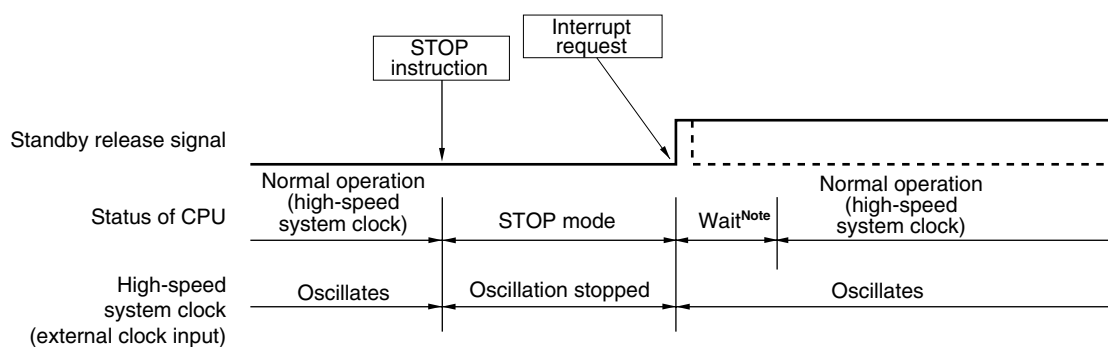
When an unmasked interrupt request is generated, the STOP mode is released. After the oscillation stabilization time has elapsed, if interrupt acknowledgment is enabled, vectored interrupt servicing is carried out. If interrupt acknowledgment is disabled, the next address instruction is executed.

Figure 17-6. STOP Mode Release by Interrupt Request Generation (1/2)

(1) When high-speed system clock (X1 oscillation) is used as CPU clock



(2) When high-speed system clock (external clock input) is used as CPU clock



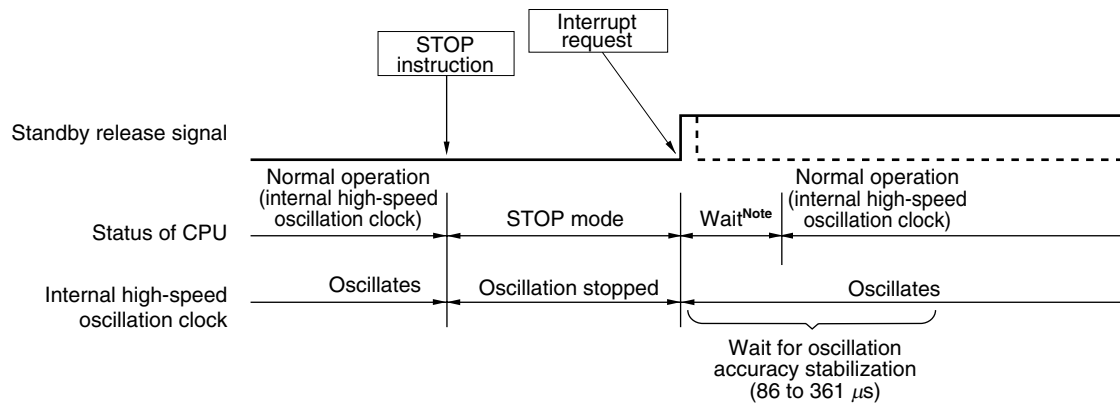
Note The wait time is as follows:

- When vectored interrupt servicing is carried out: 17 or 18 clocks
- When vectored interrupt servicing is not carried out: 11 or 12 clocks

Remark The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

Figure 17-6. STOP Mode Release by Interrupt Request Generation (2/2)

(3) When internal high-speed oscillation clock is used as CPU clock



Note The wait time is as follows:

- When vectored interrupt servicing is carried out: 17 or 18 clocks
- When vectored interrupt servicing is not carried out: 11 or 12 clocks

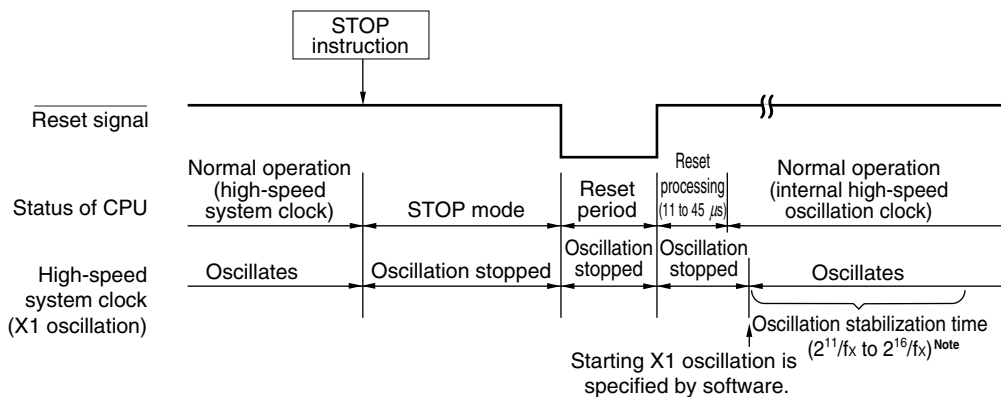
Remark The broken lines indicate the case when the interrupt request that has released the standby mode is acknowledged.

(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.

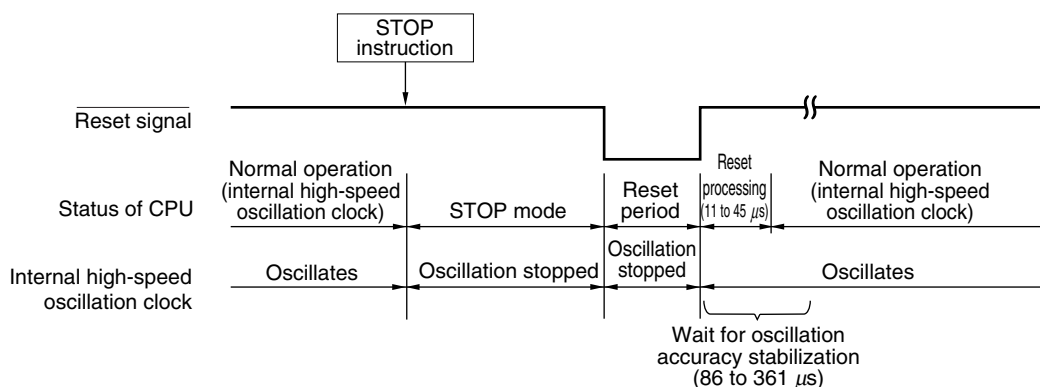
Figure 17-7. STOP Mode Release by Reset

(1) When high-speed system clock is used as CPU clock



Note Oscillation stabilization time is not required when using the external main system clock (fEXCLK) as the high-speed system clock.

(2) When internal high-speed oscillation clock is used as CPU clock



Remark fx: X1 clock oscillation frequency

Table 17-4. Operation in Response to Interrupt Request in STOP Mode

| Release Source | MK _{xx} | PR _{xx} | IE | ISP | Operation |
|----------------------------|------------------|------------------|----|-----|------------------------------------|
| Maskable interrupt request | 0 | 0 | 0 | × | Next address instruction execution |
| | 0 | 0 | 1 | × | Interrupt servicing execution |
| | 0 | 1 | 0 | 1 | Next address instruction execution |
| | 0 | 1 | × | 0 | |
| | 0 | 1 | 1 | 1 | Interrupt servicing execution |
| | 1 | × | × | × | STOP mode held |
| Reset | – | – | × | × | Reset processing |

×: don't care

CHAPTER 18 RESET FUNCTION

The following four 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-clear (POC) circuit
- (4) Internal reset by comparison of supply voltage and detection voltage of low-power-supply detector (LVI)

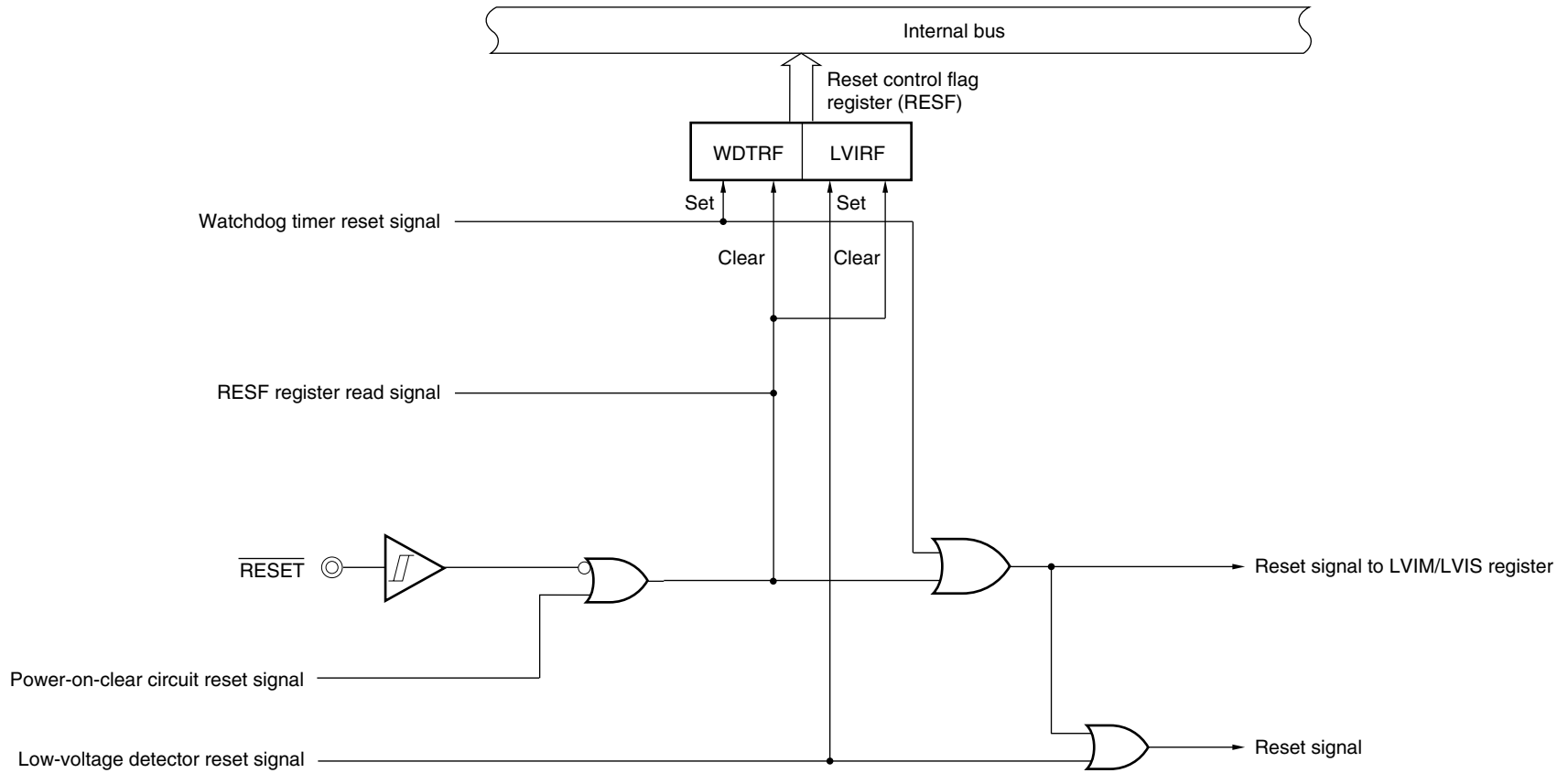
External and internal resets have no functional differences. In both cases, program execution starts at the address at 0000H and 0001H when the reset signal is generated.

A reset is applied when a low level is input to the $\overline{\text{RESET}}$ pin, the watchdog timer overflows, or by POC and LVI circuit voltage detection, and each item of hardware is set to the status shown in Tables 18-1 and 18-2. Each pin is high impedance during reset signal generation or during the oscillation stabilization time just after a reset release.

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 internal high-speed oscillation clock after reset processing. A reset by the watchdog timer is automatically released, and program execution starts using the internal high-speed oscillation clock (see **Figures 18-2 to 18-4**) after reset processing. Reset by POC and LVI circuit power supply detection is automatically released when $V_{DD} \geq V_{POC}$ or $V_{DD} \geq V_{LVI}$ after the reset, and program execution starts using the internal high-speed oscillation clock (see **CHAPTER 19 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 20 LOW-VOLTAGE DETECTOR**) after reset processing.

- Cautions**
1. For an external reset, input a low level for 10 μs or more to the $\overline{\text{RESET}}$ pin.
 2. During reset signal generation, the X1 clock, internal high-speed oscillation clock, and internal low-speed oscillation clock stop oscillating. External main system clock input becomes invalid.
 3. When the STOP mode is released by a reset, the STOP mode contents are held during reset input. However, the port pins become high-impedance.

Figure 18-1. Block Diagram of Reset Function



Caution An LVI circuit internal reset does not reset the LVI circuit.

- Remarks**
1. LVIM: Low-voltage detection register
 2. LVIS: Low-voltage detection level selection register

Figure 18-2. Timing of Reset by RESET Input

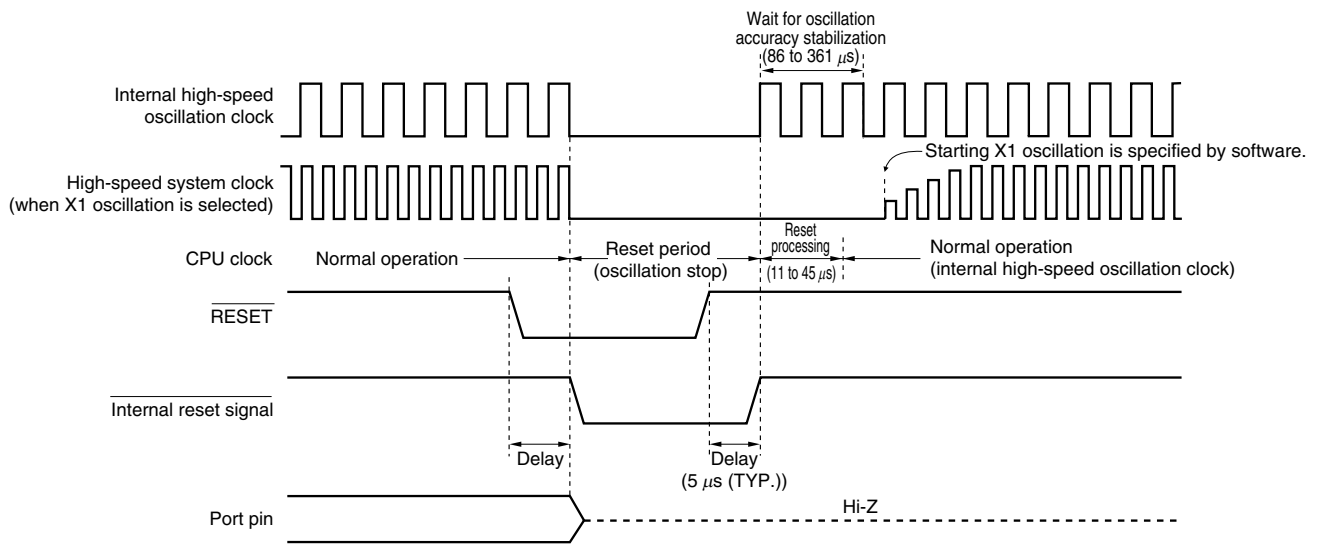
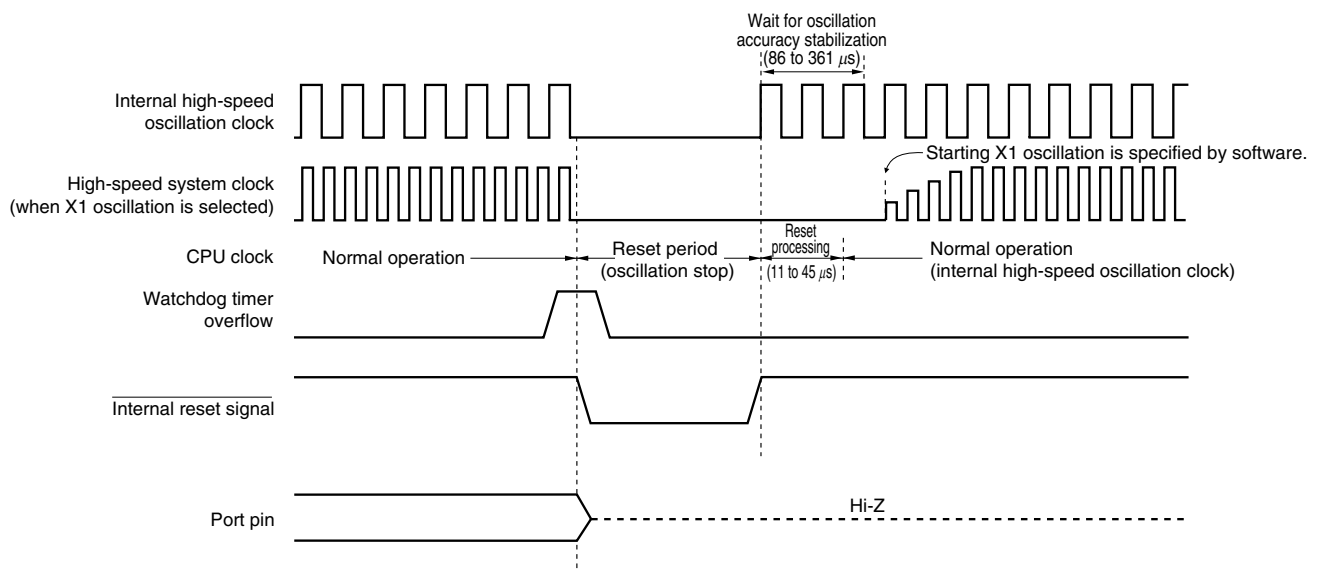
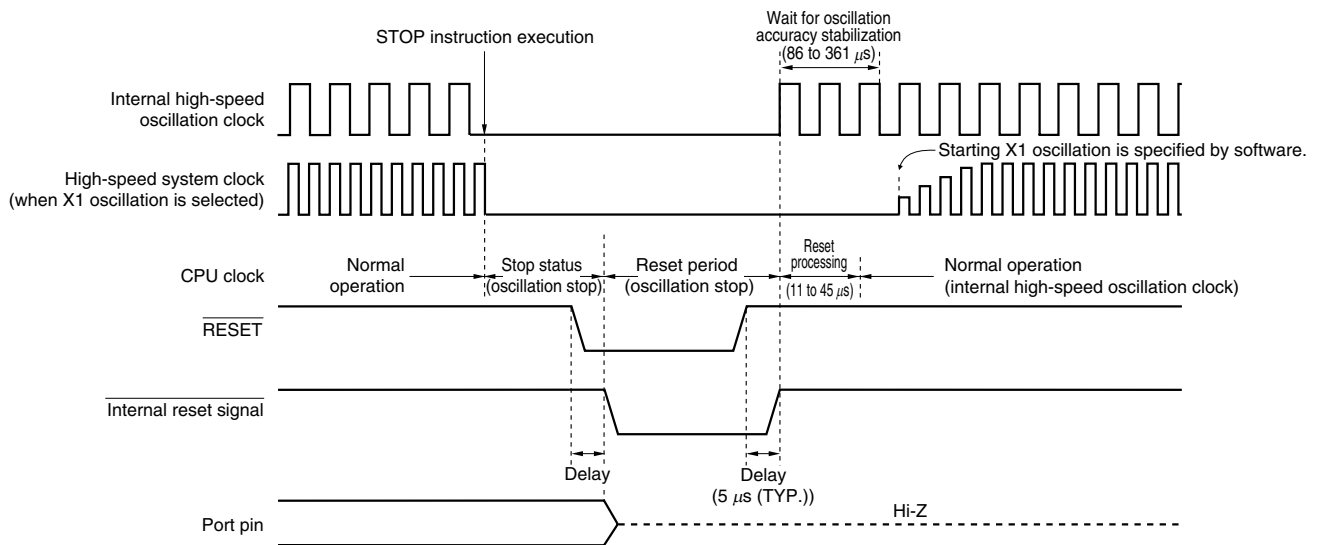


Figure 18-3. Timing of Reset Due to Watchdog Timer Overflow



Caution A watchdog timer internal reset resets the watchdog timer.

Figure 18-4. Timing of Reset in STOP Mode by $\overline{\text{RESET}}$ Input

Remark For the reset timing of the power-on-clear circuit and low-voltage detector, see **CHAPTER 19 POWER-ON-CLEAR CIRCUIT** and **CHAPTER 20 LOW-VOLTAGE DETECTOR**.

Table 18-1. Operation Statuses During Reset Period

| Item | | During Reset Period |
|--------------------------------|--------------------|--|
| System clock | | Clock supply to the CPU is stopped. |
| Main system clock | f _{RH} | Operation stopped |
| | f _x | Operation stopped (pin is I/O port mode) |
| | f _{EXCLK} | Clock input invalid (pin is I/O port mode) |
| f _{RL} | | Operation stopped |
| CPU | | |
| Flash memory | | |
| RAM | | |
| Port (latch) | | |
| 16-bit timer/event counter | 00 | |
| 8-bit timer/event counter | 50 | |
| | 51 | |
| 8-bit timer | H0 | |
| | H1 | |
| Watch timer | | |
| Watchdog timer | | |
| A/D converter | | |
| Serial interface | UART0 | |
| | UART6 | |
| | CSI10 | |
| | IIC0 | |
| Power-on-clear function | | Operable |
| Low-voltage detection function | | Operation stopped |
| External interrupt | | |

- Remarks 1.** f_{RH}: Internal high-speed oscillation clock, f_x: X1 clock
f_{EXCLK}: External main system clock, f_{RL}: Internal low-speed oscillation clock
- 2.** The functions mounted depend on the product. See 1.6 **Block Diagram** and 1.7 **Outline of Functions**.

Table 18-2. Hardware Statuses After Reset Acknowledgment (1/4)

| Hardware | | After Reset Acknowledgment ^{Note 1} |
|---|---------------------------|--|
| Program counter (PC) | | The contents of the reset vector table (0000H, 0001H) are set. |
| Stack pointer (SP) | | Undefined |
| Program status word (PSW) | | 02H |
| RAM | Data memory | Undefined ^{Note 2} |
| | General-purpose registers | Undefined ^{Note 2} |
| Port registers (P0 to P4, P6, P7, P12) (output latches) | | 00H |
| Port mode registers (PM0 to PM4, PM6, PM7, PM12) | | FFH |
| Pull-up resistor option registers (PU0, PU1, PU3, PU4, PU7, PU12) | | 00H |
| Internal memory size switching register (IMS) | | CFH ^{Note 3} |

- 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. When a reset is executed in the standby mode, the pre-reset status is held even after reset.
 3. The initial value of the internal memory size switching register (IMS) after a reset release is constant (IMS = CFH) in all products of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B, regardless of the internal memory capacity. Therefore, set the value corresponding to each product as indicated in Table 3-1.

Table 18-2. Hardware Statuses After Reset Acknowledgment (2/4)

| Hardware | | Status After Reset Acknowledgment ^{Note 1} |
|---|---|---|
| Clock operation mode select register (OSCCTL) | | 00H |
| Processor clock control register (PCC) | | 01H |
| Internal oscillation mode register (RCM) | | 80H |
| Main OSC control register (MOC) | | 80H |
| Main clock mode register (MCM) | | 00H |
| Oscillation stabilization time counter status register (OSTC) | | 00H |
| Oscillation stabilization time select register (OSTS) | | 05H |
| 16-bit timer/event counter 00 | Timer counter 00 (TM00) | 0000H |
| | Capture/compare registers 000, 010 (CR000, CR010) | 0000H |
| | Mode control register 00 (TMC00) | 00H |
| | Prescaler mode register 00 (PRM00) | 00H |
| | Capture/compare control register 00 (CRC00) | 00H |
| | Timer output control register 00 (TOC00) | 00H |
| 8-bit timer/event counters 50, 51 | Timer counters 50, 51 (TM50, TM51) | 00H |
| | Compare registers 50, 51 (CR50, CR51) | 00H |
| | Timer clock selection registers 50, 51 (TCL50, TCL51) | 00H |
| | Mode control registers 50, 51 (TMC50, TMC51) | 00H |
| 8-bit timers H0, H1 | Compare registers 00, 10, 01, 11 (CMP00, CMP10, CMP01, CMP11) | 00H |
| | Mode registers (TMHMD0, TMHMD1) | 00H |
| | Carrier control register 1 (TMCYC1) ^{Note 2} | 00H |
| Watch timer | Operation mode register (WTM) | 00H |
| Watchdog timer | Enable register (WDTE) | 1AH/9AH ^{Note 3} |
| A/D converter | 10-bit A/D conversion result register (ADCR) | 0000H |
| | 8-bit A/D conversion result register (ADCRH) | 00H |
| | Mode register (ADM) | 00H |
| | Analog input channel specification register (ADS) | 00H |
| | A/D port configuration register (ADPC) | 00H |
| Serial interface UART0 | Receive buffer register 0 (RXB0) | FFH |
| | Transmit shift register 0 (TXS0) | FFH |
| | Asynchronous serial interface operation mode register 0 (ASIM0) | 01H |
| | Asynchronous serial interface reception error status register 0 (ASIS0) | 00H |
| | Baud rate generator control register 0 (BRGC0) | 1FH |

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. 8-bit timer H1 only.

3. The reset value of WDTE is determined by the option byte setting.

Table 18-2. Hardware Statuses After Reset Acknowledgment (3/4)

| Hardware | | Status After Reset Acknowledgment ^{Note} |
|------------------------|---|---|
| Serial interface UART6 | Receive buffer register 6 (RXB6) | FFH |
| | Transmit buffer register 6 (TXB6) | FFH |
| | Asynchronous serial interface operation mode register 6 (ASIM6) | 01H |
| | Asynchronous serial interface reception error status register 6 (ASIS6) | 00H |
| | Asynchronous serial interface transmission status register 6 (ASIF6) | 00H |
| | Clock selection register 6 (CKSR6) | 00H |
| | Baud rate generator control register 6 (BRGC6) | FFH |
| | Asynchronous serial interface control register 6 (ASICL6) | 16H |
| | Input switch control register (ISC) | 00H |
| Serial interface CSI10 | Transmit buffer register 10 (SOTB10) | 00H |
| | Serial I/O shift register 10 (SIO10) | 00H |
| | Serial operation mode register 10 (CSIM10) | 00H |
| | Serial clock selection register 10 (CSIC10) | 00H |
| Serial interface IIC0 | Shift register 0 (IIC0) | 00H |
| | Control register 0 (IICC0) | 00H |
| | Slave address register 0 (SVA0) | 00H |
| | Clock selection register 0 (IICCL0) | 00H |
| | Function expansion register 0 (IICX0) | 00H |
| | Status register 0 (IICS0) | 00H |
| | Flag register 0 (IICF0) | 00H |

Note 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.

Table 18-2. Hardware Statuses After Reset Acknowledgment (4/4)

| Hardware | | Status After Reset Acknowledgment ^{Note 1} |
|----------------------|---|---|
| Reset function | Reset control flag register (RESF) | 00H ^{Note 2} |
| Low-voltage detector | Low-voltage detection register (LVIM) | 00H ^{Note 2} |
| | Low-voltage detection level selection register (LVIS) | 00H ^{Note 2} |
| Interrupt | Request flag registers 0L, 0H, 1L, 1H (IF0L, IF0H, IF1L, IF1H) | 00H |
| | Mask flag registers 0L, 0H, 1L, 1H (MK0L, MK0H, MK1L, MK1H) | FFH |
| | Priority specification flag registers 0L, 0H, 1L, 1H (PR0L, PR0H, PR1L, PR1H) | FFH |
| | External interrupt rising edge enable register (EGP) | 00H |
| | External interrupt falling edge enable register (EGN) | 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. These values vary depending on the reset source.

| Reset Source | | $\overline{\text{RESET}}$ Input | Reset by POC | Reset by WDT | Reset by LVI |
|--------------|------------|---------------------------------|---------------|---------------|--------------|
| Register | RESF | Cleared (0) | Cleared (0) | Set (1) | Held |
| | WDTRF flag | | | Held | Set (1) |
| | LVIRF flag | | | | |
| | LVIM | Cleared (00H) | Cleared (00H) | Cleared (00H) | Held |
| | LVIS | | | | |

18.1 Register for Confirming Reset Source

Many internal reset generation sources exist in the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B. The reset control flag register (RESF) is used to store which source has generated the reset request.

RESF can be read by an 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input, reset by power-on-clear (POC) circuit, and reading RESF set RESF to 00H.

Figure 18-5. Format of Reset Control Flag Register (RESF)

Address: FFACH After reset: 00H^{Note} R

| | | | | | | | | |
|--------|---|---|---|-------|---|---|---|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RESF | 0 | 0 | 0 | WDTRF | 0 | 0 | 0 | LVIRF |

| | |
|-------|--|
| WDTRF | Internal reset request by watchdog timer (WDT) |
| 0 | Internal reset request is not generated, or RESF is cleared. |
| 1 | Internal reset request is generated. |

| | |
|-------|--|
| LVIRF | Internal reset request by low-voltage detector (LVI) |
| 0 | Internal reset request is not generated, or RESF is cleared. |
| 1 | Internal reset request is generated. |

Note The value after reset varies depending on the reset source.

Caution Do not read data by a 1-bit memory manipulation instruction.

The status of RESF when a reset request is generated is shown in Table 18-3.

Table 18-3. RESF Status When Reset Request Is Generated

| Reset Source | $\overline{\text{RESET}}$ Input | Reset by POC | Reset by WDT | Reset by LVI |
|--------------|---------------------------------|--------------|--------------|--------------|
| Flag | | | | |
| WDTRF | Cleared (0) | Cleared (0) | Set (1) | Held |
| LVIRF | | | Held | Set (1) |

CHAPTER 19 POWER-ON-CLEAR CIRCUIT

19.1 Functions of Power-on-Clear Circuit

The power-on-clear circuit (POC) has the following functions.

- Generates internal reset signal at power on.
In the 1.59 V POC mode (option byte: POCMODE = 0), the reset signal is released when the supply voltage (V_{DD}) exceeds $1.59\text{ V} \pm 0.15\text{ V}$.
In the 2.7 V/1.59 V POC mode (option byte: POCMODE = 1), the reset signal is released when the supply voltage (V_{DD}) exceeds $2.7\text{ V} \pm 0.2\text{ V}$.
- Compares supply voltage (V_{DD}) and detection voltage ($V_{POC} = 1.59\text{ V} \pm 0.15\text{ V}$), generates internal reset signal when $V_{DD} < V_{POC}$.

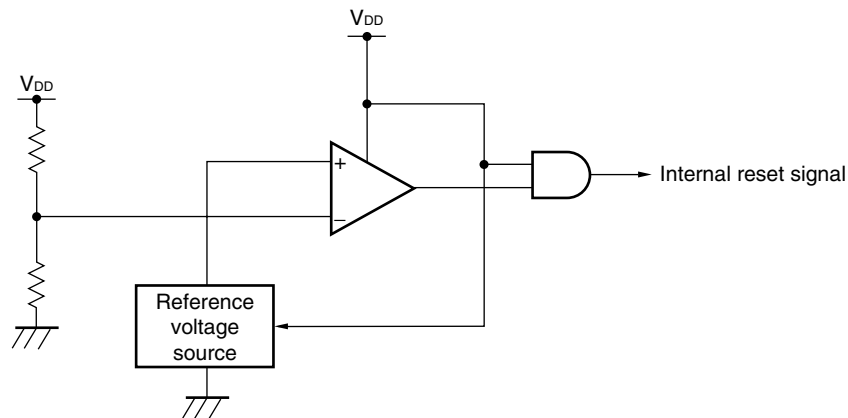
Caution If an internal reset signal is generated in the POC circuit, the reset control flag register (RESF) is cleared to 00H.

Remark R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B incorporate 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) or low-voltage-detector (LVI). RESF is not cleared to 00H and the flag is set to 1 when an internal reset signal is generated by WDT or LVI. For details of RESF, see **CHAPTER 18 RESET FUNCTION**.

19.2 Configuration of Power-on-Clear Circuit

The block diagram of the power-on-clear circuit is shown in Figure 19-1.

Figure 19-1. Block Diagram of Power-on-Clear Circuit



19.3 Operation of Power-on-Clear Circuit

(1) In 1.59 V POC mode (option byte: POCMODE = 0)

- An internal reset signal is generated on power application. When the supply voltage (V_{DD}) exceeds the detection voltage ($V_{POC} = 1.59 \text{ V} \pm 0.15 \text{ V}$), the reset status is released.
- The supply voltage (V_{DD}) and detection voltage ($V_{POC} = 1.59 \text{ V} \pm 0.15 \text{ V}$) are compared. When $V_{DD} < V_{POC}$, the internal reset signal is generated. It is released when $V_{DD} \geq V_{POC}$.

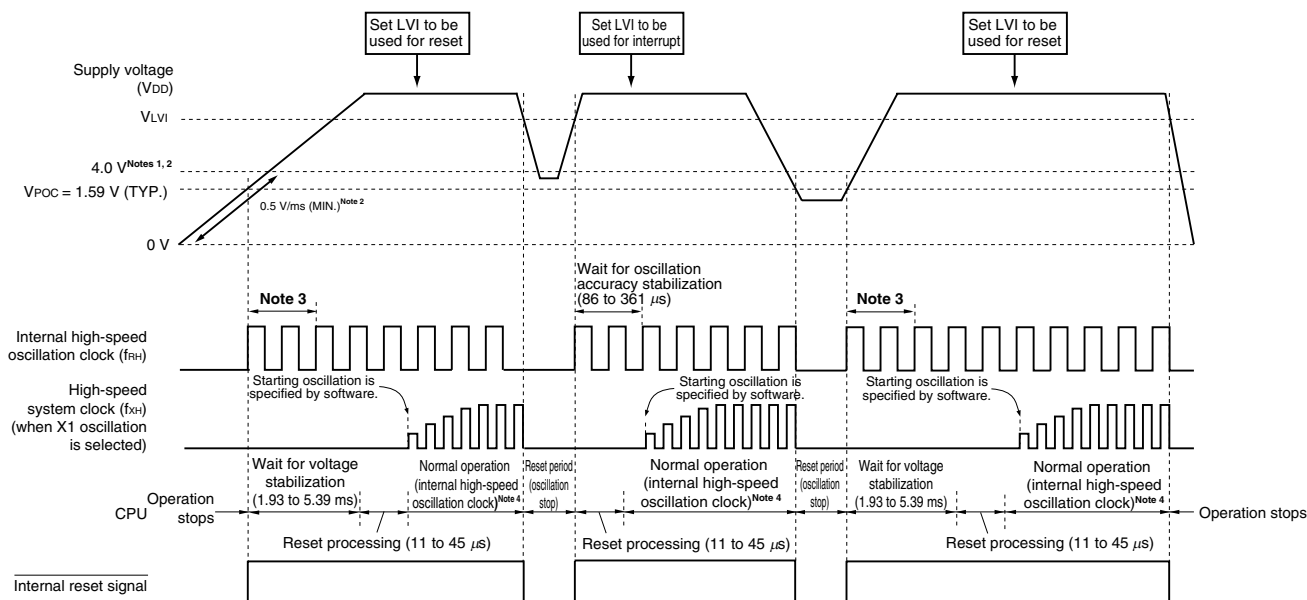
(2) In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)

- An internal reset signal is generated on power application. When the supply voltage (V_{DD}) exceeds the detection voltage ($V_{DDPOC} = 2.7 \text{ V} \pm 0.2 \text{ V}$), the reset status is released.
- The supply voltage (V_{DD}) and detection voltage ($V_{POC} = 1.59 \text{ V} \pm 0.15 \text{ V}$) are compared. When $V_{DD} < V_{POC}$, the internal reset signal is generated. It is released when $V_{DD} \geq V_{DDPOC}$.

The timing of generation of the internal reset signal by the power-on-clear circuit and low-voltage detector is shown below.

Figure 19-2. Timing of Generation of Internal Reset Signal by Power-on-Clear Circuit and Low-Voltage Detector (1/2)

(1) In 1.59 V POC mode (option byte: POCMODE = 0)



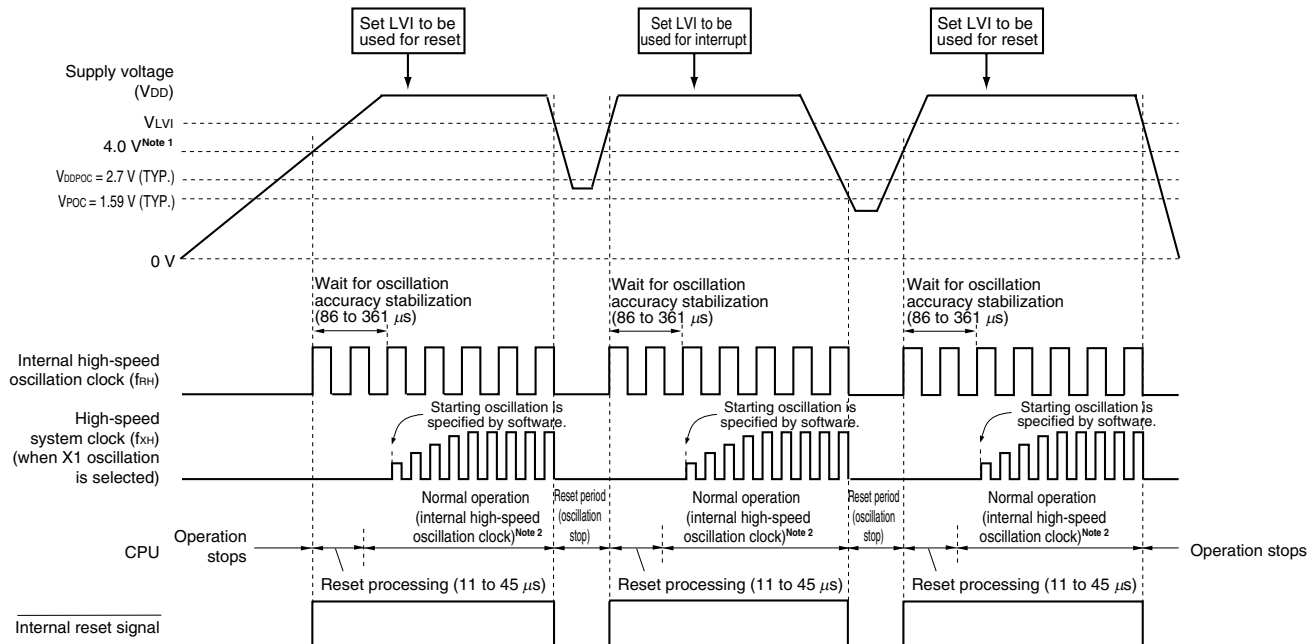
- Notes**
1. The guaranteed operation range is $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$. To set the voltage range below the guaranteed operation range to the reset state when the supply voltage falls, use the reset function of the low-voltage detector, or input a low level to the $\overline{\text{RESET}}$ pin.
 2. If the voltage rises to 4.0 V at a rate slower than 0.5 V/ms (MIN.) on power application, input a low level to the $\overline{\text{RESET}}$ pin after power application and before the voltage reaches 4.0 V, or set the 2.7 V/1.59 V POC mode by using an option byte (POCMODE = 1).
 3. The oscillation accuracy stabilization time of the internal high-speed oscillation clock is included in the internal voltage stabilization time.
 4. The CPU clock can be switched from the internal high-speed oscillation clock to the high-speed system clock. To use the X1 clock, use the OSTC register to confirm the lapse of the oscillation stabilization time.

Caution Set the low-voltage detector by software after the reset status is released (see CHAPTER 20 LOW-VOLTAGE DETECTOR).

Remark V_{LVI} : LVI detection voltage
 V_{POC} : POC detection voltage

Figure 19-2. Timing of Generation of Internal Reset Signal by Power-on-Clear Circuit and Low-Voltage Detector (2/2)

(2) In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



- Notes**
1. The guaranteed operation range is $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$. To set the voltage range below the guaranteed operation range to the reset state when the supply voltage falls, use the reset function of the low-voltage detector, or input a low level to the $\overline{\text{RESET}}$ pin.
 2. The CPU clock can be switched from the internal high-speed oscillation clock to the high-speed system clock. To use the X1 clock, use the OSTC register to confirm the lapse of the oscillation stabilization time.

- Cautions**
1. Set the low-voltage detector by software after the reset status is released (see CHAPTER 20 LOW-VOLTAGE DETECTOR).
 2. A voltage oscillation stabilization time of 1.93 to 5.39 ms is required after the supply voltage reaches 1.59 V (TYP.). If the supply voltage rises from 1.59 V (TYP.) to 2.7 V (TYP.) within 1.93 ms, the power supply oscillation stabilization time of 0 to 5.39 ms is automatically generated before reset processing.

Remark V_{LVI} : LVI detection voltage
 V_{POC} : POC detection voltage

19.4 Cautions for Power-on-Clear Circuit

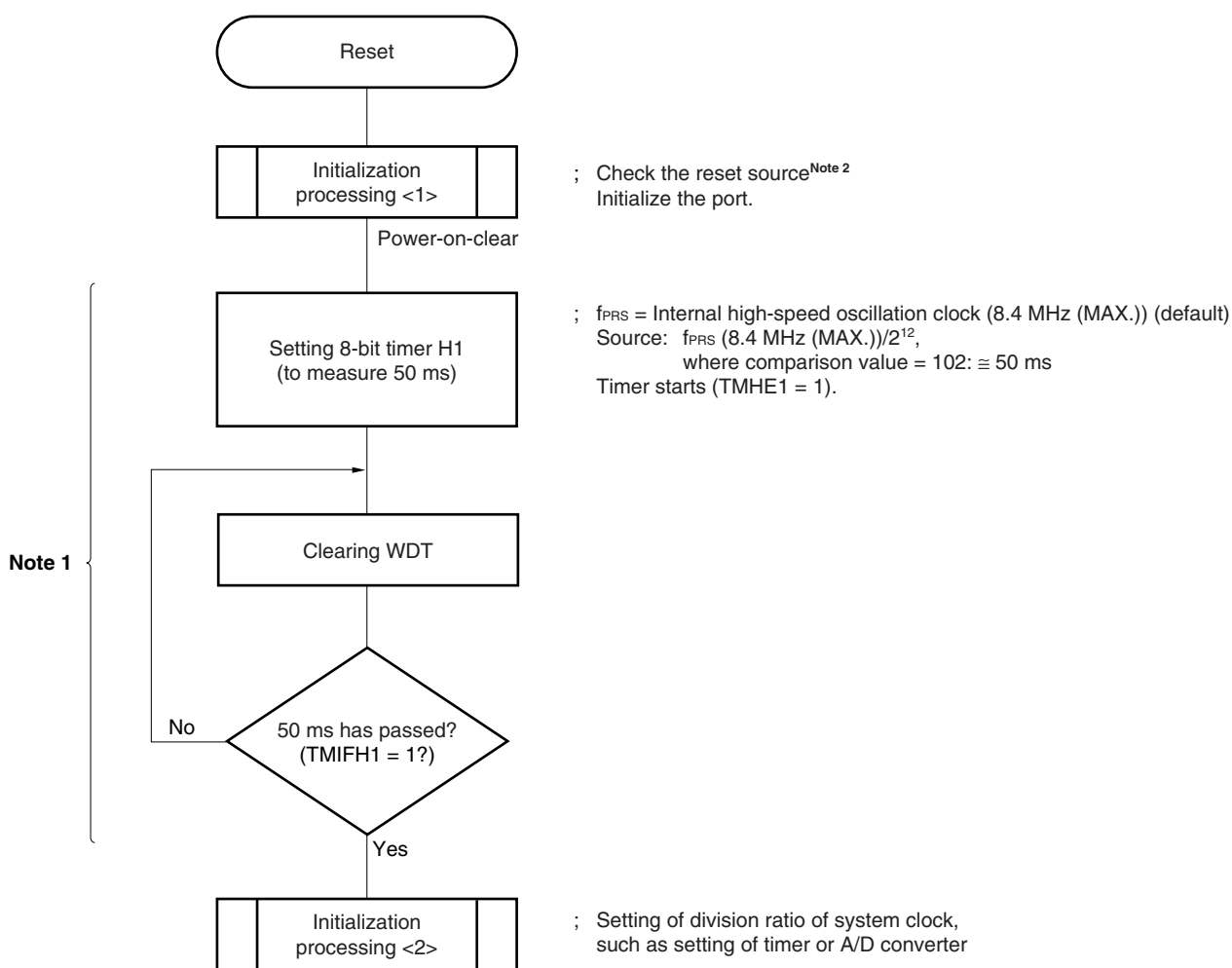
In a system where the supply voltage (V_{DD}) fluctuates for a certain period in the vicinity of the POC detection voltage (V_{POC}), the system may be repeatedly reset and released from the reset status. In this case, the time from release of reset to the start of the operation of the microcontroller 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 19-3. Example of Software Processing After Reset Release (1/2)

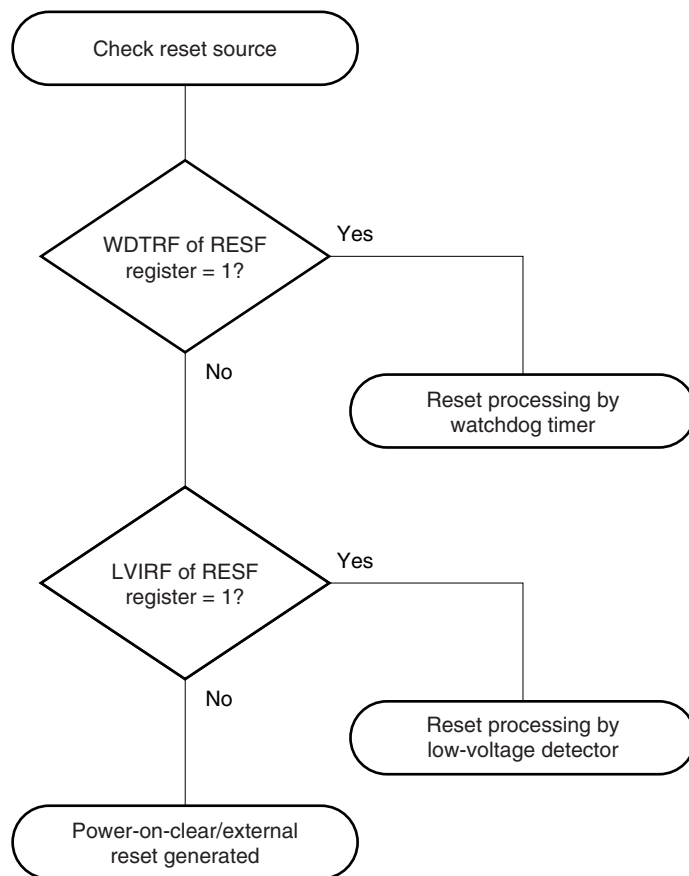
- If supply voltage fluctuation is 50 ms or less in vicinity of POC detection voltage



- 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.

Figure 19-3. Example of Software Processing After Reset Release (2/2)

- Checking reset source



CHAPTER 20 LOW-VOLTAGE DETECTOR

20.1 Functions of Low-Voltage Detector

The low-voltage detector has the following functions.

- The LVI circuit compares the supply voltage (V_{DD}) with the detection voltage (V_{LVI}) or the input voltage from an external input pin (EXLVI) with the detection voltage ($V_{EXLVI} = 1.21$ V (TYP.): fixed), and generates an internal reset or internal interrupt signal.
- The supply voltage (V_{DD}) or input voltage from an external input pin (EXLVI) can be selected by software.
- Reset or interrupt function can be selected by software.
- Detection levels (2 levels) of supply voltage can be changed by software.
- Operable in STOP mode.

The reset and interrupt signals are generated as follows depending on selection by software.

| Selection of Level Detection of Supply Voltage (V_{DD}) (LVISEL = 0) | | Selection Level Detection of Input Voltage from External Input Pin (EXLVI) (LVISEL = 1) | |
|---|--|---|--|
| Selects reset (LVIMD = 1). | Selects interrupt (LVIMD = 0). | Selects reset (LVIMD = 1). | Selects interrupt (LVIMD = 0). |
| Generates an internal reset signal when $V_{DD} < V_{LVI}$ and releases the reset signal when $V_{DD} \geq V_{LVI}$. | Generates an internal interrupt signal when V_{DD} drops lower than V_{LVI} ($V_{DD} < V_{LVI}$) or when V_{DD} becomes V_{LVI} or higher ($V_{DD} \geq V_{LVI}$). | Generates an internal reset signal when $EXLVI < V_{EXLVI}$ and releases the reset signal when $EXLVI \geq V_{EXLVI}$. | Generates an internal interrupt signal when EXLVI drops lower than V_{EXLVI} ($EXLVI < V_{EXLVI}$) or when EXLVI becomes V_{EXLVI} or higher ($EXLVI \geq V_{EXLVI}$). |

Remark LVISEL: Bit 2 of low-voltage detection register (LVIM)

LVIMD: Bit 1 of LVIM

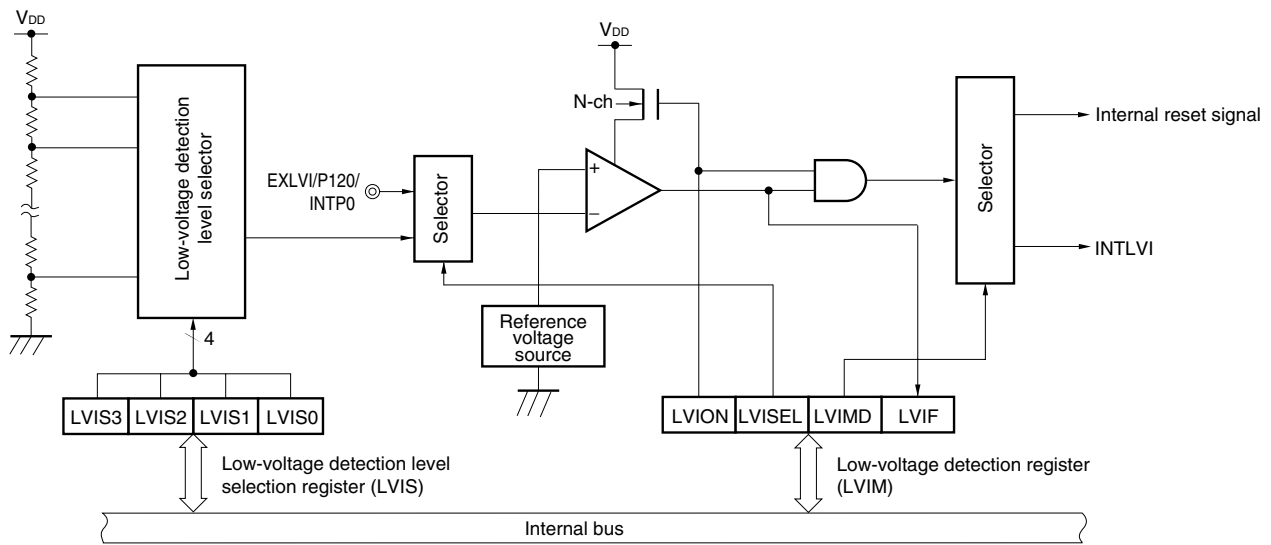
While the low-voltage detector is operating, whether the supply voltage or the input voltage from an external input pin is more than or less than the detection level can be checked by reading the low-voltage detection flag (LVIF: bit 0 of LVIM).

When the low-voltage detector is used to reset, bit 0 (LVIRF) of the reset control flag register (RESF) is set to 1 if reset occurs. For details of RESF, see **CHAPTER 18 RESET FUNCTION**.

20.2 Configuration of Low-Voltage Detector

The block diagram of the low-voltage detector is shown in Figure 20-1.

Figure 20-1. Block Diagram of Low-Voltage Detector



20.3 Registers Controlling Low-Voltage Detector

The low-voltage detector is controlled by the following registers.

- Low-voltage detection register (LVIM)
- Low-voltage detection level selection register (LVIS)
- Port mode register 12 (PM12)

(1) Low-voltage detection register (LVIM)

This register sets low-voltage detection and the operation mode.

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

The generation of a reset signal other than an LVI reset clears this register to 00H.

Figure 20-2. Format of Low-Voltage Detection Register (LVIM)

Address: FFBEH After reset: 00H^{Note 1} R/W^{Note 2}

| Symbol | <7> | 6 | 5 | 4 | 3 | <2> | <1> | <0> |
|-----------------------------|---|---|---|---|---|--------|-------|------|
| LVIM | LVION | 0 | 0 | 0 | 0 | LVISEL | LVIMD | LVIF |
| LVION ^{Notes 3, 4} | Enables low-voltage detection operation | | | | | | | |
| 0 | Disables operation | | | | | | | |
| 1 | Enables operation | | | | | | | |
| LVISEL ^{Note 3} | Voltage detection selection | | | | | | | |
| 0 | Detects level of supply voltage (V_{DD}) | | | | | | | |
| 1 | Detects level of input voltage from external input pin (EXLVI) | | | | | | | |
| LVIMD ^{Note 3} | Low-voltage detection operation mode (interrupt/reset) selection | | | | | | | |
| 0 | <ul style="list-style-type: none"> LVISEL = 0: Generates an internal interrupt signal when the supply voltage (V_{DD}) drops lower than the detection voltage (V_{LVI}) ($V_{DD} < V_{LVI}$) or when V_{DD} becomes V_{LVI} or higher ($V_{DD} \geq V_{LVI}$). LVISEL = 1: Generates an interrupt signal when the input voltage from an external input pin (EXLVI) drops lower than the detection voltage (V_{EXLVI}) ($EXLVI < V_{EXLVI}$) or when EXLVI becomes V_{EXLVI} or higher ($EXLVI \geq V_{EXLVI}$). | | | | | | | |
| 1 | <ul style="list-style-type: none"> LVISEL = 0: Generates an internal reset signal when the supply voltage (V_{DD}) < detection voltage (V_{LVI}) and releases the reset signal when $V_{DD} \geq V_{LVI}$. LVISEL = 1: Generates an internal reset signal when the input voltage from an external input pin (EXLVI) < detection voltage (V_{EXLVI}) and releases the reset signal when $EXLVI \geq V_{EXLVI}$. | | | | | | | |
| LVIF | Low-voltage detection flag | | | | | | | |
| 0 | <ul style="list-style-type: none"> LVISEL = 0: Supply voltage (V_{DD}) \geq detection voltage (V_{LVI}), or when operation is disabled LVISEL = 1: Input voltage from external input pin (EXLVI) \geq detection voltage (V_{EXLVI}), or when operation is disabled | | | | | | | |
| 1 | <ul style="list-style-type: none"> LVISEL = 0: Supply voltage (V_{DD}) < detection voltage (V_{LVI}) LVISEL = 1: Input voltage from external input pin (EXLVI) < detection voltage (V_{EXLVI}) | | | | | | | |

- Notes**
- This bit is cleared to 00H upon a reset other than an LVI reset.
 - Bit 0 is read-only.
 - LVION, LVIMD, and LVISEL are cleared to 0 in the case of a reset other than an LVI reset. These are not cleared to 0 in the case of an LVI reset.
 - When LVION is set to 1, operation of the comparator in the LVI circuit is started. Use software to wait for an operation stabilization time (10 μ s (MIN.)) from when LVION is set to 1 until operation is stabilized. After operation has stabilized, the external input of 200 μ s (MIN.) (Minimum pulse width: 200 μ s (MIN.)) is required from when a state below LVI detection voltage has been entered, until LVIF is set (1).

- Cautions**
- To stop LVI, follow either of the procedures below.
 - When using 8-bit memory manipulation instruction: Write 00H to LVIM.
 - When using 1-bit memory manipulation instruction: Clear LVION to 0.
 - Input voltage from external input pin (EXLVI) must be $EXLVI < V_{DD}$.
 - When using LVI as an interrupt, if LVION is cleared (0) in a state below the LVI detection voltage, an INTLVI signal is generated and LVIF becomes 1.

(2) Low-voltage detection level selection register (LVIS)

This register selects the low-voltage detection level.

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

The generation of a reset signal other than an LVI reset clears this register to 00H.

Figure 20-3. Format of Low-Voltage Detection Level Selection Register (LVIS)

Address: FFBFH After reset: 00H^{Note} R/W

| | | | | | | | | |
|--------|---|---|---|---|---|---|---|-------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LVIS | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LVIS0 |

| | |
|-------|---------------------------------|
| LVIS0 | Detection level |
| 0 | V_{LV10} (4.24 V \pm 0.1 V) |
| 0 | V_{LV11} (4.09 V \pm 0.1 V) |

Note The value of LVIS is not reset but retained as is, upon a reset by LVI. It is cleared to 00H upon other resets.

- Cautions**
1. Be sure to clear bits 1 to 7 to "0".
 2. Do not change the value of LVIS during LVI operation.
 3. When an input voltage from the external input pin (EXLVI) is detected, the detection voltage ($V_{EXLVI} = 1.21$ V (TYP.)) is fixed. Therefore, setting of LVIS is not necessary.

(3) Port mode register 12 (PM12)

When using the P120/EXLVI/INTP0 pin for external low-voltage detection potential input, set PM120 to 1. At this time, the output latch of P120 may be 0 or 1.

PM12 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation sets PM12 to FFH.

Figure 20-4. Format of Port Mode Register 12 (PM12)

Address: FF2CH After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|---|---|---|---|---|-------|-------|-------|
| PM12 | 1 | 1 | 1 | 1 | 1 | PM122 | PM121 | PM120 |

| PM12n | P12n pin I/O mode selection (n = 0 to 2) |
|-------|--|
| 0 | Output mode (output buffer on) |
| 1 | Input mode (output buffer off) |

20.4 Operation of Low-Voltage Detector

The low-voltage detector can be used in the following two modes.

(1) Used as reset (LVIMD = 1)

- If LVISEL = 0, compares the supply voltage (V_{DD}) and detection voltage (V_{LVI}), generates an internal reset signal when $V_{DD} < V_{LVI}$, and releases internal reset when $V_{DD} \geq V_{LVI}$.
- If LVISEL = 1, compares the input voltage from external input pin (EXLVI) and detection voltage ($V_{EXLVI} = 1.21 \text{ V}$ (TYP.)), generates an internal reset signal when $EXLVI < V_{EXLVI}$, and releases internal reset when $EXLVI \geq V_{EXLVI}$.

(2) Used as interrupt (LVIMD = 0)

- If LVISEL = 0, compares the supply voltage (V_{DD}) and detection voltage (V_{LVI}). When V_{DD} drops lower than V_{LVI} ($V_{DD} < V_{LVI}$) or when V_{DD} becomes V_{LVI} or higher ($V_{DD} \geq V_{LVI}$), generates an interrupt signal (INTLVI).
- If LVISEL = 1, compares the input voltage from external input pin (EXLVI) and detection voltage ($V_{EXLVI} = 1.21 \text{ V}$ (TYP.)). When EXLVI drops lower than V_{EXLVI} ($EXLVI < V_{EXLVI}$) or when EXLVI becomes V_{EXLVI} or higher ($EXLVI \geq V_{EXLVI}$), generates an interrupt signal (INTLVI).

While the low-voltage detector is operating, whether the supply voltage or the input voltage from an external input pin is more than or less than the detection level can be checked by reading the low-voltage detection flag (LVIF: bit 0 of LVIM).

Remark LVIMD: Bit 1 of low-voltage detection register (LVIM)
LVISEL: Bit 2 of LVIM

20.4.1 When used as reset

(1) When detecting level of supply voltage (V_{DD})

- When starting operation
 - <1> Mask the LVI interrupt ($LVIMK = 1$).
 - <2> Clear bit 2 ($LVISEL$) of the low-voltage detection register ($LVIM$) to 0 (detects level of supply voltage (V_{DD})) (default value).
 - <3> Set the detection voltage using bit 0 ($LVIS0$) of the low-voltage detection level selection register ($LVIS$).
 - <4> Set bit 7 ($LVION$) of $LVIM$ to 1 (enables LVI operation).
 - <5> Use software to wait for an operation stabilization time ($10 \mu s$ (MIN.)).
 - <6> Wait until it is checked that (supply voltage (V_{DD}) \geq detection voltage (V_{LVI})) by bit 0 ($LVIF$) of $LVIM$.
 - <7> Set bit 1 ($LVIMD$) of $LVIM$ to 1 (generates reset when the level is detected).

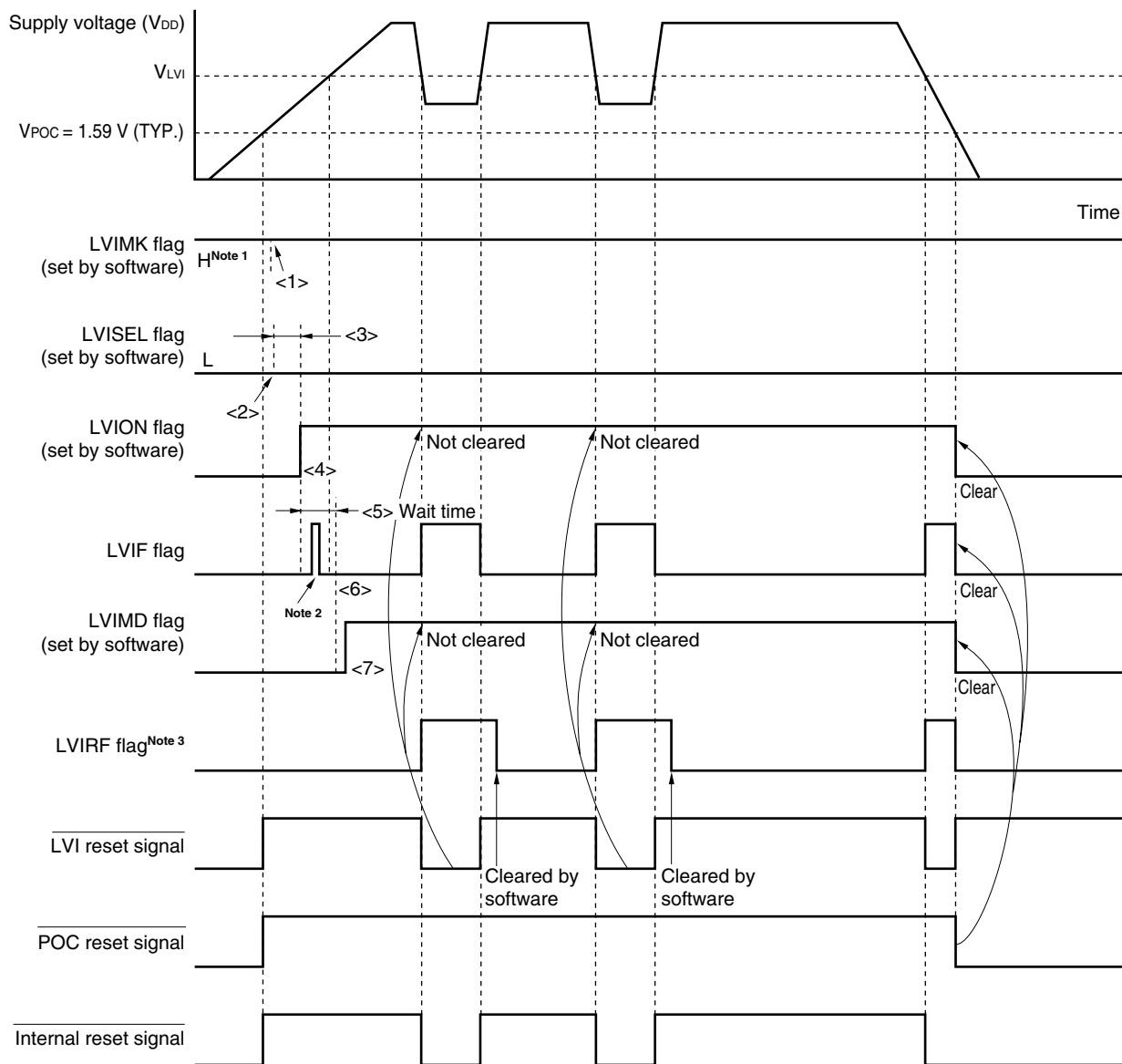
Figure 20-5 shows the timing of the internal reset signal generated by the low-voltage detector. The numbers in this timing chart correspond to <1> to <7> above.

- Cautions**
1. <1> must always be executed. When $LVIMK = 0$, an interrupt may occur immediately after the processing in <4>.
 2. If supply voltage (V_{DD}) \geq detection voltage (V_{LVI}) when $LVIMD$ is set to 1, an internal reset signal is not generated.

- When stopping operation
Either of the following procedures must be executed.
 - When using 8-bit memory manipulation instruction:
Write 00H to $LVIM$.
 - When using 1-bit memory manipulation instruction:
Clear $LVIMD$ to 0 and then $LVION$ to 0.

**Figure 20-5. Timing of Low-Voltage Detector Internal Reset Signal Generation
(Detects Level of Supply Voltage (V_{DD})) (1/2)**

(1) In 1.59 V POC mode (option byte: POCMODE = 0)

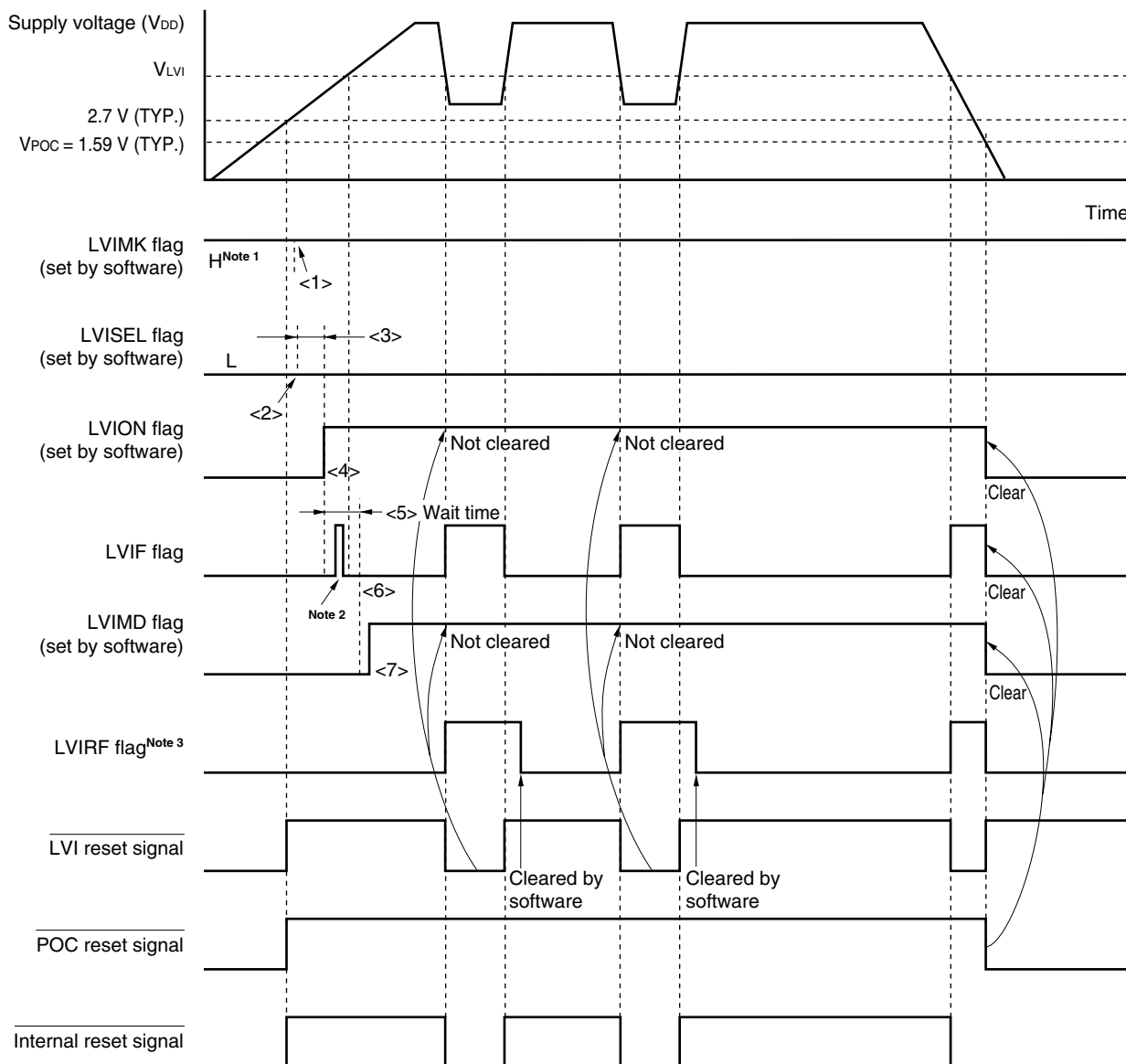


- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
 2. The LVIF flag may be set (1).
 3. LVIRF is bit 0 of the reset control flag register (RESF). For details of RESF, see **CHAPTER 18 RESET FUNCTION**.

Remark <1> to <7> in Figure 20-5 above correspond to <1> to <7> in the description of "When starting operation" in **20.4.1 (1) When detecting level of supply voltage (V_{DD})**.

**Figure 20-5. Timing of Low-Voltage Detector Internal Reset Signal Generation
(Detects Level of Supply Voltage (V_{DD})) (2/2)**

(2) In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
 2. The LVIF flag may be set (1).
 3. LVIRF is bit 0 of the reset control flag register (RESF). For details of RESF, see **CHAPTER 18 RESET FUNCTION**.

Remark <1> to <7> in Figure 20-5 above correspond to <1> to <7> in the description of "When starting operation" in **20.4.1 (1) When detecting level of supply voltage (V_{DD})**.

(2) When detecting level of input voltage from external input pin (EXLVI)

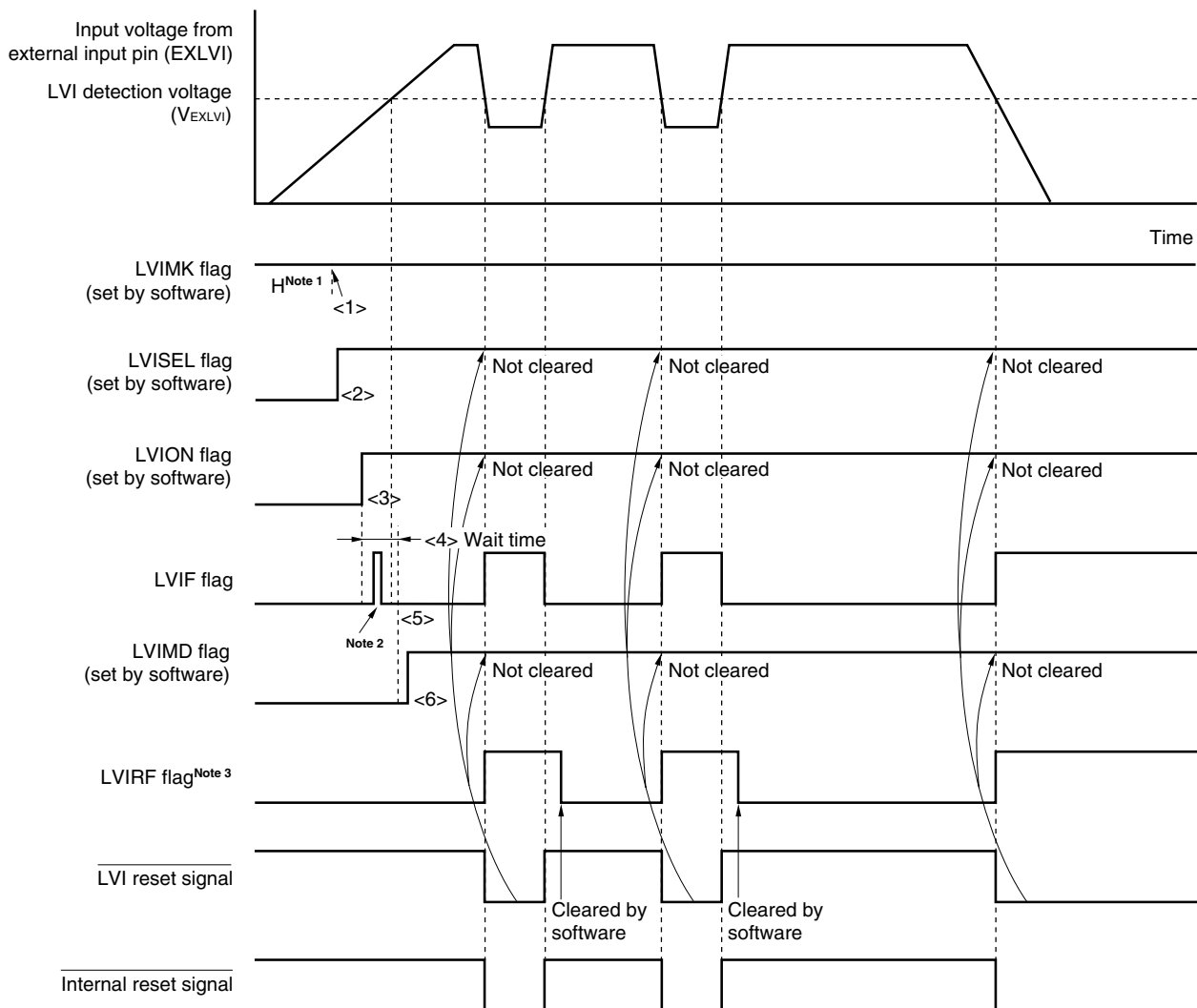
- When starting operation
 - <1> Mask the LVI interrupt (LVIMK = 1).
 - <2> Set bit 2 (LVISEL) of the low-voltage detection register (LVIM) to 1 (detects level of input voltage from external input pin (EXLVI)).
 - <3> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
 - <4> Use software to wait for an operation stabilization time (10 μ s (MIN.)).
 - <5> Wait until it is checked that (input voltage from external input pin (EXLVI) \geq detection voltage ($V_{EXLVI} = 1.21$ V (TYP.))) by bit 0 (LVIF) of LVIM.
 - <6> Set bit 1 (LVIMD) of LVIM to 1 (generates reset signal when the level is detected).

Figure 20-6 shows the timing of the internal reset signal generated by the low-voltage detector. The numbers in this timing chart correspond to <1> to <6> above.

- Cautions**
1. <1> must always be executed. When LVIMK = 0, an interrupt may occur immediately after the processing in <3>.
 2. If input voltage from external input pin (EXLVI) \geq detection voltage ($V_{EXLVI} = 1.21$ V (TYP.)) when LVIMD is set to 1, an internal reset signal is not generated.
 3. Input voltage from external input pin (EXLVI) must be $EXLVI < V_{DD}$.

- When stopping operation
Either of the following procedures must be executed.
 - When using 8-bit memory manipulation instruction:
Write 00H to LVIM.
 - When using 1-bit memory manipulation instruction:
Clear LVIMD to 0 and then LVION to 0.

**Figure 20-6. Timing of Low-Voltage Detector Internal Reset Signal Generation
(Detects Level of Input Voltage from External Input Pin (EXLVI))**



- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
 2. The LVIF flag may be set (1).
 3. LVIRF is bit 0 of the reset control flag register (RESF). For details of RESF, see **CHAPTER 18 RESET FUNCTION**.

Remark <1> to <6> in Figure 20-6 above correspond to <1> to <6> in the description of "When starting operation" in **20.4.1 (2) When detecting level of input voltage from external input pin (EXLVI)**.

20.4.2 When used as interrupt

(1) When detecting level of supply voltage (V_{DD})

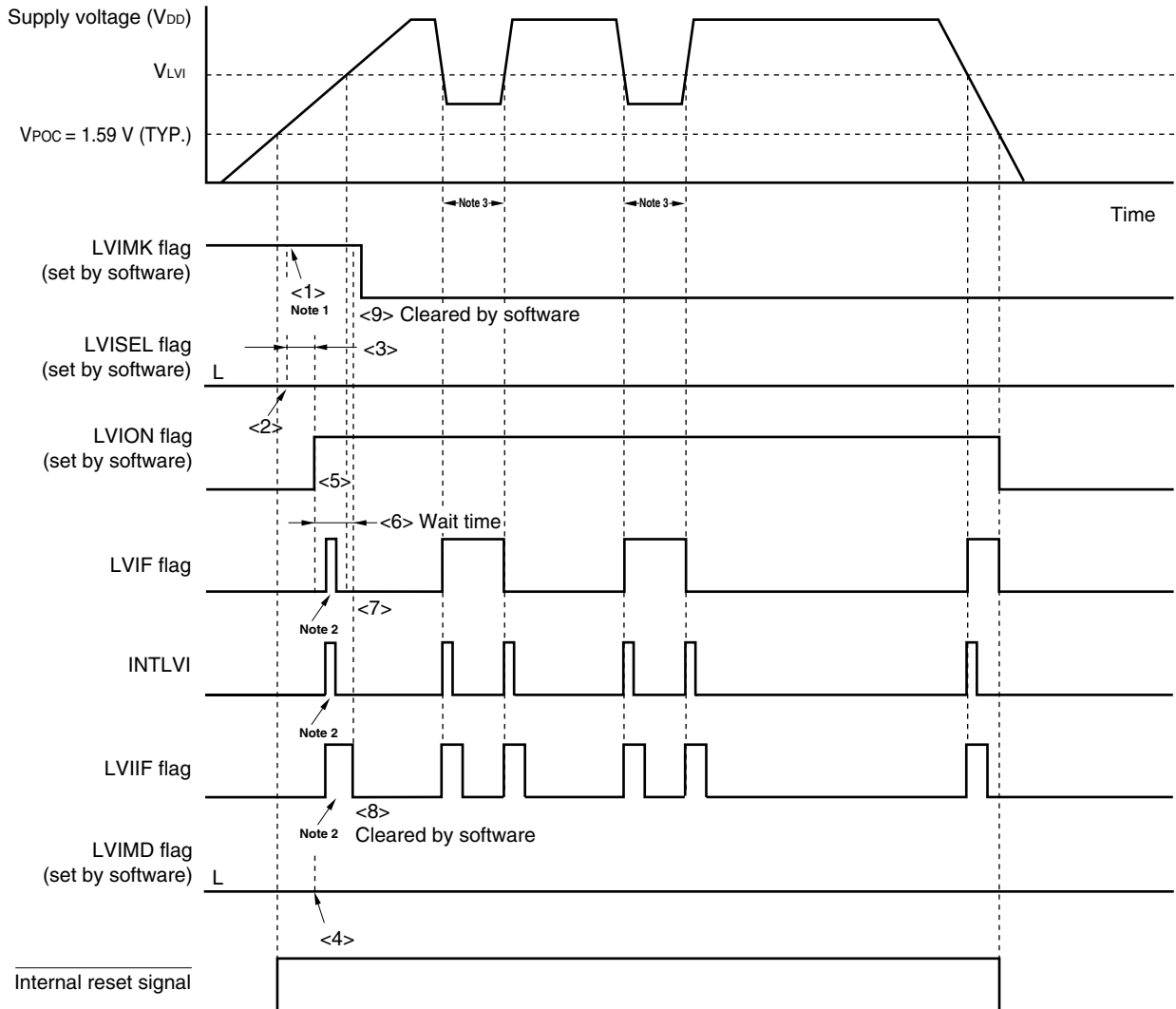
- When starting operation
 - <1> Mask the LVI interrupt ($LVIMK = 1$).
 - <2> Clear bit 2 (LVISEL) of the low-voltage detection register (LVIM) to 0 (detects level of supply voltage (V_{DD})) (default value).
 - <3> Set the detection voltage using bit 0 (LVIS0) of the low-voltage detection level selection register (LVIS).
 - <4> Clear bit 1 (LVIMD) of LVIM to 0 (generates interrupt signal when the level is detected) (default value).
 - <5> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
 - <6> Use software to wait for an operation stabilization time (10 μ s (MIN.)).
 - <7> Confirm that “supply voltage (V_{DD}) \geq detection voltage (V_{LVI})” when detecting the falling edge of V_{DD} , or “supply voltage (V_{DD}) $<$ detection voltage (V_{LVI})” when detecting the rising edge of V_{DD} , at bit 0 (LVIF) of LVIM.
 - <8> Clear the interrupt request flag of LVI (LVIIIF) to 0.
 - <9> Release the interrupt mask flag of LVI (LVIMK).
 - <10> Execute the EI instruction (when vector interrupts are used).

Figure 20-7 shows the timing of the interrupt signal generated by the low-voltage detector. The numbers in this timing chart correspond to <1> to <9> above.

- When stopping operation
Either of the following procedures must be executed.
 - When using 8-bit memory manipulation instruction:
Write 00H to LVIM.
 - When using 1-bit memory manipulation instruction:
Clear LVION to 0.

**Figure 20-7. Timing of Low-Voltage Detector Interrupt Signal Generation
(Detects Level of Supply Voltage (V_{DD})) (1/2)**

(1) In 1.59 V POC mode (option byte: POCMODE = 0)

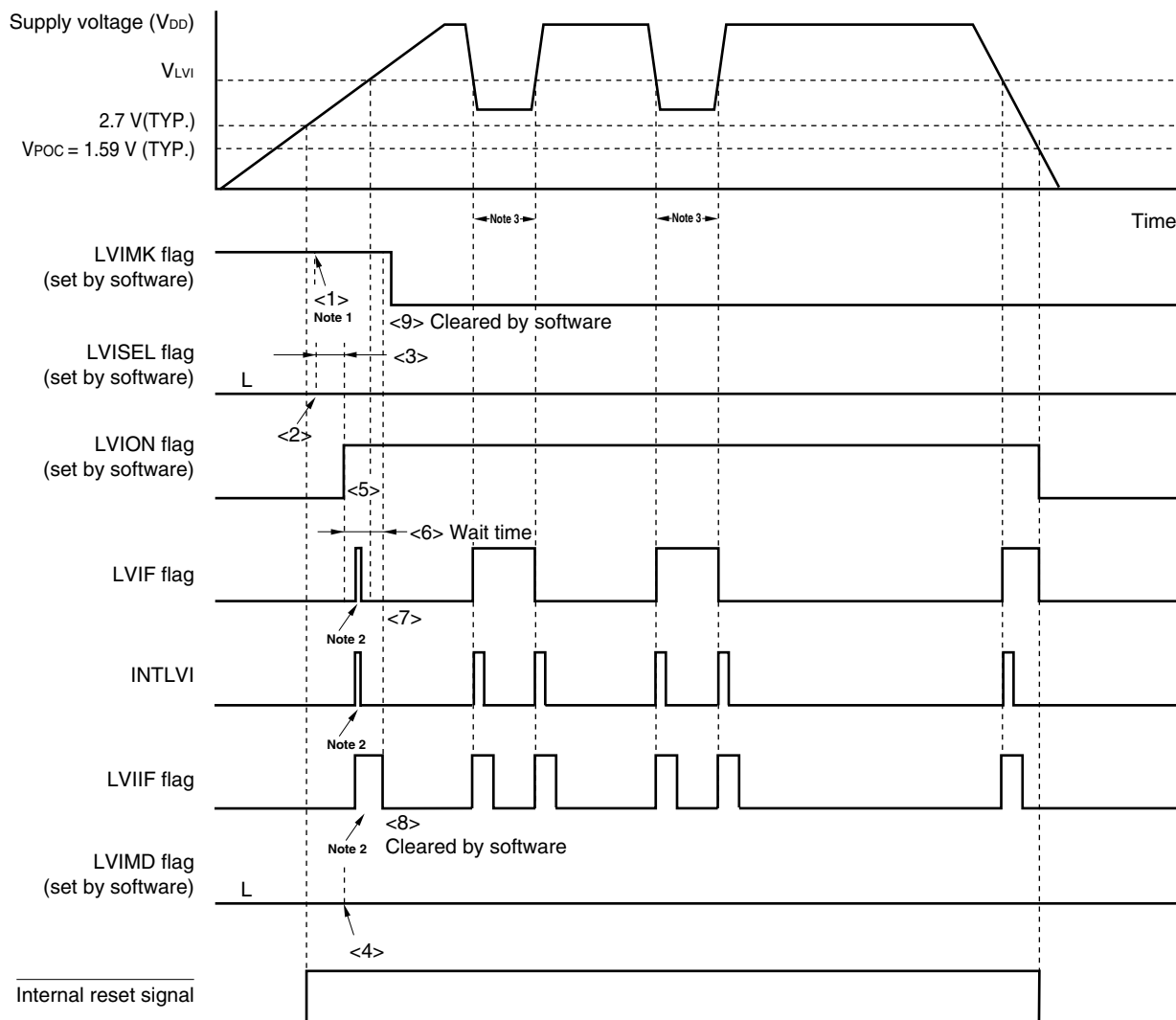


- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
 2. The interrupt request signal (INTLVI) is generated and the LVIF and LVIIIF flags may be set (1).
 3. If LVION is cleared (0) in a state below the LVI detection voltage, an INTLVI signal is generated and LVIIIF becomes 1.

Remark <1> to <9> in Figure 20-7 above correspond to <1> to <9> in the description of "When starting operation" in 20.4.2 (1) When detecting level of supply voltage (V_{DD}).

**Figure 20-7. Timing of Low-Voltage Detector Interrupt Signal Generation
(Detects Level of Supply Voltage (V_{DD})) (2/2)**

(2) In 2.7 V/1.59 V POC mode (option byte: POCMODE = 1)



- Notes**
1. The LVIMK flag is set to "1" by reset signal generation.
 2. The interrupt request signal (INTLVI) is generated and the LVIF and LVIIF flags may be set (1).
 3. If LVION is cleared (0) in a state below the LVI detection voltage, an INTLVI signal is generated and LVIIF becomes 1.

Remark <1> to <9> in Figure 20-7 above correspond to <1> to <9> in the description of "When starting operation" in 20.4.2 (1) When detecting level of supply voltage (V_{DD}).

(2) When detecting level of input voltage from external input pin (EXLVI)

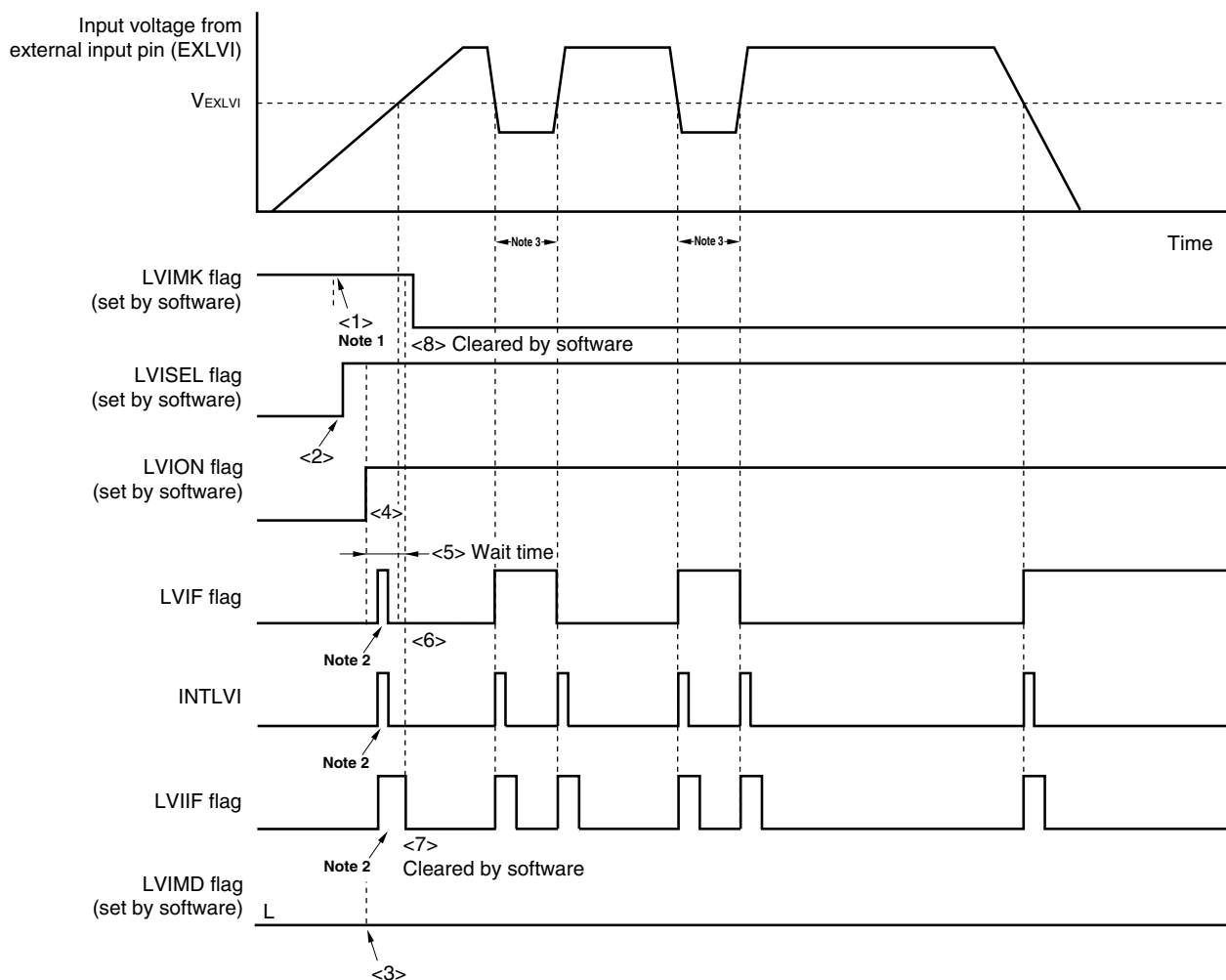
- When starting operation
 - <1> Mask the LVI interrupt (LVIMK = 1).
 - <2> Set bit 2 (LVISEL) of the low-voltage detection register (LVIM) to 1 (detects level of input voltage from external input pin (EXLVI)).
 - <3> Clear bit 1 (LVIMD) of LVIM to 0 (generates interrupt signal when the level is detected) (default value).
 - <4> Set bit 7 (LVION) of LVIM to 1 (enables LVI operation).
 - <5> Use software to wait for an operation stabilization time (10 μ s (MIN.)).
 - <6> Confirm that “input voltage from external input pin (EXLVI) \geq detection voltage ($V_{EXLVI} = 1.21$ V (TYP.))” when detecting the falling edge of EXLVI, or “input voltage from external input pin (EXLVI) $<$ detection voltage ($V_{EXLVI} = 1.21$ V (TYP.))” when detecting the rising edge of EXLVI, at bit 0 (LVIF) of LVIM.
 - <7> Clear the interrupt request flag of LVI (LVIF) to 0.
 - <8> Release the interrupt mask flag of LVI (LVIMK).
 - <9> Execute the EI instruction (when vector interrupts are used).

Figure 20-8 shows the timing of the interrupt signal generated by the low-voltage detector. The numbers in this timing chart correspond to <1> to <8> above.

Caution Input voltage from external input pin (EXLVI) must be $EXLVI < V_{DD}$.

- When stopping operation
Either of the following procedures must be executed.
 - When using 8-bit memory manipulation instruction:
Write 00H to LVIM.
 - When using 1-bit memory manipulation instruction:
Clear LVION to 0.

Figure 20-8. Timing of Low-Voltage Detector Interrupt Signal Generation (Detects Level of Input Voltage from External Input Pin (EXLVI))



- Notes**
1. The LVIMK flag is set to “1” by reset signal generation.
 2. The interrupt request signal (INTLVI) is generated and the LVIF and LVIIF flags may be set (1).
 3. If LVION is cleared (0) in a state below the LVI detection voltage, an INTLVI signal is generated and LVIIF becomes 1.

Remark <1> to <8> in Figure 20-8 above correspond to <1> to <8> in the description of “When starting operation” in **20.4.2 (2) When detecting level of input voltage from external input pin (EXLVI)**.

20.5 Cautions for Low-Voltage Detector

In a system where the supply voltage (V_{DD}) fluctuates for a certain period in the vicinity of the LVI detection voltage (V_{LVI}), the operation is as follows depending on how the low-voltage detector is used.

(1) When used as reset

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 action (1) below.

(2) When used as interrupt

Interrupt requests may be frequently generated. Take (b) of action (2) below.

<Action>

(1) When used as reset

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 (see **Figure 20-9**).

(2) When used as interrupt

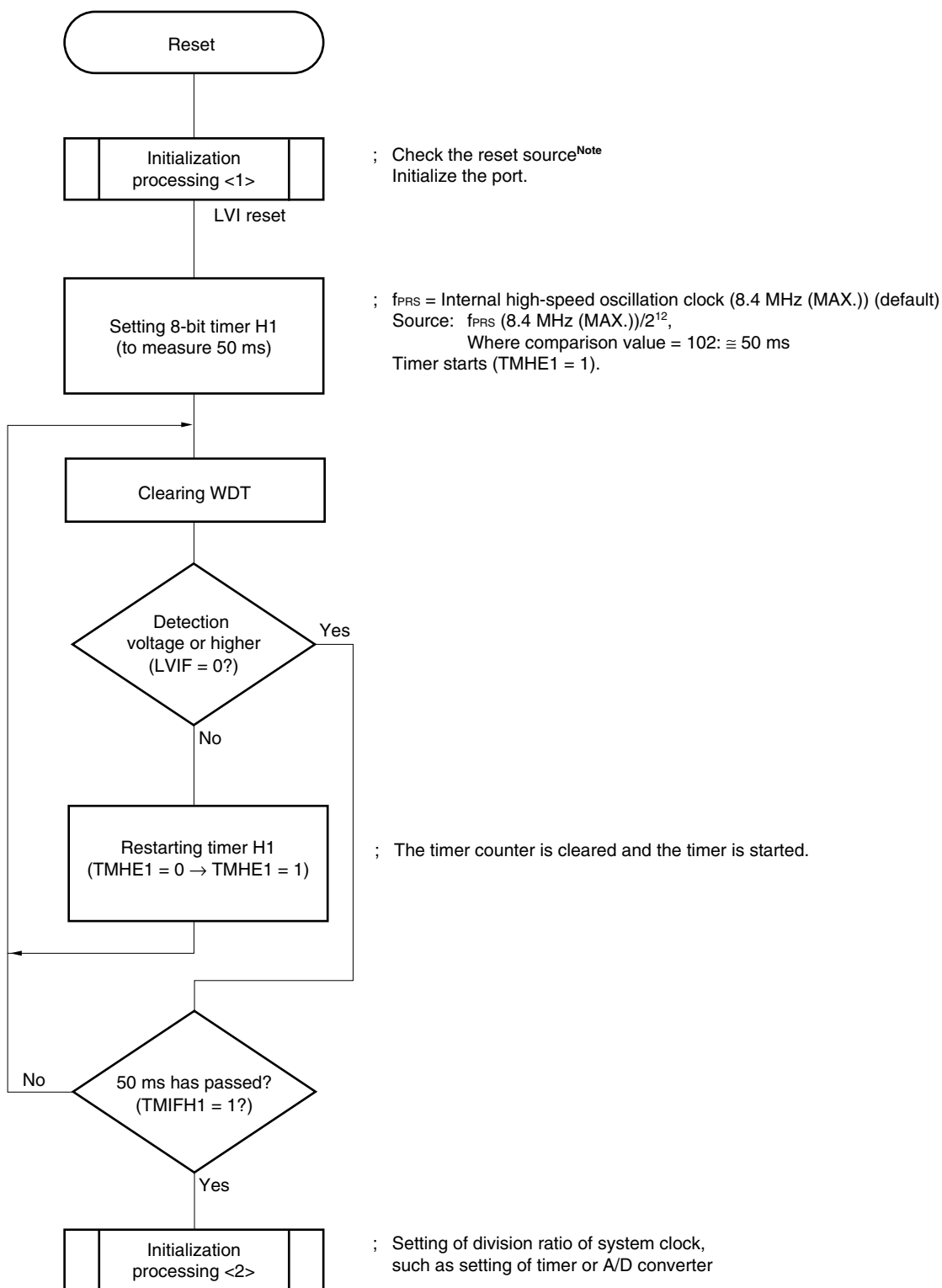
- (a) Confirm that “supply voltage (V_{DD}) \geq detection voltage (V_{LVI})” when detecting the falling edge of V_{DD} , or “supply voltage (V_{DD}) $<$ detection voltage (V_{LVI})” when detecting the rising edge of V_{DD} , in the servicing routine of the LVI interrupt by using bit 0 (LVIF) of the low-voltage detection register (LVIM). Clear bit 0 (LVIIF) of interrupt request flag register 0L (IF0L) to 0.
- (b) In a system where the supply voltage fluctuation period is long in the vicinity of the LVI detection voltage, wait for the supply voltage fluctuation period, confirm that “supply voltage (V_{DD}) \geq detection voltage (V_{LVI})” when detecting the falling edge of V_{DD} , or “supply voltage (V_{DD}) $<$ detection voltage (V_{LVI})” when detecting the rising edge of V_{DD} , using the LVIF flag, and clear the LVIIF flag to 0.

Remark If bit 2 (LVISEL) of the low voltage detection register (LVIM) is set to “1”, the meanings of the above words change as follows.

- Supply voltage (V_{DD}) → Input voltage from external input pin (EXLVI)
- Detection voltage (V_{LVI}) → Detection voltage ($V_{EXLVI} = 1.21$ V)

Figure 20-9. Example of Software Processing After Reset Release (1/2)

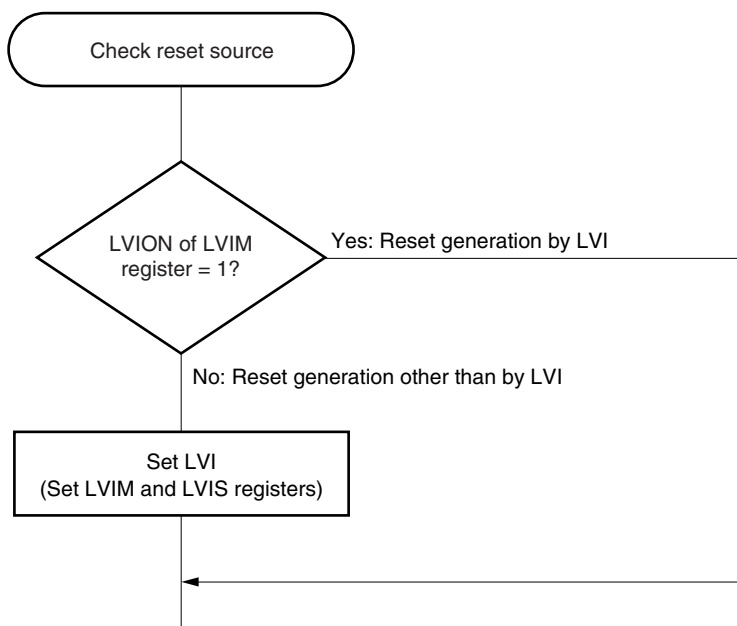
- If supply voltage fluctuation is 50 ms or less in vicinity of LVI detection voltage



Note A flowchart is shown on the next page.

Figure 20-9. Example of Software Processing After Reset Release (2/2)

- Checking reset source



CHAPTER 21 OPTION BYTE

21.1 Functions of Option Bytes

The flash memory at 0080H to 0084H of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B is an option byte area. When power is turned on or when the device is restarted from the reset status, the device automatically references the option bytes and sets specified functions. When using the product, be sure to set the following functions by using the option bytes.

(1) 0080H

- Internal low-speed oscillator operation
 - Can be stopped by software
 - Cannot be stopped
- Watchdog timer overflow time setting
- Watchdog timer counter operation
 - Enabled counter operation
 - Disabled counter operation
- Watchdog timer window open period setting

(2) 0081H

- Selecting POC mode
 - During 2.7 V/1.59 V POC mode operation (POCMODE = 1)
The device is in the reset state upon power application and until the supply voltage reaches 2.7 V (TYP.). It is released from the reset state when the voltage exceeds 2.7 V (TYP.). After that, POC is not detected at 2.7 V but is detected at 1.59 V (TYP.).
If the supply voltage rises to 4.0 V after power application at a rate slower than 0.5 V/ms (MIN.), use of the 2.7 V/1.59 V POC mode is recommended.
 - During 1.59 V POC mode operation (POCMODE = 0)
The device is in the reset state upon power application and until the supply voltage reaches 1.59 V (TYP.). It is released from the reset state when the voltage exceeds 1.59 V (TYP.). After that, POC is detected at 1.59 V (TYP.), in the same manner as on power application.

(3) 0082H

Be sure to set 00H.

(4) 0083H

Be sure to set 00H.

(5) 0084H

- On-chip debug operation control (R7F0C999B only)
 - Disabling on-chip debug operation
 - Enabling on-chip debug operation and erasing data of the flash memory in case authentication of the on-chip debug security ID fails
 - Enabling on-chip debug operation and not erasing data of the flash memory even in case authentication of the on-chip debug security ID fails

21.2 Format of Option Byte

The format of the option byte is shown below.

Figure 21-1. Format of Option Byte (1/2)

Address: 0080H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|---------|--|-----------------------------------|------------------------------|-------|-------|--------|
| 0 | WINDOW1 | WINDOW0 | WDTON | WDCS2 | WDCS1 | WDCS0 | LSROSC |
| WINDOW1 | | WINDOW0 | Watchdog timer window open period | | | | |
| 0 | | 0 | 25% | | | | |
| 0 | | 1 | 50% | | | | |
| 1 | | 0 | 75% | | | | |
| 1 | | 1 | 100% | | | | |
| WDTON | | Operation control of watchdog timer counter/illegal access detection | | | | | |
| 0 | | Counter operation disabled (counting stopped after reset), illegal access detection operation disabled | | | | | |
| 1 | | Counter operation enabled (counting started after reset), illegal access detection operation enabled | | | | | |
| WDCS2 | | WDCS1 | WDCS0 | Watchdog timer overflow time | | | |
| 0 | | 0 | 0 | $2^{10}/f_{RL}$ (3.88 ms) | | | |
| 0 | | 0 | 1 | $2^{11}/f_{RL}$ (7.76 ms) | | | |
| 0 | | 1 | 0 | $2^{12}/f_{RL}$ (15.52 ms) | | | |
| 0 | | 1 | 1 | $2^{13}/f_{RL}$ (31.03 ms) | | | |
| 1 | | 0 | 0 | $2^{14}/f_{RL}$ (62.06 ms) | | | |
| 1 | | 0 | 1 | $2^{15}/f_{RL}$ (124.12 ms) | | | |
| 1 | | 1 | 0 | $2^{16}/f_{RL}$ (248.24 ms) | | | |
| 1 | | 1 | 1 | $2^{17}/f_{RL}$ (496.48 ms) | | | |
| LSROSC | | Internal low-speed oscillator operation | | | | | |
| 0 | | Can be stopped by software (stopped when 1 is written to bit 1 (LSRSTOP) of RCM register) | | | | | |
| 1 | | Cannot be stopped (not stopped even if 1 is written to LSRSTOP bit) | | | | | |

- Cautions**
1. The combination of $WDCS2 = WDCS1 = WDCS0 = 0$ and $WINDOW1 = WINDOW0 = 0$ is prohibited.
 2. If $LSROSC = 0$ (oscillation can be stopped by software), the count clock is not supplied to the watchdog timer in the HALT and STOP modes, regardless of the setting of bit 0 (LSRSTOP) of the internal oscillation mode register (RCM).
When 8-bit timer H1 operates with the internal low-speed oscillation clock, the count clock is supplied to 8-bit timer H1 even in the HALT/STOP mode.
 3. Be sure to clear bit 7 to "0".

- Remarks**
1. f_{RL} : Internal low-speed oscillation clock frequency
 2. (): $f_{RL} = 264$ kHz (MAX.)

Figure 21-1. Format of Option Byte (2/2)

Address: 0081H^{Note}

| | | | | | | | |
|---|---|---|---|---|---|---|---------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | POCMODE |

| POCMODE | POC mode selection |
|---------|---------------------------|
| 0 | 1.59 V POC mode (default) |
| 1 | 2.7 V/1.59 V POC mode |

Note To change the setting for the POC mode, set the value to 0081H again after batch erasure (chip erasure) of the flash memory. The setting cannot be changed after the memory of the specified block is erased.

Caution Be sure to clear bits 7 to 1 to “0”.

Address: 0082H^{Note}

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Note Be sure to set 00H to 0082H, as this address is a reserved area.

Address: 0083H^{Note}

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Note Be sure to set 00H to 0083H, as this address is a reserved area.

Address: 0084H^{Note}

| | | | | | | | |
|---|---|---|---|---|---|--------|--------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | OCDEN1 | OCDEN0 |

| OCDEN1 | OCDEN0 | On-chip debug operation control |
|--------|--------|---|
| 0 | 0 | Operation disabled |
| 0 | 1 | Setting prohibited |
| 1 | 0 | Operation enabled. Does not erase data of the flash memory in case authentication of the on-chip debug security ID fails. |
| 1 | 1 | Operation enabled. Erases data of the flash memory in case authentication of the on-chip debug security ID fails. |

Note In the R7F0C011B, R7F0C012B, and R7F0C013B, be sure to set 00H to 0084H, as this address is a reserved area.

Caution In the R7F0C999B, be sure to clear bits 7 to 2 to “0”.

Remark For the on-chip debug security ID, see **CHAPTER 23 ON-CHIP DEBUG FUNCTION**.

Here is an example of description of the software for setting the option bytes.

| | | | |
|---------|------|----------|---|
| OPT | CSEG | AT 0080H | |
| OPTION: | DB | 30H | ; Enables watchdog timer operation (illegal access detection operation), ; Window open period of watchdog timer: 50%, ; Overflow time of watchdog timer: $2^{10}/f_{RL}$, ; Internal low-speed oscillator can be stopped by software. |
| | DB | 00H | ; 1.59 V POC mode |
| | DB | 00H | ; Reserved area |
| | DB | 00H | ; Reserved area |
| | DB | 00H | ; On-chip debug operation disabled |

Remark Referencing of the option byte is performed during reset processing. For the reset processing timing, see **CHAPTER 18 RESET FUNCTION**.

CHAPTER 22 FLASH MEMORY

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B incorporate the flash memory to which a program can be written, erased, and overwritten while mounted on the board.

22.1 Internal Memory Size Switching Register

Select the internal memory capacity using the internal memory size switching register (IMS).

IMS is set by an 8-bit memory manipulation instruction.

Reset signal generation sets IMS to CFH.

Caution Be sure to set each product to the values shown in Table 22-1 after a reset release.

Figure 22-1. Format of Internal Memory Size Switching Register (IMS)

Address: FFF0H After reset: CFH R/W

| | | | | | | | | |
|--------|------|------|------|---|------|------|------|------|
| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IMS | RAM2 | RAM1 | RAM0 | 0 | ROM3 | ROM2 | ROM1 | ROM0 |

| | | | |
|------------------|------|------|--|
| RAM2 | RAM1 | RAM0 | Internal high-speed RAM capacity selection |
| 0 | 0 | 0 | 768 bytes |
| 1 | 1 | 0 | 1024 bytes |
| Other than above | | | Setting prohibited |

| | | | | |
|------------------|------|------|------|---------------------------------|
| ROM3 | ROM2 | ROM1 | ROM0 | Internal ROM capacity selection |
| 0 | 1 | 0 | 0 | 16 KB |
| 0 | 1 | 1 | 0 | 24 KB |
| 1 | 0 | 0 | 0 | 32 KB |
| Other than above | | | | Setting prohibited |

Table 22-1. Internal Memory Size Switching Register Settings

| Product Name | IMS Setting |
|--------------|-------------|
| R7F0C011B | 04H |
| R7F0C012B | C6H |
| R7F0C013B | C8H |
| R7F0C999B | C8H |

22.2 Writing with Flash Memory Programmer

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 R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B 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

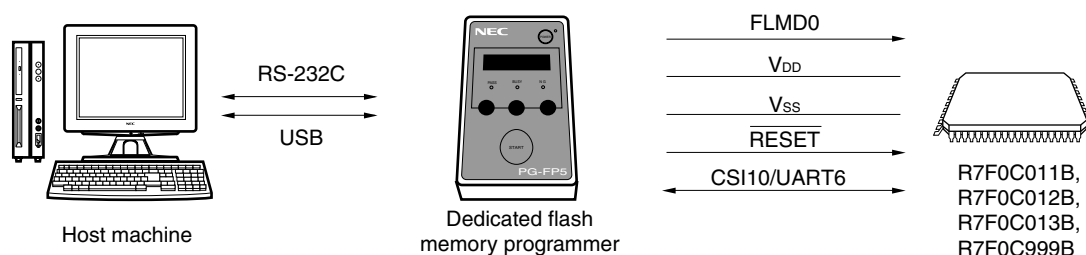
Data can be written to the flash memory with a dedicated program adapter (FA series) before the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B is mounted on the target system.

Remark The FA series is a product of Naito Densai Machida Mfg. Co., Ltd.

22.3 Programming Environment

The environment required for writing a program to the flash memory of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B is illustrated below.

Figure 22-2. 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 R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B, CSI10 or UART6 is used for manipulation such as writing and erasing. To write the flash memory off-board, a dedicated program adapter (FA series) is necessary.

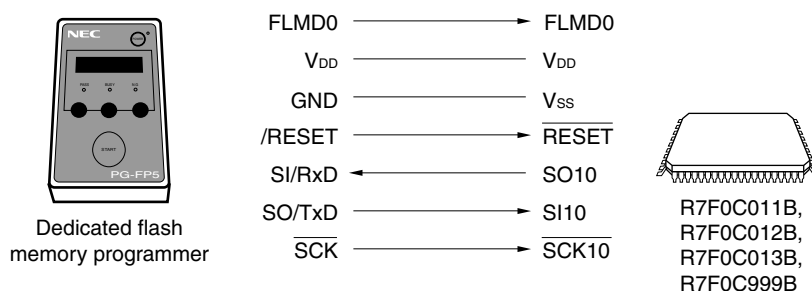
22.4 Communication Mode

Communication between the dedicated flash memory programmer and the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B is established by serial communication via CSI10 or UART6 of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B.

(1) CSI10

Transfer rate: 2.4 kHz to 2.5 MHz

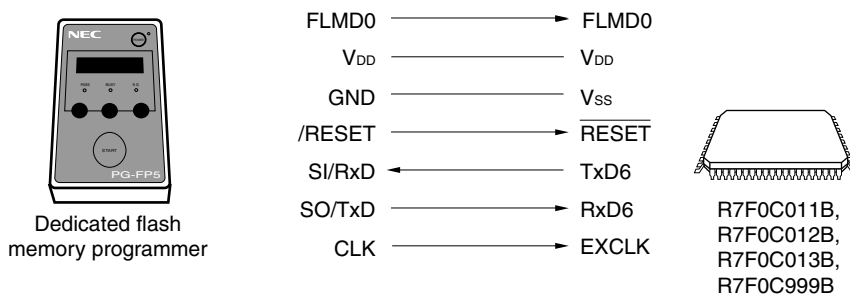
Figure 22-3. Communication with Dedicated Flash memory programmer (CSI10)



(2) UART6

Transfer rate: 115200 bps

Figure 22-4. Communication with Dedicated Flash memory programmer (UART6)



The dedicated flash memory programmer generates the following signals for the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B. For details, refer to the user's manual for the PG-FP5 or FL-PR5.

Table 22-2. Pin Connection

| Dedicated Flash memory programmer | | | R7F0C011B, R7F0C012B, R7F0C013B, R7F0C999B | Connection | |
|-----------------------------------|--------|---|---|---------------------|---------------------|
| Signal Name | I/O | Pin Function | Pin Name | CSI10 | UART6 |
| FLMD0 | Output | Mode signal | FLMD0 | ◎ | ◎ |
| V _{DD} | I/O | V _{DD} voltage generation/power monitoring | V _{DD} | ◎ | ◎ |
| GND | – | Ground | V _{SS} | ◎ | ◎ |
| CLK | Output | Clock output to R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B | EXCLK/X2/P122 | × ^{Note 1} | ○ ^{Note 2} |
| /RESET | Output | Reset signal | $\overline{\text{RESET}}$ | ◎ | ◎ |
| SI/RxD | Input | Receive signal | SO10/TxD6 | ◎ | ◎ |
| SO/TxD | Output | Transmit signal | SI10/RxD6 | ◎ | ◎ |
| SCK | Output | Transfer clock | $\overline{\text{SCK10}}$ | ◎ | × |

Notes 1. Only the internal high-speed oscillation clock (f_{RH}) can be used when CSI10 is used.

2. Only the X1 clock (f_x) or external main system clock (f_{EXCLK}) can be used when UART6 is used.

Remark ◎: Be sure to connect the pin.

○: The pin does not have to be connected if the signal is generated on the target board.

×: The pin does not have to be connected.

For the pins not to be used when the dedicated program adapter (FA series) is used, perform the processing described under the recommended connection of unused pins shown in **Table 22-3 Processing of Unused Pins When the Flash Memory Write Adapter Is Connected (Required)**.

Table 22-3. Processing of Unused Pins When the Flash Memory Write Adapter Is Connected (Required)

| Pin name | Pin processing |
|------------|--|
| P00, P01 | Independently connect to V _{SS} via a resistor. ^{Note 1} |
| P10, P11 | Independently connect to V _{SS} via a resistor. ^{Note 2} |
| P14 | Independently connect to V _{SS} via a resistor. ^{Note 3} |
| P16, P17 | Independently connect to V _{SS} via a resistor. ^{Note 1} |
| P30 to P33 | |
| P60, P61 | Independently connect to V _{SS} via a resistor, or connect directly to EV _{SS} . |
| P70, P71 | Independently connect to V _{SS} via a resistor. ^{Note 1} |
| P120 | |

- Notes 1.** These pins may be directly connected to V_{SS}, without using a resistor, when design is performed so that operation is not switched to the normal operation mode on the flash memory write adapter board during flash memory programming.
- 2.** Connect these pins with the programmer when communicating with the dedicated flash memory programmer via serial communication by CSI10.
- 3.** Connect this pin with the programmer when communicating with the dedicated flash memory programmer via serial communication by UART6.

22.5 Connection of Pins on Board

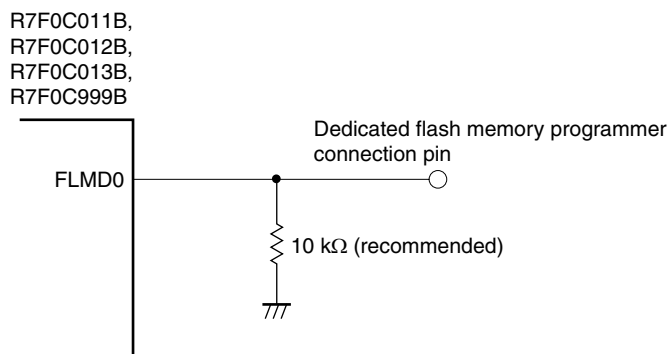
To write the flash memory on-board, connectors that connect the dedicated flash memory programmer must be provided on the target system. First provide a function that selects the normal operation mode or flash memory programming mode on the board.

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.

22.5.1 FLMD0 pin

In the normal operation mode, 0 V is input to the FLMD0 pin. In the flash memory programming mode, the V_{DD} write voltage is supplied to the FLMD0 pin. An FLMD0 pin connection example is shown below.

Figure 22-5. FLMD0 Pin Connection Example



22.5.2 Serial interface pins

The pins used by each serial interface are listed below.

Table 22-4. Pins Used by Each Serial Interface

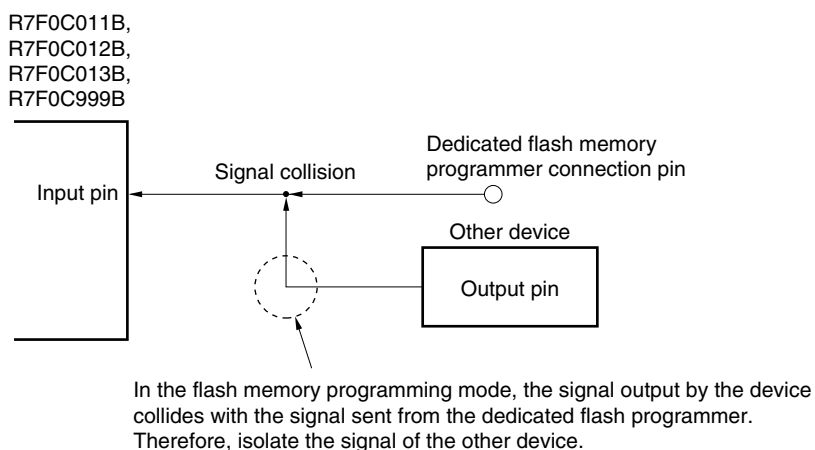
| Serial Interface | Pins Used |
|------------------|-------------------|
| CSI10 | SO10, SI10, SCK10 |
| UART6 | TxD6, RxD6 |

To connect the dedicated flash memory programmer to the pins of a serial interface that is connected to another device on the board, care must be exercised so that signals do not collide or that the other device does not malfunction.

(1) Signal collision

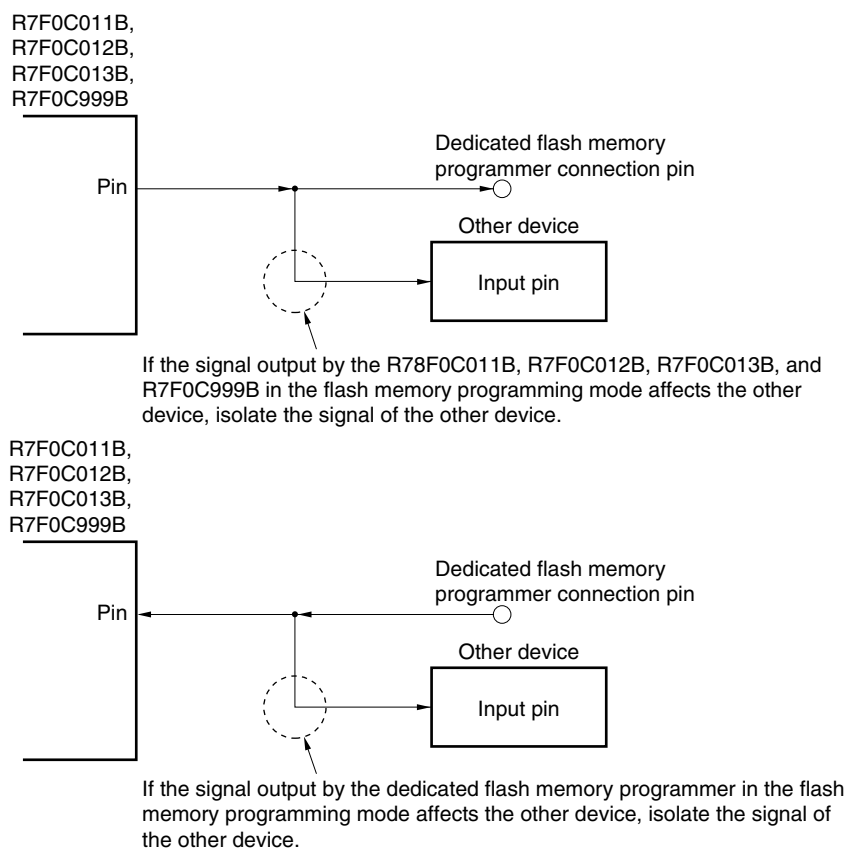
If the dedicated flash memory programmer (output) is connected to a pin (input) of a serial interface connected to another device (output), signal collision takes place. To avoid this collision, either isolate the connection with the other device, or make the other device go into an output high-impedance state.

Figure 22-6. Signal Collision (Input Pin of Serial Interface)



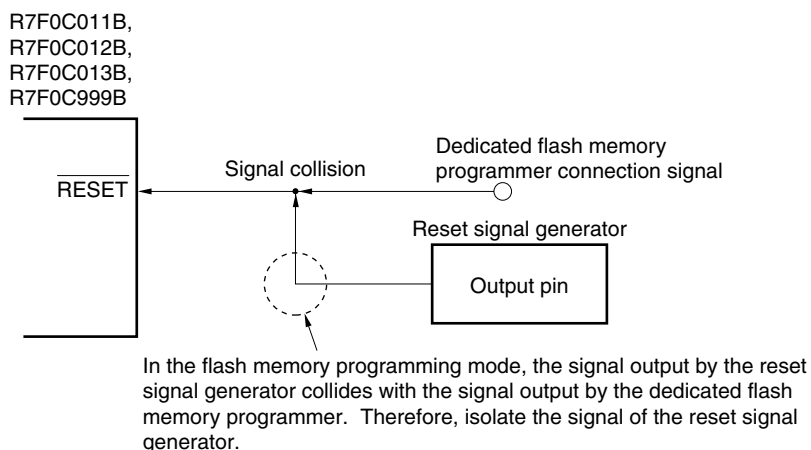
(2) Malfunction of other device

If the dedicated flash memory programmer (output or input) is connected to a pin (input or output) of a serial interface connected to another device (input), a signal may be output to the other device, causing the device to malfunction. To avoid this malfunction, isolate the connection with the other device.

Figure 22-7. Malfunction of Other Device**22.5.3 $\overline{\text{RESET}}$ pin**

If the reset signal of the dedicated flash memory programmer is connected to the $\overline{\text{RESET}}$ pin that is connected to the reset signal generator on the board, signal collision takes place. To prevent this collision, isolate the connection with the reset signal generator.

If the reset signal is input from the user system while the flash memory programming mode is set, the flash memory will not be correctly programmed. Do not input any signal other than the reset signal of the dedicated flash memory programmer.

Figure 22-8. Signal Collision ($\overline{\text{RESET}}$ Pin)

22.5.4 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 V_{DD} or V_{SS} via a resistor.

22.5.5 REGC pin

Connect the REGC pin to V_{SS} via a capacitor (0.47 to 1 μF) in the same manner as during normal operation.

22.5.6 Other signal pins

Connect X1 and X2 in the same status as in the normal operation mode when using the on-board clock.

To input the operating clock from the dedicated flash memory programmer, however, connect CLK of the programmer to EXCLK/X2/P122.

- Cautions**
1. Only the internal high-speed oscillation clock (f_{RH}) can be used when CS110 is used.
 2. Only the X1 clock (f_x) or external main system clock (f_{EXCLK}) can be used when UART6 is used.

22.5.7 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.

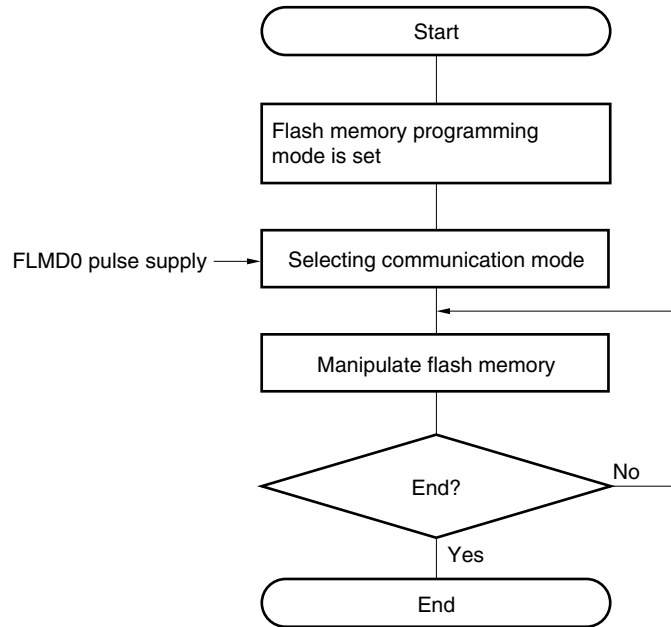
However, be sure to connect the V_{DD} and V_{SS} pins to V_{DD} and GND of the flash memory programmer to use the power monitor function with the flash memory programmer, even when using the on-board supply voltage.

22.6 Programming Method

22.6.1 Controlling flash memory

The following figure illustrates the procedure to manipulate the flash memory.

Figure 22-9. Flash Memory Manipulation Procedure



22.6.2 Flash memory programming mode

To rewrite the contents of the flash memory by using the dedicated flash memory programmer, set the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B in the flash memory programming mode. To set the mode, set the FLMD0 pin to V_{DD} and clear the reset signal.

Change the mode by using a jumper when writing the flash memory on-board.

Figure 22-10. Flash Memory Programming Mode

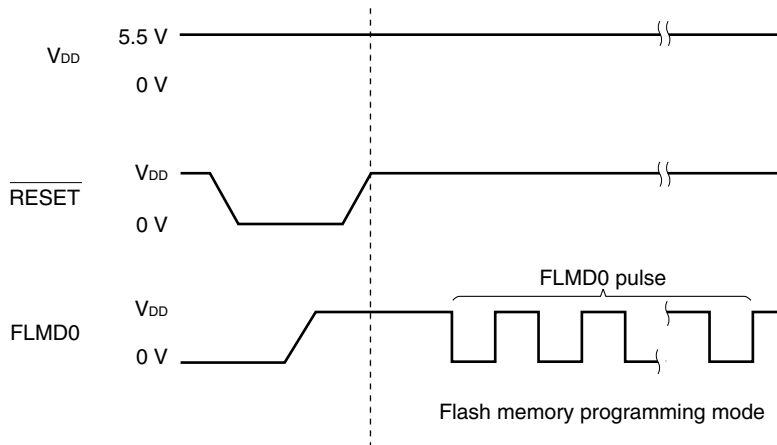


Table 22-5. Relationship Between FLMD0 Pin and Operation Mode After Reset Release

| FLMD0 | Operation Mode |
|----------|-------------------------------|
| 0 | Normal operation mode |
| V_{DD} | Flash memory programming mode |

22.6.3 Selecting communication mode

In the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B, a communication mode is selected by inputting pulses to the FLMD0 pin after the dedicated flash memory programming mode is entered. These FLMD0 pulses are generated by the flash memory programmer.

The following table shows the relationship between the number of pulses and communication modes.

Table 22-6. Communication Modes

| Communication Mode | Standard Setting ^{Note 1} | | | | Pins Used | Peripheral Clock | Number of FLMD0 Pulses |
|---------------------------|------------------------------------|-------------------------------|-------------------------------|---------------|-------------------|------------------|------------------------|
| | Port | Speed | Frequency | Multiply Rate | | | |
| UART (UART6) | UART-Ext-Osc | 115,200 bps ^{Note 3} | 2 to 20 MHz ^{Note 2} | 1.0 | TxD6, RxD6 | f_x | 0 |
| | UART-Ext-FP5CK | | | | | f_{EXCLK} | 3 |
| 3-wire serial I/O (CSI10) | CSI-Internal-OSC | 2.4 kHz to 2.5 MHz | – | | SO10, SI10, SCK10 | f_{RH} | 8 |

- Notes**
1. Selection items for Standard settings on GUI of the flash memory programmer.
 2. The possible setting range differs depending on the voltage. For details, refer to the chapter of electrical specifications.
 3. 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.

Caution When UART6 is selected, the receive clock is calculated based on the reset command sent from the dedicated flash memory programmer after the FLMD0 pulse has been received.

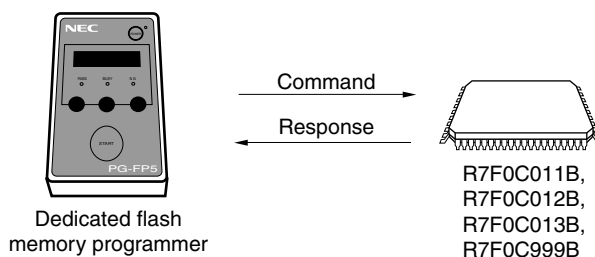
Remark

f_x : X1 clock
 f_{EXCLK} : External main system clock
 f_{RH} : Internal high-speed oscillation clock

22.6.4 Communication commands

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B communicate with the dedicated flash memory programmer by using commands. The signals sent from the flash memory programmer to the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B are called commands, and the signals sent from the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B to the dedicated flash memory programmer are called response.

Figure 22-11. Communication Commands



The flash memory control commands of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B are listed in the table below. All these commands are issued from the programmer and the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B perform processing corresponding to the respective commands.

Table 22-7. 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 | Chip Erase | Erases the entire flash memory. |
| | 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. |
| Getting information | Status | Gets the current operating status (status data). |
| | Silicon Signature | Gets R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B information (such as the part number and flash memory configuration). |
| | Version Get | Gets the product version and firmware version. |
| | Checksum | Gets the checksum data for a specified area. |
| Security | Security Set | Sets security information. |
| Others | Reset | Used to detect synchronization status of communication. |
| | Oscillating Frequency Set | Specifies an oscillation frequency. |

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B return a response for the command issued by the dedicated flash memory programmer. The response names sent from the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B are listed below.

Table 22-8. Response Names

| Response Name | Function |
|---------------|------------------------------------|
| ACK | Acknowledges command/data. |
| NAK | Acknowledges illegal command/data. |

22.7 Security Settings

The R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B 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. The security setting is valid when the programming mode is set next.

- Disabling batch erase (chip erase)

Execution of the block erase and batch erase (chip erase) commands for entire blocks in the flash memory is prohibited by this setting during on-board/off-board programming. Once execution of the batch erase (chip erase) command is prohibited, all of the prohibition settings (including prohibition of batch erase (chip erase)) can no longer be cancelled.

Caution After the security setting for the batch erase is set, erasure cannot be performed for the device. In addition, even if a write command is executed, data different from that which has already been written to the flash memory cannot be written, because the erase command is disabled.

- 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.

- Disabling write

Execution of the write and block erase commands for entire blocks in the flash memory is prohibited during on-board/off-board programming.

- Disabling rewriting boot cluster 0

Execution of the block erase command and write command on boot cluster 0 (0000H to 0FFFH) in the flash memory is prohibited by this setting. Execution of the batch erase (chip erase) command is also prohibited by this setting.

Caution If a security setting that rewrites boot cluster 0 has been applied, the rewriting of boot cluster 0 and the batch erase (chip erase) will not be executed for the device.

The batch erase (chip erase), block erase, write commands, and rewriting boot cluster 0 are enabled by the default setting when the flash memory is shipped. Security can be set by on-board/off-board programming. Each security setting can be used in combination.

Prohibition of erasing blocks and writing is cleared by executing the batch erase (chip erase) command.

Table 22-9 shows the relationship between the erase and write commands when the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B security function is enabled.

Table 22-9. Relationship Between Enabling Security Function and Command

- **During on-board/off-board programming**

| Valid Security | Executed Command | | |
|---|---------------------------|----------------------------------|------------------------------------|
| | Batch Erase (Chip Erase) | Block Erase | Write |
| Prohibition of batch erase (chip erase) | Cannot be erased in batch | Blocks cannot be erased. | Can be performed ^{Note} . |
| Prohibition of block erase | Can be erased in batch. | | Can be performed. |
| Prohibition of writing | | | Cannot be performed. |
| Prohibition of rewriting boot cluster 0 | Cannot be erased in batch | Boot cluster 0 cannot be erased. | Boot cluster 0 cannot be written. |

Note Confirm that no data has been written to the write area. Because data cannot be erased after batch erase (chip erase) is prohibited, do not write data if the data has not been erased.

Table 22-10 shows how to perform security settings in each programming mode.

Table 22-10. Setting Security in Each Programming Mode

- **On-board/off-board programming**

| Security | Security Setting | How to Disable Security Setting |
|---|--|--|
| Prohibition of batch erase (chip erase) | Set via GUI of dedicated flash memory programmer, etc. | Cannot be disabled after set. |
| Prohibition of block erase | | Execute batch erase (chip erase) command |
| Prohibition of writing | | |
| Prohibition of rewriting boot cluster 0 | | Cannot be disabled after set. |

CHAPTER 23 ON-CHIP DEBUG FUNCTION (R7F0C999B ONLY)

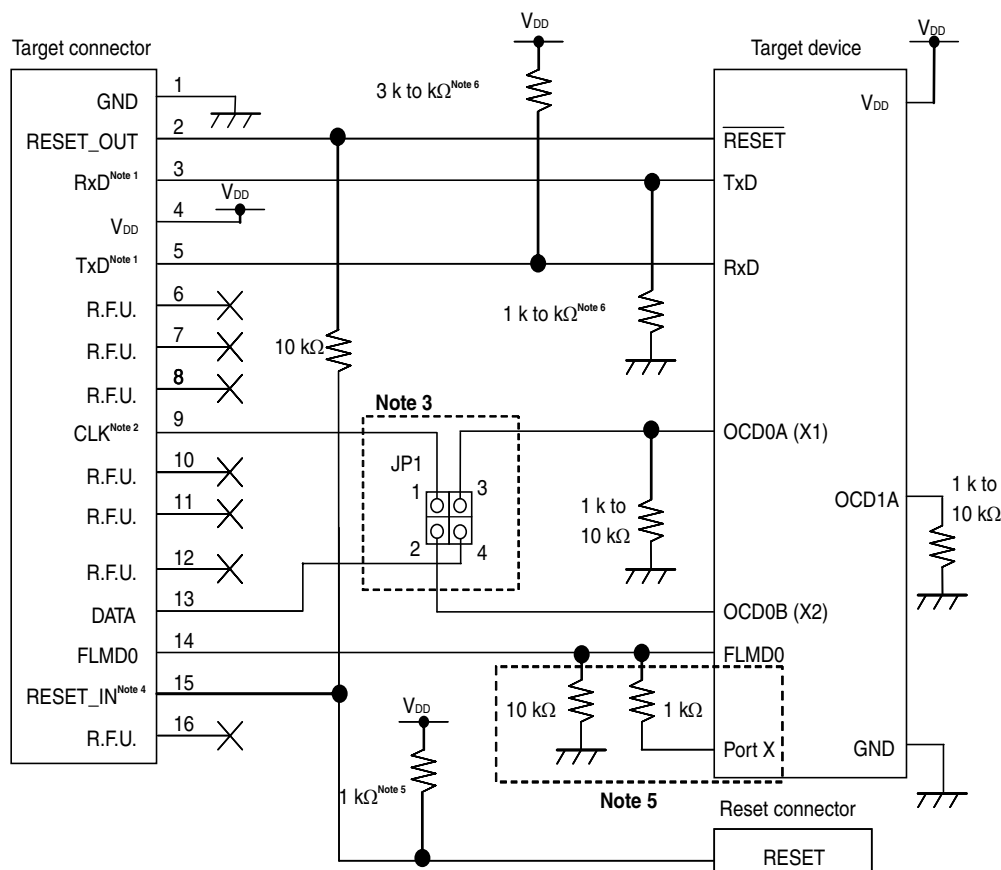
23.1 Connecting QB-MINI2 to R7F0C999B

The R7F0C999B uses the V_{DD} , $\overline{\text{RESET}}$, OCD0A/X1 (or OCD1A/P31), OCD0B/X2 (or OCD1B/P32), and V_{SS} pins to communicate with the host machine via an on-chip debug emulator (QB-MINI2).

- Cautions**
1. The R7F0C999B has 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. When transitioning to STOP mode during on-chip debugging, oscillation of the internal high-speed oscillator continues, but the on-chip debug operation is not affected.

Figure 23-1. Connection Example of QB-MINI2 and R7F0C999B (1/4)

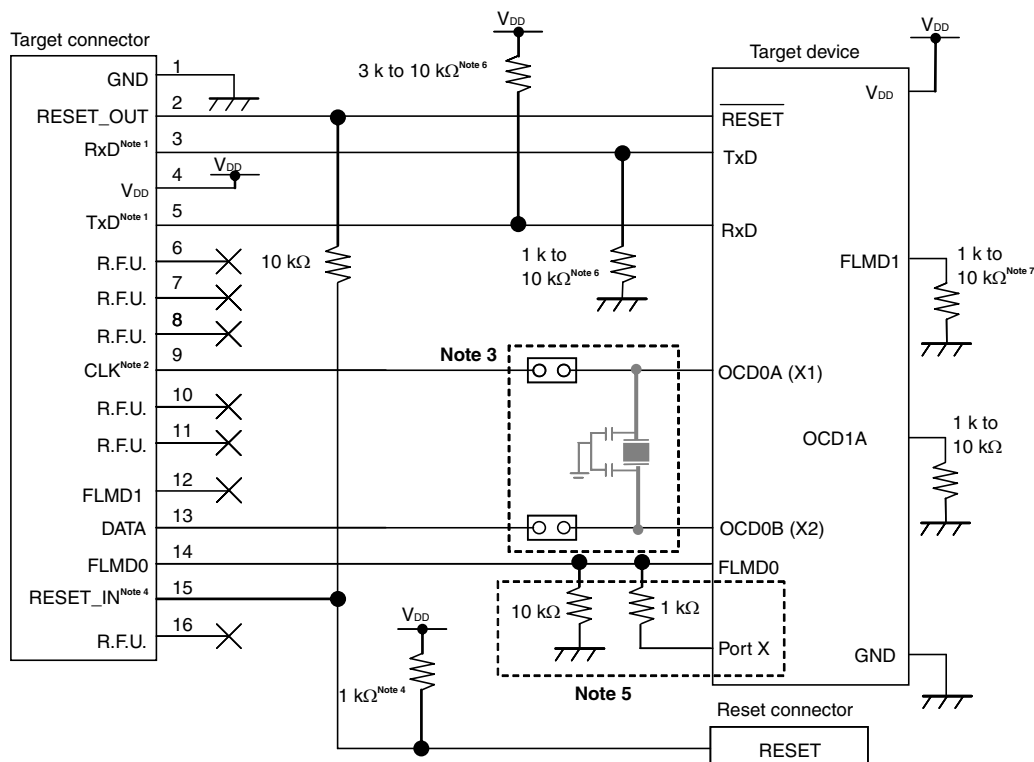
(1) When both debugging and programming are performed
(not communicating by using OCD0A and OCD0B pins or X1 oscillator)



- Notes**
1. Connect TxD (transmit side) of the target device to RxD (receive side) of the target connector, and TxD (transmit side) of the target connector to RxD (receive side) of the target device.
 2. Circuits other than the X1 oscillator can be used to generate the clock signal for the target device during on-chip debugging.
Only a 4, 8, or 16 MHz clock signal generated in QB-MINI2 can be used during flash programming.
 3. Short 1 to 3 and 2 to 4 of JP1 during on-chip debugging.
Short 1 to 2 of JP1 and leave 3 to 4 of JP1 open during flash programming.
Leave JP1 open when using the target device (when QB-MINI2 is not connected).
 4. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100Ω or less).
 5. The circuit is designed for flash self programming, which controls the FLMD0 pin by user program.
Connect any port that can output data to FLMD0 via a resistor.
When not using flash self programming, process the pins according to the device specifications.
 6. This is the processing for the pins that are unused (the inputs are left open) when the target device operates (when QB-MINI2 is not connected).

Figure 23-1. Connection Example of QB-MINI2 and R7F0C999B (2/4)

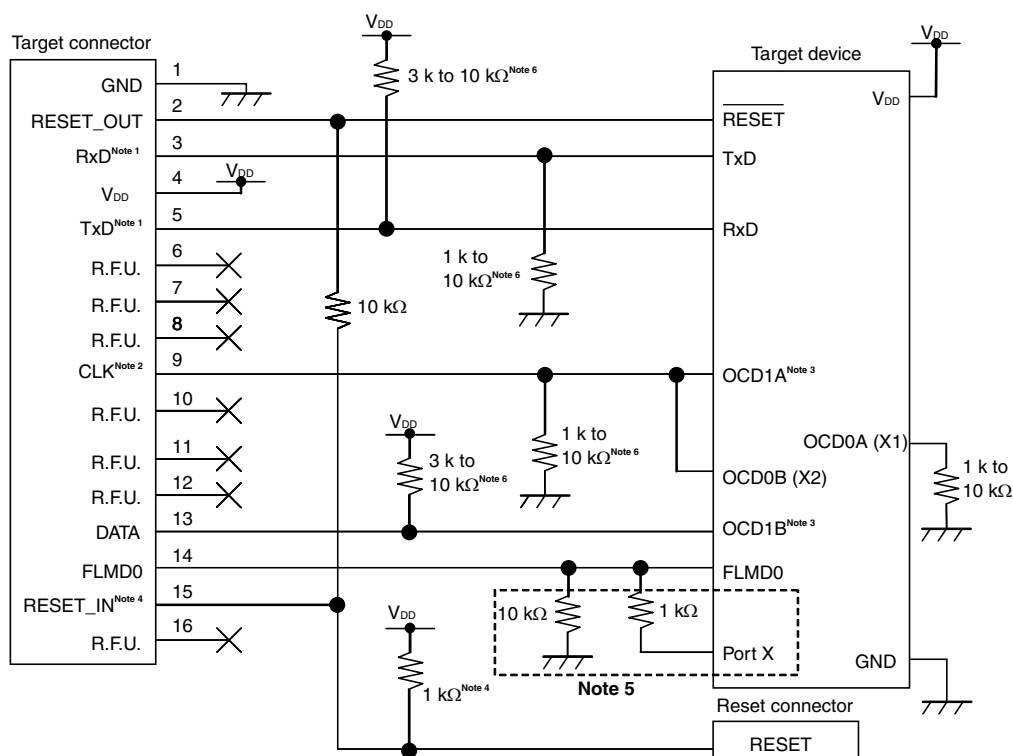
(2) When both debugging and programming are performed
(communicating by using OCD0A and OCD0B pins or X1 oscillator)



- Notes**
1. Connect TxD (transmit side) of the target device to RxD (receive side) of the target connector, and TxD (transmit side) of the target connector to RxD (receive side) of the target device.
 2. The X1 oscillator of the target system cannot be used to generate the clock signal for the target device during on-chip debugging.
Use the clock signal supplied from QB-MINI2. (The clock signal generated by an oscillator provided on the 78K0-OCD board or a 4, 8, or 16 MHz clock that can be selected as the system clock can also be used.)
 3. Short the jumper pins and disconnect the clock circuit during on-chip debugging.
Leave the jumper pins open and connect the clock circuit when the target device operates (when QB-MINI2 is not connected) during flash programming.
 4. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less).
 5. The circuit is designed for flash self programming, which controls the FLMD0 pin by user program.
Connect any port that can output data to FLMD0 via a resistor.
When not using flash self programming, process the pins according to the device specifications.
 6. This is the processing for the pins that are unused (the inputs are left open) when the target device operates (when QB-MINI2 is not connected).
 7. To use FLMD1, connect it to FLMD1 of QB-MINI2.

Figure 23-1. Connection Example of QB-MINI2 and R7F0C999B (3/4)

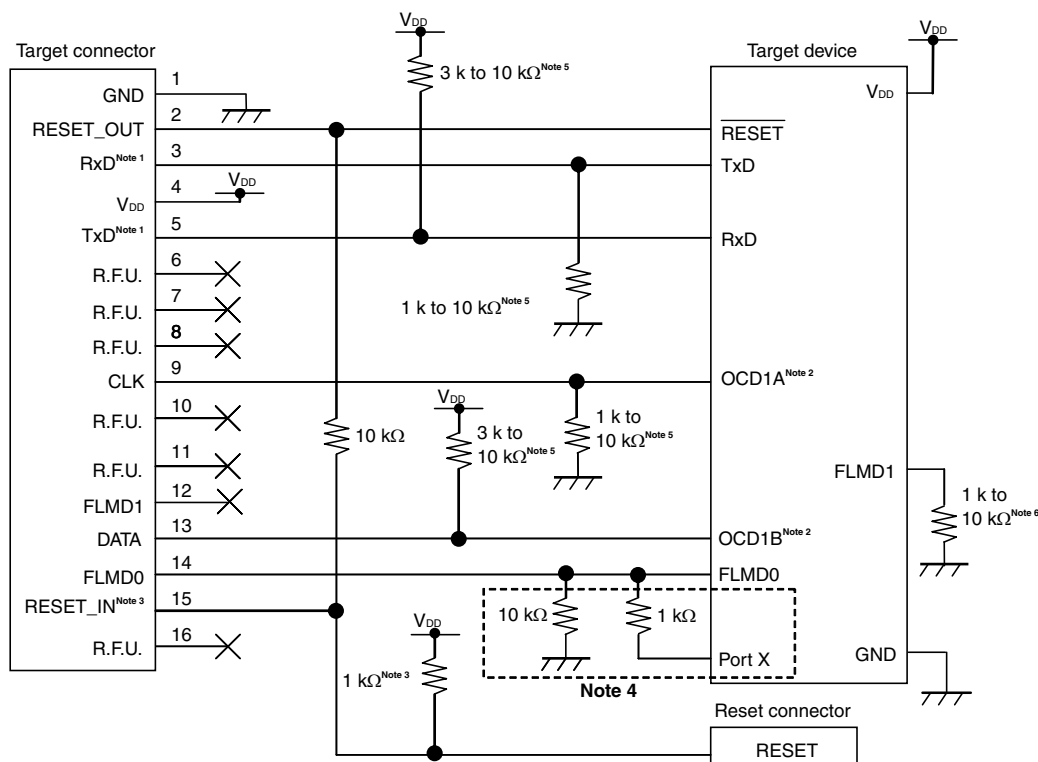
(3) When both debugging and programming are performed
(not communicating by using OCD1A and OCD1B pins or X1 oscillator)



- Notes**
1. Connect Tx D (transmit side) of the target device to Rx D (receive side) of the target connector, and Tx D (transmit side) of the target connector to Rx D (receive side) of the target device.
 2. Circuits other than the X1 oscillator can be used to generate the clock signal for the target device during on-chip debugging.
 3. During on-chip debugging, the settings specified by the user program are ignored, because these pins are used as pins dedicated to on-chip debugging. However, if the pins are specified as input pins, the pins must be processed (because they are left open when QB-MINI2 is not connected.)
 4. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less).
 5. The circuit is designed for flash self programming, which controls the FLMD0 pin by user program. When not using flash self programming, process the pins according to the device specifications.
 6. This is the processing for the pins that are unused (the inputs are left open) when the target device operates (when QB-MINI2 is not connected).

Figure 23-1. Connection Example of QB-MINI2 and R7F0C999B (4/4)

(4) When both debugging and programming are performed
(communicating by using OCD1A and OCD1B pins or X1 oscillator)



- Notes**
1. Connect TxD (transmit side) of the target device to RxD (receive side) of the target connector, and TxD (transmit side) of the target connector to RxD (receive side) of the target device.
 2. During on-chip debugging, the settings specified by the user program are ignored, because these pins are used as pins dedicated to on-chip debugging. However, if the pins are specified as input pins, the pins must be processed (because they are left open when QB-MINI2 is not connected.)
 3. This connection is designed assuming that the RESET signal is output from the N-ch open-drain buffer (output resistance: 100 Ω or less).
 4. The circuit is designed for flash self programming, which controls the FLMD0 pin by user program. Connect any port that can output data to FLMD0 via a resistor. When not using flash self programming, process the pins according to the device specifications.
 5. This is the processing for the pins that are unused (the inputs are left open) when the target device operates (when QB-MINI2 is not connected).
 6. To use FLMD1, connect it to FLMD1 of QB-MINI2.

23.2 On-Chip Debug Security ID

The R7F0C999B has an on-chip debug operation control bit in the flash memory at 0084H (refer to **CHAPTER 21 OPTION BYTE**) and an on-chip debug security ID setting area at 0085H to 008EH, to prevent third parties from reading memory content.

For details on the on-chip debug security ID, refer to the **QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual (R20UT0449E)**.

Table 23-1. On-Chip Debug Security ID

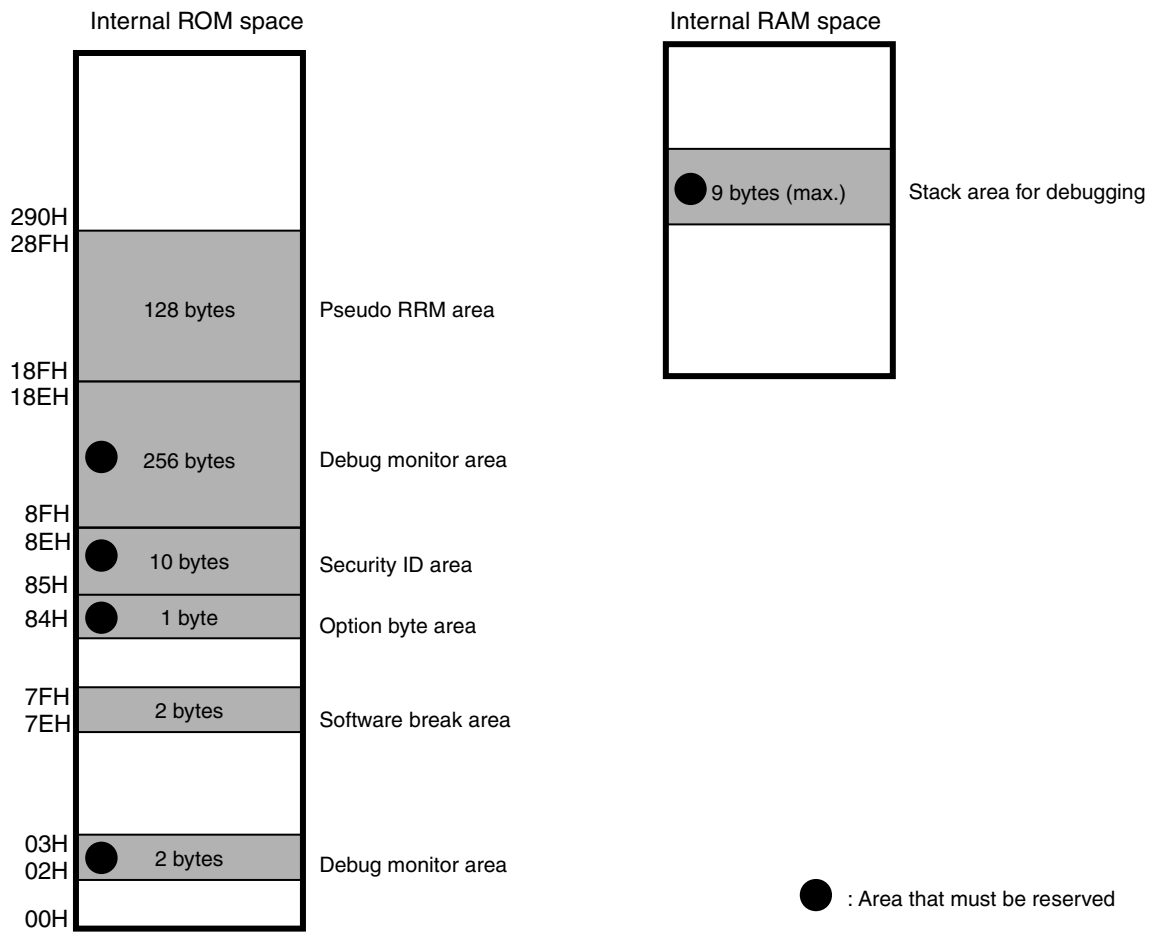
| Address | On-Chip Debug Security ID Code |
|----------------|--------------------------------|
| 0085H to 008EH | Any ID code of 10 bytes |
| 1085H to 108EH | |

23.3 Securing of User Resources

QB-MINI2 uses the user memory spaces (shaded portions in Figure 23-2) to implement communication with the target device, or each debug functions. The areas marked with a dot (•) are always used for debugging, and other areas are used for each debug function used.

These areas can be secured by using user programs or the linker option. For details on the securing of these areas, refer to the **QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual (R20UT0449E)**.

Figure 23-2. Reserved Area Used by QB-MINI2



CHAPTER 24 INSTRUCTION SET

This chapter lists each instruction set of the R7F0C011B, R7F0C012B, R7F0C013B, and R7F0C999B in table form. For details of each operation and operation code, refer to the separate document **78K/0 Series Instructions User's Manual (U12326E)**.

24.1 Conventions Used in Operation List

24.1.1 Operand identifiers and specification methods

Operands are written in the “Operand” column of each instruction in accordance with the specification method of the instruction operand identifier (refer to the assembler specifications for details). When there are two or more methods, select one of them. Uppercase letters and the symbols #, !, \$ and [] are keywords and must be written as they are. Each symbol has the following meaning.

- #: Immediate data specification
- !: Absolute address specification
- \$: Relative address specification
- []: Indirect address specification

In the case of immediate data, describe an appropriate numeric value or a label. When using a label, be sure to write the #, !, \$, and [] 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 specification.

Table 24-1. Operand Identifiers and Specification Methods

| Identifier | Specification Method |
|---------------|--|
| <i>r</i> | X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7) |
| <i>rp</i> | AX (RP0), BC (RP1), DE (RP2), HL (RP3) |
| <i>sfr</i> | Special function register symbol ^{Note} |
| <i>sfrp</i> | Special function register symbol (16-bit manipulatable register even addresses only) ^{Note} |
| <i>saddr</i> | FE20H to FF1FH Immediate data or labels |
| <i>saddrp</i> | FE20H to FF1FH Immediate data or labels (even address only) |
| <i>addr16</i> | 0000H to FFFFH Immediate data or labels (Only even addresses for 16-bit data transfer instructions) |
| <i>addr11</i> | 0800H to 0FFFH Immediate data or labels |
| <i>addr5</i> | 0040H to 007FH Immediate data or labels (even address only) |
| <i>word</i> | 16-bit immediate data or label |
| <i>byte</i> | 8-bit immediate data or label |
| <i>bit</i> | 3-bit immediate data or label |
| <i>RBn</i> | RB0 to RB3 |

Note Addresses from FFD0H to FFDFH cannot be accessed with these operands.

Remark For special function register symbols, see **Table 3-6 Special Function Register List**.

24.1.2 Description of operation column

| | |
|-----------------------------------|--|
| 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 |
| 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 _H , X _L : | Higher 8 bits and lower 8 bits of 16-bit register |
| ∧: | Logical product (AND) |
| ∨: | Logical sum (OR) |
| ⊕: | Exclusive logical sum (exclusive OR) |
| ⎯: | Inverted data |
| addr16: | 16-bit immediate data or label |
| dis8: | Signed 8-bit data (displacement value) |

24.1.3 Description of flag operation column

| | |
|----------|-------------------------------------|
| (Blank): | Not affected |
| 0: | Cleared to 0 |
| 1: | Set to 1 |
| x: | Set/cleared according to the result |
| R: | Previously saved value is restored |

24.2 Operation List

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | | |
|---------------------|-------------|----------------|--------|--------|--------|----------------|-----------------|----|----|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY | |
| 8-bit data transfer | MOV | r, #byte | 2 | 4 | – | r ← byte | | | | |
| | | saddr, #byte | 3 | 6 | 7 | (saddr) ← byte | | | | |
| | | sfr, #byte | 3 | – | 7 | sfr ← byte | | | | |
| | | A, r | Note 3 | 1 | 2 | – | A ← r | | | |
| | | r, A | Note 3 | 1 | 2 | – | r ← A | | | |
| | | A, saddr | | 2 | 4 | 5 | A ← (saddr) | | | |
| | | saddr, A | | 2 | 4 | 5 | (saddr) ← A | | | |
| | | A, sfr | | 2 | – | 5 | A ← sfr | | | |
| | | sfr, A | | 2 | – | 5 | sfr ← A | | | |
| | | A, !addr16 | | 3 | 8 | 9 | A ← (addr16) | | | |
| | | !addr16, A | | 3 | 8 | 9 | (addr16) ← A | | | |
| | | PSW, #byte | | 3 | – | 7 | PSW ← byte | x | x | x |
| | | A, PSW | | 2 | – | 5 | A ← PSW | | | |
| | | PSW, A | | 2 | – | 5 | PSW ← A | x | x | x |
| | | A, [DE] | | 1 | 4 | 5 | A ← (DE) | | | |
| | | [DE], A | | 1 | 4 | 5 | (DE) ← A | | | |
| | | A, [HL] | | 1 | 4 | 5 | A ← (HL) | | | |
| | | [HL], A | | 1 | 4 | 5 | (HL) ← A | | | |
| | | A, [HL + byte] | | 2 | 8 | 9 | A ← (HL + byte) | | | |
| | | [HL + byte], A | | 2 | 8 | 9 | (HL + byte) ← A | | | |
| | | A, [HL + B] | | 1 | 6 | 7 | A ← (HL + B) | | | |
| | | [HL + B], A | | 1 | 6 | 7 | (HL + B) ← A | | | |
| | | A, [HL + C] | | 1 | 6 | 7 | A ← (HL + C) | | | |
| | [HL + C], A | | 1 | 6 | 7 | (HL + C) ← A | | | | |
| | XCH | A, r | Note 3 | 1 | 2 | – | A ↔ r | | | |
| | | A, saddr | | 2 | 4 | 6 | A ↔ (saddr) | | | |
| | | A, sfr | | 2 | – | 6 | A ↔ (sfr) | | | |
| | | A, !addr16 | | 3 | 8 | 10 | A ↔ (addr16) | | | |
| | | A, [DE] | | 1 | 4 | 6 | A ↔ (DE) | | | |
| | | A, [HL] | | 1 | 4 | 6 | A ↔ (HL) | | | |
| | | A, [HL + byte] | | 2 | 8 | 10 | A ↔ (HL + byte) | | | |
| | | A, [HL + B] | | 2 | 8 | 10 | A ↔ (HL + B) | | | |
| A, [HL + C] | | | 2 | 8 | 10 | A ↔ (HL + C) | | | | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed
 3. Except “r = A”

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | | |
|----------------------|-------------|----------------|--------|--------|--------|-----------------------------------|------------------------------|----|----|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY | |
| 16-bit data transfer | MOVW | rp, #word | 3 | 6 | – | rp ← word | | | | |
| | | saddrp, #word | 4 | 8 | 10 | (saddrp) ← word | | | | |
| | | sfrp, #word | 4 | – | 10 | sfrp ← word | | | | |
| | | AX, saddrp | 2 | 6 | 8 | AX ← (saddrp) | | | | |
| | | saddrp, AX | 2 | 6 | 8 | (saddrp) ← AX | | | | |
| | | AX, sfrp | 2 | – | 8 | AX ← sfrp | | | | |
| | | sfrp, AX | 2 | – | 8 | sfrp ← AX | | | | |
| | | AX, rp | Note 3 | 1 | 4 | – | AX ← rp | | | |
| | | rp, AX | Note 3 | 1 | 4 | – | rp ← AX | | | |
| | | AX, !addr16 | | 3 | 10 | 12 | AX ← (addr16) | | | |
| | | !addr16, AX | | 3 | 10 | 12 | (addr16) ← AX | | | |
| | XCHW | AX, rp | Note 3 | 1 | 4 | – | AX ↔ rp | | | |
| 8-bit operation | ADD | A, #byte | 2 | 4 | – | A, CY ← A + byte | x | x | x | |
| | | saddr, #byte | 3 | 6 | 8 | (saddr), CY ← (saddr) + byte | x | x | x | |
| | | A, r | Note 4 | 2 | 4 | – | A, CY ← A + r | x | x | x |
| | | r, A | | 2 | 4 | – | r, CY ← r + A | x | x | x |
| | | A, saddr | | 2 | 4 | 5 | A, CY ← A + (saddr) | x | x | x |
| | | A, !addr16 | | 3 | 8 | 9 | A, CY ← A + (addr16) | x | x | x |
| | | A, [HL] | | 1 | 4 | 5 | A, CY ← A + (HL) | x | x | x |
| | | A, [HL + byte] | | 2 | 8 | 9 | A, CY ← A + (HL + byte) | x | x | x |
| | | A, [HL + B] | | 2 | 8 | 9 | A, CY ← A + (HL + B) | x | x | x |
| | A, [HL + C] | | 2 | 8 | 9 | A, CY ← A + (HL + C) | x | x | x | |
| | ADDC | A, #byte | 2 | 4 | – | A, CY ← A + byte + CY | x | x | x | |
| | | saddr, #byte | 3 | 6 | 8 | (saddr), CY ← (saddr) + byte + CY | x | x | x | |
| | | A, r | Note 4 | 2 | 4 | – | A, CY ← A + r + CY | x | x | x |
| | | r, A | | 2 | 4 | – | r, CY ← r + A + CY | x | x | x |
| | | A, saddr | | 2 | 4 | 5 | A, CY ← A + (saddr) + CY | x | x | x |
| | | A, !addr16 | | 3 | 8 | 9 | A, CY ← A + (addr16) + C | x | x | x |
| | | A, [HL] | | 1 | 4 | 5 | A, CY ← A + (HL) + CY | x | x | x |
| | | A, [HL + byte] | | 2 | 8 | 9 | A, CY ← A + (HL + byte) + CY | x | x | x |
| | | A, [HL + B] | | 2 | 8 | 9 | A, CY ← A + (HL + B) + CY | x | x | x |
| A, [HL + C] | | | 2 | 8 | 9 | A, CY ← A + (HL + C) + CY | x | x | x | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed
 3. Only when rp = BC, DE or HL
 4. Except "r = A"

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{cpu}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|-------------------|-------------|----------------------------|-------|--------|---------------------------|-----------------------------------|------|----|----|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit operation | SUB | A, #byte | 2 | 4 | – | A, CY ← A – byte | × | × | × |
| | | saddr, #byte | 3 | 6 | 8 | (saddr), CY ← (saddr) – byte | × | × | × |
| | | A, r <small>Note 3</small> | 2 | 4 | – | A, CY ← A – r | × | × | × |
| | | r, A | 2 | 4 | – | r, CY ← r – A | × | × | × |
| | | A, saddr | 2 | 4 | 5 | A, CY ← A – (saddr) | × | × | × |
| | | A, !addr16 | 3 | 8 | 9 | A, CY ← A – (addr16) | × | × | × |
| | | A, [HL] | 1 | 4 | 5 | A, CY ← A – (HL) | × | × | × |
| | | A, [HL + byte] | 2 | 8 | 9 | A, CY ← A – (HL + byte) | × | × | × |
| | | A, [HL + B] | 2 | 8 | 9 | A, CY ← A – (HL + B) | × | × | × |
| | A, [HL + C] | 2 | 8 | 9 | A, CY ← A – (HL + C) | × | × | × | |
| | SUBC | A, #byte | 2 | 4 | – | A, CY ← A – byte – CY | × | × | × |
| | | saddr, #byte | 3 | 6 | 8 | (saddr), CY ← (saddr) – byte – CY | × | × | × |
| | | A, r <small>Note 3</small> | 2 | 4 | – | A, CY ← A – r – CY | × | × | × |
| | | r, A | 2 | 4 | – | r, CY ← r – A – CY | × | × | × |
| | | A, saddr | 2 | 4 | 5 | A, CY ← A – (saddr) – CY | × | × | × |
| | | A, !addr16 | 3 | 8 | 9 | A, CY ← A – (addr16) – CY | × | × | × |
| | | A, [HL] | 1 | 4 | 5 | A, CY ← A – (HL) – CY | × | × | × |
| | | A, [HL + byte] | 2 | 8 | 9 | A, CY ← A – (HL + byte) – CY | × | × | × |
| | | A, [HL + B] | 2 | 8 | 9 | A, CY ← A – (HL + B) – CY | × | × | × |
| | A, [HL + C] | 2 | 8 | 9 | A, CY ← A – (HL + C) – CY | × | × | × | |
| | AND | A, #byte | 2 | 4 | – | A ← A ∧ byte | × | | |
| | | saddr, #byte | 3 | 6 | 8 | (saddr) ← (saddr) ∧ byte | × | | |
| | | A, r <small>Note 3</small> | 2 | 4 | – | A ← A ∧ r | × | | |
| | | r, A | 2 | 4 | – | r ← r ∧ A | × | | |
| | | A, saddr | 2 | 4 | 5 | A ← A ∧ (saddr) | × | | |
| | | A, !addr16 | 3 | 8 | 9 | A ← A ∧ (addr16) | × | | |
| | | A, [HL] | 1 | 4 | 5 | A ← A ∧ (HL) | × | | |
| | | A, [HL + byte] | 2 | 8 | 9 | A ← A ∧ (HL + byte) | × | | |
| | | A, [HL + B] | 2 | 8 | 9 | A ← A ∧ (HL + B) | × | | |
| | A, [HL + C] | 2 | 8 | 9 | A ← A ∧ (HL + C) | × | | | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed
 3. Except “r = A”

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{cpu}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|-------------------|------------|----------------------------|-------|--------|--------|---|------|----|----|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 8-bit operation | OR | A, #byte | 2 | 4 | – | $A \leftarrow A \vee \text{byte}$ | | x | |
| | | saddr, #byte | 3 | 6 | 8 | $(\text{saddr}) \leftarrow (\text{saddr}) \vee \text{byte}$ | | x | |
| | | A, r <small>Note 3</small> | 2 | 4 | – | $A \leftarrow A \vee r$ | | x | |
| | | r, A | 2 | 4 | – | $r \leftarrow r \vee A$ | | x | |
| | | A, saddr | 2 | 4 | 5 | $A \leftarrow A \vee (\text{saddr})$ | | x | |
| | | A, !addr16 | 3 | 8 | 9 | $A \leftarrow A \vee (\text{addr16})$ | | x | |
| | | A, [HL] | 1 | 4 | 5 | $A \leftarrow A \vee (\text{HL})$ | | x | |
| | | A, [HL + byte] | 2 | 8 | 9 | $A \leftarrow A \vee (\text{HL} + \text{byte})$ | | x | |
| | | A, [HL + B] | 2 | 8 | 9 | $A \leftarrow A \vee (\text{HL} + B)$ | | x | |
| | | A, [HL + C] | 2 | 8 | 9 | $A \leftarrow A \vee (\text{HL} + C)$ | | x | |
| | XOR | A, #byte | 2 | 4 | – | $A \leftarrow A \nabla \text{byte}$ | | x | |
| | | saddr, #byte | 3 | 6 | 8 | $(\text{saddr}) \leftarrow (\text{saddr}) \nabla \text{byte}$ | | x | |
| | | A, r <small>Note 3</small> | 2 | 4 | – | $A \leftarrow A \nabla r$ | | x | |
| | | r, A | 2 | 4 | – | $r \leftarrow r \nabla A$ | | x | |
| | | A, saddr | 2 | 4 | 5 | $A \leftarrow A \nabla (\text{saddr})$ | | x | |
| | | A, !addr16 | 3 | 8 | 9 | $A \leftarrow A \nabla (\text{addr16})$ | | x | |
| | | A, [HL] | 1 | 4 | 5 | $A \leftarrow A \nabla (\text{HL})$ | | x | |
| | | A, [HL + byte] | 2 | 8 | 9 | $A \leftarrow A \nabla (\text{HL} + \text{byte})$ | | x | |
| | | A, [HL + B] | 2 | 8 | 9 | $A \leftarrow A \nabla (\text{HL} + B)$ | | x | |
| | | A, [HL + C] | 2 | 8 | 9 | $A \leftarrow A \nabla (\text{HL} + C)$ | | x | |
| | CMP | A, #byte | 2 | 4 | – | $A - \text{byte}$ | x | x | x |
| | | saddr, #byte | 3 | 6 | 8 | $(\text{saddr}) - \text{byte}$ | x | x | x |
| | | A, r <small>Note 3</small> | 2 | 4 | – | $A - r$ | x | x | x |
| | | r, A | 2 | 4 | – | $r - A$ | x | x | x |
| | | A, saddr | 2 | 4 | 5 | $A - (\text{saddr})$ | x | x | x |
| | | A, !addr16 | 3 | 8 | 9 | $A - (\text{addr16})$ | x | x | x |
| | | A, [HL] | 1 | 4 | 5 | $A - (\text{HL})$ | x | x | x |
| | | A, [HL + byte] | 2 | 8 | 9 | $A - (\text{HL} + \text{byte})$ | x | x | x |
| | | A, [HL + B] | 2 | 8 | 9 | $A - (\text{HL} + B)$ | x | x | x |
| | | A, [HL + C] | 2 | 8 | 9 | $A - (\text{HL} + C)$ | x | x | x |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed
 3. Except “r = A”

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|---------------------|--------------|---------------|-------|---------------|--------|--|------|----|----|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| 16-bit operation | ADDW | AX, #word | 3 | 6 | – | AX, CY ← AX + word | × | × | × |
| | SUBW | AX, #word | 3 | 6 | – | AX, CY ← AX – word | × | × | × |
| | CMPW | AX, #word | 3 | 6 | – | AX – word | × | × | × |
| Multiply/divide | MULU | X | 2 | 16 | – | AX ← A × X | | | |
| | DIVUW | C | 2 | 25 | – | AX (Quotient), C (Remainder) ← AX ÷ C | | | |
| Increment/decrement | INC | r | 1 | 2 | – | r ← r + 1 | × | × | |
| | | saddr | 2 | 4 | 6 | (saddr) ← (saddr) + 1 | × | × | |
| | DEC | r | 1 | 2 | – | r ← r – 1 | × | × | |
| | | saddr | 2 | 4 | 6 | (saddr) ← (saddr) – 1 | × | × | |
| | INCW | rp | 1 | 4 | – | rp ← rp + 1 | | | |
| | DECW | rp | 1 | 4 | – | rp ← rp – 1 | | | |
| Rotate | ROR | A, 1 | 1 | 2 | – | (CY, A ₇ ← A ₀ , A _{m-1} ← A _m) × 1 time | | | × |
| | ROL | A, 1 | 1 | 2 | – | (CY, A ₀ ← A ₇ , A _{m+1} ← A _m) × 1 time | | | × |
| | RORC | A, 1 | 1 | 2 | – | (CY ← A ₀ , A ₇ ← CY, A _{m-1} ← A _m) × 1 time | | | × |
| | ROLC | A, 1 | 1 | 2 | – | (CY ← A ₇ , A ₀ ← CY, A _{m+1} ← A _m) × 1 time | | | × |
| | ROR4 | [HL] | 2 | 10 | 12 | A ₃₋₀ ← (HL) ₃₋₀ , (HL) ₇₋₄ ← A ₃₋₀ , (HL) ₃₋₀ ← (HL) ₇₋₄ | | | |
| | ROL4 | [HL] | 2 | 10 | 12 | A ₃₋₀ ← (HL) ₇₋₄ , (HL) ₃₋₀ ← A ₃₋₀ , (HL) ₇₋₄ ← (HL) ₃₋₀ | | | |
| BCD adjustment | ADJBA | | 2 | 4 | – | Decimal Adjust Accumulator after Addition | × | × | × |
| | ADJBS | | 2 | 4 | – | Decimal Adjust Accumulator after Subtract | × | × | × |
| Bit manipulate | MOV1 | CY, saddr.bit | 3 | 6 | 7 | CY ← (saddr.bit) | | | × |
| | | CY, sfr.bit | 3 | – | 7 | CY ← sfr.bit | | | × |
| | | CY, A.bit | 2 | 4 | – | CY ← A.bit | | | × |
| | | CY, PSW.bit | 3 | – | 7 | CY ← PSW.bit | | | × |
| | | CY, [HL].bit | 2 | 6 | 7 | CY ← (HL).bit | | | × |
| | | saddr.bit, CY | 3 | 6 | 8 | (saddr.bit) ← CY | | | |
| | | sfr.bit, CY | 3 | – | 8 | sfr.bit ← CY | | | |
| | | A.bit, CY | 2 | 4 | – | A.bit ← CY | | | |
| | | PSW.bit, CY | 3 | – | 8 | PSW.bit ← CY | | | × |
| [HL].bit, CY | 2 | 6 | 8 | (HL).bit ← CY | | | | | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{cpu}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|-------------------|-------------|---------------|-------|--------|--------|--|------|----|----|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Bit manipulate | AND1 | CY, saddr.bit | 3 | 6 | 7 | $CY \leftarrow CY \wedge (\text{saddr.bit})$ | | | x |
| | | CY, sfr.bit | 3 | – | 7 | $CY \leftarrow CY \wedge \text{sfr.bit}$ | | | x |
| | | CY, A.bit | 2 | 4 | – | $CY \leftarrow CY \wedge A.\text{bit}$ | | | x |
| | | CY, PSW.bit | 3 | – | 7 | $CY \leftarrow CY \wedge \text{PSW.bit}$ | | | x |
| | | CY, [HL].bit | 2 | 6 | 7 | $CY \leftarrow CY \wedge (\text{HL}).\text{bit}$ | | | x |
| | OR1 | CY, saddr.bit | 3 | 6 | 7 | $CY \leftarrow CY \vee (\text{saddr.bit})$ | | | x |
| | | CY, sfr.bit | 3 | – | 7 | $CY \leftarrow CY \vee \text{sfr.bit}$ | | | x |
| | | CY, A.bit | 2 | 4 | – | $CY \leftarrow CY \vee A.\text{bit}$ | | | x |
| | | CY, PSW.bit | 3 | – | 7 | $CY \leftarrow CY \vee \text{PSW.bit}$ | | | x |
| | | CY, [HL].bit | 2 | 6 | 7 | $CY \leftarrow CY \vee (\text{HL}).\text{bit}$ | | | x |
| | XOR1 | CY, saddr.bit | 3 | 6 | 7 | $CY \leftarrow CY \oplus (\text{saddr.bit})$ | | | x |
| | | CY, sfr.bit | 3 | – | 7 | $CY \leftarrow CY \oplus \text{sfr.bit}$ | | | x |
| | | CY, A.bit | 2 | 4 | – | $CY \leftarrow CY \oplus A.\text{bit}$ | | | x |
| | | CY, PSW.bit | 3 | – | 7 | $CY \leftarrow CY \oplus \text{PSW.bit}$ | | | x |
| | | CY, [HL].bit | 2 | 6 | 7 | $CY \leftarrow CY \oplus (\text{HL}).\text{bit}$ | | | x |
| | SET1 | saddr.bit | 2 | 4 | 6 | $(\text{saddr.bit}) \leftarrow 1$ | | | |
| | | sfr.bit | 3 | – | 8 | $\text{sfr.bit} \leftarrow 1$ | | | |
| | | A.bit | 2 | 4 | – | $A.\text{bit} \leftarrow 1$ | | | |
| | | PSW.bit | 2 | – | 6 | $\text{PSW.bit} \leftarrow 1$ | x | x | x |
| | | [HL].bit | 2 | 6 | 8 | $(\text{HL}).\text{bit} \leftarrow 1$ | | | |
| | CLR1 | saddr.bit | 2 | 4 | 6 | $(\text{saddr.bit}) \leftarrow 0$ | | | |
| | | sfr.bit | 3 | – | 8 | $\text{sfr.bit} \leftarrow 0$ | | | |
| | | A.bit | 2 | 4 | – | $A.\text{bit} \leftarrow 0$ | | | |
| | | PSW.bit | 2 | – | 6 | $\text{PSW.bit} \leftarrow 0$ | x | x | x |
| | | [HL].bit | 2 | 6 | 8 | $(\text{HL}).\text{bit} \leftarrow 0$ | | | |
| | SET1 | CY | 1 | 2 | – | $CY \leftarrow 1$ | | | 1 |
| | CLR1 | CY | 1 | 2 | – | $CY \leftarrow 0$ | | | 0 |
| | NOT1 | CY | 1 | 2 | – | $CY \leftarrow \overline{CY}$ | | | x |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|----------------------|--------------|-----------|-------|--------|----------------------------------|---|------|----|----|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Call/return | CALL | !addr16 | 3 | 7 | – | $(SP - 1) \leftarrow (PC + 3)_H, (SP - 2) \leftarrow (PC + 3)_L,$ $PC \leftarrow \text{addr16}, SP \leftarrow SP - 2$ | | | |
| | CALLF | !addr11 | 2 | 5 | – | $(SP - 1) \leftarrow (PC + 2)_H, (SP - 2) \leftarrow (PC + 2)_L,$ $PC_{15-11} \leftarrow 00001, PC_{10-0} \leftarrow \text{addr11},$ $SP \leftarrow SP - 2$ | | | |
| | CALLT | [addr5] | 1 | 6 | – | $(SP - 1) \leftarrow (PC + 1)_H, (SP - 2) \leftarrow (PC + 1)_L,$ $PC_H \leftarrow (\text{addr5} + 1), PC_L \leftarrow (\text{addr5}),$ $SP \leftarrow SP - 2$ | | | |
| | BRK | | 1 | 6 | – | $(SP - 1) \leftarrow \text{PSW}, (SP - 2) \leftarrow (PC + 1)_H,$ $(SP - 3) \leftarrow (PC + 1)_L, PC_H \leftarrow (003FH),$ $PC_L \leftarrow (003EH), SP \leftarrow SP - 3, IE \leftarrow 0$ | | | |
| | RET | | 1 | 6 | – | $PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $SP \leftarrow SP + 2$ | | | |
| | RETI | | 1 | 6 | – | $PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $PSW \leftarrow (SP + 2), SP \leftarrow SP + 3$ | R | R | R |
| | RETB | | 1 | 6 | – | $PC_H \leftarrow (SP + 1), PC_L \leftarrow (SP),$ $PSW \leftarrow (SP + 2), SP \leftarrow SP + 3$ | R | R | R |
| Stack manipulate | PUSH | PSW | 1 | 2 | – | $(SP - 1) \leftarrow \text{PSW}, SP \leftarrow SP - 1$ | | | |
| | | rp | 1 | 4 | – | $(SP - 1) \leftarrow \text{rp}_H, (SP - 2) \leftarrow \text{rp}_L,$ $SP \leftarrow SP - 2$ | | | |
| | POP | PSW | 1 | 2 | – | $\text{PSW} \leftarrow (SP), SP \leftarrow SP + 1$ | R | R | R |
| | | rp | 1 | 4 | – | $\text{rp}_H \leftarrow (SP + 1), \text{rp}_L \leftarrow (SP),$ $SP \leftarrow SP + 2$ | | | |
| | MOVW | SP, #word | 4 | – | 10 | $SP \leftarrow \text{word}$ | | | |
| | | SP, AX | 2 | – | 8 | $SP \leftarrow \text{AX}$ | | | |
| AX, SP | | 2 | – | 8 | $\text{AX} \leftarrow \text{SP}$ | | | | |
| Unconditional branch | BR | !addr16 | 3 | 6 | – | $PC \leftarrow \text{addr16}$ | | | |
| | | \$addr16 | 2 | 6 | – | $PC \leftarrow PC + 2 + \text{jdisp8}$ | | | |
| | | AX | 2 | 8 | – | $PC_H \leftarrow A, PC_L \leftarrow X$ | | | |
| Conditional branch | BC | \$addr16 | 2 | 6 | – | $PC \leftarrow PC + 2 + \text{jdisp8}$ if CY = 1 | | | |
| | BNC | \$addr16 | 2 | 6 | – | $PC \leftarrow PC + 2 + \text{jdisp8}$ if CY = 0 | | | |
| | BZ | \$addr16 | 2 | 6 | – | $PC \leftarrow PC + 2 + \text{jdisp8}$ if Z = 1 | | | |
| | BNZ | \$addr16 | 2 | 6 | – | $PC \leftarrow PC + 2 + \text{jdisp8}$ if Z = 0 | | | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag |
|--------------------|--------------|---------------------|-------|--------|--------|--|---------|
| | | | | Note 1 | Note 2 | | Z AC CY |
| Conditional branch | BT | saddr.bit, \$addr16 | 3 | 8 | 9 | PC ← PC + 3 + jdisp8 if (saddr.bit) = 1 | |
| | | sfr.bit, \$addr16 | 4 | – | 11 | PC ← PC + 4 + jdisp8 if sfr.bit = 1 | |
| | | A.bit, \$addr16 | 3 | 8 | – | PC ← PC + 3 + jdisp8 if A.bit = 1 | |
| | | PSW.bit, \$addr16 | 3 | – | 9 | PC ← PC + 3 + jdisp8 if PSW.bit = 1 | |
| | | [HL].bit, \$addr16 | 3 | 10 | 11 | PC ← PC + 3 + jdisp8 if (HL).bit = 1 | |
| | BF | saddr.bit, \$addr16 | 4 | 10 | 11 | PC ← PC + 4 + jdisp8 if (saddr.bit) = 0 | |
| | | sfr.bit, \$addr16 | 4 | – | 11 | PC ← PC + 4 + jdisp8 if sfr.bit = 0 | |
| | | A.bit, \$addr16 | 3 | 8 | – | PC ← PC + 3 + jdisp8 if A.bit = 0 | |
| | | PSW.bit, \$addr16 | 4 | – | 11 | PC ← PC + 4 + jdisp8 if PSW.bit = 0 | |
| | | [HL].bit, \$addr16 | 3 | 10 | 11 | PC ← PC + 3 + jdisp8 if (HL).bit = 0 | |
| | BTCLR | saddr.bit, \$addr16 | 4 | 10 | 12 | PC ← PC + 4 + jdisp8 if (saddr.bit) = 1 then reset (saddr.bit) | |
| | | sfr.bit, \$addr16 | 4 | – | 12 | PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit | |
| | | A.bit, \$addr16 | 3 | 8 | – | PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit | |
| | | PSW.bit, \$addr16 | 4 | – | 12 | PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit | × × × |
| | | [HL].bit, \$addr16 | 3 | 10 | 12 | PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit | |
| | DBNZ | B, \$addr16 | 2 | 6 | – | B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0 | |
| | | C, \$addr16 | 2 | 6 | – | C ← C – 1, then PC ← PC + 2 + jdisp8 if C ≠ 0 | |
| | | saddr, \$addr16 | 3 | 8 | 10 | (saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0 | |
| CPU control | SEL | RBn | 2 | 4 | – | RBS1, 0 ← n | |
| | NOP | | 1 | 2 | – | No Operation | |
| | EI | | 2 | – | 6 | IE ← 1 (Enable Interrupt) | |
| | DI | | 2 | – | 6 | IE ← 0 (Disable Interrupt) | |
| | HALT | | 2 | 6 | – | Set HALT Mode | |
| | STOP | | 2 | 6 | – | Set STOP Mode | |

- Notes**
1. When the internal high-speed RAM area is accessed or for an instruction with no data access
 2. When an area except the internal high-speed RAM area is accessed

- Remarks**
1. One instruction clock cycle is one cycle of the CPU clock (f_{CPU}) selected by the processor clock control register (PCC).
 2. This clock cycle applies to the internal ROM program.

24.3 Instructions Listed by Addressing Type

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

| Second Operand First Operand | #byte | A | r ^{Note} | sfr | saddr | !addr16 | PSW | [DE] | [HL] | [HL + byte] [HL + B] [HL + C] | \$addr16 | 1 | None |
|-------------------------------------|--|--|---|------------|---|------------|-----|------------|---|---|----------|----------------------------|--------------|
| A | ADD ADDC SUB SUBC AND OR XOR CMP | | MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP | MOV XCH | MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP | MOV XCH | MOV | MOV XCH | MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP | MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP | | ROR ROL RORC ROL4 | |
| r | MOV | MOV ADD ADDC SUB SUBC AND OR XOR CMP | | | | | | | | | | | INC DEC |
| B, C | | | | | | | | | | | DBNZ | | |
| sfr | MOV | MOV | | | | | | | | | | | |
| saddr | MOV ADD ADDC SUB SUBC AND OR XOR CMP | MOV | | | | | | | | | DBNZ | | INC DEC |
| !addr16 | | MOV | | | | | | | | | | | |
| PSW | MOV | MOV | | | | | | | | | | | PUSH POP |
| [DE] | | MOV | | | | | | | | | | | |
| [HL] | | MOV | | | | | | | | | | | ROR4 ROL4 |
| [HL + byte] [HL + B] [HL + C] | | MOV | | | | | | | | | | | |
| X | | | | | | | | | | | | | MULU |
| C | | | | | | | | | | | | | DIVUW |

Note Except "r = A"

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

| Second Operand First Operand | #word | AX | rp ^{Note} | sfrp | saddrp | !addr16 | SP | None |
|---------------------------------|----------------------|----------------------|--------------------|------|--------|---------|------|-----------------------------|
| AX | ADDW SUBW CMPW | | MOVW XCHW | MOVW | MOVW | MOVW | MOVW | |
| rp | MOVW | MOVW ^{Note} | | | | | | INCW DECW PUSH POP |
| sfrp | MOVW | MOVW | | | | | | |
| saddrp | MOVW | MOVW | | | | | | |
| !addr16 | | MOVW | | | | | | |
| SP | MOVW | MOVW | | | | | | |

Note Only when rp = BC, DE, HL**(3) Bit manipulation instructions**

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

| Second Operand First Operand | A.bit | sfr.bit | saddr.bit | PSW.bit | [HL].bit | CY | \$addr16 | None |
|---------------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|-----------------------------|------|-------------------|----------------------|
| A.bit | | | | | | MOV1 | BT BF BTCLR | SET1 CLR1 |
| sfr.bit | | | | | | MOV1 | BT BF BTCLR | SET1 CLR1 |
| saddr.bit | | | | | | MOV1 | BT BF BTCLR | SET1 CLR1 |
| PSW.bit | | | | | | MOV1 | BT BF BTCLR | SET1 CLR1 |
| [HL].bit | | | | | | MOV1 | BT BF BTCLR | SET1 CLR1 |
| CY | MOV1 AND1 OR1 XOR1 | MOV1 AND1 OR1 XOR1 | MOV1 AND1 OR1 XOR1 | MOV1 AND1 OR1 XOR1 | MOV1 AND1 OR1 XOR1 | | | SET1 CLR1 NOT1 |

(4) Call instructions/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

| Second Operand First Operand | AX | !addr16 | !addr11 | [addr5] | \$addr16 |
|---------------------------------|----|------------|---------|---------|------------------------------|
| Basic instruction | BR | CALL BR | CALLF | CALLT | BR BC BNC BZ BNZ |
| Compound instruction | | | | | BT BF BTCLR DBNZ |

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

CHAPTER 25 ELECTRICAL SPECIFICATIONS (TARGET)

Caution These specifications show target values, which may change after device evaluation.

Absolute Maximum Ratings (T_A = 25°C) (1/2)

| Parameter | Symbol | Conditions | Ratings | Unit |
|------------------------|-------------------------------|--|---|------|
| Supply voltage | V _{DD} | | -0.5 to +6.5 | V |
| | V _{SS} | | -0.5 to +0.3 | V |
| REGC pin input voltage | V _{I_{REGC}} | | -0.5 to + 3.6 and -0.5 to V _{DD} | V |
| Input voltage | V _{I1} | P00, P01, P10 to P17, P20 to P23, P30 to P33, P40, P41, P70, P71, P120 to P122, X1, X2, RESET, FLMD0 | -0.3 to V _{DD} + 0.3 ^{Note} | V |
| | V _{I2} | P60, P61 (N-ch open drain) | -0.3 to +6.5 | V |
| Output voltage | V _O | | -0.3 to V _{DD} + 0.3 ^{Note} | V |
| Analog input voltage | V _{AN} | ANI0 to ANI3 | -0.3 to V _{DD} + 0.3 ^{Note} | V |

Note Must be 6.5 V or lower.

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 port pins.

Absolute Maximum Ratings (T_A = 25°C) (2/2)

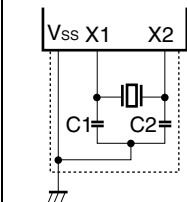
| Parameter | Symbol | Conditions | | Ratings | Unit |
|-------------------------------|------------------|----------------------------------|---|-------------|------|
| Output current, high | I _{OH1} | Per pin | P00, P01, P10 to P17, P30 to P33, P40, P41, P70, P71, P120 | -10 | mA |
| | | Total of all pins -80 mA | P00, P01, P40, P41, P120 | -25 | mA |
| | | | P10 to P17, P30 to P33, P70, P71 | -55 | mA |
| | I _{OH2} | Per pin | P20 to P23 | -0.5 | mA |
| | | Total of all pins | | -2 | mA |
| | I _{OH3} | Per pin | P121, P122 | -1 | mA |
| | | Total of all pins | | -2 | mA |
| Output current, low | I _{OL1} | Per pin | P00, P01, P10 to P17, P30 to P33, P40, P41, P60, P61, P70, P71, P120 | 30 | mA |
| | | Total of all pins 200 mA | P00, P01, P40, P41, P120 | 60 | mA |
| | | | P10 to P17, P30 to P33, P60, P61, P70, P71 | 140 | mA |
| | I _{OL2} | Per pin | P20 to P23 | 1 | mA |
| | | Total of all pins | | 4 | mA |
| | I _{OL3} | Per pin | P121, P122 | 4 | mA |
| | | Total of all pins | | 8 | mA |
| Operating ambient temperature | T _A | In normal operating mode | | -40 to +85 | °C |
| | | In flash memory programming mode | | | |
| Storage temperature | T _{stg} | | | -65 to +150 | °C |

Cautions 1. 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.

- 2.** The value of the current that can be run per pin must satisfy the value of the current per pin and the total value of the currents of all pins.

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

X1 Oscillator Characteristics**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Resonator | Recommended Circuit | Parameter | Conditions | MIN. | TYP. | MAX. | Unit |
|---|---|--|------------|-----------------------|------|------|------|
| Ceramic resonator, Crystal resonator |  | X1 clock oscillation frequency (f_x) ^{Note 1} | | 1.0 ^{Note 2} | | 10.0 | MHz |

Notes 1. Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.**2.** It is 2.0 MHz (MIN.) when programming on the board via UART6.**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 internal high-speed oscillation 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 oscillation stabilization time select register (OSTS) after sufficiently evaluating the oscillation stabilization time with the resonator to be used.

Internal Oscillator Characteristics**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Resonator | Parameter | Conditions | MIN. | TYP. | MAX. | Unit |
|-----------------------------|--|------------|------|------|------|------|
| 8 MHz internal oscillator | Internal high-speed oscillation clock frequency (f_{RH}) ^{Note} | RSTS = 1 | 7.6 | 8.0 | 8.4 | MHz |
| | | RSTS = 0 | 2.48 | 5.6 | 9.86 | MHz |
| 240 kHz internal oscillator | Internal low-speed oscillation clock frequency (f_{RL}) | | 216 | 240 | 264 | kHz |

Note Indicates only oscillator characteristics. Refer to **AC Characteristics** for instruction execution time.**Remark** RSTS: Bit 7 of the internal oscillation mode register (RCM)

DC Characteristics (1/5)**(T_A = -40 to +85°C, 4.0 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|--|------------------|--|------|------|-------|------|
| Output current, high ^{Note 1} | I _{OH1} | Per pin for P00, P01, P10 to P17, P30 to P33, P40, P41, P70, P71, P120 | | | -3.0 | mA |
| | | Total of P00, P01, P40, P41, P120 ^{Note 3} | | | -20.0 | mA |
| | | Total of P10 to P17, P30 to P33, P70, P71 ^{Note 3} | | | -30.0 | mA |
| | | Total of all the pins above ^{Note 3} | | | -50.0 | mA |
| | I _{OH2} | Per pin for P20 to P23 | | | -0.1 | mA |
| | | Per pin for P121, P122 | | | -0.1 | mA |
| Output current, low ^{Note 2} | I _{OL1} | Per pin for P00, P01, P10 to P17, P30 to P33, P40, P41, P70, P71, P120 | | | 8.5 | mA |
| | | Per pin for P60, P61 | | | 15.0 | mA |
| | | Total of P00, P01, P40, P41, P120 ^{Note 3} | | | 20.0 | mA |
| | | Total of P10 to P17, P30 to P33, P60, P61, P70, P71 ^{Note 3} | | | 45.0 | mA |
| | | Total of all the pins above ^{Note 3} | | | 65.0 | mA |
| | I _{OL2} | Per pin for P20 to P23 | | | 0.4 | mA |
| | | Per pin for P121, P122 | | | 0.4 | mA |

- Notes**
- Value of current at which the device operation is guaranteed even if the current flows from V_{DD} to an output pin.
 - Value of current at which the device operation is guaranteed even if the current flows from an output pin to GND.
 - Specification under conditions where the duty factor is 70% (time for which current is output is 0.7 × t and time for which current is not output is 0.3 × t, where t is a specific time). The total output current of the pins at a duty factor of other than 70% can be calculated by the following expression.
 - Where the duty factor of I_{OH} is n%: Total output current of pins = (I_{OH} × 0.7)/(n × 0.01)

<Example> Where the duty factor is 50%, I_{OH} = -20.0 mA

$$\text{Total output current of pins} = (-20.0 \times 0.7)/(50 \times 0.01) = -28.0 \text{ mA}$$

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 port pins.

DC Characteristics (2/5)**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|----------------------|-----------|--|-------------------------------|------|----------------|------|
| Input voltage, high | V_{IH1} | P40, P41, P121, P122 | $0.7V_{DD}$ | | V_{DD} | V |
| | V_{IH2} | P00, P01, P10 to P17, P30 to P33, P70, P71, P120, RESET | $0.8V_{DD}$ | | V_{DD} | V |
| | V_{IH3} | P20 to P23 | $0.7V_{DD}$ | | V_{DD} | V |
| | V_{IH4} | P60, P61 | $0.7V_{DD}$ | | 6.0 | V |
| Input voltage, low | V_{IL1} | P40, P41, P60, P61, P121, P122 | 0 | | $0.3V_{DD}$ | V |
| | V_{IL2} | P00, P01, P10 to P17, P30 to P33, P70, P71, P120, RESET | 0 | | $0.2V_{DD}$ | V |
| | V_{IL3} | P20 to P23 | 0 | | $0.3V_{DD}$ | V |
| Output voltage, high | V_{OH1} | P00, P01, P10 to P17, P30 to P33, P40, P41, P70, P71, P120 | $I_{OH1} = -3.0\text{ mA}$ | | $V_{DD} - 0.7$ | V |
| | V_{OH2} | P20 to P23 P121, P122 | $I_{OH2} = -100\ \mu\text{A}$ | | $V_{DD} - 0.5$ | V |

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (3/5)**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit | |
|-----------------------------|------------|---|----------------------------|---------------|------|-------------|------------------|---------------|
| Output voltage, low | V_{OL1} | P00, P01, P10 to P17, P30 to P33, P40, P41, P70, P71, P120 | $I_{OL1} = 8.5\text{ mA}$ | | | 0.7 | V | |
| | V_{OL2} | P20 to P23 | $I_{OL2} = 0.4\text{ mA}$ | | | 0.4 | V | |
| | | P121, P122 | | | | | | |
| | V_{OL3} | P60, P61 | $I_{OL1} = 15.0\text{ mA}$ | | | 2.0 | V | |
| $I_{OL1} = 5.0\text{ mA}$ | | | | | 0.4 | V | | |
| Input leakage current, high | I_{LIH1} | P00, P01, P10 to P17, P30 to P33, P40, P41, P60, P61, P70, P71, P120, FLMD0, RESET | $V_I = V_{DD}$ | | | 1 | μA | |
| | I_{LIH2} | P20, P23 | $V_I = V_{DD}$ | | | 1 | μA | |
| | I_{LIH3} | P121, P122 (X1, X2) | $V_I = V_{DD}$ | I/O port mode | | | 1 | μA |
| | | | | OSC mode | | | 20 | μA |
| Input leakage current, low | I_{LIL1} | P00, P01, P10 to P17, P30 to P33, P40, P41, P60, P61, P70, P71, P120, FLMD0, RESET | $V_I = V_{SS}$ | | | -1 | μA | |
| | I_{LIL2} | P20 to P23 | $V_I = V_{SS}$ | | | -1 | μA | |
| | I_{LIL3} | P121, P122 (X1, X2) | $V_I = V_{SS}$ | I/O port mode | | | -1 | μA |
| | | | | OSC mode | | | -20 | μA |
| Pull-up resistor | R_U | $V_I = V_{SS}$ | | 10 | 20 | 100 | $\text{k}\Omega$ | |
| FLMD0 supply voltage | V_{IL} | In normal operation mode | | 0 | | $0.2V_{DD}$ | V | |
| | V_{IH} | | | $0.8V_{DD}$ | | V_{DD} | V | |

Remark Unless specified otherwise, the characteristics of alternate-function pins are the same as those of port pins.

DC Characteristics (4/5)**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Parameter | Symbol | Conditions | | MIN. | TYP. | MAX. | Unit | |
|------------------------------------|------------------|--------------------------------|--|----------------------|------|------|------|----|
| Supply current ^{Note 1} | I _{DD1} | Operating mode | f _{XH} = 10 MHz, V _{DD} = 5.0 V ^{Note 2} | Square wave input | | 1.6 | 2.8 | mA |
| | | | | Resonator connection | | 2.3 | 3.9 | mA |
| | | | f _{RH} = 8 MHz, V _{DD} = 5.0 V ^{Note 3} | | | 1.4 | 2.5 | mA |
| | I _{DD2} | HALT mode | f _{XH} = 10 MHz, V _{DD} = 5.0 V ^{Note 2} | Square wave input | | 0.4 | 1.3 | mA |
| | | | | Resonator connection | | 1.0 | 2.4 | mA |
| | | | f _{RH} = 8 MHz, V _{DD} = 5.0 V ^{Note 3} | | | 0.4 | 1.2 | mA |
| I _{DD3} ^{Note 4} | STOP mode | T _A = +70 to +85 °C | | | – | 20 | μA | |
| | | T _A = –40 to +70 °C | | | 1 | 10 | μA | |

- Notes**
1. Total current flowing into the internal power supply (V_{DD}), including the peripheral operation current and the input leakage current flowing when the level of the input pin is fixed to V_{DD} or V_{SS}. However, the current flowing into the pull-up resistors and the output current of the port are not included.
 2. Not including the operating current of the 8 MHz internal oscillator and 240 kHz internal oscillator, and the current flowing into the A/D converter, watchdog timer, and LVI circuit.
 3. Not including the operating current of the X1 oscillator and 240 kHz internal oscillator, and the current flowing into the A/D converter, watchdog timer, and LVI circuit.
 4. Not including the operating current of the 240 kHz internal oscillator and the current flowing into the A/D converter, watchdog timer, and LVI circuit.

- Remarks**
1. f_{XH}: High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)
 2. f_{RH}: Internal high-speed oscillation clock frequency

DC Characteristics (5/5)**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|----------------------------------|-----------------------------|---|------|------|------|---------------|
| A/D converter operating current | I_{ADC} ^{Note 1} | ADCS = 1 | | 0.86 | 1.9 | mA |
| Watchdog timer operating current | I_{WDT} ^{Note 2} | During 240 kHz internal low-speed oscillation clock operation | | 5 | 10 | μA |
| LVI operating current | I_{LVI} ^{Note 3} | | | 9 | 18 | μA |

- Notes**
1. Current flowing only to the A/D converter. The current value of the R7F0C011B, R7F0C012B, and R7F0C013B is the sum of I_{DD1} or I_{DD2} and I_{ADC} when the A/D converter operates in an operation mode or the HALT mode.
 2. Current flowing only to the watchdog timer (including the operating current of the 240 kHz internal oscillator). The current value of the R7F0C011B, R7F0C012B, and R7F0C013B is the sum of I_{DD1} , I_{DD2} or I_{DD3} and I_{WDT} when the watchdog timer operates.
 3. Current flowing only to the LVI circuit. The current value of the R7F0C011B, R7F0C012B, and R7F0C013B is the sum of I_{DD1} , I_{DD2} or I_{DD3} and I_{LVI} when the LVI circuit operates.

- Remarks**
1. f_{XH} : High-speed system clock frequency (X1 clock oscillation frequency or external main system clock frequency)
 2. f_{RH} : Internal high-speed oscillation clock frequency

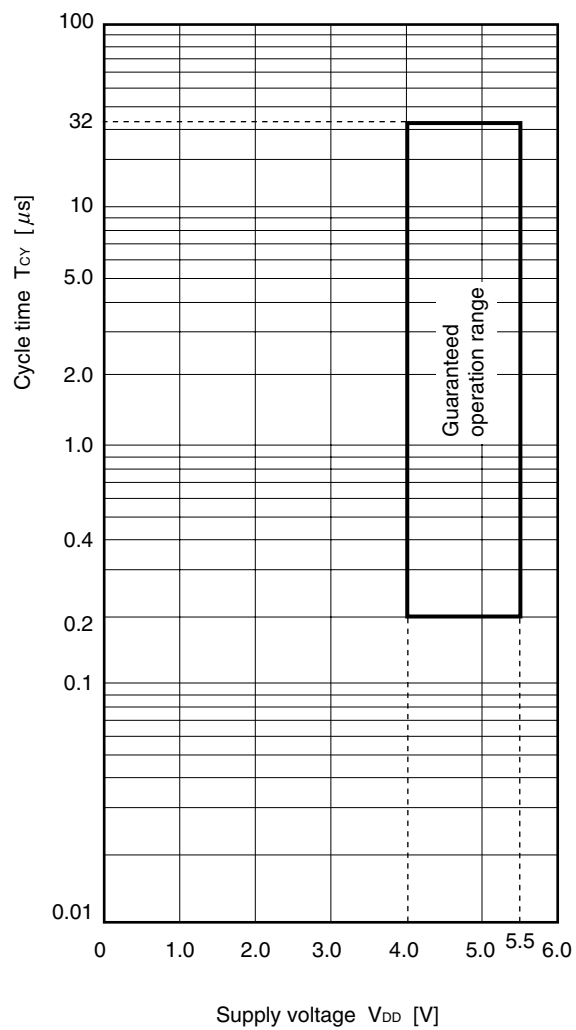
AC Characteristics

(1) Basic operation

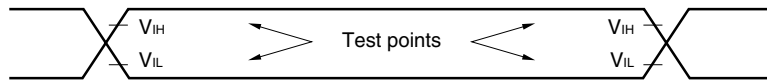
(T_A = -40 to +85°C, 4.0 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|--|--|--|---|------|------|------|
| Instruction cycle (minimum instruction execution time) | T _{CY} | Main system clock (f _{XP}) operation | 0.2 | | 32 | μs |
| Peripheral hardware clock frequency | f _{PRS} | f _{PRS} = f _{XH} (XSEL = 1) | | | 10 | MHz |
| | | f _{PRS} = f _{RH} (XSEL = 0) | 7.6 | | 8.4 | MHz |
| External main system clock frequency | f _{EXCLK} | | 1.0 ^{Note 1} | | 10.0 | MHz |
| External main system clock input high-level width, low-level width | t _{EXCLKH} , t _{EXCLKL} | | 48 | | | ns |
| TI000, TI010 input high-level width, low-level width | t _{TIH0} , t _{TIL0} | | 2/f _{sam} + 0.1 ^{Note 2} | | | μs |
| TI50, TI51 input frequency | f _{TI5} | | | | 10 | MHz |
| TI50, TI51 input high-level width, low-level width | t _{TIH5} , t _{TIL5} | | 50 | | | ns |
| Interrupt input high-level width, low-level width | t _{INTH} , t _{INTL} | | 1 | | | μs |
| RESET low-level width | t _{RSL} | | 10 | | | μs |

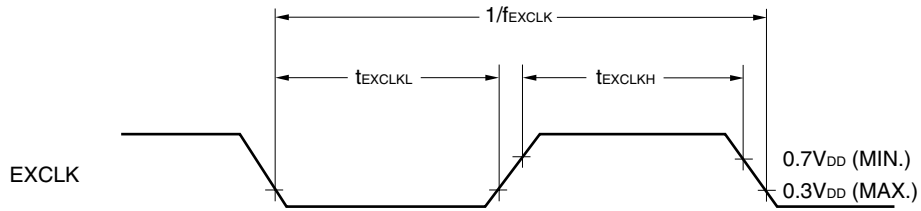
- Notes**
- 2.0 MHz (MIN.) when using UART6 during on-board programming.
 - Selection of f_{sam} = f_{PRS}, f_{PRS}/4, f_{PRS}/256, or f_{PRS}, f_{PRS}/16, f_{PRS}/64 is possible using bits 0 and 1 (PRM000, PRM001) of prescaler mode register 00 (PRM00). Note that when selecting the TI000 valid edge as the count clock, f_{sam} = f_{PRS}.

T_{cy} vs. V_{DD} (Main System Clock Operation)

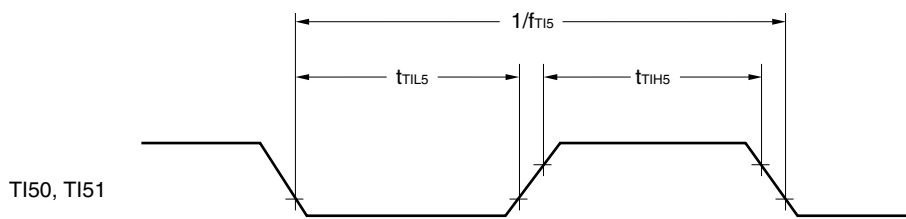
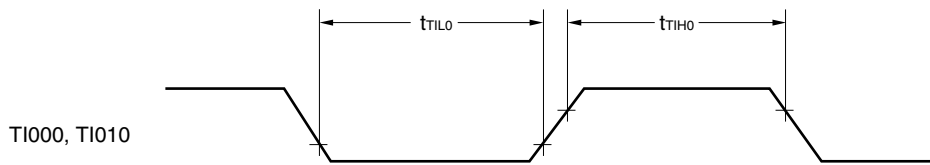
AC Timing Test Points



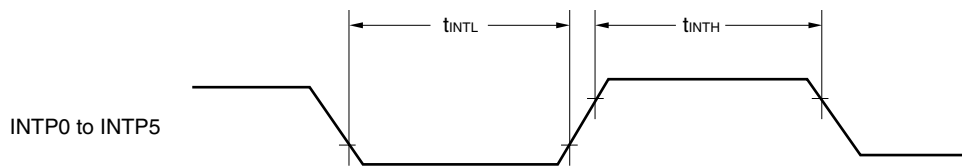
External Main System Clock Timing



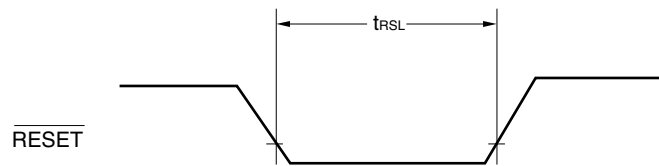
TI Timing



Interrupt Request Input Timing



RESET Input Timing



(2) Serial interface(T_A = -40 to +85°C, 4.0 V ≤ V_{DD} ≤ 5.5 V, V_{SS} = 0 V)**(a) UART6 (dedicated baud rate generator output)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---------------|--------|------------|------|------|------|------|
| Transfer rate | | | | | 625 | kbps |

(b) UART0 (dedicated baud rate generator output)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---------------|--------|------------|------|------|------|------|
| Transfer rate | | | | | 625 | kbps |

(c) IIC0

| Parameter | Symbol | Conditions | Standard Mode | | High-Speed Mode | | Unit | |
|---|----------------------|--|---------------|------|-----------------|-------------------------|------------------------|----|
| | | | MIN. | MAX. | MIN. | MAX. | | |
| SCL0 clock frequency | f _{SCL} | | 0 | 100 | 0 | 400 | kHz | |
| Setup time of restart condition | t _{SU: STA} | | 4.7 | – | 0.6 | – | μs | |
| Hold time ^{Note 1} | t _{HD: STA} | | 4.0 | – | 0.6 | – | μs | |
| Hold time when SCL0 = “L” | t _{LOW} | Internal clock operation | 4.7 | – | 1.3 | – | μs | |
| Hold time when SCL0 = “H” | t _{HIGH} | | 4.0 | – | 0.6 | – | μs | |
| Data setup time (reception) | t _{SU: DAT} | | 250 | – | 100 | – | ns | |
| Data hold time (transmission) ^{Note 2} | t _{HD: DAT} | f _w = f _{XH} /2 ^N selected ^{Note 3} | DFC0 = 0 | 0 | 3.45 | 0 | 0.9 ^{Note 4} | μs |
| | | | | | | | 1.00 ^{Note 5} | |
| | | DFC0 = 1 | – | – | 0 | 0.9 ^{Note 6} | μs | |
| | | | | | | 1.125 ^{Note 7} | | |
| | | f _w = f _{RH} /2 ^N selected ^{Note 3} | DFC0 = 0 | 0 | 3.45 | 0 | 1.05 | μs |
| | | | DFC0 = 1 | – | – | 0 | 1.184 | |
| Setup time of stop condition | t _{SU: STO} | | 4.0 | – | 0.6 | – | μs | |
| Bus free time | t _{BUF} | | 4.7 | – | 1.3 | – | μs | |

- Notes**
- The first clock pulse is generated after this period when the start/restart condition is detected.
 - The maximum value (MAX.) of t_{HD: DAT} is during normal transfer and a wait state is inserted in the $\overline{\text{ACK}}$ (acknowledge) timing.
 - f_w indicates the IIC0 transfer clock selected by the IICCL0 register.
 - When f_w ≥ 4.4 MHz is selected
 - When f_w < 4.4 MHz is selected
 - When f_w ≥ 5.0 MHz is selected
 - When f_w < 5.0 MHz is selected

(d) CSI10 (master mode, $\overline{\text{SCK10}}$... internal clock output)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|--|---------------------------------------|-------------------------------------|--|------|------|------|
| $\overline{\text{SCK10}}$ cycle time | t_{CY1} | | 200 | | | ns |
| $\overline{\text{SCK10}}$ high-/low-level width | $t_{\text{KH1}},$ t_{KL1} | | $t_{\text{CY1}}/2 -$ $15^{\text{Note 1}}$ | | | ns |
| SI10 setup time (to $\overline{\text{SCK10}}\uparrow$) | t_{SIK1} | | 55 | | | ns |
| SI10 hold time (from $\overline{\text{SCK10}}\uparrow$) | t_{SII1} | | 30 | | | ns |
| Delay time from $\overline{\text{SCK10}}\downarrow$ to SO10 output | t_{KSO1} | $C = 50 \text{ pF}^{\text{Note 2}}$ | | | 40 | ns |

- Notes**
1. This value is when high-speed system clock (f_{XH}) is used.
 2. C is the load capacitance of the $\overline{\text{SCK10}}$ and SO10 output lines.

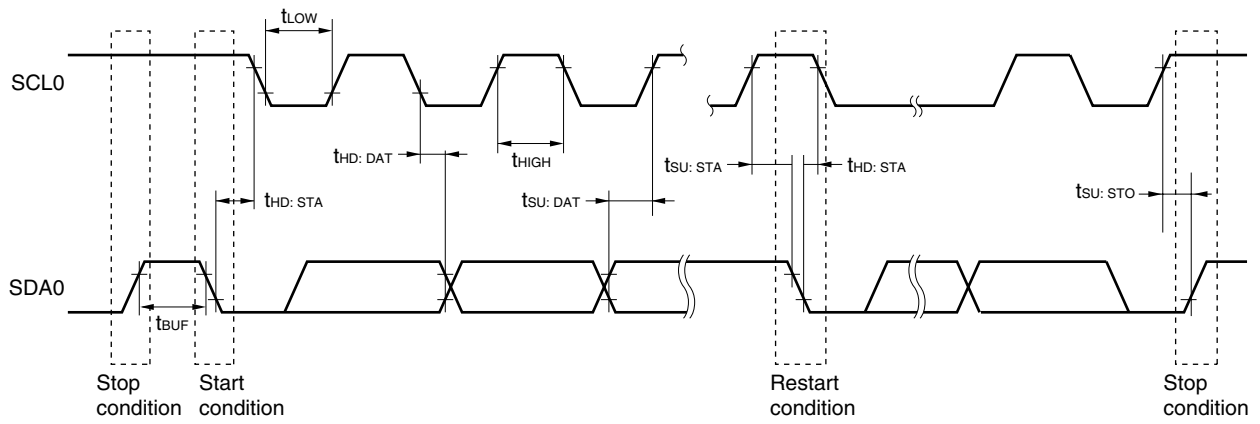
(e) CSI10 (slave mode, $\overline{\text{SCK10}}$... external clock input)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|--|---------------------------------------|-----------------------------------|--------------------|------|------|------|
| $\overline{\text{SCK10}}$ cycle time | t_{CY2} | | 400 | | | ns |
| $\overline{\text{SCK10}}$ high-/low-level width | $t_{\text{KH2}},$ t_{KL2} | | $t_{\text{CY2}}/2$ | | | ns |
| SI10 setup time (to $\overline{\text{SCK10}}\uparrow$) | t_{SIK2} | | 80 | | | ns |
| SI10 hold time (from $\overline{\text{SCK10}}\uparrow$) | t_{SII2} | | 50 | | | ns |
| Delay time from $\overline{\text{SCK10}}\downarrow$ to SO10 output | t_{KSO2} | $C = 50 \text{ pF}^{\text{Note}}$ | | | 120 | ns |

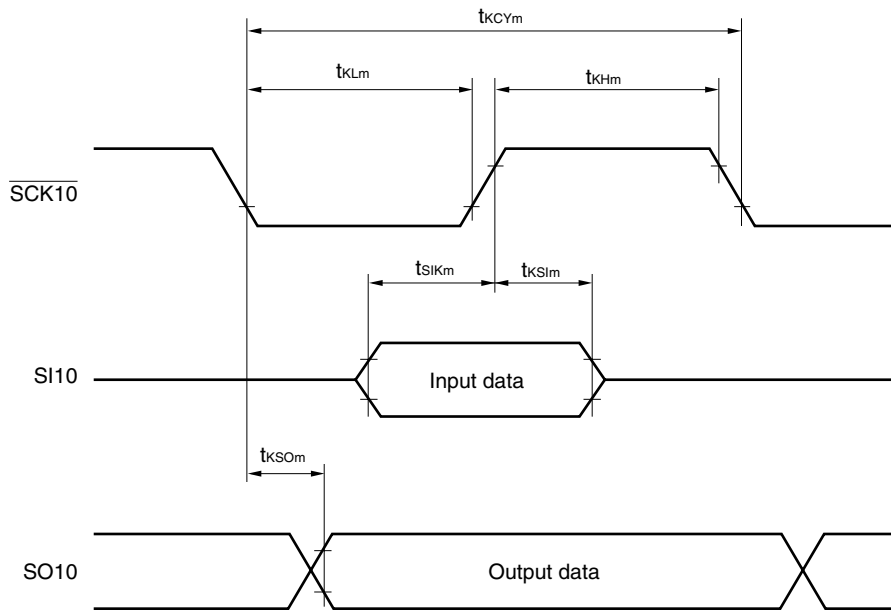
Note C is the load capacitance of the SO10 output line.

Serial Transfer Timing

IIC0:



CSI10:



Remark m = 1, 2

A/D Converter Characteristics**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)**

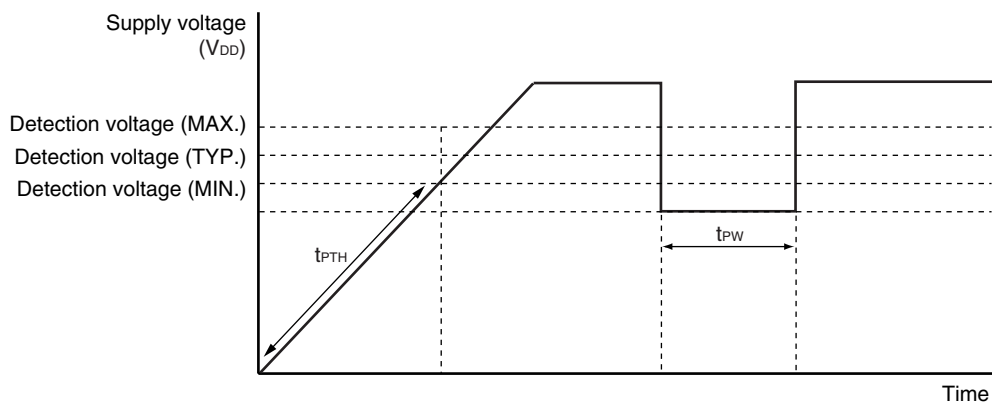
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|-------------------------------------|-------------------|------------|-----------------|------|-----------------|------|
| Resolution | RES | | 10 | | | bit |
| Overall error ^{Notes 1, 2} | A _{INL} | | | | ±0.4 | %FSR |
| Conversion time | t _{CONV} | | 6.1 | | 66.6 | μs |
| Analog input voltage | V _{AIN} | | V _{SS} | | V _{DD} | V |

Notes 1. Excludes quantization error ($\pm 1/2$ LSB).

2. This value is indicated as a ratio (%FSR) to the full-scale value.

1.59 V POC Circuit Characteristics ($T_A = -40$ to $+85^\circ\text{C}$, $V_{SS} = V_{SS} = 0\text{ V}$)

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---------------------------------------|------------------|--|------|------|------|------|
| Detection voltage | V _{POC} | | 1.44 | 1.59 | 1.74 | V |
| Power supply voltage rise inclination | t _{PTH} | V _{DD} : 0 V → change inclination of V _{POC} | 0.5 | | | V/ms |
| Minimum pulse width | t _{PW} | | 200 | | | μs |

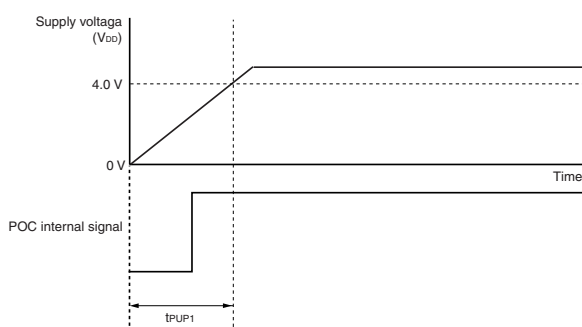
1.59 V POC Circuit Timing

Supply Voltage Rise Time ($T_A = -40$ to $+85^\circ\text{C}$, $V_{SS} = 0$ V)

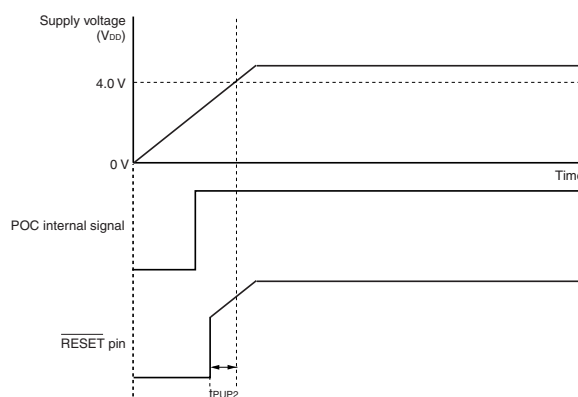
| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|------------|---|------|------|------|------|
| Maximum time to rise to 4.0 V (V_{DD} (MIN.)) (V_{DD} : 0 V \rightarrow 4.0 V) | t_{PUP1} | POCMODE (option byte) = 0, when $\overline{\text{RESET}}$ input is not used | | | 3.6 | ms |
| Maximum time to rise to 4.0 V (V_{DD} (MIN.)) (releasing $\overline{\text{RESET}}$ input \rightarrow V_{DD} : 4.0 V) | t_{PUP2} | POCMODE (option byte) = 0, when $\overline{\text{RESET}}$ input is used | | | 1.9 | ms |

Supply Voltage Rise Time Timing

- When $\overline{\text{RESET}}$ pin input is not used



- When $\overline{\text{RESET}}$ pin input is used

**2.7 V POC Circuit Characteristics ($T_A = -40$ to $+85^\circ\text{C}$, $V_{SS} = 0$ V)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|--|-------------|---------------------------|------|------|------|------|
| Detection voltage on application of supply voltage | V_{DDPOC} | POCMODE (option byte) = 1 | 2.50 | 2.70 | 2.90 | V |

Remark The operations of the POC circuit are as described below, depending on the POCMODE (option byte) setting.

| Option Byte Setting | POC Mode | Operation |
|---------------------|-----------------------------|--|
| POCMODE = 0 | 1.59 V mode operation | A reset state is retained until $V_{POC} = 1.59$ V (TYP.) is reached after the power is turned on, and the reset is released when V_{POC} is exceeded. After that, POC detection is performed at V_{POC} , similarly as when the power was turned on. The power supply voltage must be raised at a time of t_{PUP1} or t_{PUP2} when POCMODE is 0. |
| POCMODE = 1 | 2.7 V/1.59 V mode operation | A reset state is retained until $V_{DDPOC} = 2.7$ V (TYP.) is reached after the power is turned on, and the reset is released when V_{DDPOC} is exceeded. After that, POC detection is performed at $V_{POC} = 1.59$ V (TYP.) and not at V_{DDPOC} . The use of the 2.7 V/1.59 V POC mode is recommended when the rise of the voltage, after the power is turned on and until the voltage reaches 4.0 V, is more relaxed than t_{PTH} . |

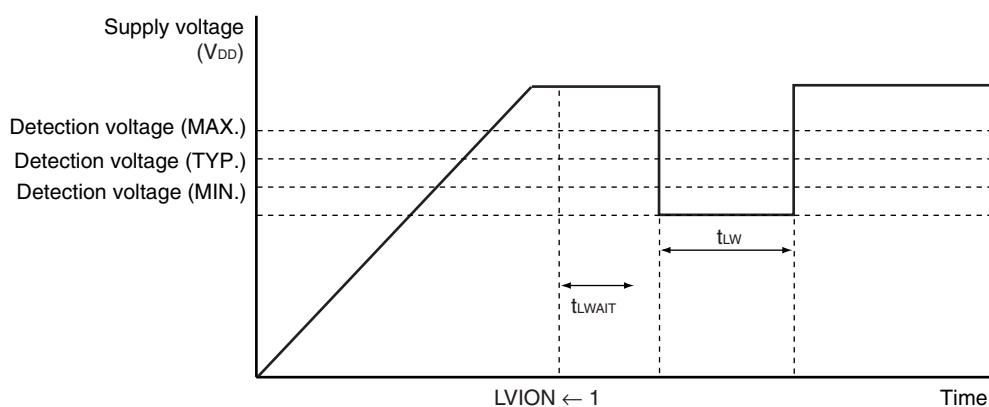
LVI Circuit Characteristics ($T_A = -40$ to $+85^\circ\text{C}$, $V_{POC} \leq V_{DD} \leq 5.5$ V, $V_{SS} = 0$ V)

| Parameter | | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|---|--------------------------------------|-------------|---|------|------|------|---------------|
| Detection voltage | Supply voltage level | V_{LV10} | | 4.14 | 4.24 | 4.34 | V |
| | | V_{LV11} | | 3.99 | 4.09 | 4.19 | V |
| | External input pin ^{Note 1} | EXLVI | $EXLVI < V_{DD}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ | 1.11 | 1.21 | 1.31 | V |
| Minimum pulse width | | t_{LW} | | 200 | | | μs |
| Operation stabilization wait time ^{Note 2} | | t_{LWAIT} | | 10 | | | μs |

Notes 1. The EXLVI/P120/INTP0 pin is used.

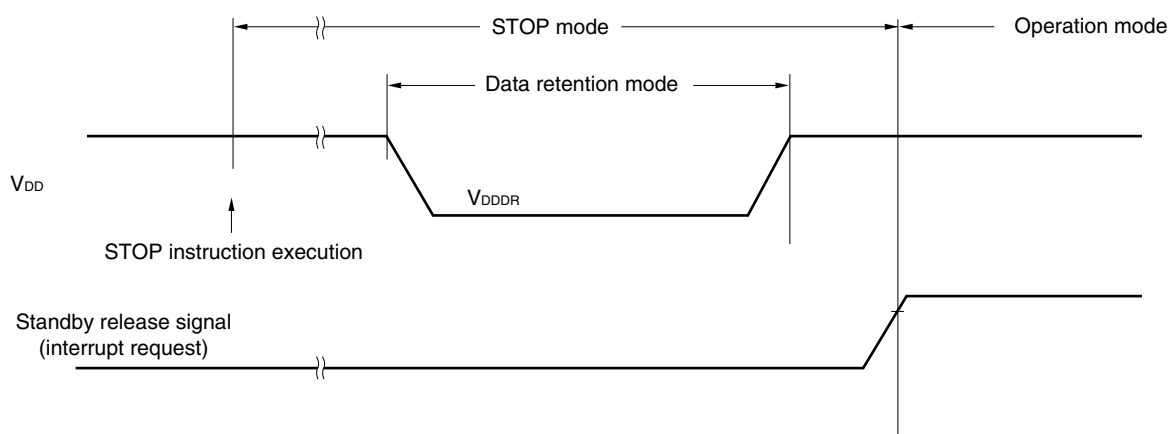
2. Time required from setting bit 7 (LVION) of the low-voltage detection register (LVIM) to 1 to operation stabilization

Remark $V_{LV10} > V_{LV11}$

LVI Circuit Timing**Data Memory STOP Mode Low Supply Voltage Data Retention Characteristics ($T_A = -40$ to $+85^\circ\text{C}$)**

| Parameter | Symbol | Conditions | MIN. | TYP. | MAX. | Unit |
|-------------------------------|------------|------------|----------------------|------|------|------|
| Data retention supply voltage | V_{DDDR} | | 1.44 ^{Note} | | 5.5 | V |

Note The value depends on the POC detection voltage. When the voltage drops, the data is retained until a POC reset is effected, but data is not retained when a POC reset is effected.



Flash Memory Programming Characteristics**($T_A = -40$ to $+85^\circ\text{C}$, $4.0\text{ V} \leq V_{DD} \leq 5.5\text{ V}$, $V_{SS} = 0\text{ V}$)****• Basic characteristics**

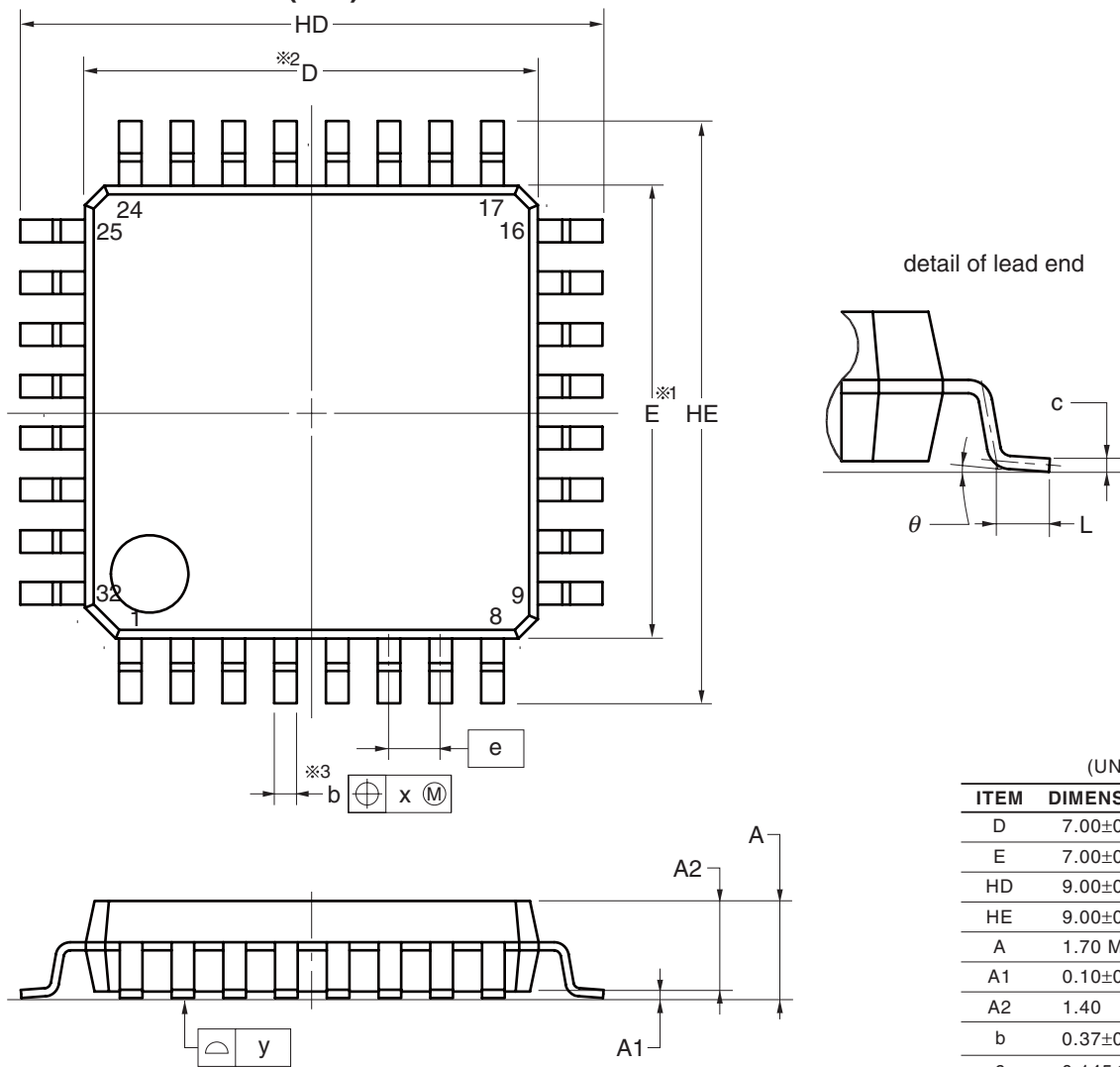
| Parameter | Symbol | Conditions | | | MIN. | TYP. | MAX. | Unit |
|-----------------------------|------------|---|--|---------------------|------|------|------|-------|
| V_{DD} supply current | I_{DD} | | | | | 4.5 | 11.0 | mA |
| Number of rewrites per chip | C_{erwr} | 1 erase + 1 write after erase = 1 rewrite ^{Note} | When a flash memory programmer is used, for program update | Retention: 15 years | 10 | | | Times |

Note When a product is first written after shipment, “erase → write” and “write only” are both taken as one rewrite.

CHAPTER 26 PACKAGE DRAWINGS

- R7F0C011B2001DFP
R7F0C012B2001DFP
R7F0C013B2001DFP
R7F0C999B2DFP

32-PIN PLASTIC LQFP(7x7)



NOTE

1. Dimensions "※1" and "※2" do not include mold flash.
2. Dimension "※3" does not include trim offset.

(UNIT:mm)

| ITEM | DIMENSIONS |
|------|-------------|
| D | 7.00±0.10 |
| E | 7.00±0.10 |
| HD | 9.00±0.20 |
| HE | 9.00±0.20 |
| A | 1.70 MAX. |
| A1 | 0.10±0.10 |
| A2 | 1.40 |
| b | 0.37±0.05 |
| c | 0.145±0.055 |
| L | 0.50±0.20 |
| θ | 0° to 8° |
| e | 0.80 |
| x | 0.20 |
| y | 0.10 |

P32GA-80-GBT

CHAPTER 27 CAUTIONS FOR WAIT

27.1 Cautions for Wait

This product has two internal system buses.

One is a CPU bus and the other is a peripheral bus that interfaces with the low-speed peripheral hardware.

Because the clock of the CPU bus and the clock of the peripheral bus are asynchronous, unexpected illegal data may be passed if an access to the CPU conflicts with an access to the peripheral hardware.

When accessing the peripheral hardware that may cause a conflict, therefore, the CPU repeatedly executes processing, until the correct data is passed.

As a result, the CPU does not start the next instruction processing but waits. If this happens, the number of execution clocks of an instruction increases by the number of wait clocks (for the number of wait clocks, see **Table 27-1**). This must be noted when real-time processing is performed.

27.2 Peripheral Hardware That Generates Wait

Table 27-1 lists the registers that issue a wait request when accessed by the CPU, and the number of CPU wait clocks.

Table 27-1. Registers That Generate Wait and Number of CPU Wait Clocks

| Peripheral Hardware | Register | Access | Number of Wait Clocks |
|---|----------|--------|---|
| Serial interface UART0 | ASIS0 | Read | 1 clock (fixed) |
| Serial interface UART6 | ASIS6 | Read | 1 clock (fixed) |
| Serial interface IIC0 | IICS0 | Read | 1 clock (fixed) |
| A/D converter | ADM | Write | 1 to 5 clocks (when $f_{AD} = f_{PRS}/2$ is selected) |
| | ADS | Write | 1 to 7 clocks (when $f_{AD} = f_{PRS}/3$ is selected) |
| | ADPC | Write | 1 to 9 clocks (when $f_{AD} = f_{PRS}/4$ is selected) |
| | ADCR | Read | 2 to 13 clocks (when $f_{AD} = f_{PRS}/6$ is selected) 2 to 17 clocks (when $f_{AD} = f_{PRS}/8$ is selected) 2 to 25 clocks (when $f_{AD} = f_{PRS}/12$ is selected) |
| <p>The above number of clocks is when the same source clock is selected for f_{CPU} and f_{PRS}. The number of wait clocks can be calculated by the following expression and under the following conditions.</p> <p><Calculating number of wait clocks></p> <ul style="list-style-type: none"> Number of wait clocks = $\frac{2 f_{CPU}}{f_{AD}} + 1$ <p>* Fraction is truncated if the number of wait clocks ≤ 0.5 and rounded up if the number of wait clocks > 0.5.</p> <p>f_{AD}: A/D conversion clock frequency ($f_{PRS}/2$ to $f_{PRS}/12$) f_{CPU}: CPU clock frequency f_{PRS}: Peripheral hardware clock frequency f_{XP}: Main system clock frequency</p> <p><Conditions for maximum/minimum number of wait clocks></p> <ul style="list-style-type: none"> Maximum number of times: Maximum speed of CPU (f_{XP}), lowest speed of A/D conversion clock ($f_{PRS}/12$) Minimum number of times: Minimum speed of CPU ($f_{XP}/16$), highest speed of A/D conversion clock ($f_{PRS}/2$) | | | |

Caution When the peripheral hardware clock (f_{PRS}) is stopped, do not access the registers listed above using an access method in which a wait request is issued.

Remark The clock is the CPU clock (f_{CPU}).

R7F0C011B, R7F0C012B, R7F0C013B
User's Manual: Hardware

Publication Date: Rev.1.00 Jan 31, 2013

Published by: Renesas Electronics Corporation

**SALES OFFICES**

Renesas Electronics Corporation

<http://www.renesas.com>Refer to "<http://www.renesas.com/>" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852-2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, 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.
11F., Samik Lavied or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

R7F0C011B, R7F0C012B, R7F0C013B



Renesas Electronics Corporation

R01UH0408EJ0100